Cálculo de Programas

Class T01

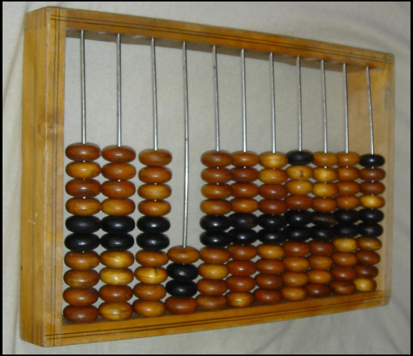
J.N. Oliveira

(Introduction to the course. Website and other administrative details.)

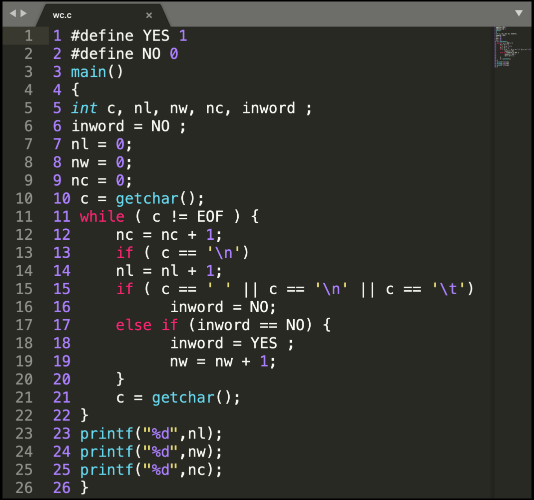
Programming
by
Calculation

Program versus Calculus

- **Calculus** — abacus **pebble**
- **Program** — *pro* (‘before’) + *graphein* (‘write’)

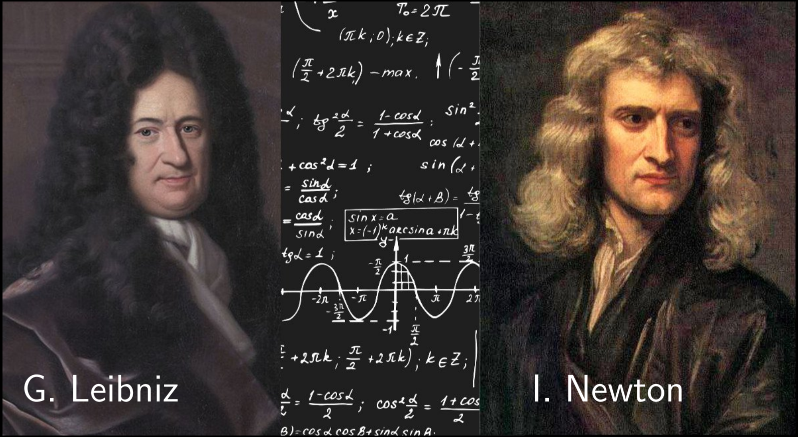


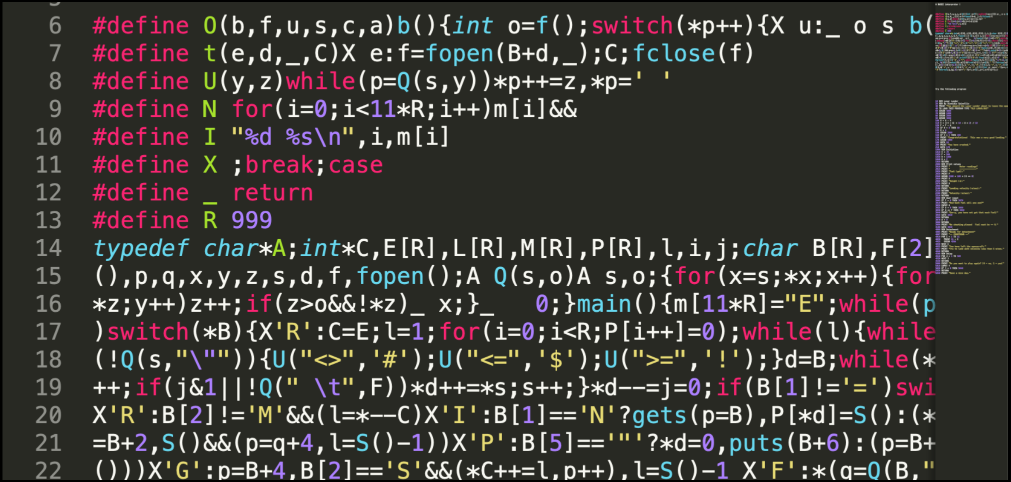
Programs…



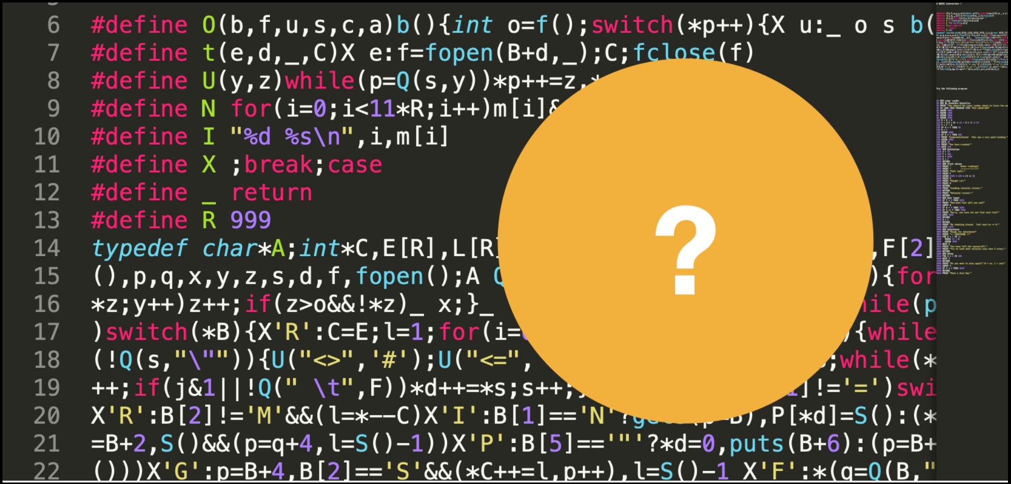
```c
#define YES 1
#define NO 0
main()
{
int c, nl, nw, nc, inword ;
inword = NO ;
nl = 0;
nw = 0;
nc = 0;
c = getchar();
while ( c != EOF ) {
    nc = nc + 1;
    if ( c == '\n')
    nl = nl + 1;
    if ( c == ' ' || c == '\n' || c == '\t')
          inword = NO;
    else if (inword == NO) {
          inword = YES ;
          nw = nw + 1;
    }
    c = getchar();
}
printf("%d",nl);
printf("%d",nw);
printf("%d",nc);
}
```

Calculus





```
#define O(b,f,u,s,c,a)b(){int o=f();switch(*p++){X u:_ o s b(
#define t(e,d,_,C)X e:f=fopen(B+d,_);C;fclose(f)
#define U(y,z)while(p=Q(s,y))*p++=z,*p=' '
#define N for(i=0;i<11*R;i++)m[i]&&
#define I "%d %s\n",i,m[i]
#define X ;break;case
#define _ return
#define R 999
typedef char*A;int*C,E[R],L[R],M[R],P[R],l,i,j;char B[R],F[2]
(),p,q,x,y,z,s,d,f,fopen();A Q(s,o)A s,o;{for(x=s;*x;x++){for
*z;y++)z++;if(z>o&&!*z)_ x;}_   0;}main(){m[11*R]="E";while(p
)switch(*B){X'R':C=E;l=1;for(i=0;i<R;P[i++]=0);while(l){while
(!Q(s,"\"")){U("<>",'#');U("<=",'$');U(">=",'!');}d=B;while(*
++;if(j&1||!Q(" \t",F))*d++=*s;s++;}*d--=j=0;if(B[1]!='=')swi
X'R':B[2]!='M'&&(l=*--C)X'I':B[1]=='N'?gets(p=B),P[*d]=S():(*
=B+2,S()&&(p=q+4,l=S()-1))X'P':B[5]=='"'?*d=0,puts(B+6):(p=B+
()))X'G':p=B+4,B[2]=='S'&&(*C++=l,p++),l=S()-1 X'F':*(q=Q(B,"
```

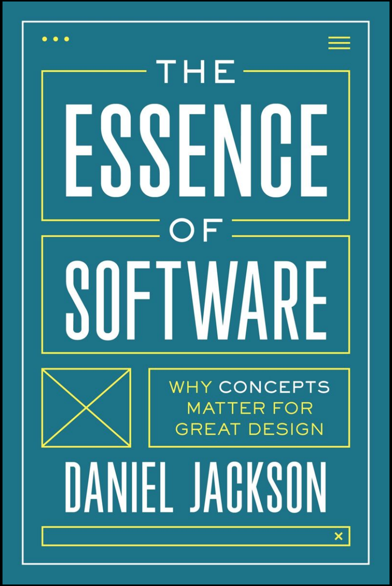


```
#define O(b,f,u,s,c,a)b(){int o=f();switch(*p++){X u:_ o s b(
#define t(e,d,_,C)X e:f=fopen(B+d,_);C;fclose(f)
#define U(y,z)while(p=Q(s,y))*p++=z,
#define N for(i=0;i<11*R;i++)m[i]&
#define I "%d %s\n",i,m[i]
#define X ;break;case
#define _ return
#define R 999
typedef char*A;int*C,E[R],L[R                                  ,F[2]
(),p,q,x,y,z,s,d,f,fopen();A Q                                 ){for
*z;y++)z++;if(z>o&&!*z)_ x;}_                                  hile(p
)switch(*B){X'R':C=E;l=1;for(i=                                ){while
(!Q(s,"\""))){U("<>",'#');U("<=",                              ;while(*
++;if(j&1||!Q(" \t",F))*d++=*s;s++;                           ]!='=')swi
X'R':B[2]!='M'&&(l=*--C)X'I':B[1]=='N'?g          B),P[*d]=S():(*
=B+2,S()&&(p=q+4,l=S()-1))X'P':B[5]=='"'?*d=0,puts(B+6):(p=B+
()))X'G':p=B+4,B[2]=='S'&&(*C++=l,p++),l=S()-1 X'F':*(q=Q(B,"
```

Software as a problem

Software

- Process — ☑
- Product — ?

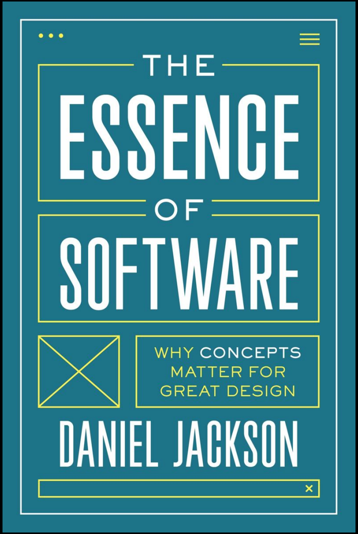


THE ESSENCE OF SOFTWARE

WHY CONCEPTS MATTER FOR GREAT DESIGN

DANIEL JACKSON

"(...) The best services revolve around **a small number of concepts** that are **well designed and easy** (...) **to understand and use**, and their innovations often involve simple but compelling new concepts."



In
The Essence of Software
by
Prof. Daniel Jackson, MIT
(2021)

… small number

… small number

… well defined

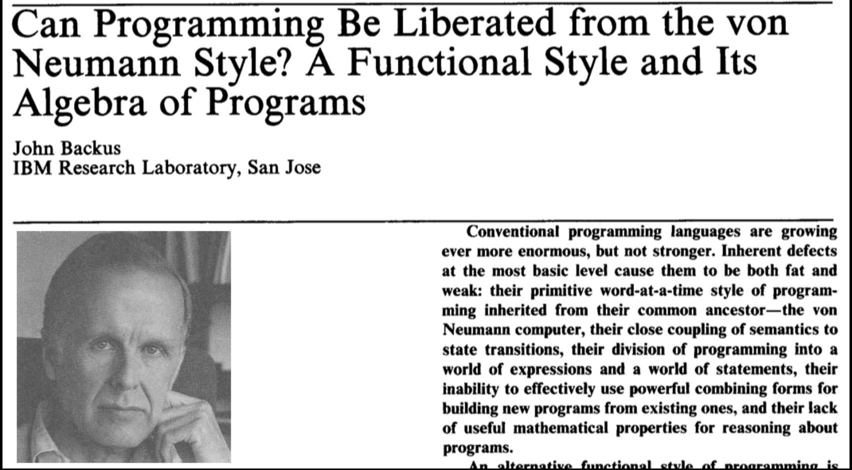
... small number

... well defined

... easy to understand

Urgently needed in software design

John Backus – Turing Award (1978)



Can Programming Be Liberated from the von Neumann Style? A Functional Style and Its Algebra of Programs

John Backus
IBM Research Laboratory, San Jose

Conventional programming languages are growing ever more enormous, but not stronger. Inherent defects at the most basic level cause them to be both fat and weak: their primitive word-at-a-time style of programming inherited from their common ancestor—the von Neumann computer, their close coupling of semantics to state transitions, their division of programming into a world of expressions and a world of statements, their inability to effectively use powerful combining forms for building new programs from existing ones, and their lack of useful mathematical properties for reasoning about programs.

An alternative functional style of programming is

Part I

Motivation — back to the late 1990s

From a mobile phone manufacturer

(...) For each **list of calls** *stored in the mobile phone (e.g. numbers dialed, SMS messages, lost calls), the* **store** *operation should work in a way such that* **(a)** *the more recently a* **call** *is made the more accessible it is;* **(b)** *no number appears twice in a list;* **(c)** *only the last 10 entries in each list are stored.*

From a mobile phone manufacturer

```
store ::  Call -> [Call] -> [Call]
store c l = take 10 (nub (c:l))
```

From a mobile phone manufacturer

```
store ::  Call -> [Call] -> [Call]
```

$$\texttt{store c l = }\underbrace{\texttt{take 10}}_{(c)}\ \texttt{(}\underbrace{\texttt{nub}}_{(b)}\ \underbrace{\texttt{(c:l)}}_{(a)}\texttt{)}$$

Compare with ...

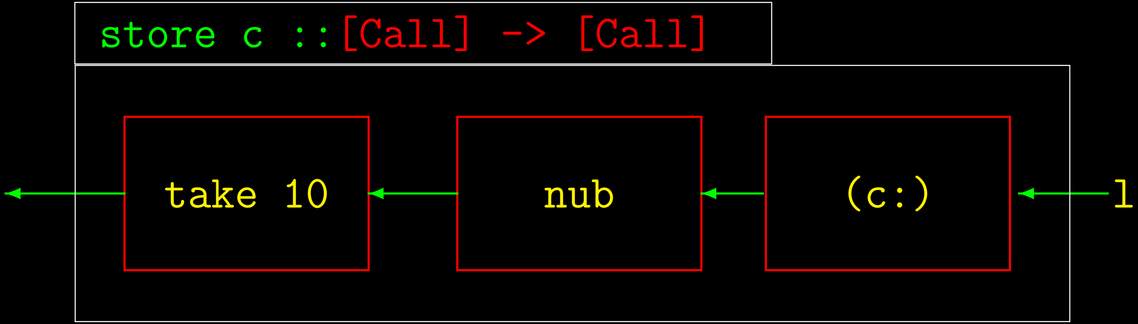
```
public void store10(string phoneNumber)
{
  System.Collections.ArrayList auxList =
      new System.Collections.ArrayList();
  auxList.Add(phoneNumber);
  auxList.AddRange(
      this.filteratmost9(phoneNumber) );
  this.callList = auxList;
}
```

+ `filteratmost9` (next slide)

Compare with ...

```
public System.Collections.ArrayList filteratmost9(string n)
{
  System.Collections.ArrayList retList =
      new System.Collections.ArrayList();
      int i=0, m=0;
  while((i < this.callList.Count) && (m < 9))
  {
      if ((string)this.callList[i] != n)
      {
          retList.Add(this.callList[i]);
          m++;
      }
      i++;
  }
  return retList;
}
```

From a mobile phone manufacturer



`store c :: [Call] -> [Call]`

take 10

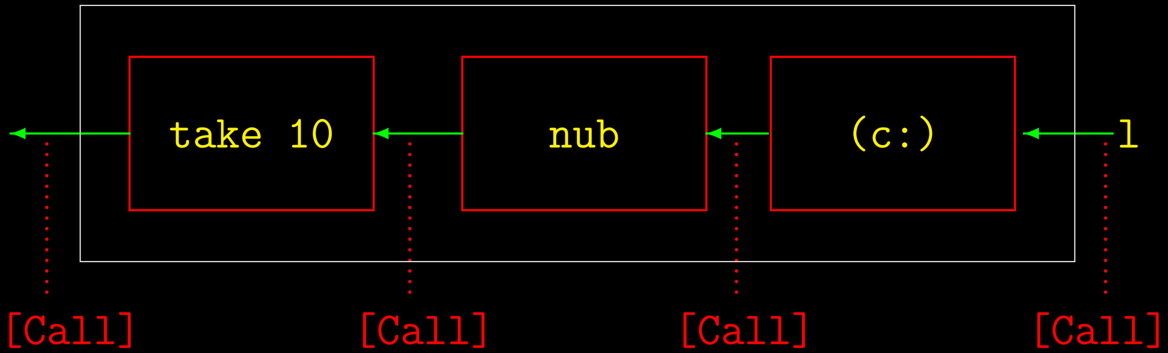
nub

(c:)

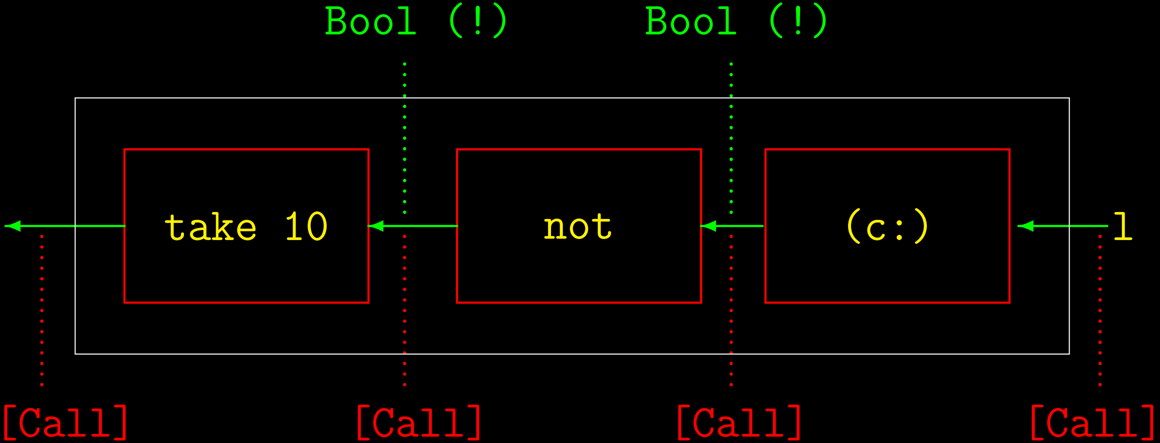
l

From a mobile phone manufacturer

`store c :: [Call] -> [Call]`



Uups!



Bool (!)

Bool (!)

take 10

not

(c:)

l

[Call]

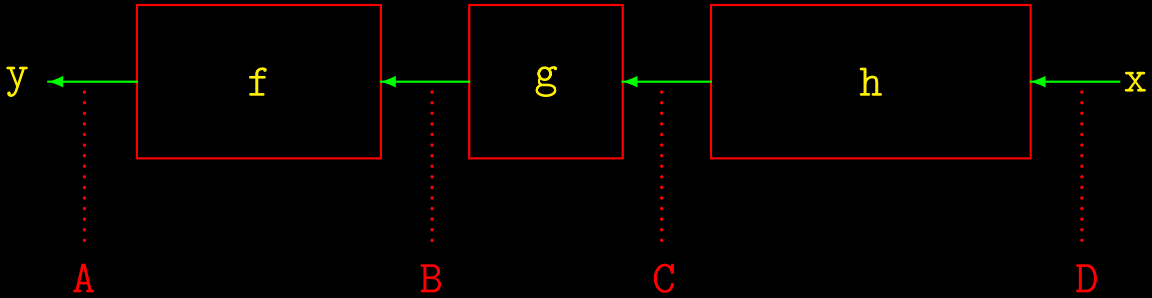
[Call]

[Call]

[Call]

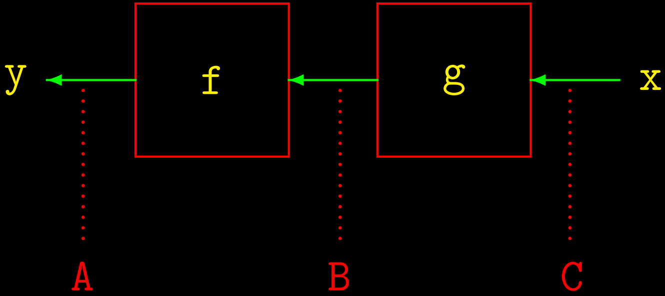
In general

`y = f(g(h x))`



In general

`y = f(g x)`



Simplification

```
y = f(g x)
```

$$\begin{tikzcd}
A & B \arrow[l, "f"'] & C \arrow[l, "g"']
\end{tikzcd}$$

Composition

$$y = f(g\ x)$$

$$\begin{tikzcd}
A & B \arrow[l, "f"'] & C \arrow[l, "g"'] \\
A & & C \arrow[ll, "f \cdot g"']
\end{tikzcd}$$

$$y = (f \cdot g)\ x$$

Composition

$$(f \cdot g) \cdot h = f \cdot (g \cdot h)$$

Composition

$$(f \cdot g) \cdot h \;=\; f \cdot (g \cdot h)$$

$$(a + b) + c \;=\; a + (b + c)$$

Composition

$$(f \cdot g) \cdot h = f \cdot (g \cdot h)$$

$$(a + b) + c = a + (b + c)$$

$$f \cdot g \cdot h$$

$$a + b + c$$

Composition

$$\mathit{store}\ c = \mathit{take}\ 10 \cdot \underbrace{\mathit{nub} \cdot (c:)}_{\mathit{store}'\ c}$$

Composition

$$store\ c = take\ 10 \cdot \underbrace{nub \cdot (c:)}_{store'\ c}$$

i.e.

$$take\ 10 \cdot (nub \cdot (c:))$$

Composition

$$store\ c = take\ 10 \cdot \underbrace{nub \cdot (c:)}_{store'\ c}$$

i.e.

$$take\ 10 \cdot (nub \cdot (c:))$$

the same as

$$(take\ 10 \cdot nub) \cdot (c:)$$

Composition

$$(f \cdot g) \cdot h \;=\; f \cdot (g \cdot h)$$

$$(a + b) + c \;=\; a + (b + c)$$

Composition

$$(f \cdot g) \cdot h \;=\; f \cdot (g \cdot h)$$

$$(a + b) + c \;=\; a + (b + c)$$

$$a + 0 = 0 + a = a$$

Composition

$$(f \cdot g) \cdot h \;=\; f \cdot (g \cdot h)$$

$$(a + b) + c \;=\; a + (b + c)$$

$$a + 0 = 0 + a = a$$

$$f \cdot ? = ? \cdot f = f$$

$$\begin{tikzcd} A & B \arrow[l, "f"'] & C \arrow[l, "g"'] \end{tikzcd}$$

$$\begin{tikzcd}C \arrow[r, "g"] & B \arrow[r, "f"] & A\end{tikzcd}$$

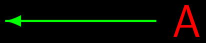
$$\begin{tikzcd}
A & & \\
& B \arrow[ul, "f"'] & \\
& & C \arrow[ul, "g"']
\end{tikzcd}$$

Cf. Unix/Linux pipes

$$\begin{tikzcd}C \arrow[r, "g"] & B \arrow[r, "f"] & A\end{tikzcd}$$

$$g \mid f$$

A


A

$A \leftarrow A$

$$\begin{tikzcd} A & A \arrow[l, "id"'] \end{tikzcd}$$

$$\begin{tikzcd}A & A \arrow[l, "id"']\end{tikzcd}$$

$$id\ a = a$$

Identity

$$\begin{tikzcd}
A \arrow[d, "f"'] & A \arrow[l, "id"'] \\
B &
\end{tikzcd}$$

Identity

$$\begin{tikzcd}
A \arrow[d, "f"'] & A \arrow[l, "id"'] \arrow[d, "f"] \\
B & B
\end{tikzcd}$$

Identity

$$\begin{tikzcd}
A \arrow[d, "f"'] & A \arrow[l, "id"'] \arrow[d, "f"] \\
B & B \arrow[l, "id"]
\end{tikzcd}$$

Identity

$$\begin{tikzcd}
A \arrow[d, "f"'] & A \arrow[l, "id"'] \arrow[d, "f"] \arrow[dl, "f"'] \\
B & B \arrow[l, "id"]
\end{tikzcd}$$

Identity

$$\begin{tikzcd}
A \arrow[d, "f"'] & A \arrow[l, "id"'] \arrow[d, "f"] \arrow[dl, "f"'] \\
B & B \arrow[l, "id"]
\end{tikzcd}$$

$$f \cdot id = f = id \cdot f$$

Composition and identity

Associativity:

$$(f \cdot g) \cdot h = f \cdot (g \cdot h)$$

Composition and identity

Associativity:

$$(f \cdot g) \cdot h = f \cdot (g \cdot h)$$

" Natural-*id*" :

$$f \cdot id = f = id \cdot f$$

Cálculo
de
Programas

Class T02

$$f \cdot g$$

$$f \cdot g$$

$$f \times g\ ?$$

$$f \cdot g$$

$$f \times g \ ?$$

$$f + g \ ?$$

$$C \xrightarrow{f} B \quad \text{and} \quad A \xrightarrow{g} C$$

$C \xrightarrow{f} B$ and $A \xrightarrow{g} C$

Composition: $A \xrightarrow{f \cdot g} B$

$C \xrightarrow{f} B$ and $A \xrightarrow{g} C$

Composition: $A \xrightarrow{f \cdot g} B$

$B \xleftarrow{f} C$ e $C \xleftarrow{g} A$

$$\begin{tikzcd}
D \arrow[r, "f"] & B \\
A \arrow[r, "g"] & C
\end{tikzcd}$$

?

$D = A$?

$$\begin{tikzcd}
B & & C \\
& A \arrow[ul, "f"] \arrow[ur, "g"'] &
\end{tikzcd}$$

$D = A$?

$$\begin{tikzcd} B & & C \\ & A \arrow[ul, "f"'] \arrow[ur, "g"] & \end{tikzcd}$$

$f\ a \ \ldots \ g\ a$

$$\begin{tikzcd}
B & & C \\
& A \arrow[ul, "f"] \arrow[ur, "g"'] &
\end{tikzcd}$$

$(f\ a, g\ a)$

Cartesian product

$$B \times C = \{(b, c) \mid b \in B \wedge c \in C\}$$

Cartesian product

$$B \times C = \{(b, c) \mid b \in B \wedge c \in C\}$$

$$f\ a \in B$$

Cartesian product

$$B \times C = \{(b, c) \mid b \in B \land c \in C\}$$

$$f\ a \in B$$

$$g\ a \in C$$

Cartesian product

$$B \times C = \{(b, c) \mid b \in B \land c \in C\}$$

$$\frac{f\ a \in B \qquad g\ a \in C}{(f\ a, g\ a) \in B \times C}$$

“Split”

$$\begin{tikzcd}
B & B \times C & C \\
& A \arrow[ul, "f"] \arrow[u, "\langle f, g \rangle"'] \arrow[ur, "g"'] &
\end{tikzcd}$$

$$\langle f, g \rangle \ a = (f \ a, g \ a)$$

Product

$$A \times B = \{(a, b) \mid a \in A \wedge b \in B\}$$

Product

$$A \times B = \{(a, b) \mid a \in A \land b \in B\}$$

$$\pi_1 : A \times B \to A$$
$$\pi_1\,(a, b) = a$$

Product

$$A \times B = \{(a, b) \mid a \in A \wedge b \in B\}$$

$$\pi_1 : A \times B \to A \qquad \pi_2 : A \times B \to B$$

$$\pi_1 (a, b) = a \qquad \pi_2 (a, b) = b$$

Product

$$\begin{tikzcd}
B & B \times C \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & C \\
& A \arrow[ul, "f"] \arrow[u, "\langle f{,}g \rangle"'] \arrow[ur, "g"'] &
\end{tikzcd}$$

Product

$$\begin{tikzcd}
B & B \times C \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & C \\
& A \arrow[ul, "f"] \arrow[u, "\langle f,g \rangle"'] \arrow[ur, "g"'] &
\end{tikzcd}$$

$$\pi_1 \cdot \langle f, g \rangle = f$$

Product

$$\begin{tikzcd}
B & B \times C \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & C \\
& A \arrow[ul, "f"] \arrow[u, "\langle f,g \rangle"'] \arrow[ur, "g"'] &
\end{tikzcd}$$

$\pi_1 \cdot \langle f, g \rangle = f$ $\pi_2 \cdot \langle f, g \rangle = g$

Product

$$\langle f, g \rangle$$

Product

$$\langle f, g \rangle$$

f and g in parallel

f “split” g

Product

$$\langle f, g \rangle$$

f and g in parallel

f “split” g

$$\langle f, g \rangle\ a = (f\ a, g\ a)$$

Product

$$\begin{tikzcd} D \\ B \arrow[u, "f"] \end{tikzcd}$$

Product

$$\begin{tikzcd}
D & E \\
B \arrow[u, "f"] & C \arrow[u, "g"']
\end{tikzcd}$$

Product

$$\begin{tikzcd}
D & & E \\
B \arrow[u, "f"] & B \times C \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & C \arrow[u, "g"']
\end{tikzcd}$$

Product

$$\begin{tikzcd}
D & & E \\
B \arrow[u, "f"] & B \times C \arrow[l, "\pi_1"] \arrow[r, "\pi_2"'] \arrow[ul] & C \arrow[u, "g"']
\end{tikzcd}$$

Product

$$\begin{tikzcd}
D & & E \\
B \arrow[u, "f"] & B \times C \arrow[l, "\pi_1"] \arrow[ul] \arrow[r, "\pi_2"'] \arrow[ur] & C \arrow[u, "g"']
\end{tikzcd}$$

Product

$$\begin{tikzcd}
D & D \times E & E \\
B \arrow[u, "f"] & B \times C \arrow[l, "\pi_1"] \arrow[r, "\pi_2"'] \arrow[u, "\langle f \cdot \pi_1, g \cdot \pi_2 \rangle" description] \arrow[ul] \arrow[ur] & C \arrow[u, "g"']
\end{tikzcd}$$

Product

$$\begin{tikzcd}
D & D \times E \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & E \\
B \arrow[u, "f"] & B \times C \arrow[l, "\pi_1"] \arrow[r, "\pi_2"'] \arrow[u, "\langle f \cdot \pi_1, g \cdot \pi_2 \rangle" description] \arrow[ul] \arrow[ur] & C \arrow[u, "g"']
\end{tikzcd}$$

Product

$$\begin{tikzcd}
D & D \times E \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & E \\
B \arrow[u, "f"] & B \times C \arrow[l, "\pi_1"] \arrow[r, "\pi_2"'] \arrow[u, "f \times g" description] \arrow[ul] \arrow[ur] & C \arrow[u, "g"']
\end{tikzcd}$$

$$f \times g = \langle f \cdot \pi_1, g \cdot \pi_2 \rangle$$

Summing up

$f \cdot g$

Summing up

$f \cdot g$ Sequential composition

Summing up

$f \cdot g$ Sequential composition

$\langle f, g \rangle$

Summing up

$f \cdot g$	Sequential composition
$\langle f, g \rangle$	Parallel composition

Summing up

$f \cdot g$	Sequential composition
$\langle f, g \rangle$	Parallel composition (synchronous)

Summing up

$f \cdot g$	Sequential composition
$\langle f, g \rangle$	Parallel composition (synchronous)
$f \times g$	

Summing up

$f \cdot g$	Sequential composition
$\langle f, g \rangle$	Parallel composition (synchronous)
$f \times g$	Parallel composition

Summing up

$f \cdot g$	Sequential composition
$\langle f, g \rangle$	Parallel composition (synchronous)
$f \times g$	Parallel composition (asynchronous)

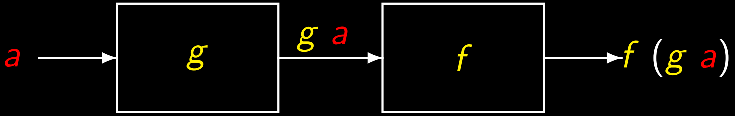
Summing up

$f \cdot g$	Sequential composition
$\langle f, g \rangle$	Parallel composition (synchronous)
$f \times g$	Parallel composition (asynchronous)

Compositional programming

In pictures

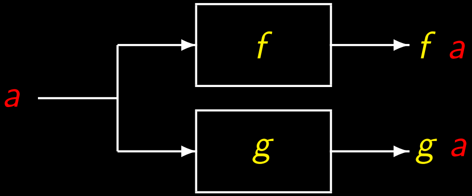
$$(f \cdot g)\ a = f\ (g\ a) \qquad (2.6)$$



Function **composition**

In pictures

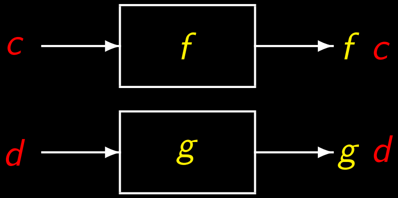
$$\langle f, g \rangle \ a = (f \ a, g \ a) \tag{2.20}$$



Functional "splits"

In pictures

$$f \times g = \langle f \cdot \pi_1, g \cdot \pi_2 \rangle \qquad (2.24)$$

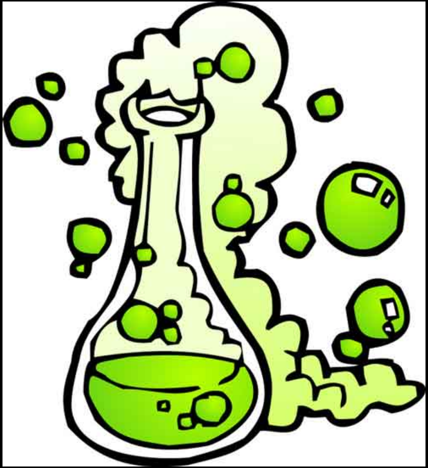


Functional **products**

$f \cdot g$	Sequential composition
$\langle f, g \rangle$	Parallel composition (synchronous)
$f \times g$	Parallel composition (asynchronous)

Compositional programming

Calculus?



$$\begin{tikzcd}
B & B \times C \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & C \\
& A \arrow[ul, "f"] \arrow[u, "{\langle f, g \rangle}"] \arrow[ur, "g"'] &
\end{tikzcd}$$

$$\begin{tikzcd}
B & B \times C \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & C \\
& A \arrow[ul, "f"] \arrow[u, "{\langle f , g \rangle}" description] \arrow[ur, "g"'] & \\
& D \arrow[u, "h"] &
\end{tikzcd}$$

$$\begin{tikzcd}
B & B \times C \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & C \\
 & D \arrow[u, "\langle f , g \rangle \cdot h" description] &
\end{tikzcd}$$

$$\begin{tikzcd}
B & B \times C \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & C \\
 & A \arrow[lu, "f"] \arrow[u, "\langle f , g \rangle" description] \arrow[ru, "g"'] & \\
 & D \arrow[u, "h"] &
\end{tikzcd}$$

$$\begin{tikzcd}
B & B \times C \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & C \\
 & A \arrow[ul, "f"] \arrow[u, "\langle f, g \rangle"] \arrow[ur, "g"'] & \\
 & D \arrow[u, "h"] \arrow[uul, "f \cdot h"] &
\end{tikzcd}$$

$$\begin{tikzcd}
B & B \times C \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & C \\
& A \arrow[ul, "f"] \arrow[u, "\langle f , g \rangle"] \arrow[ur, "g"'] & \\
& D \arrow[u, "h"] \arrow[uul, "f \cdot h"] \arrow[uur, "g \cdot h"'] &
\end{tikzcd}$$

$$\begin{tikzcd}
B & B \times C \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & C \\
& D \arrow[ul, "f \cdot h"] \arrow[u, "\langle f \cdot h, g \cdot h \rangle" description] \arrow[ur, "g \cdot h"'] &
\end{tikzcd}$$

$$\begin{tikzcd}
B & B \times C \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & C \\
& D \arrow[u, "\langle f , g \rangle \cdot h = \langle f \cdot h, g \cdot h \rangle" description] &
\end{tikzcd}$$

×-Fusion

$$\langle f, g \rangle \cdot h = \langle f \cdot h, g \cdot h \rangle \tag{2.26}$$

$$\begin{tikzcd}
B & B \times C \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & C \\
& A \arrow[ul, "f"'] \arrow[u, "\langle f,g \rangle"] \arrow[ur, "g"] & \\
& D \arrow[u, "h"] \arrow[uul, "f \cdot h"] \arrow[uur, "g \cdot h"'] &
\end{tikzcd}$$

$$\begin{tikzcd}
B & B \times C \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & C \\
& A \arrow[ul, "f"] \arrow[u, "\langle f, g \rangle"] \arrow[ur, "g"'] &
\end{tikzcd}$$

$$\begin{tikzcd}
D & & \\
B \arrow[u, "h"] & B \times C \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & C \\
& A \arrow[ul, "f"] \arrow[u, "\langle f, g \rangle"] \arrow[ur, "g"'] &
\end{tikzcd}$$

$$\begin{tikzcd}
D & & E \\
B \arrow[u, "h"] & B \times C \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & C \arrow[u, "k"'] \\
& A \arrow[ul, "f"] \arrow[u, "\langle f, g \rangle" description] \arrow[ur, "g"'] &
\end{tikzcd}$$

$$\begin{tikzcd}
D & D \times E & E \\
B \arrow[u, "h"] & B \times C \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] \arrow[u, "h \times k" description] & C \arrow[u, "k"'] \\
& A \arrow[ul, "f"] \arrow[u, "\langle f , g \rangle" description] \arrow[ur, "g"'] &
\end{tikzcd}$$

$$\begin{tikzcd}
D & D \times E \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & E \\
B \arrow[u, "h"] & B \times C \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] \arrow[u, "h \times k" description] & C \arrow[u, "k"'] \\
& A \arrow[ul, "f"] \arrow[u, "\langle f , g \rangle" description] \arrow[ur, "g"'] &
\end{tikzcd}$$

$$\begin{tikzcd}
D & D \times E \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & E \\
B \arrow[u, "h"] & & C \arrow[u, "k"'] \\
& A \arrow[ul, "f"] \arrow[ur, "g"'] \arrow[uu, "(h \times k) \cdot \langle f, g \rangle" description] &
\end{tikzcd}$$

$$\begin{tikzcd}
D & & D \times E \arrow[ll, "\pi_1"'] \arrow[rr, "\pi_2"] & & E \\
B \arrow[u, "h"] & & & & C \arrow[u, "k"'] \\
& & A \arrow[llu, "f"] \arrow[rru, "g"'] \arrow[lluu, "h \cdot f"'] & &
\end{tikzcd}$$

$$\begin{tikzcd}
D & D \times E \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & E \\
B \arrow[u, "h"] & & C \arrow[u, "k"'] \\
& A \arrow[ul, "f"] \arrow[ur, "g"'] \arrow[uul, "h \cdot f"'] \arrow[uur, "k \cdot g"] &
\end{tikzcd}$$

$$\begin{tikzcd}
D & D \times E \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & E \\
B \arrow[u, "h"] & & C \arrow[u, "k"'] \\
& A \arrow[ul, "f"] \arrow[uu, "\langle h \cdot f, k \cdot g \rangle" description] \arrow[ur, "g"'] &
\end{tikzcd}$$

$$\begin{tikzcd}
D & D \times E \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & E \\
B \arrow[u, "h"] & & C \arrow[u, "k"'] \\
& A \arrow[ul, "f"] \arrow[ur, "g"'] \arrow[uu, "(h \times k) \cdot \langle f, g \rangle" description] &
\end{tikzcd}$$

$\times$-Absorption

$$(h \times k) \cdot \langle f, g \rangle = \langle h \cdot f, k \cdot g \rangle \qquad (2.27)$$

$$\begin{tikzcd}
A & A \times B \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & B \\
D \arrow[u, "h"] & D \times E \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] \arrow[u, "h \times k"] & E \arrow[u, "k"'] \\
 & C \arrow[ul, "f"] \arrow[u, "\langle f, g \rangle"] \arrow[ur, "g"'] &
\end{tikzcd}$$

$$\begin{tikzcd}
D & D \times E \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & E \\
B \arrow[u, "h"] & B \times C \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] \arrow[u, "h \times k" description] & C \arrow[u, "k"'] \\
& A \arrow[ul, "f"] \arrow[u, "\langle f , g \rangle" description] \arrow[ur, "g"'] &
\end{tikzcd}$$

$$\begin{tikzcd}
D & D \times E \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & E \\
B \arrow[u, "h"] & B \times C \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] \arrow[u, "h \times k" description] & C \arrow[u, "k"']
\end{tikzcd}$$

$$\begin{tikzcd}
D & D \times E \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & E \\
B \arrow[u, "h"] & B \times C \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] \arrow[u, "h \times k"] & C \arrow[u, "k"']
\end{tikzcd}$$

$$\pi_1 \cdot (h \times k) = h \cdot \pi_1$$

$$\begin{tikzcd}
D & D \times E \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & E \\
B \arrow[u, "h"] & B \times C \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] \arrow[u, "h \times k"] & C \arrow[u, "k"']
\end{tikzcd}$$

$$\pi_1 \cdot (h \times k) = h \cdot \pi_1 \qquad \pi_2 \cdot (h \times k) = k \cdot \pi_2$$

Natural-π_1, natural-π_2

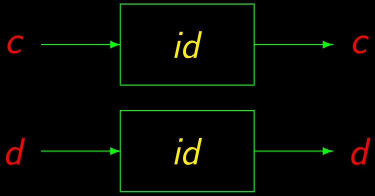
$$\pi_1 \cdot (h \times k) \;=\; h \cdot \pi_1 \tag{2.28}$$

$$\pi_2 \cdot (h \times k) = k \cdot \pi_2 \tag{2.29}$$

$$\begin{tikzcd}
D & D \times E \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & E \\
B \arrow[u, "h"] & B \times C \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] \arrow[u, "h \times k"'] & C \arrow[u, "k"']
\end{tikzcd}$$

Functor-id-$\times$

$$id \times id = id \tag{2.31}$$



Product of two identities is an identity.

$\times$-Functor

$$(f \times h) \cdot (g \times k) = (f \cdot g) \times (h \cdot k) \tag{2.30}$$

Composition of products is a **product** of compositions.

Two laws still missing

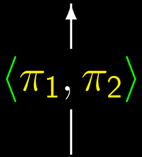
$\times$-Reflexion $$\langle \pi_1, \pi_2 \rangle = \mathit{id} \tag{2.32}$$

Two laws still missing

$\times$-Reflexion $$\langle \pi_1, \pi_2 \rangle = id \tag{2.32}$$

$\times$-Eq $$\langle i, j \rangle = \langle f, g \rangle \quad \Leftrightarrow \quad \left\{ \begin{array}{l} i = f \\ j = g \end{array} \right. \tag{2.64}$$

$\times$-Reflexion



$$\langle \pi_1, \pi_2 \rangle$$

×-Reflexion

$$\begin{tikzcd}
A & {} \\
& A \times B \arrow[ul, "\pi_1"'] \arrow[u, "{\langle \pi_1, \pi_2 \rangle}" description]
\end{tikzcd}$$

×-Reflexion

$$\begin{tikzcd}
A & {} & B \\
& A \times B \arrow[ul, "\pi_1"'] \arrow[u, "\langle \pi_1, \pi_2 \rangle" description] \arrow[ur, "\pi_2"] &
\end{tikzcd}$$

$\times$-Reflexion

$$\begin{tikzcd}
A & A \times B & B \\
 & A \times B \arrow[ul, "\pi_1"'] \arrow[u, "\langle \pi_1, \pi_2 \rangle" description] \arrow[ur, "\pi_2"] &
\end{tikzcd}$$

$\times$-Reflexion

$$\begin{tikzcd}
A & A \times B \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & B \\
& A \times B \arrow[ul, "\pi_1"] \arrow[u, "\langle \pi_1, \pi_2 \rangle" description] \arrow[ur, "\pi_2"'] &
\end{tikzcd}$$

$$\begin{tikzcd}
A & A \times B \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & B \\
& A \times B \arrow[ul, "\pi_1"] \arrow[u, "id"] \arrow[ur, "\pi_2"'] &
\end{tikzcd}$$

Finally the most important...

Recall $\times$-**cancellation**:

$$\begin{tikzcd}
B & B \times C \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & C \\
& A \arrow[ul, "f"] \arrow[u, "\langle f,g \rangle"'] \arrow[ur, "g"'] &
\end{tikzcd}$$

$\pi_1 \cdot \langle f, g \rangle = f$

$\pi_2 \cdot \langle f, g \rangle = g$

$$\begin{cases} \pi_1 \cdot \langle f, g \rangle = f \\ \pi_2 \cdot \langle f, g \rangle = g \end{cases}$$

$$\begin{cases} \pi_1 \cdot \langle f, g \rangle = f \\ \pi_2 \cdot \langle f, g \rangle = g \end{cases}$$

$$k = \langle f, g \rangle \Rightarrow \begin{cases} \pi_1 \cdot k = f \\ \pi_2 \cdot k = g \end{cases}$$

$$\begin{cases} \pi_1 \cdot \langle f, g \rangle = f \\ \pi_2 \cdot \langle f, g \rangle = g \end{cases}$$

$$k = \langle f, g \rangle \Leftarrow \begin{cases} \pi_1 \cdot k = f \\ \pi_2 \cdot k = g \end{cases}$$

$$\begin{cases} \pi_1 \cdot \langle f, g \rangle = f \\ \pi_2 \cdot \langle f, g \rangle = g \end{cases}$$

$$k = \langle f, g \rangle \Leftrightarrow \begin{cases} \pi_1 \cdot k = f \\ \pi_2 \cdot k = g \end{cases}$$

$\times$-Universal

$$k = \langle f, g \rangle \Leftrightarrow \begin{cases} \pi_1 \cdot k = f \\ \pi_2 \cdot k = g \end{cases}$$

$\times$-Universal

Existence

$$k = \langle f, g \rangle \Rightarrow \begin{cases} \pi_1 \cdot k = f \\ \pi_2 \cdot k = g \end{cases}$$

“There exists a solution — $k = \langle f, g \rangle$ — for the equations on the right”

$\times$-Universal

Unicity

$$k = \langle f, g \rangle \quad \Leftarrow \quad \left\{ \begin{array}{l} \pi_1 \cdot k = f \\ \pi_2 \cdot k = g \end{array} \right.$$

“Such a solution, $k = \langle f, g \rangle$, is unique”

Equations!

$$\begin{cases} x = 2\, y \\ z = \frac{y}{3} \\ x + y + z = 10 \end{cases}$$

Equations!

$$\begin{cases} x = 2\,y \\ z = \frac{y}{3} \\ x + y + z = 10 \end{cases} \Leftrightarrow \begin{cases} x = 6 \\ z = 1 \\ y = 3 \end{cases}$$

Equations!

Problem

Solve the equation

$$\langle f, g \rangle = id$$

for f *and* g.

Equations!

Calculation

In $$k = \langle f, g \rangle \Leftrightarrow \left\{ \begin{array}{l} \pi_1 \cdot k = f \\ \pi_2 \cdot k = g \end{array} \right.$$

Equations!

Calculation

In $k = \langle f, g \rangle \Leftrightarrow \begin{cases} \pi_1 \cdot k = f \\ \pi_2 \cdot k = g \end{cases}$ let $k = id$

Equations!

Calculation

In $$k = \langle f, g \rangle \Leftrightarrow \begin{cases} \pi_1 \cdot k = f \\ \pi_2 \cdot k = g \end{cases}$$ let $k = id$

$$id = \langle f, g \rangle \Leftrightarrow \begin{cases} \pi_1 \cdot id = f \\ \pi_2 \cdot id = g \end{cases}$$

Equations!

Calculation

In $k = \langle f, g \rangle \Leftrightarrow \begin{cases} \pi_1 \cdot k = f \\ \pi_2 \cdot k = g \end{cases}$ let $k = id$

$$id = \langle f, g \rangle \Leftrightarrow \begin{cases} \pi_1 = f \\ \pi_2 = g \end{cases}$$

Equations!

$$id = \langle f, g \rangle \Leftrightarrow \begin{cases} \pi_1 = f \\ \pi_2 = g \end{cases}$$

Equations!

$$id = \langle f, g \rangle \Leftrightarrow \begin{cases} \pi_1 = f \\ \pi_2 = g \end{cases}$$

Substituting:

$$id = \langle \pi_1, \pi_2 \rangle$$

Equations!

$$id = \langle f, g \rangle \;\Leftrightarrow\; \begin{cases} \pi_1 = f \\ \pi_2 = g \end{cases}$$

Substituting:

$$id = \langle \pi_1, \pi_2 \rangle$$

×-Reflexion 🙂

Problem

Solve the equation

$$\langle h, k \rangle = \langle f, g \rangle$$

Problem

Solve the equation

$$\langle h, k \rangle = \langle f, g \rangle$$

(1 equation, 4 unknowns)

Calculation

$$\langle h, k \rangle = \langle f, g \rangle$$

Calculation

$$\langle h, k\rangle = \langle f, g\rangle$$

$$\Leftrightarrow \qquad \{ \ \times\text{-universal} \ \}$$

$$\begin{cases} \pi_1 \cdot \langle h, k\rangle = f \\ \pi_2 \cdot \langle h, k\rangle = g \end{cases}$$

Calculation

$$\langle h, k \rangle = \langle f, g \rangle$$

$$\Leftrightarrow \quad \{ \ \times\text{-universal} \ \}$$

$$\begin{cases} \pi_1 \cdot \langle h, k \rangle = f \\ \pi_2 \cdot \langle h, k \rangle = g \end{cases}$$

$$\Leftrightarrow \quad \{ \ \times\text{-cancellation} \ \}$$

$$\begin{cases} h = f \\ k = g \end{cases}$$

Calculation

$$\langle h, k \rangle = \langle f, g \rangle$$

$$\Leftrightarrow \qquad \{ \ \times\text{-universal} \ \}$$

$$\begin{cases} \pi_1 \cdot \langle h, k \rangle = f \\ \pi_2 \cdot \langle h, k \rangle = g \end{cases}$$

$$\Leftrightarrow \qquad \{ \ \times\text{-cancellation} \ \}$$

$$\begin{cases} h = f \\ k = g \end{cases}$$

×-Eq !

$$\langle \pi_1, \pi_2 \rangle = \mathit{id}$$

$\langle \pi_1, \pi_2 \rangle = id$

$\langle \pi_2, \pi_1 \rangle$?

$\langle \pi_1, \pi_2 \rangle = id$

$\langle \pi_2, \pi_1 \rangle$?

$$\begin{tikzcd} {} \\ \langle \pi_2, \pi_1 \rangle \arrow[u] \\ {} \arrow[u, dash] \end{tikzcd}$$

$\langle \pi_1, \pi_2 \rangle = id$

$\langle \pi_2, \pi_1 \rangle$?

$$\begin{tikzcd}
B & {} \\
& A \times B \arrow[ul, "\pi_2"] \arrow[u, "\langle \pi_2, \pi_1 \rangle"']
\end{tikzcd}$$

$\langle \pi_1, \pi_2 \rangle = id$

$\langle \pi_2, \pi_1 \rangle$?

$$\begin{tikzcd}
B & {} & A \\
& A \times B \arrow[ul, "\pi_2"'] \arrow[ur, "\pi_1"] \arrow[u, "\langle \pi_2, \pi_1 \rangle"] &
\end{tikzcd}$$

$\langle \pi_1, \pi_2 \rangle = id$

$\langle \pi_2, \pi_1 \rangle$?

$$\begin{tikzcd}
B & B \times A & A \\
& A \times B \arrow[ul, "\pi_2"'] \arrow[u, "\langle \pi_2, \pi_1 \rangle"] \arrow[ur, "\pi_1"] &
\end{tikzcd}$$

$\langle \pi_1, \pi_2 \rangle = id$

$\langle \pi_2, \pi_1 \rangle$?

$$\begin{tikzcd}
B & B \times A \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & A \\
& A \times B \arrow[ul, "\pi_2"] \arrow[u, "\langle \pi_2, \pi_1 \rangle"] \arrow[ur, "\pi_1"'] &
\end{tikzcd}$$

Problem

Solve

$$\langle \pi_2, \pi_1 \rangle \cdot k = id$$

for k

Problem

Solve

$$\langle \pi_2, \pi_1 \rangle \cdot k = id$$

for k

$$\begin{tikzcd}
A \times B \arrow[rr, bend left, "\langle \pi_2, \pi_1 \rangle"] & & B \times A \arrow[ll, bend left, "k"]
\end{tikzcd}$$

Calculation

$$\langle \pi_2, \pi_1 \rangle \cdot k = id$$

Calculation

$$\langle \pi_2, \pi_1 \rangle \cdot k = id$$

$$\Leftrightarrow \qquad \{ \ \times\text{-fusion} \ \}$$

$$\langle \pi_2 \cdot k, \pi_1 \cdot k \rangle = id$$

Calculation

$$\langle \pi_2, \pi_1 \rangle \cdot k = id$$

$$\Leftrightarrow \quad \{ \ \times\text{-fusion} \ \}$$

$$\langle \pi_2 \cdot k, \pi_1 \cdot k \rangle = id$$

$$\Leftrightarrow \quad \{ \ \times\text{-universal} \ \}$$

$$\left\{ \begin{array}{l} \pi_2 \cdot k = \pi_1 \\ \pi_1 \cdot k = \pi_2 \end{array} \right.$$

Calculation

$$\langle \pi_2, \pi_1 \rangle \cdot k = id$$

$$\Leftrightarrow \quad \{ \times\text{-fusion} \}$$

$$\langle \pi_2 \cdot k, \pi_1 \cdot k \rangle = id$$

$$\Leftrightarrow \quad \{ \times\text{-universal} \}$$

$$\begin{cases} \pi_2 \cdot k = \pi_1 \\ \pi_1 \cdot k = \pi_2 \end{cases}$$

$$\Leftrightarrow \quad \{ \text{trivial} \}$$

$$\begin{cases} \pi_1 \cdot k = \pi_2 \\ \pi_2 \cdot k = \pi_1 \end{cases}$$

Calculation

$$
\begin{array}{ll}
 & \langle \pi_2, \pi_1 \rangle \cdot k = id \\
\Leftrightarrow & \quad \{ \ \times\text{-fusion} \ \} \\
 & \langle \pi_2 \cdot k, \pi_1 \cdot k \rangle = id \\
\Leftrightarrow & \quad \{ \ \times\text{-universal} \ \} \\
 & \left\{ \begin{array}{l} \pi_2 \cdot k = \pi_1 \\ \pi_1 \cdot k = \pi_2 \end{array} \right.
\end{array}
$$

$$
\begin{array}{ll}
\Leftrightarrow & \quad \{ \ \text{trivial} \ \} \\
 & \left\{ \begin{array}{l} \pi_1 \cdot k = \pi_2 \\ \pi_2 \cdot k = \pi_1 \end{array} \right. \\
\Leftrightarrow & \quad \{ \ \times\text{-universal} \ \} \\
 & k = \langle \pi_2, \pi_1 \rangle
\end{array}
$$

Swap

$$\begin{tikzcd}
A \times B \arrow[r, bend left, "\langle \pi_2, \pi_1 \rangle"] & B \times A \arrow[l, bend left, "\langle \pi_2, \pi_1 \rangle"]
\end{tikzcd}$$

$$\mathit{swap} = \langle \pi_2, \pi_1 \rangle$$

Swap

$$\begin{tikzcd}A \times B \arrow[r, bend left, "\langle \pi_2, \pi_1 \rangle"] & B \times A \arrow[l, bend left, "\langle \pi_2, \pi_1 \rangle"]\end{tikzcd}$$

$$swap = \langle \pi_2, \pi_1 \rangle$$

$$swap \cdot swap = id$$

Até agora

$f \cdot g$ Sequential composition

Até agora

$f \cdot g$	Sequential composition
$\langle f, g \rangle$	Parallel composition

Até agora

$f \cdot g$ Sequential composition

$\langle f, g \rangle$ Parallel composition

Associativity

$$(f \cdot g) \cdot h = f \cdot (g \cdot h)$$

Até agora

$f \cdot g$ Sequential composition

$\langle f, g \rangle$ Parallel composition

Associativity

$$(f \cdot g) \cdot h = f \cdot (g \cdot h)$$

Associativity?

$$\langle\langle f, g \rangle, h \rangle = \langle f, \langle g, h \rangle\rangle ?$$

Até agora

$f \cdot g$ Sequential composition

$\langle f, g \rangle$ Parallel composition

Associativity

$$(f \cdot g) \cdot h = f \cdot (g \cdot h)$$

Associativity?

$$\langle \langle f, g \rangle, h \rangle = \langle f, \langle g, h \rangle \rangle ?$$

No! but...

$\langle\langle f, g\rangle, h\rangle$

$$\begin{tikzcd}
A & \\
& D \arrow[ul, "f"]
\end{tikzcd}$$

$\langle\langle f, g\rangle, h\rangle$

$$\begin{tikzcd}
A & & B \\
& D \arrow[ul, "f"] \arrow[ur, "g"'] &
\end{tikzcd}$$

$\langle \langle f, g \rangle, h \rangle$

$$\begin{tikzcd}
A & A \times B \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & B \\
& D \arrow[ul, "f"] \arrow[u, "\langle f, g \rangle" description] \arrow[ur, "g"'] &
\end{tikzcd}$$

$\langle\langle f, g\rangle, h\rangle$

$$\begin{tikzcd}
A \times B & & C \\
& D \arrow[ul, "{\langle f, g\rangle}"] \arrow[ur, "h"'] &
\end{tikzcd}$$

$\langle\langle f, g\rangle, h\rangle$

$$\begin{tikzcd}
A \times B & & C \\
& D \arrow[ul, "\langle f, g\rangle"'] \arrow[ur, "h"'] \arrow[u, "\langle\langle f, g\rangle, h\rangle" description] &
\end{tikzcd}$$

$\langle\langle f, g\rangle, h\rangle$

$$\begin{tikzcd}
A \times B & (A \times B) \times C \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & C \\
& D \arrow[ul, "\langle f, g\rangle"] \arrow[u, "\langle\langle f, g\rangle, h\rangle"] \arrow[ur, "h"'] &
\end{tikzcd}$$

$\langle f, \langle g, h \rangle \rangle$

$$\begin{tikzcd}
B & & C \\
& D \arrow[ul, "g"] \arrow[ur, "h"'] &
\end{tikzcd}$$

$\langle f, \langle g, h\rangle\rangle$

$$\begin{tikzcd}
B & B \times C \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & C \\
& D \arrow[ul, "g"] \arrow[u, "{\langle g, h\rangle}"] \arrow[ur, "h"'] &
\end{tikzcd}$$

$\langle f, \langle g, h \rangle \rangle$

$$\begin{tikzcd}
A & & B \times C \\
& D \arrow[ul, "f"] \arrow[ur, "\langle g, h \rangle"'] &
\end{tikzcd}$$

$\langle f, \langle g, h \rangle \rangle$

$$\begin{tikzcd}
A & & B \times C \\
& D \arrow[ul, "f"] \arrow[u, "\langle f, \langle g, h \rangle \rangle"] \arrow[ur, "\langle g, h \rangle"'] &
\end{tikzcd}$$

$\langle f, \langle g, h\rangle\rangle$

$$\begin{tikzcd}
A & A \times (B \times C) \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & B \times C \\
& D \arrow[ul, "f"] \arrow[u, "{\langle f, \langle g, h\rangle\rangle}" description] \arrow[ur, "{\langle g, h\rangle}"'] &
\end{tikzcd}$$

$\langle\langle f, g\rangle, h\rangle$

$$\begin{tikzcd}
A \times B & (A \times B) \times C \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & C \\
& D \arrow[ul, "{\langle f, g\rangle}"] \arrow[u, "{\langle\langle f, g\rangle, h\rangle}" description] \arrow[ur, "h"'] &
\end{tikzcd}$$

$$\begin{tikzcd}
(A \times B) \times C \arrow[rr, bend left, "k"] & & A \times (B \times C) \\
& D \arrow[ul, "\langle \langle f , g \rangle , h \rangle"] \arrow[ur, "\langle f , \langle g , h \rangle \rangle"'] &
\end{tikzcd}$$

$$\begin{tikzcd}
(A \times B) \times C \arrow[rr, bend left, "k"] & & A \times (B \times C) \\
& D \arrow[ul, "\langle\langle f, g\rangle, h\rangle"] \arrow[ur, "\langle f, \langle g, h\rangle\rangle"'] &
\end{tikzcd}$$

$$k \cdot \langle\langle f, g\rangle, h\rangle = \langle f, \langle g, h\rangle\rangle$$

$$k \cdot \langle \langle f, g \rangle, h \rangle = \langle f, \langle g, h \rangle \rangle$$

$$k \cdot \langle \langle f, g \rangle, h \rangle = \langle f, \langle g, h \rangle \rangle$$

$$k \cdot \underbrace{\langle \langle f, g \rangle, h \rangle}_{id?} = \langle f, \langle g, h \rangle \rangle$$

$$k \cdot \langle\langle f, g\rangle, h\rangle = \langle f, \langle g, h\rangle\rangle$$

$$k \cdot \underbrace{\langle\langle f, g\rangle, h\rangle}_{id?} = \langle f, \langle g, h\rangle\rangle$$

Solve $\langle\langle f, g\rangle, h\rangle = id$

$$\langle\langle f, g\rangle, h\rangle = id$$

$$\langle\langle f, g\rangle, h\rangle = id$$

$$\Leftrightarrow \quad \{\ \times\text{-universal}\ \}$$

$$\begin{cases} \pi_1 = \langle f, g\rangle \\ \pi_2 = h \end{cases}$$

$$\langle\langle f, g\rangle, h\rangle = id$$

$$\Leftrightarrow \qquad \{\ \times\text{-universal}\ \}$$

$$\begin{cases} \pi_1 = \langle f, g\rangle \\ \pi_2 = h \end{cases}$$

$$\Leftrightarrow \qquad \{\ \times\text{-universal}\ \}$$

$$\begin{cases} \pi_1 \cdot \pi_1 = f \\ \pi_2 \cdot \pi_1 = g \\ \pi_2 = h \end{cases}$$

Substitute solutions

$$\begin{cases} \pi_1 \cdot \pi_1 = f \\ \pi_2 \cdot \pi_1 = g \\ \pi_2 = h \end{cases}$$

Substitute solutions

$$\begin{cases} \pi_1 \cdot \pi_1 = f \\ \pi_2 \cdot \pi_1 = g \\ \pi_2 = h \end{cases}$$

$$k \cdot \underbrace{\langle\langle f, g\rangle, h\rangle}_{id} = \langle f, \langle g, h\rangle\rangle$$

🙂

Substitute solutions

$$\begin{cases} \pi_1 \cdot \pi_1 = f \\ \pi_2 \cdot \pi_1 = g \\ \pi_2 = h \end{cases}$$

$$k = \langle \pi_1 \cdot \pi_1, \langle \pi_2 \cdot \pi_1, \pi_2 \rangle \rangle$$

Sligth improvement...

$$k = \langle \pi_1 \cdot \pi_1, \langle \pi_2 \cdot \pi_1, \pi_2 \rangle \rangle$$

Sligth improvement...

$$
\begin{array}{ll}
 & k = \langle \pi_1 \cdot \pi_1, \langle \pi_2 \cdot \pi_1, \pi_2 \rangle\rangle \\
\Leftrightarrow & \quad \{ \ \pi_2 = id \cdot \pi_2 \ \} \\
 & k = \langle \pi_1 \cdot \pi_1, \langle \pi_2 \cdot \pi_1, id \cdot \pi_2 \rangle\rangle
\end{array}
$$

Sligth improvement...

$$k = \langle \pi_1 \cdot \pi_1, \langle \pi_2 \cdot \pi_1, \pi_2 \rangle \rangle$$

$$\Leftrightarrow \quad \{ \ \pi_2 = id \cdot \pi_2 \ \}$$

$$k = \langle \pi_1 \cdot \pi_1, \langle \pi_2 \cdot \pi_1, id \cdot \pi_2 \rangle \rangle$$

$$\Leftrightarrow \quad \{ \ f \times g = \langle f \cdot \pi_1, g \cdot \pi_2 \rangle \ \}$$

$$k = \langle \pi_1 \cdot \pi_1, \pi_2 \times id \rangle$$

$$\begin{tikzcd}
(A \times B) \times C \arrow[rr, bend left, "\langle \pi_1 \cdot \pi_1 , \pi_2 \times id \rangle"] & & A \times (B \times C) \\
& D \arrow[ul, "\langle \langle f , g \rangle , h \rangle"] \arrow[ur, "\langle f , \langle g , h \rangle \rangle"'] &
\end{tikzcd}$$

$$\begin{tikzcd}
(A \times B) \times C \arrow[rr, bend left, "\langle \pi_1 \cdot \pi_1, \pi_2 \times id \rangle"] & & A \times (B \times C) \arrow[ll, bend left, "j"] \\
& D \arrow[ul, "\langle \langle f, g \rangle, h \rangle"] \arrow[ur, "\langle f, \langle g, h \rangle \rangle"'] &
\end{tikzcd}$$

$$\begin{tikzcd}
(A \times B) \times C \arrow[r, bend left, "\langle \pi_1 \cdot \pi_1 , \pi_2 \times id \rangle"] & A \times (B \times C) \arrow[l, bend left, "\langle id \times \pi_1 , \pi_2 \cdot \pi_2 \rangle"]
\end{tikzcd}$$

$$\begin{tikzcd}
(A \times B) \times C \arrow[r, bend left, "assocr"] & A \times (B \times C) \arrow[l, bend left, "assocl"]
\end{tikzcd}$$

$$assocr = \langle \pi_1 \cdot \pi_1, \pi_2 \times id \rangle$$
$$assocl = \langle id \times \pi_1, \pi_2 \cdot \pi_2 \rangle$$

$$\begin{tikzcd}
(A \times B) \times C \arrow[r, bend left, "assocr"] & A \times (B \times C) \arrow[l, bend left, "assocl"]
\end{tikzcd}$$

$$assocr \cdot assocl = id$$

$$\begin{tikzcd}(A \times B) \times C \arrow[r, bend left, "assocr"] & A \times (B \times C) \arrow[l, bend left, "assocl"]\end{tikzcd}$$

$$assocr \cdot assocl = id$$

$$assocl \cdot assocr = id$$

Isomorphism

$$\begin{tikzcd}
(A \times B) \times C \arrow[rr, bend left, "\mathit{assocr}"] & \cong & A \times (B \times C) \arrow[ll, bend left, "\mathit{assocl}"]
\end{tikzcd}$$

$$\mathit{assocr} \cdot \mathit{assocl} = \mathit{id}$$

$$\mathit{assocl} \cdot \mathit{assocr} = \mathit{id}$$

Isomorphism

$$\begin{tikzcd} A \times B \arrow[r, bend left, "swap"] \arrow[r, phantom, "\cong"] & B \times A \arrow[l, bend left, "swap"] \end{tikzcd}$$

$$swap \cdot swap = id$$

Isomorphism

$$\begin{tikzcd}
A \arrow[r, bend left, "f"] \arrow[r, phantom, "\cong"] & B \arrow[l, bend left, "g"]
\end{tikzcd}$$

$$f \cdot g = id$$

$$g \cdot f = id$$

Isomorphism

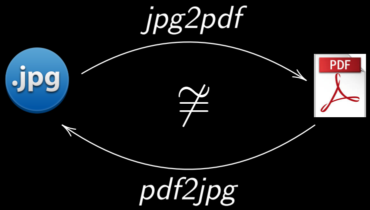
$$\underbrace{\textit{iso}}_{\textit{the same}}\ (\iota\sigma o)$$

Isomorphism

$$\underbrace{\textit{iso}}_{\textit{the same}}\ (\iota\sigma o) + \underbrace{\textit{morphism}}_{\textit{shape}}\ \ (\mu o\rho\phi\iota\sigma\mu o\zeta)$$

“Similar shape”

Practical problem!

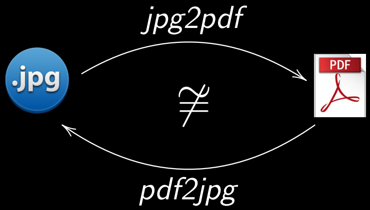


jpg2pdf

$\ncong$

pdf2jpg

Practical problem!



$$\mathit{jpg2pdf} \cdot \mathit{pdf2jpg} \neq \mathit{id}$$

$$\mathit{pdf2jpg} \cdot \mathit{jpg2pdf} \neq \mathit{id}$$

Format conversion

Need

$$\begin{tikzcd}
A \arrow[d, "f"'] \\
B
\end{tikzcd}$$

Format conversion

Need

Reusable

$$\begin{tikzcd} A \arrow[d, "f"'] & & C \arrow[d, "r"] \\ B & & D \end{tikzcd}$$

Format conversion

$$\begin{tikzcd}
\text{Need} & \text{Reusable} \\
A \arrow[r, "export"] \arrow[d, "f"'] & C \arrow[d, "r"] \\
B & D
\end{tikzcd}$$

Format conversion

$$\begin{tikzcd}
\text{Need} & \text{Reusable} \\
A \arrow[r, "export"] \arrow[d, "f"'] & C \arrow[d, "r"] \\
B & D \arrow[l, "import"]
\end{tikzcd}$$

Format conversion

$$\begin{tikzcd}
\text{Need} & \text{Reusable} \\
A \arrow[r, "\mathit{export}"] \arrow[d, "f"'] & C \arrow[d, "r"] \\
B & D \arrow[l, "\mathit{import}"]
\end{tikzcd}$$

$$f = \mathit{import} \cdot r \cdot \mathit{export}$$

Format conversion

$$\begin{tikzcd}
\text{Need} & \text{Reusable} \\
A \arrow[r, "export"] \arrow[d, "f"'] \arrow[dr, phantom, "(\cong)"] & C \arrow[d, "r"] \\
B & D \arrow[l, "import"]
\end{tikzcd}$$

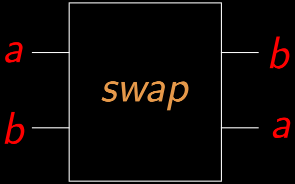
$$f = import \cdot r \cdot export$$

By the way — *swap*

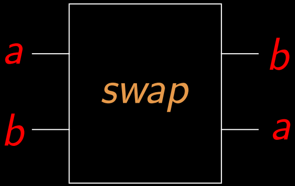
$$\begin{tikzcd}
A \times B \arrow[r, bend left, "\mathit{swap}"] \arrow[r, phantom, "\cong"] & B \times A \arrow[l, bend left, "\mathit{swap}"]
\end{tikzcd}$$

Isomorphisms are **reversible** computations

By the way — *swap*

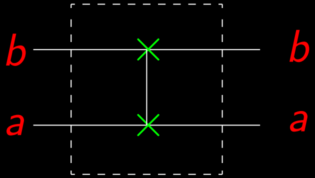


By the way — *swap*



swap is a basic gate in **quantum programming**.

By the way — *swap*



Cálculo
de
Programas

Class T03 [vídeos]

Problem

Retrieve the address of a civil servant, knowing that she/he can be identified either by a citizen card number (CC) or a fiscal number (NIF).

Problem

Retrieve the address of a civil servant, knowing that she/he can be identified either by a citizen card number (CC) or a fiscal number (NIF).

$$address : Iden \rightarrow Address$$

$$Iden = CC \cup NIF$$

Problem!

$$CC = \mathbb{N}$$

$$NIF = \mathbb{N}$$

Problem!

$$CC = \mathbb{N}$$
$$NIF = \mathbb{N}$$

$$Iden = CC \cup NIF = \mathbb{N} \cup \mathbb{N} = \mathbb{N}$$

Problem!

$$CC = \mathbb{N}$$
$$NIF = \mathbb{N}$$
$$Iden = CC \cup NIF = \mathbb{N} \cup \mathbb{N} = \mathbb{N}$$

$$address : \mathbb{N} \to Address \ (!)$$

In general

We need to fix:

$$m : A \cup B \to C$$

In general

We need to fix:

$$m : A \cup B \to C$$

Let us start from

$$A \cup B = \{ a \mid a \in A \} \cup \{ b \mid b \in B \}$$

Disjoint union

In need of something like...

$$\{(1, a) \mid a \in A\} \cup \{(2, b) \mid b \in B\}$$

Disjoint union

In need of something like...

$$\{(1, a) \mid a \in A\} \cup \{(2, b) \mid b \in B\}$$

... we define:

$$A + B = \{(1, a) \mid a \in A\} \cup \{(2, b) \mid b \in B\}$$

Disjoint union

Clearly,

$$A + B = \{\, i_1\ a \mid a \in A \,\} \cup \{\, i_2\ b \mid b \in B \,\}$$

once we define:

$$i_1\ a = (1, a)$$
$$i_2\ b = (2, b)$$

Disjoint union

Types:

$$i_1 : A \to A + B$$

$$i_2 : B \to A + B$$

Disjoint union

Types:

$$i_1 : A \rightarrow A + B$$

$$i_2 : B \rightarrow A + B$$

Now think of:

$$m : A + B \rightarrow C$$

Disjoint union

Types:

$$i_1 : A \to A + B$$

$$i_2 : B \to A + B$$

Now think of:

$$m : A + B \to C$$

$$\begin{tikzcd}
A + B \arrow[d, "m" description] \\
C
\end{tikzcd}$$

Disjoint union

Types:

$$i_1 : A \to A + B$$

$$i_2 : B \to A + B$$

Now think of:

$$m : A + B \to C$$

$$\begin{tikzcd} A \arrow[r, "i_1"] & A + B \arrow[d, "m"] \\ & C \end{tikzcd}$$

Disjoint union

Types:

$$i_1 : A \to A + B$$

$$i_2 : B \to A + B$$

Now think of:

$$m : A + B \to C$$

$$\begin{tikzcd}
A \arrow[r, "i_1"] & A + B \arrow[d, "m"] & B \arrow[l, "i_2"'] \\
 & C &
\end{tikzcd}$$

Disjoint union

Types:

$$i_1 : A \to A + B$$

$$i_2 : B \to A + B$$

Now think of:

$$m : A + B \to C$$

$$\begin{tikzcd}
A \arrow[r, "i_1"] \arrow[dr, "f"'] & A + B \arrow[d, "m"] & B \arrow[l, "i_2"'] \\
& C &
\end{tikzcd}$$

$$f = m \cdot i_1$$

Disjoint union

Types:

$$i_1 : A \to A + B$$

$$i_2 : B \to A + B$$

Now think of:

$$m : A + B \to C$$

$$\begin{tikzcd}
A \arrow[r, "i_1"] \arrow[dr, "f"'] & A + B \arrow[d, "m"] & B \arrow[l, "i_2"'] \arrow[dl, "g"] \\
& C &
\end{tikzcd}$$

$$f = m \cdot i_1 \qquad g = m \cdot i_2$$

Compare...

$$\begin{tikzcd}
A \arrow[r, "i_1"] \arrow[dr, "f"'] & A + B \arrow[d, "{[f, g]}"] & B \arrow[l, "i_2"'] \arrow[dl, "g"] \\
& C &
\end{tikzcd}$$

Compare…

$$\begin{tikzcd}
A & A \times B \arrow[l, "\pi_1"'] \arrow[r, "\pi_2"] & B \\
& C \arrow[ul, "f"] \arrow[u, "\langle f, g \rangle" description] \arrow[ur, "g"'] &
\end{tikzcd}$$

$+$-Universal

$$k = [f, g] \Leftrightarrow \left\{ \begin{array}{l} k \cdot i_1 = f \\ k \cdot i_2 = g \end{array} \right.$$

$+$-Universal

$$k = [f, g] \Leftrightarrow \left\{ \begin{array}{l} k \cdot i_1 = f \\ k \cdot i_2 = g \end{array} \right.$$

Compare with

$$k = \langle f, g \rangle \Leftrightarrow \left\{ \begin{array}{l} \pi_1 \cdot k = f \\ \pi_2 \cdot k = g \end{array} \right.$$

“Alternating” functions

$$[f, g] : A + B \to C$$

$$[f, g]\ x \ = \ \begin{cases} x = i_1\ a \Rightarrow f\ a \\ x = i_2\ b \Rightarrow g\ b \end{cases}$$

“Alternating” functions

$$[f, g] : A + B \to C$$

$$[f, g]\ x \;=\; \begin{cases} x = i_1\ a \Rightarrow f\ a \\ x = i_2\ b \Rightarrow g\ b \end{cases}$$

$f + g$?

Sum of two functions

$$\begin{tikzcd}
D \arrow[d, "f"'] \\
B
\end{tikzcd}$$

Sum of two functions

$$\begin{tikzcd}
D \arrow[d, "f"'] & & E \arrow[d, "g"] \\
B & & C
\end{tikzcd}$$

Sum of two functions

$$\begin{tikzcd}
D \arrow[d, "f"'] & & E \arrow[d, "g"] \\
B \arrow[r, "i_1"'] & B + C & C \arrow[l, "i_2"]
\end{tikzcd}$$

Sum of two functions

$$\begin{tikzcd}
D \arrow[d, "f"'] \arrow[dr] & & E \arrow[d, "g"] \\
B \arrow[r, "i_1"'] & B + C & C \arrow[l, "i_2"]
\end{tikzcd}$$

Sum of two functions

$$\begin{tikzcd}
D \arrow[d, "f"'] \arrow[dr] & & E \arrow[dl] \arrow[d, "g"] \\
B \arrow[r, "i_1"'] & B + C & C \arrow[l, "i_2"]
\end{tikzcd}$$

Sum of two functions

$$\begin{tikzcd}
D \arrow[r, "i_1"] \arrow[d, "f"'] \arrow[dr] & D + E \arrow[d, "f + g"] & E \arrow[l, "i_2"'] \arrow[d, "g"] \arrow[dl] \\
B \arrow[r, "i_1"'] & B + C & C \arrow[l, "i_2"]
\end{tikzcd}$$

$$f + g = [i_1 \cdot f, i_2 \cdot g]$$

Coproduct laws

“Just reverse the arrows”, cf.

$+$-Absorption $$[h, k] \cdot (f + g) = [h \cdot f, k \cdot g] \qquad (2.43)$$

Coproduct laws

“Just reverse the arrows”, cf.

$+$-Absorption	$[h, k] \cdot (f + g) = [h \cdot f, k \cdot g]$	(2.43)
$+$-Fusion	$f \cdot [h, k] = [f \cdot h, f \cdot k]$	(2.42)

Coproduct laws

“Just reverse the arrows”, cf.

$+$-Absorption $$[h, k] \cdot (f + g) = [h \cdot f, k \cdot g] \quad (2.43)$$

$+$-Fusion $$f \cdot [h, k] = [f \cdot h, f \cdot k] \quad (2.42)$$

$+$-Reflexion $$[i_1, i_2] = id \quad (2.41)$$

Coproduct laws

$+$-Equality

$$[h, k] = [f, g] \Leftrightarrow \left\{ \begin{array}{l} h = f \\ k = g \end{array} \right. \tag{2.66}$$

Coproduct laws

$+$-Equality
$$[h, k] = [f, g] \Leftrightarrow \left\{ \begin{array}{l} h = f \\ k = g \end{array} \right. \tag{2.66}$$

$+$-Functor
$$(h + k) \cdot (f + g) = h \cdot f + k \cdot g \tag{2.44}$$

Coproduct laws

+-Equality $$[h, k] = [f, g] \Leftrightarrow \begin{cases} h = f \\ k = g \end{cases} \qquad (2.66)$$

+-Functor $$(h + k) \cdot (f + g) = h \cdot f + k \cdot g \qquad (2.44)$$

+-Functor-*id* $$id + id = id \qquad (2.45)$$

and so on

All these laws can be found in the reference sheet

(WWW → **Material**)

Summary

$f \cdot g$	Sequential composition
$\langle f, g \rangle$	Parallel composition
$f \times g$	Product composition

Summary

$f \cdot g$	Sequential composition
$\langle f, g \rangle$	Parallel composition
$f \times g$	Product composition
$[f, g]$	Alternative composition

Summary

$f \cdot g$	Sequential composition
$\langle f, g \rangle$	Parallel composition
$f \times g$	Product composition
$[f, g]$	Alternative composition
$f + g$	Structural alternation (coproduct)

Summary

$f \cdot g$	Sequential composition
$\langle f, g \rangle$	Parallel composition
$f \times g$	Product composition

$[f, g]$	Alternative composition
$f + g$	Structural alternation (coproduct)

Compositional programming 🙂

What about...

$$f \times (g + h) \overset{?}{=} f \times g + f \times h$$

What about...

$$f \times (g + h) \overset{?}{=} f \times g + f \times h$$

...?

$$A \times (B + C) \overset{?}{\cong} A \times B + A \times C$$

What about...

$$f \times (g + h) \overset{?}{=} f \times g + f \times h$$

...?

$$A \times (B + C) \overset{?}{\cong} A \times B + A \times C$$

Moreover, any "0" such that

What about...

$$f \times (g + h) \overset{?}{=} f \times g + f \times h$$

...?

$$A \times (B + C) \overset{?}{\cong} A \times B + A \times C$$

Moreover, any "0" such that

$$A \times 0 = 0 \quad ? \; A + 0 = A \quad ??$$

Recall

$$\begin{cases} id : A \rightarrow A \\ id\ a = a \end{cases}$$

$$\begin{tikzcd}
A \arrow[r, bend left, "id"] \arrow[r, phantom, "\cong"] & A \arrow[l, bend left, "id"]
\end{tikzcd}$$

Recall

$$\begin{cases} id : A \to A \\ id\ a = a \end{cases}$$

$$\begin{tikzcd} A \arrow[r, bend left, "id"] \arrow[r, phantom, "\cong"] & A \arrow[l, bend left, "id"] \end{tikzcd}$$

$$\begin{tikzcd} A \arrow[r, bend left, "f"] \arrow[r, phantom, "\cong"] & B \arrow[l, bend left, "g"] \end{tikzcd}$$

$$f \cdot g = id$$

$$g \cdot f = id$$

Recall

$$\begin{cases} swap : A \times B \to B \times A \\ swap\ (a, b) = (b, a) \end{cases}$$

Recall

$$\begin{cases} \mathit{swap} : A \times B \to B \times A \\ \mathit{swap}\,(a, b) = (b, a) \end{cases}$$

$$\begin{tikzcd}
A \times B \arrow[r, bend left, "\mathit{swap}"] \arrow[r, phantom, "\cong"] & B \times A \arrow[l, bend left, "\mathit{swap}"]
\end{tikzcd}$$

Recall

$$\begin{cases} swap : A \times B \to B \times A \\ swap\ (a, b) = (b, a) \end{cases}$$

$$\begin{tikzcd}
A \times B \arrow[rr, bend left, "swap"] \arrow[rr, phantom, "\cong"] & & B \times A \arrow[ll, bend left, "swap"]
\end{tikzcd}$$

$$\begin{tikzcd}
A \arrow[rr, bend left, "?"] \arrow[rr, phantom, "?"] & & A \times B \arrow[ll, bend left, "\pi_1"]
\end{tikzcd}$$

Recall

$$\begin{cases} \mathit{swap} : A \times B \to B \times A \\ \mathit{swap}\ (a, b) = (b, a) \end{cases}$$

$$\begin{tikzcd} A \times B \arrow[r, bend left, "\mathit{swap}"] \arrow[r, phantom, "\cong"] & B \times A \arrow[l, bend left, "\mathit{swap}"] \end{tikzcd}$$

$$\begin{tikzcd} A \arrow[r, bend left, "?"] \arrow[r, phantom, "?"] & A \times B \arrow[l, bend left, "\pi_1"] \end{tikzcd}$$

$$\pi_1 \cdot \langle \mathit{id}, h \rangle = \mathit{id}$$

Recall

$$\begin{cases} swap : A \times B \to B \times A \\ swap\ (a, b) = (b, a) \end{cases}$$

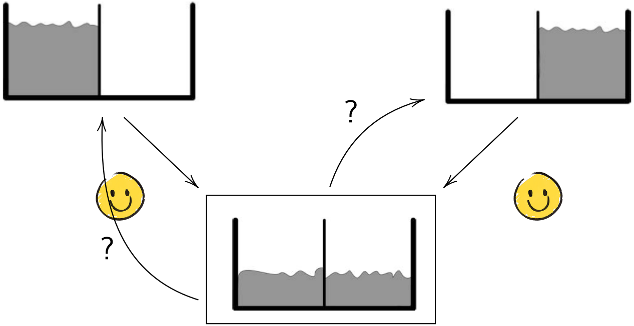
$$\begin{tikzcd} A \times B \arrow[rr, bend left, "swap"] \arrow[rr, phantom, "\cong"] & & B \times A \arrow[ll, bend left, "swap"] \end{tikzcd}$$

$$\begin{tikzcd} A \arrow[rr, bend left, "?"] \arrow[rr, phantom, "?"] & & A \times B \arrow[ll, bend left, "\pi_1"] \end{tikzcd}$$

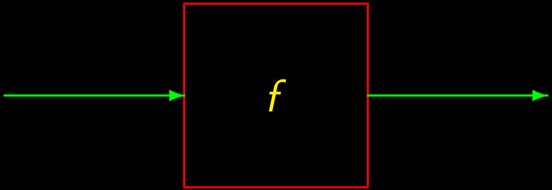
$$\pi_1 \cdot \langle id, h \rangle = id$$

$$\langle id, h \rangle \cdot \pi_1 \neq id$$

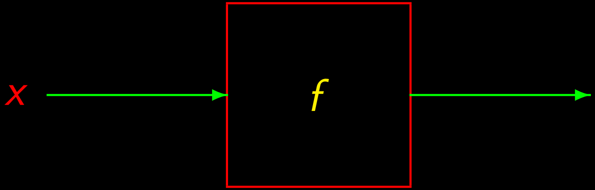
Irreversibility...



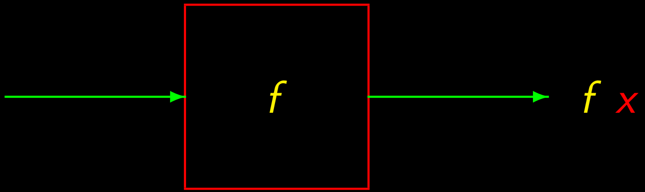
Data flow



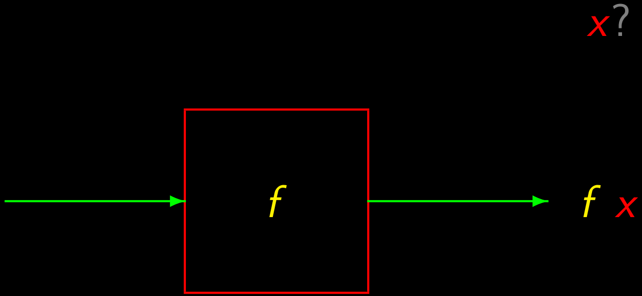
Data flow



Data flow



Data flow

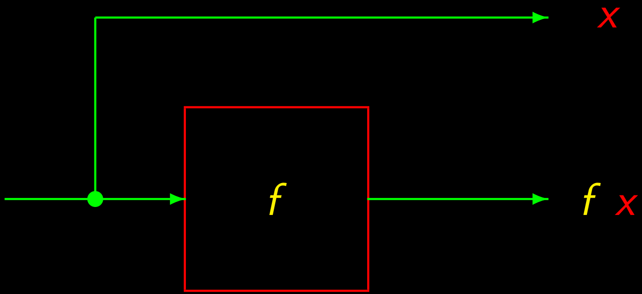


x?

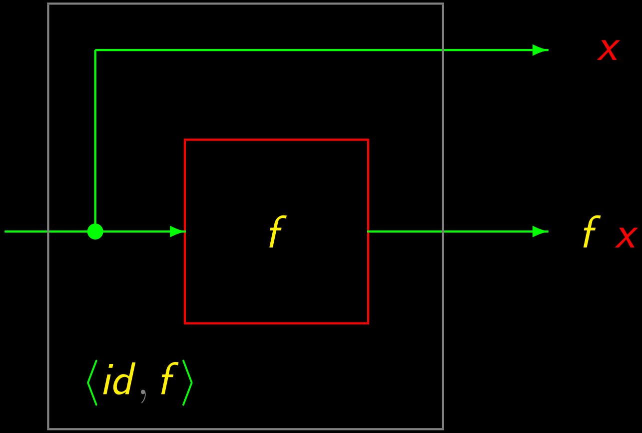
f

$f\ x$

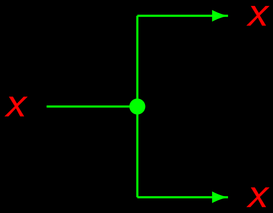
Data flow



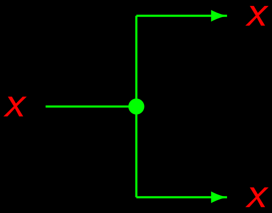
Data flow



Data duplication

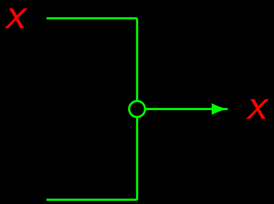


Data duplication



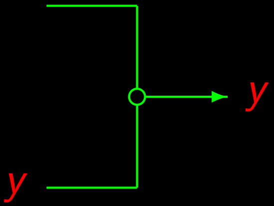
$$dup = \langle id, id \rangle$$

Data joins



$$join = [id, id]$$

Data joins

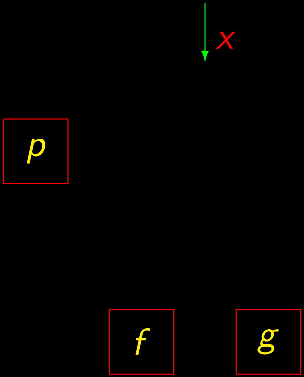


$$join = [id\,,\,id]$$

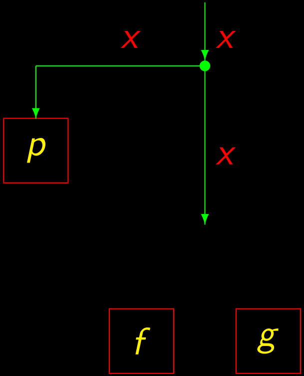
if-then-else

$$y = \textbf{if } p\ x \textbf{ then } f\ x \textbf{ else } g\ x$$

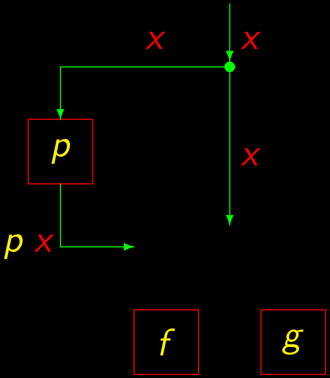
if-then-else



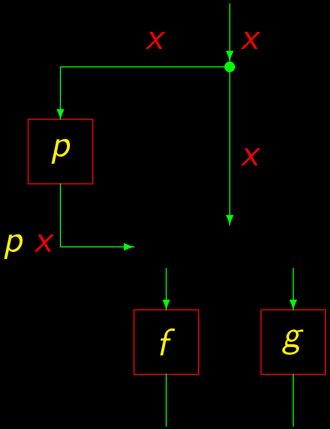
if-then-else



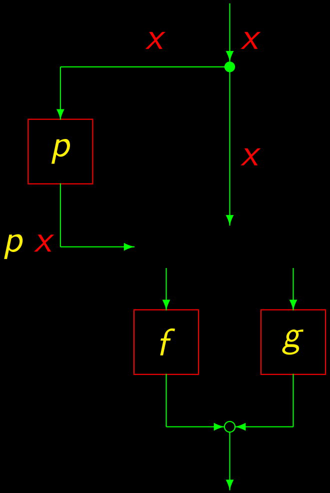
if-then-else



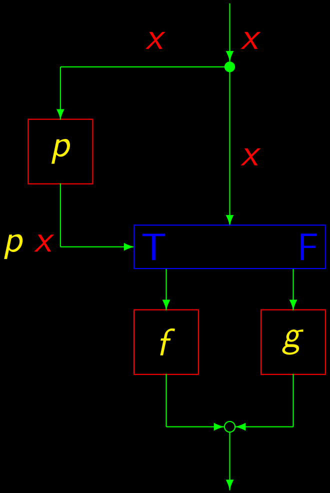
if-then-else



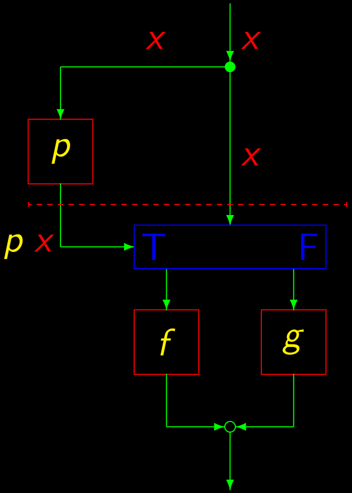
if-then-else



if-then-else



if-then-else



if-then-else (“point-free”)

$x \in A$

if-then-else ("point-free")

$x \in A$

$(p\ x, x) \in \mathbb{B} \times A$

if-then-else (“point-free”)

$$x \in A$$

$$(p\ x, x) \in \mathbb{B} \times A$$

$$\mathbb{B} = \{F, T\}$$

if-then-else (“point-free”)

$x \in A$

$(p\ x, x) \in \mathbb{B} \times A$

$\mathbb{B} = \{F, T\}$

$\mathbb{B}$ in Haskell:

```
data Bool = False | True
```

if-then-else (“point-free”)

$x \in A$

$(p\ x, x) \in \mathbb{B} \times A$

$\mathbb{B} = \{F, T\}$

$\mathbb{B}$ in Haskell:

```
data Bool = False | True
```

T represented by `True`

F represented by `False`

The type 2

$$\mathbf{B} = \{\,F, T\,\} \cong \{\texttt{False}, \texttt{True}\}$$

The type 2

$$\mathbb{B} = \{F, T\} \cong \{\texttt{False}, \texttt{True}\} \cong \{0, 1\} \cong \cdots \cong 2$$

$$\begin{tikzcd}
\mathbb{B} \arrow[r, bend left, "g"] \arrow[r, phantom, "\cong"] & \{0, 1\} \arrow[l, bend left, "f"]
\end{tikzcd}$$

The type 2

$$\mathbb{B} = \{F, T\} \cong \{\texttt{False}, \texttt{True}\} \cong \{0, 1\} \cong \cdots \cong 2$$

$$\begin{tikzcd} \mathbb{B} \arrow[rr, bend left, "g"] \arrow[rr, phantom, "\cong"] & & \{0,1\} \arrow[ll, bend left, "f"] \end{tikzcd}$$

$$\begin{cases} f\ 0 = F \\ f\ 1 = T \end{cases}$$

The type 2

$$\mathbb{B} = \{F, T\} \cong \{\texttt{False}, \texttt{True}\} \cong \{0, 1\} \cong \cdots \cong 2$$

$$\begin{tikzcd}
\mathbb{B} \arrow[rr, bend left, "g"] \arrow[rr, phantom, "\cong"] & & \{0, 1\} \arrow[ll, bend left, "f"]
\end{tikzcd}$$

$$\begin{cases} f\ 0 = F \\ f\ 1 = T \end{cases} \qquad \begin{cases} g\ F = 0 \\ g\ T = 1 \end{cases}$$

$$a + a = 2\,a$$

$$A + A \xrightarrow{join} A$$

$$a + a = 2\,a$$

$$A + A \xrightarrow{join} A$$

$$A \xrightarrow{\langle p, id \rangle} 2 \times A$$

$a + a = 2\ a$

$$\begin{tikzcd} A + A \arrow[r, "join"] & A \end{tikzcd}$$

$$\begin{tikzcd} A \arrow[r, "\langle p, id \rangle"] & 2 \times A \end{tikzcd}$$

Is it the case that...?

$$\begin{tikzcd} 2 \times A \arrow[rr, bend left, "g"] \arrow[rr, phantom, "\cong"] & & A + A \arrow[ll, bend left, "f"] \end{tikzcd}$$

$$\begin{tikzcd}
2 \times A \arrow[rr, bend left, "g"] & \cong & A + A \arrow[ll, bend left, "f"]
\end{tikzcd}$$

$$f = [\langle \underline{T}, id \rangle, \langle \underline{F}, id \rangle]$$

$$g = \cdots (?)$$

Notation

$$\begin{tikzcd}
2 \times A \arrow[rr, bend left, "\alpha^\circ"] \arrow[r, phantom, "\cong"] & {} & A + A \arrow[ll, bend left, "\alpha"]
\end{tikzcd}$$

$$\alpha = [\langle \underline{T}, id \rangle , \langle \underline{F}, id \rangle]$$

$$\alpha^\circ = \cdots$$

Notation

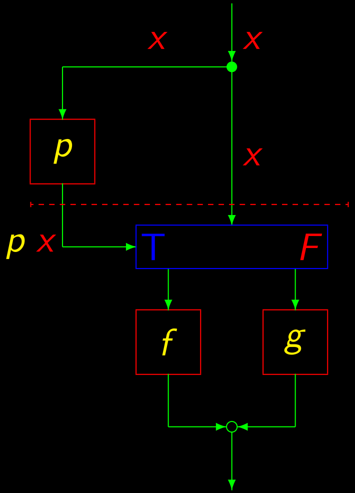
$$\begin{tikzcd} 2 \times A \arrow[rr, bend left, "\alpha^{\circ}"] \arrow[rr, phantom, "\cong"] & & A + A \arrow[ll, bend left, "\alpha"] \end{tikzcd}$$

$$\alpha \cdot \alpha^{\circ} = id$$

$$\alpha^{\circ} \cdot \alpha = id$$

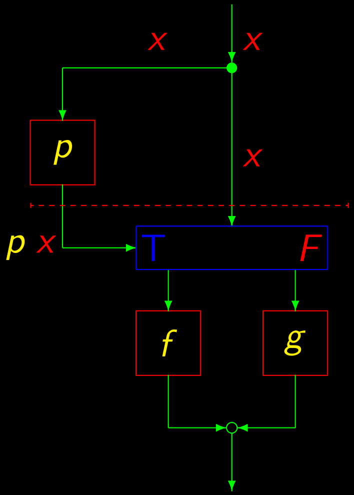
α determines α°

if-then-else



if-then-else

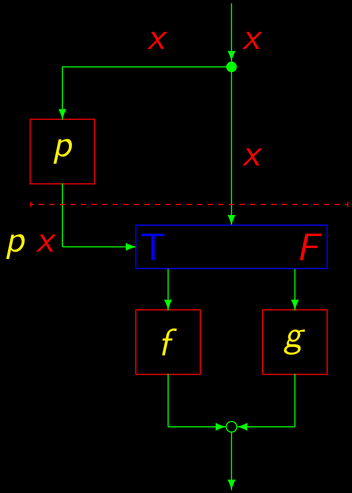
$$\begin{tikzcd}
A \arrow[d, "\langle p, id \rangle"] \\
2 \times A
\end{tikzcd}$$



if-then-else

$$\begin{tikzcd} A \arrow[d, "\langle p, id \rangle"] \\ 2 \times A \end{tikzcd}$$

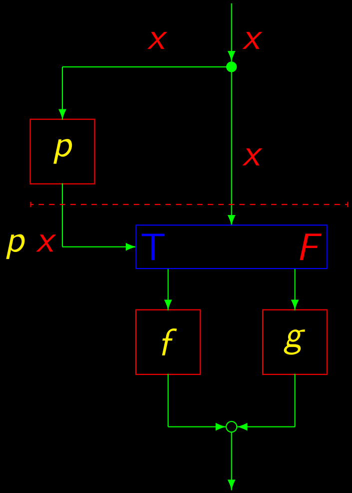
$$\begin{tikzcd} A \arrow[d, "f"'] \\ B \end{tikzcd} \qquad \begin{tikzcd} A \arrow[d, "g"] \\ B \end{tikzcd}$$



if-then-else

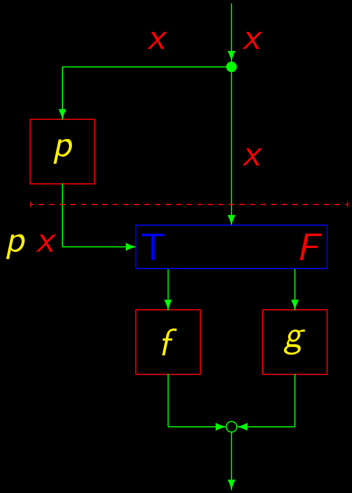
$$\begin{tikzcd} A \arrow[d, "\langle p, id \rangle"] \\ 2 \times A \end{tikzcd}$$

$$\begin{tikzcd} A \arrow[r, "i_1"] \arrow[d, "f"'] & A + A \arrow[d, dashed, "f + g"] & A \arrow[l, "i_2"'] \arrow[d, "g"] \\ B \arrow[r, "i_1"'] & B + B & B \arrow[l, "i_2"] \end{tikzcd}$$



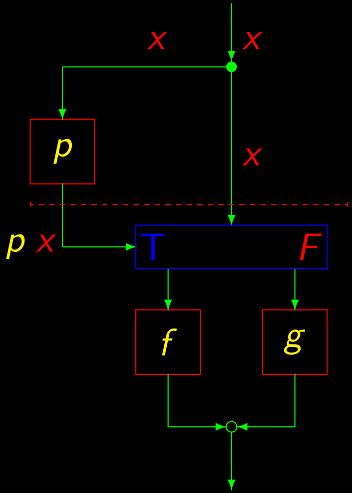
if-then-else

$$\begin{tikzcd}
& A \arrow[d, "\langle p, id \rangle"] & \\
& 2 \times A & \\
A \arrow[r, "i_1"] \arrow[d, "f"'] & A + A \arrow[d, dashed, "f + g"] & A \arrow[l, "i_2"'] \arrow[d, "g"] \\
B \arrow[r, "i_1"'] & B + B \arrow[d, "join"] & B \arrow[l, "i_2"] \\
& B &
\end{tikzcd}$$



if-then-else

$$\begin{tikzcd}
 & A \arrow[d, "\langle p, id \rangle"] & \\
 & 2 \times A \arrow[d, "\alpha^\circ"] & \\
A \arrow[r, "i_1"] \arrow[d, "f"'] & A + A \arrow[d, dashed, "f + g"] & A \arrow[l, "i_2"'] \arrow[d, "g"] \\
B \arrow[r, "i_1"'] & B + B \arrow[d, "join"] & B \arrow[l, "i_2"] \\
 & B &
\end{tikzcd}$$



McCarthy Conditional $p \rightarrow f\ ,\ g$

From the diagram:

$$p \rightarrow f\ ,\ g \quad = \quad join \cdot (f + g) \cdot \alpha^{\circ} \cdot \langle p, id \rangle$$

McCarthy Conditional $p \rightarrow f\ ,\ g$

From the diagram:

$$p \rightarrow f\ ,\ g \quad = \quad join \cdot (f + g) \cdot \alpha^{\circ} \cdot \langle p, id \rangle$$

Since $$join \cdot (f + g) = [id, id] \cdot (f + g) = [f, g]$$

McCarthy Conditional $p \rightarrow f\ ,\ g$

From the diagram:

$$p \rightarrow f\ ,\ g \quad = \quad join \cdot (f + g) \cdot \alpha^{\circ} \cdot \langle p, id \rangle$$

Since

$$join \cdot (f + g) = [id, id] \cdot (f + g) = [f, g]$$

we have:

$$p \rightarrow f\ ,\ g \; = \; [f, g] \cdot \underbrace{\alpha^{\circ} \cdot \langle p, id \rangle}_{p?}$$

McCarthy Conditional $p \rightarrow f\ ,\ g$

p-guard:

$$\begin{tikzcd} A \arrow[r, "\langle p, id \rangle"] \arrow[rr, bend right, "p?"'] & 2 \times A \arrow[r, "\alpha^\circ"] & A + A \end{tikzcd}$$

Conditional:

$$p \rightarrow f\ ,\ g \;=\; [f\,, g] \cdot p?$$

Pointwise

$$p? : A \rightarrow A + A$$

$$p?\ a = \begin{cases} p\ a \Rightarrow i_1\ a \\ \neg\,(p\ a) \Rightarrow i_2\ a \end{cases}$$

Pointwise

$$p? : A \rightarrow A + A$$

$$p?\ a = \begin{cases} p\ a \Rightarrow i_1\ a \\ \neg\ (p\ a) \Rightarrow i_2\ a \end{cases}$$

wherefrom

$$(p \rightarrow f\ ,\ g)\ a = \begin{cases} p\ a \Rightarrow f\ a \\ \neg\ (p\ a) \Rightarrow g\ a \end{cases}$$

Pointwise

$$p? : A \rightarrow A + A$$

$$p?\ a = \begin{cases} p\ a \Rightarrow i_1\ a \\ \neg\ (p\ a) \Rightarrow i_2\ a \end{cases}$$

wherefrom

$$(p \rightarrow f\ ,\ g)\ a = \begin{cases} p\ a \Rightarrow f\ a \\ \neg\ (p\ a) \Rightarrow g\ a \end{cases}$$

Case analysis?

Pointwise

$$p? : A \rightarrow A + A$$
$$p?\ a = \begin{cases} p\ a \Rightarrow i_1\ a \\ \neg\ (p\ a) \Rightarrow i_2\ a \end{cases}$$

wherefrom

$$(p \rightarrow f\ ,\ g)\ a = \begin{cases} p\ a \Rightarrow f\ a \\ \neg\ (p\ a) \Rightarrow g\ a \end{cases}$$

Case analysis?

No!

Between conditional and composition

Any "chemistry"?

$$h \cdot (p \rightarrow f \ , g) = ?$$

Between conditional and composition

Any "chemistry"?

$$h \cdot (p \rightarrow f\ ,\ g) = ?$$

$$(p \rightarrow f\ ,\ g) \cdot h = ?$$

“Calculemus”

$$h \cdot (p \rightarrow f \; , \; g)$$

“Calculemus”

$$h \cdot (p \rightarrow f \;,\; g)$$

$$\Leftrightarrow \qquad \{ \text{ definition of conditional } \}$$

$$h \cdot [f\,, g] \cdot p?$$

“Calculemus”

$$
\begin{array}{ll}
 & h \cdot (p \rightarrow f \;,\; g) \\
\Leftrightarrow & \qquad \{ \text{ definition of conditional } \} \\
 & h \cdot [f, g] \cdot p? \\
\Leftrightarrow & \qquad \{ \; + \text{-fusion } \} \\
 & [h \cdot f, h \cdot g] \cdot p?
\end{array}
$$

“Calculemus”

$$h \cdot (p \rightarrow f \,,\, g)$$

$$\Leftrightarrow \quad \{ \text{ definition of conditional } \}$$

$$h \cdot [f, g] \cdot p?$$

$$\Leftrightarrow \quad \{ \text{ $+$-fusion } \}$$

$$[h \cdot f, h \cdot g] \cdot p?$$

$$\Leftrightarrow \quad \{ \text{ definition of conditional } \}$$

$$p \rightarrow (h \cdot f) \,,\, (h \cdot g)$$

Conditional — 1st fusion law

Altogether:

$$h \cdot (p \rightarrow f \, , g) \; = \; p \rightarrow (h \cdot f) \, , (h \cdot g)$$

Conditional — 1st fusion law

Altogether:

$$h \cdot (p \rightarrow f\ ,\ g) \ = \ p \rightarrow (h \cdot f)\ ,\ (h \cdot g)$$

What about running h **before** the conditional?

$$(p \rightarrow f\ ,\ g) \cdot h = ?$$

Conditional — 2nd fusion law

Intuitively, one has:

$$(p \to f\ ,\ g) \cdot h \ = \ (p \cdot h) \to (f \cdot h)\ ,\ (g \cdot h)$$

Conditional — 2nd fusion law

Intuitively, one has:

$$(p \rightarrow f\ ,\ g) \cdot h \;=\; (p \cdot h) \rightarrow (f \cdot h)\ ,\ (g \cdot h)$$

Intuitively?

Conditional — 2nd fusion law

Intuitively, one has:

$$(p \rightarrow f \ , g) \cdot h \ = \ (p \cdot h) \rightarrow (f \cdot h) \ , (g \cdot h)$$

Intuitively?

Proof ?

Conditional — 2nd fusion law

Intuitively, one has:

$$(p \rightarrow f\ ,\ g) \cdot h \ = \ (p \cdot h) \rightarrow (f \cdot h)\ ,\ (g \cdot h)$$

Intuitively?

Proof ? Later on...

(We need to know more about $p?$)

John McCarthy (1927-2011)

Turing Award

Founding father of **Artificial Intelligence**

PhD in mathematics (Princeton, 1951)

Inventor of **LISP**

Cálculo de Programas

Class T04

Rolf Landauer (1927–99)

“Information is physical”

Even worse than π_1...

$$\begin{cases} \mathit{zero}\ x = 0 \\ \mathit{one}\ x = 1 \end{cases}$$

Even worse than π_1...

$$\begin{cases} \mathit{zero}\ x = 0 \\ \mathit{one}\ x = 1 \end{cases}$$

$$\mathit{one}\ 10 = 1$$

Even worse than π_1...

$$\begin{cases} \mathit{zero}\ x = 0 \\ \mathit{one}\ x = 1 \end{cases}$$

$$\mathit{one}\ 10 = 1$$

$$\mathit{one}\ \texttt{"string"} = 1$$

Even worse than π_1...

$$\begin{cases} \mathit{zero}\ x = 0 \\ \mathit{one}\ x = 1 \end{cases}$$

$$\mathit{one}\ 10 = 1$$

$$\mathit{one}\ \texttt{"string"} = 1$$

$$\mathit{zero}\ [50\,..\,1000] = 0$$

Even worse than π_1...

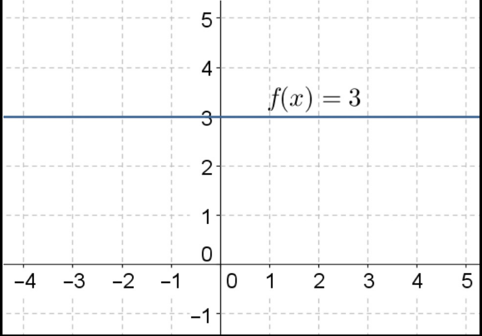
$$\begin{cases} zero\ x = 0 \\ one\ x = 1 \end{cases}$$

$$one\ 10 = 1$$

$$one\ \texttt{"string"} = 1$$

$$zero\ [50\,..\,1000] = 0$$

$$f\ x = 3$$



Constant functions

$3 \in \mathbb{N}_0$

$f\ x = 3$

Constant functions

$3 \in \mathbb{N}_0$

$f\ x = 3$

$A \xrightarrow{f} \mathbb{N}_0$

Constant functions

$3 \in \mathbb{N}_0$

$b_0 \in B$

$f\ x = 3$

$A \xrightarrow{f} \mathbb{N}_0$

Constant functions

$3 \in \mathbb{N}_0$

$b_0 \in B$

$f\ x = 3$

$$\left\{ \begin{array}{l} \underline{b_0} : A \to B \\ \underline{b_0}\ a = b_0 \end{array} \right.$$

$A \xrightarrow{f} \mathbb{N}_0$

Constant functions

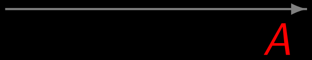
$3 \in \mathbb{N}_0$

$b_0 \in B$

$f\ x = 3$

$$\left\{ \begin{array}{l} \underline{b_0} : A \to B \\ \underline{b_0}\ a = b_0 \end{array} \right.$$

$A \xrightarrow{f} \mathbb{N}_0$



Constant functions

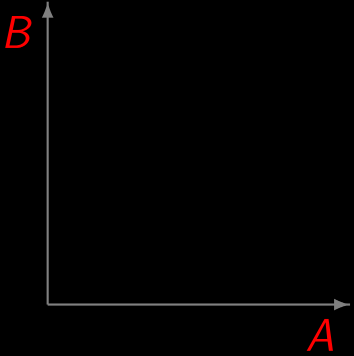
$3 \in \mathbb{N}_0$

$f\ x = 3$

$A \xrightarrow{f} \mathbb{N}_0$

$b_0 \in B$

$$\begin{cases} \underline{b_0} : A \to B \\ \underline{b_0}\ a = b_0 \end{cases}$$



Constant functions

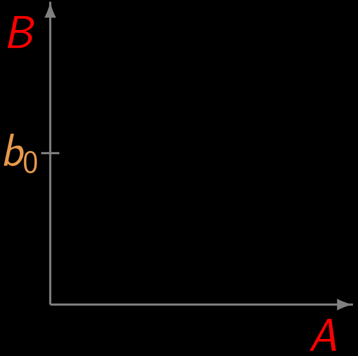
$$3 \in \mathbb{N}_0$$

$$f\ x = 3$$

$$A \xrightarrow{f} \mathbb{N}_0$$

$$b_0 \in B$$

$$\begin{cases} \underline{b_0} : A \to B \\ \underline{b_0}\ a = b_0 \end{cases}$$



Constant functions

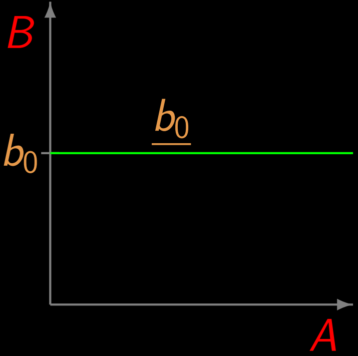
$3 \in \mathbb{N}_0$

$f\ x = 3$

$A \xrightarrow{f} \mathbb{N}_0$

$b_0 \in B$

$$\begin{cases} \underline{b_0} : A \to B \\ \underline{b_0}\ a = b_0 \end{cases}$$



Constant functions

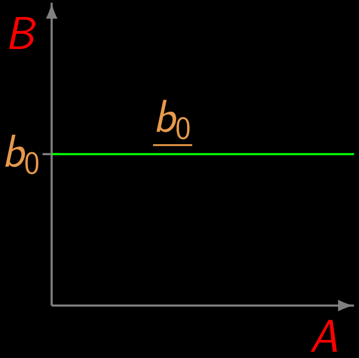
$3 \in \mathbb{N}_0$

$f\ x = 3$

$A \xrightarrow{f} \mathbb{N}_0$

$b_0 \in B$

$$\left\{ \begin{array}{l} \underline{b_0} : A \to B \\ \underline{b_0}\ a = b_0 \end{array} \right.$$



(**Haskell**: `const` b_0)

Properties

$$\underline{b} \cdot f = \underline{b}$$

Properties

$$\underline{b} \cdot f = \underline{b}$$

$$g \cdot \underline{b} = \underline{g \; b}$$

Properties

$$\underline{b} \cdot f = \underline{b}$$

$$g \cdot \underline{b} = \underline{g\ b}$$

$$\begin{tikzcd} B \arrow[r, leftarrow, "\underline{b}"] & A \end{tikzcd}$$

Properties

$$\underline{b} \cdot f = \underline{b}$$

$$g \cdot \underline{b} = \underline{g\ b}$$

$$\begin{tikzcd}
 & C \arrow[d, "f"] \\
B & A \arrow[l, "\underline{b}"']
\end{tikzcd}$$

Properties

$$\underline{b} \cdot f = \underline{b}$$

$$g \cdot \underline{b} = \underline{g\ b}$$

$$\begin{tikzcd}
 & C \arrow[dl, "\underline{b}"'] \arrow[d, "f"] \\
B & A \arrow[l, "\underline{b}"']
\end{tikzcd}$$

Properties

$$\underline{b} \cdot f = \underline{b}$$

$$g \cdot \underline{b} = \underline{g\ b}$$

$$\begin{tikzcd}
& C \arrow[dl, "\underline{b}"'] \arrow[d, "f"] \\
B \arrow[d, "g"'] & A \arrow[l, "\underline{b}"'] \\
D &
\end{tikzcd}$$

Properties

$$\underline{b} \cdot f = \underline{b}$$

$$g \cdot \underline{b} = \underline{g\ b}$$

$$\begin{tikzcd}
 & C \arrow[dl, "\underline{b}"'] \arrow[d, "f"] \\
B \arrow[d, "g"'] & A \arrow[l, "\underline{b}"'] \arrow[dl, "\underline{g\ b}"] \\
D &
\end{tikzcd}$$

Properties

$$\underline{b} \cdot f = \underline{b}$$

$$g \cdot \underline{b} = \underline{g\ b}$$

$$\begin{tikzcd}
 & C \arrow[dl, "\underline{b}"'] \arrow[d, "f"] \\
B \arrow[d, "g"'] & A \arrow[l, "\underline{b}"'] \arrow[dl, "\underline{g\ b}"] \\
D &
\end{tikzcd}$$

No loss of information?

Isomorphisms shining...

Isomorphisms

$$A \times (B \times C) \cong (A \times B) \times C$$ 🙂

$$A \times B \cong B \times A$$ 🙂

Isomorphisms

$$A \times (B \times C) \cong (A \times B) \times C$$ 🙂

$$A \times B \cong B \times A$$ 🙂

$$A + (B + C) \cong (A + B) + C$$

$$A + B \cong B + A$$

Isomorphisms

$$A + A \cong 2 \times A$$

Isomorphisms

$$A + A \cong 2 \times A$$ 🙂

$$A \times A \cong A^2$$?

Isomorphisms

$$A + A \cong 2 \times A$$ 🙂

$$A \times A \cong A^2$$?

$$A \times 1 \cong A$$?

Isomorphisms

$$A + A \cong 2 \times A$$ 🙂

$$A \times A \cong A^2$$?

$$A \times 1 \cong A$$?

$$A \times 0 \cong 0$$?

Isomorphisms

$$A + A \cong 2 \times A$$ 🙂

$$A \times A \cong A^2$$?

$$A \times 1 \cong A$$?

$$A \times 0 \cong 0$$?

$$A + 0 \cong A$$?

Isomorphisms

$$A + A \cong 2 \times A$$ 🙂

$$A \times A \cong A^2$$?

$$A \times 1 \cong A$$?

$$A \times 0 \cong 0$$?

$$A + 0 \cong A$$?

$$A \times (B + C) \cong A \times B + A \times C$$?

Distributivity

$$\begin{tikzcd}
A \times (B + C) \arrow[r, bend left, "\mathit{distr}"] \arrow[r, phantom, "\cong"] & A \times B + A \times C \arrow[l, bend left, "\mathit{undistr}"]
\end{tikzcd}$$

Distributivity

$$\begin{tikzcd}
A \times (B + C) \arrow[rr, bend left, "distr"] \arrow[r, phantom, "\cong"] & & A \times B + A \times C \arrow[ll, bend left, "undistr"]
\end{tikzcd}$$

$$\begin{tikzcd}
(B + C) \times A \arrow[rr, bend left, "distl"] \arrow[r, phantom, "\cong"] & & B \times A + C \times A \arrow[ll, bend left, "undistl"]
\end{tikzcd}$$

Distributivity

$$\begin{tikzcd} A \times (B + C) \arrow[r, bend left, "distr"] \arrow[r, phantom, "\cong"] & A \times B + A \times C \arrow[l, bend left, "undistr"] \end{tikzcd}$$

Distributivity

$$\begin{tikzcd} A \times (B + C) \arrow[r, phantom, "\cong"] \arrow[rr, bend left, "distr"] & \; & A \times B + A \times C \arrow[ll, bend left, "undistr"] \end{tikzcd}$$

$$undistr = [\langle f, g\rangle, \langle h, k\rangle]$$

Distributivity

$$\begin{tikzcd}
A \times (B + C) \arrow[rr, bend left, "distr"] & \cong & A \times B + A \times C \arrow[ll, bend left, "undistr"]
\end{tikzcd}$$

$$undistr = [\langle f, g \rangle, \langle h, k \rangle]$$

$$A \xleftarrow{f} A \times B$$

Distributivity

$$\begin{tikzcd}
A \times (B + C) \arrow[r, bend left, "distr"] & \cong & A \times B + A \times C \arrow[l, bend left, "undistr"]
\end{tikzcd}$$

$$undistr = [\langle f, g \rangle, \langle h, k \rangle]$$

$$A \xleftarrow{f} A \times B$$

$$B + C \xleftarrow{g} A \times B$$

Distributivity

$$\begin{tikzcd}
A \times (B + C) \arrow[rr, bend left, "distr"] & \cong & A \times B + A \times C \arrow[ll, bend left, "undistr"]
\end{tikzcd}$$

$$undistr = [\langle f, g \rangle, \langle h, k \rangle]$$

$$A \xleftarrow{f} A \times B \qquad\qquad A \xleftarrow{h} A \times C$$

$$B + C \xleftarrow{g} A \times B$$

Distributivity

$$\begin{tikzcd}
A \times (B + C) \arrow[rr, bend left, "distr"] & \cong & A \times B + A \times C \arrow[ll, bend left, "undistr"]
\end{tikzcd}$$

$$undistr = [\langle f, g\rangle, \langle h, k\rangle]$$

$$A \xleftarrow{f} A \times B \qquad A \xleftarrow{h} A \times C$$

$$B + C \xleftarrow{g} A \times B \qquad B + C \xleftarrow{k} A \times C$$

Distributivity

$$undistr = [\langle f, g\rangle, \langle h, k\rangle]$$

$$A \xleftarrow{f} A \times B \qquad A \xleftarrow{h} A \times C$$

$$B + C \xleftarrow{g} A \times B \qquad B + C \xleftarrow{k} A \times C$$

Distributivity

$$undistr = [\langle f, g\rangle, \langle h, k\rangle]$$

$$A \xleftarrow{f} A \times B \qquad\qquad A \xleftarrow{h} A \times C$$

$$B + C \xleftarrow{g} A \times B \qquad\qquad B + C \xleftarrow{k} A \times C$$

$$A \xleftarrow{\pi_1} A \times B$$

Distributivity

$$undistr = [\langle f, g \rangle, \langle h, k \rangle]$$

$$A \xleftarrow{f} A \times B \qquad A \xleftarrow{h} A \times C$$

$$B + C \xleftarrow{g} A \times B \qquad B + C \xleftarrow{k} A \times C$$

$$A \xleftarrow{\pi_1} A \times B$$

$$B + C \xleftarrow{i_1 \cdot \pi_2} A \times B$$

Distributivity

$$undistr = [\langle f, g \rangle, \langle h, k \rangle]$$

$$\begin{tikzcd} A & A \times B \arrow[l, "f"'] \end{tikzcd} \qquad \begin{tikzcd} A & A \times C \arrow[l, "h"'] \end{tikzcd}$$

$$\begin{tikzcd} B + C & A \times B \arrow[l, "g"'] \end{tikzcd} \qquad \begin{tikzcd} B + C & A \times C \arrow[l, "k"'] \end{tikzcd}$$

$$\begin{tikzcd} A & A \times B \arrow[l, "\pi_1"'] \end{tikzcd} \qquad \begin{tikzcd} A & A \times C \arrow[l, "\pi_1"'] \end{tikzcd}$$

$$\begin{tikzcd} B + C & A \times B \arrow[l, "i_1 \cdot \pi_2"'] \end{tikzcd}$$

Distributivity

$$undistr = [\langle f, g\rangle, \langle h, k\rangle]$$

$$\begin{tikzcd} A & A \times B \arrow[l, "f"'] \end{tikzcd} \qquad \begin{tikzcd} A & A \times C \arrow[l, "h"'] \end{tikzcd}$$

$$\begin{tikzcd} B + C & A \times B \arrow[l, "g"'] \end{tikzcd} \qquad \begin{tikzcd} B + C & A \times C \arrow[l, "k"'] \end{tikzcd}$$

$$\begin{tikzcd} A & A \times B \arrow[l, "\pi_1"'] \end{tikzcd} \qquad \begin{tikzcd} A & A \times C \arrow[l, "\pi_1"'] \end{tikzcd}$$

$$\begin{tikzcd} B + C & A \times B \arrow[l, "i_1 \cdot \pi_2"'] \end{tikzcd} \qquad \begin{tikzcd} B + C & A \times C \arrow[l, "i_2 \cdot \pi_2"'] \end{tikzcd}$$

$$undistr = [\langle \pi_1, i_1 \cdot \pi_2\rangle, \langle \pi_1, i_2 \cdot \pi_2\rangle]$$

Distributivity

$$\mathit{undistr} = [\langle \pi_1, i_1 \cdot \pi_2 \rangle, \langle \pi_1, i_2 \cdot \pi_2 \rangle]$$

$$\begin{tikzcd}
A \times (B + C) \arrow[rr, bend left, "\mathit{distr}"] & \cong & A \times B + A \times C \arrow[ll, bend left, "\mathit{undistr}"]
\end{tikzcd}$$

Distributivity

$$\mathit{undistr} = [\langle \pi_1, i_1 \cdot \pi_2 \rangle, \langle \pi_1, i_2 \cdot \pi_2 \rangle]$$

$$\begin{tikzcd}
A \times (B + C) \arrow[rr, bend left, "\mathit{distr}"] & \cong & A \times B + A \times C \arrow[ll, bend left, "\mathit{undistr}"]
\end{tikzcd}$$

By definition of $f \times g$

$$\mathit{undistr} = [\mathit{id} \times i_1, \mathit{id} \times i_2]$$

The exchange law

The exchange law

$$\begin{tikzcd}
A + B \arrow[d, "\alpha"] \\
{}
\end{tikzcd}$$

The exchange law

$$\begin{tikzcd}A + B \arrow[d, "\alpha"] \\ C \times D\end{tikzcd}$$

Exchange law

$$\begin{tikzcd}
A \arrow[r, "i_1"] \arrow[dr, "i"'] & A + B \arrow[d, "{[i, j]}"] & B \arrow[l, "i_2"'] \arrow[dl, "j"] \\
& C \times D &
\end{tikzcd}$$

Exchange law

$$\begin{tikzcd}
 & A + B \arrow[dl, "m"'] \arrow[d, "\langle m, n \rangle"] \arrow[dr, "n"] & \\
C & C \times D \arrow[l, "\pi_1"] \arrow[r, "\pi_2"'] & D
\end{tikzcd}$$

Exchange law

$$\langle m, n \rangle = [i, j]$$

One equation, four unknowns

Exchange law

$$\begin{tikzcd}
 & A + B \arrow[dl, "m"'] \arrow[d, "\langle m, n \rangle"] \arrow[dr, "n"] & \\
C & C \times D \arrow[l, "\pi_1"] \arrow[r, "\pi_2"'] & D
\end{tikzcd}$$

Exchange law

$$\begin{tikzcd}
A \arrow[r, "i_1"] \arrow[d, "f"] & A + B & B \arrow[l, "i_2"'] \arrow[dll, "g"'] \\
C & C \times D \arrow[l, "\pi_1"] \arrow[r, "\pi_2"'] & D
\end{tikzcd}$$

Exchange law

$$\begin{tikzcd}
A \arrow[r, "i_1"] \arrow[drr, "h"'] & A + B & B \arrow[l, "i_2"'] \arrow[d, "k"] \\
C & C \times D \arrow[l, "\pi_1"] \arrow[r, "\pi_2"'] & D
\end{tikzcd}$$

Exchange law

$$\begin{tikzcd}
A \arrow[r, "i_1"] & A + B \arrow[dl, "{m = [f , g]}"'] \arrow[d, "{\langle m, n \rangle}" description] \arrow[dr, "{n = [h, k]}"] & B \arrow[l, "i_2"'] \\
C & C \times D \arrow[l, "\pi_1"] \arrow[r, "\pi_2"'] & D
\end{tikzcd}$$

Exchange law

$$\begin{tikzcd}
A \arrow[r, "i_1"] \arrow[dr, "i = \langle f , h \rangle"'] & A + B \arrow[d, "{[i , j]}" description] & B \arrow[l, "i_2"'] \arrow[dl, "j = \langle g , k \rangle"] \\
C & C \times D \arrow[l, "\pi_1"] \arrow[r, "\pi_2"'] & D
\end{tikzcd}$$

Summing up

Exchange law:

$$\langle [f, g], [h, k] \rangle = [\langle f, h \rangle, \langle g, k \rangle]$$

"I have a functional **split** but need an **alternative**..."

... and vice-versa

Exchange law — Example

$$undistr = [\langle \pi_1, i_1 \cdot \pi_2 \rangle, \langle \pi_1, i_2 \cdot \pi_2 \rangle]$$

$$\begin{tikzcd}
A \times (B + C) \arrow[r, bend left, "distr"] \arrow[r, phantom, "\cong"] & A \times B + A \times C \arrow[l, bend left, "undistr"]
\end{tikzcd}$$

Exchange law — Example

$$undistr = [\langle \pi_1, i_1 \cdot \pi_2 \rangle, \langle \pi_1, i_2 \cdot \pi_2 \rangle]$$

$$\begin{tikzcd} A \times (B + C) \arrow[r, phantom, "\cong"] \arrow[r, bend left, "distr"] & A \times B + A \times C \arrow[l, bend left, "undistr"] \end{tikzcd}$$

$$undistr = \langle [\pi_1, \pi_1], [i_1 \cdot \pi_2, i_2 \cdot \pi_2] \rangle$$

The types 0, 1 and 2...

The type 1

In Haskell:

$$() \; :: \; ()$$

Then

$$1 = \{()\}$$

Isomorphism

$$\begin{tikzcd} A \times 1 \arrow[rr, bend left, "\pi_1"] \arrow[rr, phantom, "\cong"] & & A \arrow[ll, bend left, "\langle id, \mathbf{!} \rangle"] \end{tikzcd}$$

The type 1

In Haskell:

$$() :: ()$$

Then

$$1 = \{()\}$$

Isomorphism

$$\begin{tikzcd} A \times 1 \arrow[rr, bend left, "\pi_1"] \arrow[rr, phantom, "\cong"] & & A \arrow[ll, bend left, "\langle id, \mathbf{!} \rangle"] \end{tikzcd}$$

... "**!**" ?

The type 1

Fact functions involving type 1
are necessarily **constant**

The type 1

Fact functions involving type 1 are necessarily **constant**

"Bang":

$$A \xrightarrow{!} 1$$

$$! = \underline{()}$$

The type 1

Fact functions involving type 1
are necessarily **constant**

"Bang":

$$A \xrightarrow{!} 1$$

$$! = \underline{()}$$

"Points":

$$1 \xrightarrow{\underline{a}} A$$

$$\underline{a}\,() = a$$

(provided $a \in A$)

The type 0

$$0 = \{\,\}$$

The type 0

$$0 = \{\,\}$$

Immediate in e.g. Haskell:

```
data Zero
```

The type 0

$$0 = \{\,\}$$

Immediate in e.g. Haskell:

```
data Zero
```

0 is not inhabited: $\neg\,(a \in 0)$ for all a

The type 0

$$0 = \{\,\}$$

Immediate in e.g. Haskell:

```
data Zero
```

0 is not inhabited: $\neg\,(a \in 0)$ for all a

Fact impossible $f : A \to 0$ whenever $A \neq 0$

The type 0

$$\begin{tikzcd} A \times 0 \arrow[rr, bend left, "id"] & = & 0 \arrow[ll, bend left, "id"] \end{tikzcd}$$

The type 0

$$\begin{tikzcd} A \times 0 \arrow[rr, bend left, "id"] \arrow[rr, phantom, "="] & & 0 \arrow[ll, bend left, "id"] \end{tikzcd}$$

In fact

$$A \times 0 = \{(a, b) \mid a \in A \wedge b \in 0\}$$

The type 0

$$\begin{tikzcd} A \times 0 \arrow[rr, bend left, "id"] \arrow[rr, phantom, "="] & & 0 \arrow[ll, bend left, "id"] \end{tikzcd}$$

In fact

$$A \times 0 = \{(a, b) \mid a \in A \wedge b \in 0\}$$

$$A \times 0 = \{(a, b) \mid a \in A \wedge F\}$$

The type 0

$$\begin{tikzcd} A \times 0 \arrow[r, bend left, "id"] \arrow[r, phantom, "="] & 0 \arrow[l, bend left, "id"] \end{tikzcd}$$

In fact

$$A \times 0 = \{(a, b) \mid a \in A \wedge b \in 0\}$$

$$A \times 0 = \{(a, b) \mid a \in A \wedge F\}$$

$$A \times 0 = \{\}$$

The type $1 + A$

$$1 + A$$

The type $1 + A$

$$\begin{tikzcd}1 \arrow[r, "i_1"] & 1 + A & A \arrow[l, "i_2"']\end{tikzcd}$$

The type $1 + A$

$$\begin{tikzcd}
1 \arrow[r, "i_1"] & 1 + A & A \arrow[l, "i_2"'] \\
 & B &
\end{tikzcd}$$

The type $1 + A$

$$\begin{tikzcd}
1 \arrow[r, "i_1"] & 1 + A & A \arrow[l, "i_2"'] \arrow[dl, "f"] \\
 & B &
\end{tikzcd}$$

The type $1 + A$

$$\begin{tikzcd}
1 \arrow[r, "i_1"] \arrow[dr, "\underline{b_0}"'] & 1 + A & A \arrow[l, "i_2"'] \arrow[dl, "f"] \\
& B &
\end{tikzcd}$$

The type $1 + A$

$$\begin{tikzcd}
1 \arrow[r, "i_1"] \arrow[dr, "\underline{b_0}"'] & 1 + A \arrow[d, "{[\underline{b_0}, f]}"] & A \arrow[l, "i_2"'] \arrow[dl, "f"] \\
& B &
\end{tikzcd}$$

The type $1 + A$

$$\begin{tikzcd}
1 \arrow[r, "i_1"] \arrow[dr, "\underline{b_0}"'] & 1 + A \arrow[d, "{[\underline{b_0}, f]}"] & A \arrow[l, "i_2"'] \arrow[dl, "f"] \\
 & B &
\end{tikzcd}$$

$1 + A$ ”pointer to A”

More isomorphisms!

More isomorphisms!

Back to

> *Retrieve the address of a civil servant, knowing that she/he can be identified either by a citizen card number (CC) or a fiscal number (NIF).*

More isomorphisms!

Back to

> *Retrieve the address of a civil servant, knowing that she/he can be identified either by a citizen card number (CC) or a fiscal number (NIF).*

In Haskell:

```
data Iden = CC Int | NIF Int
```

More isomorphisms!

> *Obter a morada of a funcionário público, knowing that este se can identificar através d its number of the cartão of cidadão (CC) ou d its number of identificação fiscal (NIF).*

In Haskell (Portugal):

```
data Iden = CC Int | NIF Int
```

In Haskell (Germany):

```
data Iden = IDK Int | SZN Int
```

More isomorphisms!

$$\begin{tikzcd}
Int \arrow[r, "i_1"] \arrow[dr, "\mathtt{CC}"'] & Int + Int \arrow[d, "{[\mathtt{CC}, \mathtt{NIF}]}"] & Int \arrow[l, "i_2"'] \arrow[dl, "\mathtt{NIF}"] \\
& \mathtt{Iden} &
\end{tikzcd}$$

```
data Iden = CC Int | NIF Int
```

More isomorphisms!

$$\begin{tikzcd}\mathtt{Iden} \arrow[rr, bend left, "\mathbf{out}=\mathbf{in}^{\circ}"] & \cong & Int + Int \arrow[ll, bend left, "\mathbf{in}={[\mathtt{CC},\mathtt{NIF}]}"]\end{tikzcd}$$

More isomorphisms!

$$\begin{tikzcd}
\mathtt{Iden} \arrow[r, bend left, "\mathbf{out}=\mathbf{in}^\circ"] \arrow[r, phantom, "\cong"] & \mathit{Int}+\mathit{Int} \arrow[l, bend left, "\mathbf{in}={[\mathtt{CC},\mathtt{NIF}]}"]
\end{tikzcd}$$

Concrete syntax

More isomorphisms!

$$\begin{tikzcd}
\mathtt{Iden} \arrow[r, bend left, "\mathbf{out}=\mathbf{in}^{\circ}"] \arrow[r, phantom, "\cong"] & \mathit{Int} + \mathit{Int} \arrow[l, bend left, "\mathbf{in}=[\mathtt{CC},\mathtt{NIF}]"]
\end{tikzcd}$$

Concrete syntax

Abstract syntax

More isomorphisms!

$$\begin{tikzcd}
\mathtt{Iden} \arrow[rr, bend left, "\mathbf{out}{=}\mathbf{in}^{\circ}"] \arrow[rr, phantom, "\cong"] & & \mathit{Int} + \mathit{Int} \arrow[ll, bend left, "\mathbf{in}{=}{[\mathtt{CC},\mathtt{NIF}]}"]
\end{tikzcd}$$

Concrete syntax

Abstract syntax

in (“get in”)

More isomorphisms!

$$\begin{tikzcd}
\texttt{Iden} \arrow[rr, bend left, "\mathbf{out}=\mathbf{in}^{\circ}"] \arrow[rr, phantom, "\cong"] & & \mathit{Int}+\mathit{Int} \arrow[ll, bend left, "\mathbf{in}=[\texttt{CC},\texttt{NIF}]"]
\end{tikzcd}$$

Concrete syntax

Abstract syntax

in ("get in")

out ("get out")

Maybe

$$\begin{tikzcd}
1 \arrow[r, "i_1"] \arrow[dr, swap, "\underline{\texttt{Nothing}}"] & 1 + A \arrow[d, "{[\underline{\texttt{Nothing}}, \texttt{Just}]}"] & A \arrow[l, swap, "i_2"] \arrow[dl, "\texttt{Just}"] \\
 & \texttt{Maybe}\ A &
\end{tikzcd}$$

```
data Maybe a = Nothing | Just a
```

Maybe

$$\begin{tikzcd}
\mathtt{Maybe}\ A \arrow[rr, bend left, "\mathsf{out}=\mathsf{in}^{\circ}"] & \cong & 1 + A \arrow[ll, bend left, "\mathsf{in} = {[\underline{\mathtt{Nothing}}, \mathtt{Just}]}"]
\end{tikzcd}$$

Maybe

$$\begin{tikzcd}
\mathtt{Maybe}\ A \arrow[rr, bend left, "\mathsf{out} = \mathsf{in}^{\circ}"] & \cong & 1 + A \arrow[ll, bend left, "\mathsf{in} = {[\underline{\mathtt{Nothing}}, \mathtt{Just}]}"]
\end{tikzcd}$$

$$\mathsf{out} \cdot \mathsf{in} = id$$

Maybe — calculate out

$$\mathsf{out} \cdot \mathsf{in} = id$$

Maybe — calculate **out**

$$
\begin{aligned}
& \mathsf{out} \cdot \mathsf{in} = \mathit{id} \\
\Leftrightarrow \quad & \{\ \mathsf{in} = [\underline{\mathtt{Nothing}}, \mathtt{Just}]\ \} \\
& \mathsf{out} \cdot [\underline{\mathtt{Nothing}}, \mathtt{Just}] = \mathit{id}
\end{aligned}
$$

Maybe — calculate out

$$\mathsf{out} \cdot \mathsf{in} = id$$

$$\Leftrightarrow \quad \{ \ \mathsf{in} = [\underline{\mathtt{Nothing}}, \mathtt{Just}] \ \}$$

$$\mathsf{out} \cdot [\underline{\mathtt{Nothing}}, \mathtt{Just}] = id$$

$$\Leftrightarrow \quad \{ \ +\text{-fusion} \ \}$$

$$[\mathsf{out} \cdot \underline{\mathtt{Nothing}}, \mathsf{out} \cdot \mathtt{Just}] = id$$

Maybe — calculate **out**

$$\mathsf{out} \cdot \mathsf{in} = id$$

$$\Leftrightarrow \qquad \{ \ \mathsf{in} = [\underline{\mathtt{Nothing}}, \mathtt{Just}] \ \}$$

$$\mathsf{out} \cdot [\underline{\mathtt{Nothing}}, \mathtt{Just}] = id$$

$$\Leftrightarrow \qquad \{ \ +\text{-fusion} \ \}$$

$$[\mathsf{out} \cdot \underline{\mathtt{Nothing}}, \mathsf{out} \cdot \mathtt{Just}] = id$$

$$\Leftrightarrow \qquad \{ \ \text{universal-}+\text{; constant function} \ \}$$

$$\left\{ \begin{array}{l} \underline{\mathsf{out}\ \mathtt{Nothing}} = i_1 \\ \mathsf{out} \cdot \mathtt{Just} = i_2 \end{array} \right.$$

Maybe

$$\Leftrightarrow \qquad \{ \ \text{variables} \ ; \underline{\texttt{Nothing}} : 1 \rightarrow \texttt{Maybe} \ A \ \}$$

$$\begin{cases} \underline{\textbf{out} \ \texttt{Nothing}} \ () = i_1 \ () \\ (\textbf{out} \cdot \texttt{Just}) \ a = i_2 \ a \end{cases}$$

Maybe

$$\Leftrightarrow \qquad \{ \text{ variables ; } \underline{\mathtt{Nothing}} : 1 \to \mathtt{Maybe}\ A \ \}$$

$$\left\{ \begin{array}{l} \underline{\mathbf{out}\ \mathtt{Nothing}}\ () = i_1\ () \\ (\mathbf{out} \cdot \mathtt{Just})\ a = i_2\ a \end{array} \right.$$

$$\Leftrightarrow \qquad \{ \text{ composition } (\textit{pointwise}) \ \}$$

$$\left\{ \begin{array}{l} \mathbf{out}\ \mathtt{Nothing} = i_1\ () \\ \mathbf{out}\ (\mathtt{Just}\ a) = i_2\ a \end{array} \right.$$

Maybe

$\Leftrightarrow \quad \{ \text{ variables ; } \underline{\mathtt{Nothing}} : 1 \rightarrow \mathtt{Maybe}\ A \ \}$

$$\begin{cases} \underline{\mathbf{out}\ \mathtt{Nothing}}\ () = i_1\ () \\ (\mathbf{out} \cdot \mathtt{Just})\ a = i_2\ a \end{cases}$$

$\Leftrightarrow \quad \{ \text{ composition } (\textit{pointwise}) \ \}$

$$\begin{cases} \mathbf{out}\ \mathtt{Nothing} = i_1\ () \\ \mathbf{out}\ (\mathtt{Just}\ a) = i_2\ a \end{cases}$$

$$\begin{tikzcd}
\mathtt{Maybe}\ A \arrow[r, bend left, "\mathbf{out}"] \arrow[r, phantom, "\cong"] & 1 + A \arrow[l, bend left, "\mathbf{in} = {[\underline{\mathtt{Nothing}}, \mathtt{Just}]}"]
\end{tikzcd}$$

Either

$$\begin{tikzcd}
\texttt{Either}\ A\ B \arrow[rr, bend left, "\mathsf{out}"] \arrow[rr, phantom, "\cong"] & & A + B \arrow[ll, bend left, "\mathsf{in} = {[}\texttt{Left}{,}\texttt{Right}{]}"]
\end{tikzcd}$$

Either

$$\begin{tikzcd}
\texttt{Either}\ A\ B \arrow[r, bend left, "\mathsf{out}"] \arrow[r, phantom, "\cong"] & A + B \arrow[l, bend left, "\mathsf{in} = {[\texttt{Left}, \texttt{Right}]}"]
\end{tikzcd}$$

$$\begin{tikzcd}
A \arrow[r, "i_1"] \arrow[dr, "\texttt{Left}"'] & A + B \arrow[d, "{[\texttt{Left}, \texttt{Right}]}"] & B \arrow[l, "i_2"'] \arrow[dl, "\texttt{Right}"] \\
& \texttt{Either}\ A\ B &
\end{tikzcd}$$

$[f, g]$

$\texttt{either}\ f\ g$

Cálculo
de
Programas

Class T05

Polymorphism
(and why it matters!)

Polymorphism

$$reverse :: [a] \to [a]$$

Polymorphism

$$reverse :: [a] \to [a]$$

$$reverse :: [Int] \to [Int]$$

Polymorphism

$$reverse :: [a] \rightarrow [a]$$

$$reverse :: [Int] \rightarrow [Int]$$
$$reverse :: String \rightarrow String$$

Polymorphism

$$reverse :: [a] \to [a]$$

$$reverse :: [Int] \to [Int]$$
$$reverse :: String \to String$$
$$\ldots$$

Polymorphism

$$reverse :: [a] \rightarrow [a]$$

$$reverse :: [Int] \rightarrow [Int]$$
$$reverse :: String \rightarrow String$$
$$\ldots$$

"From the **type** of a **polymorphic function** we can derive a **theorem** that it satisfies. (...) How useful are the theorems so generated? Only time and experience will tell (...)" [**Philip Wadler** 1989]

Natural properties

$$[2, 5, 0, 5, 1]$$

Natural properties

$$\begin{tikzcd}[column sep=huge] {[1, 5, 0, 5, 2]} & {[2, 5, 0, 5, 1]} \arrow[l, mapsto, "\mathit{reverse}"'] \end{tikzcd}$$

Natural properties

$$\begin{tikzcd}
{[1, 5, 0, 5, 2]} \arrow[d, mapsto, "\mathit{succs}"'] & {[2, 5, 0, 5, 1]} \arrow[l, mapsto, "\mathit{reverse}"'] \\
{[2, 6, 1, 6, 3]} &
\end{tikzcd}$$

Natural properties

$$\begin{tikzcd}
{[1, 5, 0, 5, 2]} \arrow[d, mapsto, "succs"'] & {[2, 5, 0, 5, 1]} \arrow[l, mapsto, "reverse"'] \arrow[d, mapsto, "succs"] \\
{[2, 6, 1, 6, 3]} & {[3, 6, 1, 6, 2]}
\end{tikzcd}$$

Natural properties

$$\begin{tikzcd}
{[1,5,0,5,2]} \arrow[d, mapsto, "\mathit{succs}"'] & {[2,5,0,5,1]} \arrow[l, mapsto, "\mathit{reverse}"'] \arrow[d, mapsto, "\mathit{succs}"] \\
{[2,6,1,6,3]} & {[3,6,1,6,2]} \arrow[l, mapsto, "\mathit{reverse}"]
\end{tikzcd}$$

Natural properties

$$\begin{tikzcd}
{[1, 5, 0, 5, 2]} \arrow[d, mapsto, "\mathit{map\ succ}"'] & {[2, 5, 0, 5, 1]} \arrow[l, mapsto, "\mathit{reverse}"'] \arrow[d, mapsto, "\mathit{map\ succ}"] \\
{[2, 6, 1, 6, 3]} & {[3, 6, 1, 6, 2]} \arrow[l, mapsto, "\mathit{reverse}"]
\end{tikzcd}$$

Natural properties

$$\begin{tikzcd}
{[1, 5, 0, 5, 2]} & {[2, 5, 0, 5, 1]} \arrow[l, mapsto, "\textit{reverse}"']
\end{tikzcd}$$

Natural properties

$$\begin{tikzcd}
{[1, 5, 0, 5, 2]} \arrow[d, mapsto, "map\ even"'] & {[2, 5, 0, 5, 1]} \arrow[l, mapsto, "reverse"'] \\
{[F, F, T, F, T]} &
\end{tikzcd}$$

Natural properties

$$\begin{tikzcd}
{[1, 5, 0, 5, 2]} \arrow[d, mapsto, "\mathit{map}\ \mathit{even}"'] & & {[2, 5, 0, 5, 1]} \arrow[ll, mapsto, "\mathit{reverse}"'] \arrow[d, mapsto, "\mathit{map}\ \mathit{even}"] \\
{[F, F, T, F, T]} & & {[T, F, T, F, F]}
\end{tikzcd}$$

Natural properties

$$\begin{tikzcd}[1, 5, 0, 5, 2] \arrow[d, mapsto, "\mathit{map}\ \mathit{even}"'] & & {[2, 5, 0, 5, 1]} \arrow[ll, mapsto, "\mathit{reverse}"'] \arrow[d, mapsto, "\mathit{map}\ \mathit{even}"] \\ {[F, F, T, F, T]} & & {[T, F, T, F, F]} \arrow[ll, mapsto, "\mathit{reverse}"]\end{tikzcd}$$

Natural properties

$$\begin{tikzcd}[2, 5, 0] & {[2, 5, 0, 5, 1]} \arrow[l, mapsto, "\mathit{take}\ 3"']\end{tikzcd}$$

Natural properties

$$\begin{tikzcd}
{[2, 5, 0]} \arrow[d, mapsto, "\mathit{map}\ (2*)"'] & {[2, 5, 0, 5, 1]} \arrow[l, mapsto, "\mathit{take}\ 3"'] \\
{[4, 10, 0]} &
\end{tikzcd}$$

Natural properties

$$\begin{tikzcd}
{[2, 5, 0]} \arrow[d, mapsto, "\mathit{map}\ (2*)"'] & & {[2, 5, 0, 5, 1]} \arrow[ll, mapsto, "\mathit{take}\ 3"'] \arrow[d, mapsto, "\mathit{map}\ (2*)"] \\
{[4, 10, 0]} & & {[4, 10, 0, 10, 2]}
\end{tikzcd}$$

Natural properties

$$\begin{tikzcd}
{[2, 5, 0]} \arrow[d, mapsto, "\mathit{map}\ (2*)"'] & & {[2, 5, 0, 5, 1]} \arrow[ll, mapsto, "\mathit{take}\ 3"'] \arrow[d, mapsto, "\mathit{map}\ (2*)"] \\
{[4, 10, 0]} & & {[4, 10, 0, 10, 2]} \arrow[ll, mapsto, "\mathit{take}\ 3"]
\end{tikzcd}$$

Natural properties

$$\begin{tikzcd}
\texttt{"ogramas"} & \texttt{"Programas"} \arrow[l, mapsto, "\mathit{drop}\ 2"']
\end{tikzcd}$$

Natural properties

$$\begin{tikzcd}
\texttt{"ogramas"} \arrow[d, mapsto, "\mathit{map}\ \mathit{upper}"'] & \texttt{"Programas"} \arrow[l, mapsto, "\mathit{drop}\ 2"'] \\
\texttt{"OGRAMAS"} &
\end{tikzcd}$$

Natural properties

$$\begin{tikzcd}
\texttt{"ogramas"} \arrow[d, mapsto, "\mathit{map}\ \mathit{upper}"'] & \texttt{"Programas"} \arrow[l, mapsto, "\mathit{drop}\ 2"'] \arrow[d, mapsto, "\mathit{map}\ \mathit{upper}"] \\
\texttt{"OGRAMAS"} & \texttt{"PROGRAMAS"}
\end{tikzcd}$$

Natural properties

$$\begin{tikzcd}
\texttt{"ogramas"} \arrow[d, mapsto, "\mathit{map}\ \mathit{upper}"'] & \texttt{"Programas"} \arrow[l, mapsto, "\mathit{drop}\ 2"'] \arrow[d, mapsto, "\mathit{map}\ \mathit{upper}"] \\
\texttt{"OGRAMAS"} & \texttt{"PROGRAMAS"} \arrow[l, mapsto, "\mathit{drop}\ 2"]
\end{tikzcd}$$

Back to...

$$\begin{tikzcd}(A \times B) \times C \arrow[rr, bend left, "assocr"] & \cong & A \times (B \times C) \arrow[ll, bend left, "assocl"]\end{tikzcd}$$

$$assocr \cdot assocl = id$$

$$assocl \cdot assocr = id$$

“Natural” property

$$((2, T), \texttt{"CP"}) \xleftarrow{\mathit{assocl}} (2, (T, \texttt{"CP"}))$$

“Natural” property

$$\begin{tikzcd} ((2, T), \texttt{"CP"}) \arrow[d, mapsto, "(\mathit{succ} \times \neg) \times \,!"'] & (2, (T, \texttt{"CP"})) \arrow[l, mapsto, "\mathit{assocl}"'] \\ ((3, F), ()) & \end{tikzcd}$$

“Natural” property

$$\begin{tikzcd}
((2, T), \texttt{"CP"}) \arrow[d, mapsto, "(\mathit{succ} \times \neg) \times \mathbf{!}"'] & & (2, (T, \texttt{"CP"})) \arrow[ll, mapsto, "\mathit{assocl}"'] \arrow[d, mapsto, "\mathit{succ} \times (\neg \times \mathbf{!})"] \\
((3, F), ()) & & (3, (F, ()))
\end{tikzcd}$$

“Natural” property

$$\begin{tikzcd}
((2, T), \texttt{"CP"}) \arrow[d, mapsto, "(\mathit{succ} \times \neg) \times \mathbf{!}"'] & (2, (T, \texttt{"CP"})) \arrow[l, mapsto, "\mathit{assocl}"'] \arrow[d, mapsto, "\mathit{succ} \times (\neg \times \mathbf{!})"] \\
((3, F), ()) & (3, (F, ())) \arrow[l, mapsto, "\mathit{assocl}"]
\end{tikzcd}$$

“Natural” property

$$\begin{tikzcd} ((a, b), c) & (a, (b, c)) \arrow[l, mapsto, "\mathit{assocl}"'] \end{tikzcd}$$

“Natural” property

$$\begin{tikzcd}
((a, b), c) \arrow[d, mapsto, swap, "(f \times g) \times h"] & (a, (b, c)) \arrow[l, mapsto, swap, "assocl"] \\
((f\ a, g\ b), h\ c) &
\end{tikzcd}$$

“Natural” property

$$\begin{tikzcd}
((a, b), c) \arrow[d, mapsto, "(f \times g) \times h"'] & (a, (b, c)) \arrow[l, mapsto, "\mathit{assocl}"'] \arrow[d, mapsto, "f \times (g \times h)"] \\
((f\ a, g\ b), h\ c) & (f\ a, (g\ b, h\ c))
\end{tikzcd}$$

“Natural” property

$$\begin{tikzcd}
((a, b), c) \arrow[d, mapsto, "(f \times g) \times h"'] & & (a, (b, c)) \arrow[ll, mapsto, "\mathit{assocl}"'] \arrow[d, mapsto, "f \times (g \times h)"] \\
((f\ a, g\ b), h\ c) & & (f\ a, (g\ b, h\ c)) \arrow[ll, mapsto, "\mathit{assocl}"]
\end{tikzcd}$$

Natural properties

$$A \times (B \times C)$$

Natural properties

$$\begin{tikzcd}(A \times B) \times C & & A \times (B \times C) \arrow[ll, "assocl"']\end{tikzcd}$$

Natural properties

$$\begin{tikzcd}
(A \times B) \times C \arrow[d, "(f \times g) \times h"'] & A \times (B \times C) \arrow[l, "assocl"'] \\
(D \times E) \times F &
\end{tikzcd}$$

Natural properties

$$\begin{tikzcd}
(A \times B) \times C \arrow[d, "(f \times g) \times h"'] & & A \times (B \times C) \arrow[ll, "assocl"'] \arrow[d, "f \times (g \times h)"] \\
(D \times E) \times F & & D \times (E \times F)
\end{tikzcd}$$

Natural properties

$$\begin{tikzcd}
(A \times B) \times C \arrow[d, "(f \times g) \times h"'] & & A \times (B \times C) \arrow[ll, "assocl"'] \arrow[d, "f \times (g \times h)"] \\
(D \times E) \times F & & D \times (E \times F) \arrow[ll, "assocl"]
\end{tikzcd}$$

$$((f \times g) \times h) \cdot assocl = assocl \cdot (f \times (g \times h))$$

Back to...

$$\begin{tikzcd}
A \times B \arrow[rr, bend left, "swap"] & \cong & B \times A \arrow[ll, bend left, "swap"]
\end{tikzcd}$$

$$swap \cdot swap = id$$

Practical rule

$$\begin{tikzcd} B \times A & A \times B \arrow[l, "swap"'] \end{tikzcd}$$

Practical rule

$$\begin{tikzcd}
B \times A & A \times B \arrow[l, "\mathit{swap}"'] \\
\\
B' \times A' & A' \times B' \arrow[l, "\mathit{swap}"]
\end{tikzcd}$$

Practical rule

$$\begin{tikzcd}
B \times A \arrow[d, "{''B \times A''}"'] & & A \times B \arrow[ll, "swap"'] \arrow[d, "{''A \times B''}"] \\
B' \times A' & & A' \times B' \arrow[ll, "swap"]
\end{tikzcd}$$

Practical rule

$$\begin{tikzcd}
B \times A \arrow[d, "g \times f"'] & A \times B \arrow[l, "swap"'] \arrow[d, "f \times g"] \\
B' \times A' & A' \times B' \arrow[l, "swap"]
\end{tikzcd}$$

$$(g \times f) \cdot swap = swap \cdot (f \times g)$$

Practical rule

In the vertical arrows,

- Substitute $A := f$, $B := g$, etc

Practical rule

In the vertical arrows,

- Substitute $A := f$, $B := g$, etc
- In case of concrete types, replace by id, e.g. $2 := id$

Practical rule

In the vertical arrows,

- Substitute $A := f$, $B := g$, etc
- In case of concrete types, replace by id, e.g. $2 := id$
- Remove the quotes.

Simplest case...

$$\begin{tikzcd}
A & A \arrow[l, "id"']
\end{tikzcd}$$

Simplest case...

$$\begin{tikzcd}
A & A \arrow[l, "id"'] \\
A' & A' \arrow[l, "id"]
\end{tikzcd}$$

Simplest case...

$$\begin{tikzcd}
A \arrow[d, "{''A''}"'] & A \arrow[l, "id"'] \arrow[d, "{''A''}"] \\
A' & A' \arrow[l, "id"]
\end{tikzcd}$$

Natural-*id*

$$\begin{tikzcd}
A \arrow[d, "f"'] & A \arrow[l, "id"'] \arrow[d, "f"] \\
A' & A' \arrow[l, "id"]
\end{tikzcd}$$

$$f \cdot id = id \cdot f$$

Natural-π_1

$$\begin{tikzcd}
A & A \times B \arrow[l, "\pi_1"']
\end{tikzcd}$$

Natural-π_1

$$\begin{tikzcd}
A & A \times B \arrow[l, "\pi_1"'] \\
A' & A' \times B' \arrow[l, "\pi_1"]
\end{tikzcd}$$

Natural-π_1

$$\begin{tikzcd}
A \arrow[d, "\text{''}A\text{''}"'] & A \times B \arrow[l, "\pi_1"'] \arrow[d, "\text{''}A \times B\text{''}"] \\
A' & A' \times B' \arrow[l, "\pi_1"]
\end{tikzcd}$$

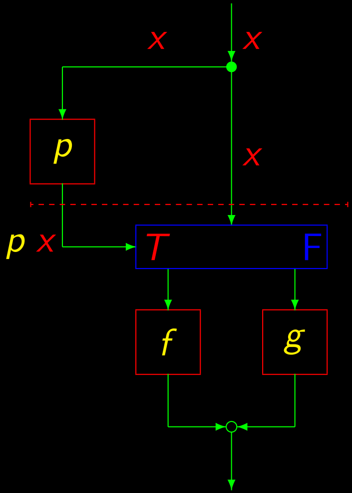
Natural-π_1

$$\begin{tikzcd}
A \arrow[d, "f"'] & A \times B \arrow[l, "\pi_1"'] \arrow[d, "f \times g"] \\
A' & A' \times B' \arrow[l, "\pi_1"]
\end{tikzcd}$$

$$(f) \cdot \pi_1 = \pi_1 \cdot (f \times g)$$

Back to "if-then-else"

$$\begin{tikzcd}
 & A \arrow[d, "\langle p, id \rangle"] & \\
 & 2 \times A \arrow[d, "\alpha^{\circ}"] & \\
A \arrow[r, "i_1"] \arrow[dr, "f"'] & A + A & A \arrow[l, "i_2"'] \arrow[dl, "g"] \\
 & B &
\end{tikzcd}$$



2ª fusion law of conditionals

$$(p \to f\ ,\ g) \cdot h \ = \ p \cdot h \to f \cdot h\ ,\ g \cdot h$$

2ª fusion law of conditionals

$$(p \rightarrow f\ ,\ g) \cdot h \ = \ p \cdot h \rightarrow f \cdot h\ ,\ g \cdot h$$

follows from

$$p? \cdot f \ = \ (f + f) \cdot (p \cdot f)?$$

2ª fusion law of conditionals

$$(p \to f\ ,\ g) \cdot h \;=\; p \cdot h \to f \cdot h\ ,\ g \cdot h$$

follows from

$$p? \cdot f \;=\; (f + f) \cdot (p \cdot f)?$$

So we are left with the **proof** of the equality above...

Let us do it now

Prove

$$p? \cdot f = (f + f) \cdot (p \cdot f)?$$

knowing:

$$\begin{tikzcd}
A \arrow[r, "\langle p, id \rangle"] \arrow[rr, bend right, "p?"'] & 2 \times A \arrow[r, "\alpha^\circ"] & A + A
\end{tikzcd}$$

Recall

$$\begin{tikzcd}
2 \times A \arrow[rr, bend left, "\alpha^{\circ}"] & \cong & A + A \arrow[ll, bend left, "\alpha"]
\end{tikzcd}$$

$$\alpha = [\langle \underline{T}, id \rangle, \langle \underline{F}, id \rangle]$$

$$\alpha^{\circ} = \cdots ?$$

Let us try it

$p? \cdot f$

Let us try it

$$
\begin{aligned}
& p? \cdot f \\
= \quad & \{ \text{ substitution } p? = \alpha^{\circ} \cdot \langle p, id \rangle \ \} \\
& \alpha^{\circ} \cdot \langle p, id \rangle \cdot f
\end{aligned}
$$

Let us try it

$$
\begin{array}{cl}
 & p? \cdot f \\
= & \{ \text{ substitution } p? = \alpha^{\circ} \cdot \langle p, id \rangle \ \} \\
 & \alpha^{\circ} \cdot \langle p, id \rangle \cdot f \\
= & \{ \ \times\text{-fusion } \} \\
 & \alpha^{\circ} \cdot \langle p \cdot f, id \cdot f \rangle
\end{array}
$$

Let us try it

$$p? \cdot f$$

$$= \quad \{ \text{ substitution } p? = \alpha^{\circ} \cdot \langle p, id \rangle \ \}$$

$$\alpha^{\circ} \cdot \langle p, id \rangle \cdot f$$

$$= \quad \{ \ \times\text{-fusion } \}$$

$$\alpha^{\circ} \cdot \langle p \cdot f , id \cdot f \rangle$$

$$= \quad \{ \text{ natural-}id \text{ twice } \}$$

$$\alpha^{\circ} \cdot \langle id \cdot p \cdot f , f \cdot id \rangle$$

Let us try it

$$p? \cdot f$$

$$= \quad \{ \text{ substitution } p? = \alpha^{\circ} \cdot \langle p, id \rangle \ \}$$

$$\alpha^{\circ} \cdot \langle p, id \rangle \cdot f$$

$$= \quad \{ \ \times\text{-fusion } \}$$

$$\alpha^{\circ} \cdot \langle p \cdot f, id \cdot f \rangle$$

$$= \quad \{ \text{ natural-}id \text{ twice } \}$$

$$\alpha^{\circ} \cdot \langle id \cdot p \cdot f, f \cdot id \rangle$$

$$= \quad \{ \ \times\text{-absorption } \}$$

$$\alpha^{\circ} \cdot (id \times f) \cdot \langle p \cdot f, id \rangle$$

Let us try it

$$
\begin{array}{ll}
& p? \cdot f \\
= & \{ \text{ substitution } p? = \alpha^{\circ} \cdot \langle p, id \rangle \ \} \\
& \alpha^{\circ} \cdot \langle p, id \rangle \cdot f \\
= & \{ \ \times\text{-fusion } \} \\
& \alpha^{\circ} \cdot \langle p \cdot f , id \cdot f \rangle \\
= & \{ \text{ natural-}id \text{ twice } \} \\
& \alpha^{\circ} \cdot \langle id \cdot p \cdot f , f \cdot id \rangle
\end{array}
$$

$$
\begin{array}{ll}
= & \{ \ \times\text{-absorption } \} \\
& \alpha^{\circ} \cdot (id \times f) \cdot \langle p \cdot f , id \rangle \\
= & \{ \ ?? \ \} \\
& ?? \\
= & \{ \ ?? \ \} \\
& (f + f) \cdot (p \cdot f)?
\end{array}
$$

Recall

$$\begin{tikzcd}
2 \times A \arrow[rr, bend left, "\alpha^\circ"] & \cong & A + A \arrow[ll, bend left, "\alpha"]
\end{tikzcd}$$

$$\alpha = [\langle \underline{T}, id \rangle , \langle \underline{F}, id \rangle]$$

$$\alpha^\circ = \cdots ?$$

Natural-α°

$$\begin{tikzcd} A + A & 2 \times A \arrow[l, "\alpha^\circ"'] \end{tikzcd}$$

Natural-α°

$$\begin{tikzcd}
A + A & 2 \times A \arrow[l, "\alpha^\circ"'] \\
A' + A' & 2 \times A' \arrow[l, "\alpha^\circ"]
\end{tikzcd}$$

Natural-α°

$$\begin{tikzcd}
A + A \arrow[d, "{''A + A''}"'] & & 2 \times A \arrow[ll, "\alpha^\circ"'] \arrow[d, "{''2 \times A''}"] \\
A' + A' & & 2 \times A' \arrow[ll, "\alpha^\circ"]
\end{tikzcd}$$

Natural-α°

$$\begin{tikzcd}
A + A \arrow[d, "f + f"'] & & 2 \times A \arrow[ll, "\alpha^\circ"'] \arrow[d, "id \times f"] \\
A' + A' & & 2 \times A' \arrow[ll, "\alpha^\circ"]
\end{tikzcd}$$

$$(f + f) \cdot \alpha^\circ = \alpha^\circ \cdot (id \times f)$$

… and this finishes the proof!

$$p? \cdot f$$

$$= \quad \{ \ p? = \alpha^\circ \cdot \langle p, id \rangle \ \}$$

$$\alpha^\circ \cdot \langle p, id \rangle \cdot f$$

$$= \quad \{ \ \times\text{-fusion} \ \}$$

$$\alpha^\circ \cdot \langle p \cdot f, id \cdot f \rangle$$

$$= \quad \{ \ \text{natural-}id \text{ twice} \ \}$$

$$\alpha^\circ \cdot \langle id \cdot p \cdot f, f \cdot id \rangle$$

$$= \quad \{ \ \times\text{-absorption} \ \}$$

$$\alpha^\circ \cdot (id \times f) \cdot \langle p \cdot f, id \rangle$$

… and this finishes the proof!

$$
\begin{aligned}
& p? \cdot f \\
= & \quad \{ \; p? = \alpha^{\circ} \cdot \langle p, id \rangle \; \} \\
& \alpha^{\circ} \cdot \langle p, id \rangle \cdot f \\
= & \quad \{ \; \times\text{-fusion} \; \} \\
& \alpha^{\circ} \cdot \langle p \cdot f, id \cdot f \rangle \\
= & \quad \{ \; \text{natural-}id \text{ twice} \; \} \\
& \alpha^{\circ} \cdot \langle id \cdot p \cdot f, f \cdot id \rangle
\end{aligned}
$$

$$
\begin{aligned}
= & \quad \{ \; \times\text{-absorption} \; \} \\
& \alpha^{\circ} \cdot (id \times f) \cdot \langle p \cdot f, id \rangle \\
= & \quad \{ \; \text{free thef } \alpha^{\circ} \; \} \\
& (f + f) \cdot \alpha^{\circ} \cdot \langle p \cdot f, id \rangle
\end{aligned}
$$

… and this finishes the proof!

$$
\begin{array}{ll}
 & p? \cdot f \\
= & \{\ p? = \alpha^{\circ} \cdot \langle p, \mathit{id} \rangle\ \} \\
 & \alpha^{\circ} \cdot \langle p, \mathit{id} \rangle \cdot f \\
= & \{\ \times\text{-fusion}\ \} \\
 & \alpha^{\circ} \cdot \langle p \cdot f, \mathit{id} \cdot f \rangle \\
= & \{\ \text{natural-}\mathit{id}\ \text{twice}\ \} \\
 & \alpha^{\circ} \cdot \langle \mathit{id} \cdot p \cdot f, f \cdot \mathit{id} \rangle
\end{array}
$$

$$
\begin{array}{ll}
= & \{\ \times\text{-absorption}\ \} \\
 & \alpha^{\circ} \cdot (\mathit{id} \times f) \cdot \langle p \cdot f, \mathit{id} \rangle \\
= & \{\ \text{free thef}\ \alpha^{\circ}\ \} \\
 & (f + f) \cdot \alpha^{\circ} \cdot \langle p \cdot f, \mathit{id} \rangle \\
= & \{\ p? = \alpha^{\circ} \cdot \langle p, \mathit{id} \rangle\ \} \\
 & (f + f) \cdot (p \cdot f)?
\end{array}
$$

☺

Higher-order functions (and why they matter!)

Notation variants

$2 + 3$	**infix** notation	operator between

Notation variants

$2 + 3$	**infix** notation	operator between
$+\ 2\ 3$	**prefix** notation	operator before

Notation variants

$2 + 3$	**infix** notation	operator between
$+ \ 2 \ 3$	**prefix** notation	operator before
$2 \ 3 \ +$	**posfix** notation	operator after

Notation variants

$2+3$	**infix** notation	operator between
$+\ 2\ 3$	**prefix** notation	operator before
$2\ 3\ +$	**posfix** notation	operator after
$+\ (2,3)$	**prefix** “uncurried”	grouped inputs

Notation variants

$f\ a\ b$ **prefix** notation “curried” separated inputs

Notation variants

$f\ a\ b$	**prefix** notation “curried”	separated inputs
$f\ (a, b)$	**prefix** “uncurried”	grouped inputs

“Curried”? “uncurried”

Terminology in honour of mathematician **Haskell Curry** (1900-1982) who developed the theory behind **functional programming** (1936).

Example

$$\pi_1 : A \times B \to A$$

Example

$$\pi_1 : A \times B \to A$$

$$\pi_1 \left(a, b\right) = a$$

Example

$$\pi_1 : A \times B \to A$$

$$\pi_1\left(a, b\right) = a$$

(notation: prefix “uncurried”)

Exponentials

```
Prelude> p1(a,b)=a
Prelude> p1’ a b = a
Prelude> :t p1
p1 :: (a, b) -> a
Prelude> :t p1’
p1’ :: a -> b -> a
Prelude>
```

In Haskell

$$\pi_1{}' = \texttt{curry}\ \pi_1$$

In Haskell

$$\pi_1{}' = \texttt{curry}\ \pi_1$$

$$\pi_1 = \texttt{uncurry}\ \pi_1{}'$$

In Haskell

$$\pi_1{}' = \texttt{curry}\ \pi_1$$

$$\pi_1 = \texttt{uncurry}\ \pi_1{}'$$

Important:

- `curry` (e `uncurry`) accept functions as **arguments**
- `curry` (e `uncurry`) yield functions as **outputs**

In Haskell

$$\pi_1{}' = \texttt{curry}\ \pi_1$$

$$\pi_1 = \texttt{uncurry}\ \pi_1{}'$$

Important:

- `curry` (e `uncurry`) accept functions as **arguments**
- `curry` (e `uncurry`) yield functions as **outputs**

They are said to be

higher order functions.

Curry | uncurry

$$\texttt{curry} \; :: ((a, b) \to c) \to (a \to b \to c)$$

$$\texttt{uncurry} \; :: (a \to b \to c) \to ((a, b) \to c)$$

Curry | uncurry

$$\texttt{curry} :: \boxed{((a, b) \to c)} \to (a \to b \to c)$$

$$\texttt{uncurry} :: (a \to b \to c) \to \boxed{((a, b) \to c)}$$

Curry | uncurry

$$\texttt{curry} :: \boxed{((a, b) \to c)} \to \boxed{(a \to b \to c)}$$

$$\texttt{uncurry} :: \boxed{(a \to b \to c)} \to \boxed{((a, b) \to c)}$$

Curry | uncurry

$$\texttt{curry} \; :: \boxed{((a, b) \to c)} \to \boxed{(a \to b \to c)}$$

$$\texttt{uncurry} \; :: \boxed{(a \to b \to c)} \to \boxed{((a, b) \to c)}$$

each other inverses?

Curry | uncurry

$$\texttt{curry} :: ((a, b) \rightarrow c) \rightarrow a \rightarrow b \rightarrow c$$

Curry | uncurry

$\texttt{curry} :: ((a, b) \to c) \to a \to b \to c$

`curry`

Curry | uncurry

$$\texttt{curry} :: ((a, b) \rightarrow c) \rightarrow a \rightarrow b \rightarrow c$$

$$\texttt{curry}\ f$$

Curry | uncurry

$$\texttt{curry} \; :: ((a, b) \rightarrow c) \rightarrow a \rightarrow b \rightarrow c$$

$$\texttt{curry} \; f \; a$$

Curry | uncurry

$$\texttt{curry} :: ((a, b) \to c) \to a \to b \to c$$

$$\texttt{curry}\ f\ a\ b$$

Curry | uncurry

$$\texttt{curry} :: ((a, b) \to c) \to a \to b \to c$$

$$\texttt{curry}\ f\ a\ b\ =$$

Curry | uncurry

$$\texttt{curry} :: ((a, b) \to c) \to a \to b \to c$$

$$\texttt{curry}\ f\ a\ b = f\ (a, b)$$

Curry | uncurry

$$\texttt{curry} :: ((a, b) \to c) \to a \to b \to c$$

$$\texttt{curry}\ f\ a\ b = f\ (a, b)$$

$$\texttt{uncurry} :: (a \to b \to c) \to (a, b) \to c$$

Curry | uncurry

$$\texttt{curry} :: ((a, b) \to c) \to a \to b \to c$$

$$\texttt{curry}\ f\ a\ b = f\ (a, b)$$

$$\texttt{uncurry} :: (a \to b \to c) \to (a, b) \to c$$

$$\texttt{uncurry}$$

Curry | uncurry

$$\texttt{curry} :: ((a, b) \to c) \to a \to b \to c$$

$$\texttt{curry}\ f\ a\ b = f\ (a, b)$$

$$\texttt{uncurry} :: (a \to b \to c) \to (a, b) \to c$$

$$\texttt{uncurry}\ g$$

Curry | uncurry

$$\texttt{curry} :: ((a, b) \to c) \to a \to b \to c$$

$$\texttt{curry}\ f\ a\ b\ = f\ (a, b)$$

$$\texttt{uncurry} :: (a \to b \to c) \to (a, b) \to c$$

$$\texttt{uncurry}\ g\ (a, b)$$

Curry | uncurry

$$\texttt{curry} :: ((a, b) \to c) \to a \to b \to c$$

$$\texttt{curry}\ f\ a\ b\ = f\ (a, b)$$

$$\texttt{uncurry} :: (a \to b \to c) \to (a, b) \to c$$

$$\texttt{uncurry}\ g\ (a, b)\ =$$

Curry | uncurry

$$\texttt{curry} :: ((a, b) \to c) \to a \to b \to c$$

$$\texttt{curry}\ f\ a\ b\ = f\ (a, b)$$

$$\texttt{uncurry} :: (a \to b \to c) \to (a, b) \to c$$

$$\texttt{uncurry}\ g\ (a, b)\ = g\ a\ b$$

Curry | uncurry

$$\begin{cases} \texttt{curry}\ (\texttt{uncurry}\ f) = f? \\ \texttt{uncurry}\ (\texttt{curry}\ f) = f? \end{cases}$$

Curry | uncurry

$$\texttt{uncurry}\ g\ (a, b) = g\ a\ b$$

Curry | uncurry

$$\texttt{uncurry}\ (\texttt{curry}\ f)\ (a, b) = \texttt{curry}\ f\ a\ b$$

$$\texttt{curry}\ f\ a\ b = f\ (a, b)$$

Curry | uncurry

$$\texttt{uncurry}\ (\texttt{curry}\ f)\ (a, b) = f\ (a, b)$$

$$f\ x = g\ x \Leftrightarrow f = g$$

Curry | uncurry

$$\texttt{uncurry}\ (\texttt{curry}\ f) = f$$

$$f\ (g\ x) = (f \cdot g)\ x$$

Curry | uncurry

$$(\texttt{uncurry} \cdot \texttt{curry})\ f = f$$

$$id\ x = x$$

Curry | uncurry

$$(\texttt{uncurry} \cdot \texttt{curry})\, f = id\ f$$

$$f\ x = g\ x \Leftrightarrow f = g$$

Curry | uncurry

$$\texttt{uncurry} \cdot \texttt{curry} = \mathit{id}$$

□

Curry | uncurry

$$\texttt{curry}\ (\texttt{uncurry}\ f) = f\ ?$$

Curry | uncurry

$$\texttt{curry}\ f\ a\ b = f\ (a, b)$$

Curry | uncurry

$$\texttt{curry}\ (\texttt{uncurry}\ g)\ a\ b = \texttt{uncurry}\ g\ (a, b)$$

$$\texttt{uncurry}\ g\ (a, b) = g\ a\ b$$

Curry | uncurry

$$\texttt{curry}\ (\texttt{uncurry}\ g)\ a\ b = g\ a\ b$$

$$f\ x = g\ x \Leftrightarrow f = g$$

Curry | uncurry

$$\texttt{curry}\ (\texttt{uncurry}\ g)\ a = g\ a$$

$$f\ x = g\ x \Leftrightarrow f = g$$

Curry | uncurry

$$\texttt{curry}\ (\texttt{uncurry}\ g) = g$$

$$f\ (g\ x) = (f \cdot g)\ x$$

Curry | uncurry

$$(\texttt{curry} \cdot \texttt{uncurry})\, g = g$$

$$\mathit{id}\ x = x$$

Curry | uncurry

$$(\texttt{curry} \cdot \texttt{uncurry})\ g = id\ g$$

$$f\ x = g\ x \Leftrightarrow f = g$$

Curry | uncurry

$$\texttt{curry} \cdot \texttt{uncurry} = id$$

□

Isomorphism

$$\begin{tikzcd}(a, b) \to c \arrow[r, phantom, "\cong"] \arrow[r, bend left, "\texttt{curry}"] & a \to b \to c \arrow[l, bend left, "\texttt{uncurry}"]\end{tikzcd}$$

The type B^A

The type B^A

$$B^A = \{ f \mid f : A \to B \}$$

NB: functions that map A to B

Isomorphism

$$\begin{tikzcd}
A \times B \to C \arrow[r, bend left, "\texttt{curry}"] \arrow[r, phantom, "\cong"] & A \to (B \to C) \arrow[l, bend left, "\texttt{uncurry}"]
\end{tikzcd}$$

Isomorphism

$$\begin{tikzcd} A \times B \to C \arrow[r, bend left, "\texttt{curry}"] \arrow[r, phantom, "\cong"] & A \to C^B \arrow[l, bend left, "\texttt{uncurry}"] \end{tikzcd}$$

Isomorphism

$$\begin{tikzcd}
C^{A\times B} \arrow[r, bend left, "\texttt{curry}"] \arrow[r, phantom, "\cong"] & \left(C^{B}\right)^{A} \arrow[l, bend left, "\texttt{uncurry}"]
\end{tikzcd}$$

Cf. arithmetic exponentials: $c^{a\,b} = \left(c^{b}\right)^{a}$

More isomorphisms

$a^1 = a$ $\quad$ $1^a = 1$ $\quad$ $a^0 = 1$

$$\begin{tikzcd}
A^1 \arrow[r, bend left, "g"] \arrow[r, phantom, "\cong"] & A \arrow[l, bend left, "f"]
\end{tikzcd}
\qquad
\begin{tikzcd}
1^A \arrow[r, bend left, "g"] \arrow[r, phantom, "\cong"] & 1 \arrow[l, bend left, "f"]
\end{tikzcd}
\qquad
\begin{tikzcd}
A^0 \arrow[r, bend left, "g"] \arrow[r, phantom, "\cong"] & 1 \arrow[l, bend left, "f"]
\end{tikzcd}$$

$$\begin{cases} f \ a = \underline{a} \\ g \ \underline{a} = a \end{cases}
\qquad
\begin{cases} f \ () = ! \\ g = ! \end{cases}
\qquad
\begin{cases} f \ () = \bot \\ g = ! \end{cases}$$

Curry | uncurry

$$\begin{tikzcd}
A \times B \to C \arrow[r, bend left, "\texttt{curry}"] \arrow[r, phantom, "\cong"] & A \to C^B \arrow[l, bend left, "\texttt{uncurry}"]
\end{tikzcd}$$

Curry | uncurry

f

$$\begin{tikzcd}
A \times B \to C \arrow[rr, bend left, "\texttt{curry}"] & \cong & A \to C^{B} \arrow[ll, bend left, "\texttt{uncurry}"]
\end{tikzcd}$$

Curry | uncurry

$$\begin{tikzcd} f \arrow[rr, mapsto, "\texttt{curry}"] & & k \end{tikzcd}$$

$$\begin{tikzcd} A \times B \to C \arrow[rr, bend left, "\texttt{curry}"] & \cong & A \to C^{B} \arrow[ll, bend left, "\texttt{uncurry}"] \end{tikzcd}$$

Curry | uncurry

$$\begin{tikzcd} f \arrow[r, mapsto, "\texttt{curry}"] & k \end{tikzcd}$$

$$\begin{tikzcd} A \times B \to C \arrow[r, bend left, "\texttt{curry}"] \arrow[r, phantom, "\cong"] & A \to C^{B} \arrow[l, bend left, "\texttt{uncurry}"] \end{tikzcd}$$

k

Curry | uncurry

$$\begin{tikzcd}
f \arrow[r, mapsto, "\texttt{curry}"] & k
\end{tikzcd}$$

$$\begin{tikzcd}
A \times B \to C \arrow[r, bend left, "\texttt{curry}"] \arrow[r, phantom, "\cong"] & A \to C^B \arrow[l, bend left, "\texttt{uncurry}"]
\end{tikzcd}$$

$$\begin{tikzcd}
f & k \arrow[l, mapsto, "\texttt{uncurry}"']
\end{tikzcd}$$

Curry | uncurry

$$k = \texttt{curry}\ f$$

$$\begin{tikzcd} A \times B \to C \arrow[rr, bend left, "\texttt{curry}"] \arrow[rr, phantom, "\cong"] & & A \to C^B \arrow[ll, bend left, "\texttt{uncurry}"] \end{tikzcd}$$

$$\texttt{uncurry}\ k = f$$

Curry | uncurry

$$\begin{tikzcd} A \times B \to C \arrow[r, phantom, "\cong"] \arrow[r, bend left, "\texttt{curry}"] & A \to C^B \arrow[l, bend left, "\texttt{uncurry}"] \end{tikzcd}$$

$$k = \texttt{curry}\ f \iff \texttt{uncurry}\ k = f$$

Curry | uncurry

$$k = \texttt{curry}\ f$$

$$\Leftrightarrow \qquad \{ \text{ isomorphism (last slide) } \}$$

$$\texttt{uncurry}\ k = f$$

Curry | uncurry

$$k = \texttt{curry}\ f$$

$\Leftrightarrow$ { isomorphism (last slide) }

$$\texttt{uncurry}\ k = f$$

$\Leftrightarrow$ { add variables }

$$\texttt{uncurry}\ k\ (a, b) = f\ (a, b)$$

Curry | uncurry

$$(k\ a)\ b = f\ (a, b)$$

$k = \texttt{curry}\ f$

$\Leftrightarrow$ { isomorphism (last slide) }

$\texttt{uncurry}\ k = f$

$\Leftrightarrow$ { add variables }

$\texttt{uncurry}\ k\ (a, b) = f\ (a, b)$

$\Leftrightarrow$ { definition of $\texttt{uncurry}\ k$ }

$k\ a\ b = f\ (a, b)$

Curry | uncurry

$$k = \mathtt{curry}\ f$$

$$\Leftrightarrow \quad \{ \text{ isomorphism (last slide) } \}$$

$$\mathtt{uncurry}\ k = f$$

$$\Leftrightarrow \quad \{ \text{ add variables } \}$$

$$\mathtt{uncurry}\ k\ (a, b) = f\ (a, b)$$

$$\Leftrightarrow \quad \{ \text{ definition of } \mathtt{uncurry}\ k \ \}$$

$$k\ a\ b = f\ (a, b)$$

$$(k\ a)\ b = f\ (a, b)$$

$$\Leftrightarrow \quad \{ \text{ introduce } ap\ (f, x) = f\ x \ \}$$

$$ap\ (k\ a, b) = f\ (a, b)$$

Curry | uncurry

$$k = \texttt{curry}\ f$$

$$\Leftrightarrow \quad \{\ \text{isomorphism (last slide)}\ \}$$

$$\texttt{uncurry}\ k = f$$

$$\Leftrightarrow \quad \{\ \text{add variables}\ \}$$

$$\texttt{uncurry}\ k\ (a, b) = f\ (a, b)$$

$$\Leftrightarrow \quad \{\ \text{definition of } \texttt{uncurry}\ k\ \}$$

$$k\ a\ b = f\ (a, b)$$

$$(k\ a)\ b = f\ (a, b)$$

$$\Leftrightarrow \quad \{\ \text{introduce } ap\ (f, x) = f\ x\ \}$$

$$ap\ (k\ a, b) = f\ (a, b)$$

$$\Leftrightarrow \quad \{\ \text{add } id\ \}$$

$$ap\ (k\ a, id\ b) = f\ (a, b)$$

Curry | uncurry

$$k = \texttt{curry}\ f$$

$$\Leftrightarrow \qquad \{\ \text{isomorphism (last slide)}\ \}$$

$$\texttt{uncurry}\ k = f$$

$$\Leftrightarrow \qquad \{\ \text{add variables}\ \}$$

$$\texttt{uncurry}\ k\ (a, b) = f\ (a, b)$$

$$\Leftrightarrow \qquad \{\ \text{definition of}\ \texttt{uncurry}\ k\ \}$$

$$k\ a\ b = f\ (a, b)$$

$$(k\ a)\ b = f\ (a, b)$$

$$\Leftrightarrow \qquad \{\ \text{introduce}\ ap\ (f, x) = f\ x\ \}$$

$$ap\ (k\ a, b) = f\ (a, b)$$

$$\Leftrightarrow \qquad \{\ \text{add}\ id\ \}$$

$$ap\ (k\ a, id\ b) = f\ (a, b)$$

$$\Leftrightarrow \qquad \{\ \text{composition and}\ \times\ \}$$

$$(ap \cdot (k \times id))\ (a, b) = f\ (a, b)$$

Curry | uncurry

$$k = \texttt{curry}\ f$$

$$\Leftrightarrow \quad \{\ \text{isomorphism (last slide)}\ \}$$

$$\texttt{uncurry}\ k = f$$

$$\Leftrightarrow \quad \{\ \text{add variables}\ \}$$

$$\texttt{uncurry}\ k\ (a, b) = f\ (a, b)$$

$$\Leftrightarrow \quad \{\ \text{definition of } \texttt{uncurry}\ k\ \}$$

$$k\ a\ b = f\ (a, b)$$

$$(k\ a)\ b = f\ (a, b)$$

$$\Leftrightarrow \quad \{\ \text{introduce } ap\ (f, x) = f\ x\ \}$$

$$ap\ (k\ a, b) = f\ (a, b)$$

$$\Leftrightarrow \quad \{\ \text{add } id\ \}$$

$$ap\ (k\ a, id\ b) = f\ (a, b)$$

$$\Leftrightarrow \quad \{\ \text{composition and } \times\ \}$$

$$(ap \cdot (k \times id))\ (a, b) = f\ (a, b)$$

$$\Leftrightarrow \quad \{\ \text{drop variables}\ \}$$

$$ap \cdot (k \times id) = f$$

Universal property

$$\begin{tikzcd}
A \times B \to C \arrow[rr, bend left, "\mathtt{curry}"] \arrow[rr, phantom, "\cong"] & & A \to C^{B} \arrow[ll, bend left, "\mathtt{uncurry}"]
\end{tikzcd}$$

$$k = \mathtt{curry}\ f \quad \Leftrightarrow \quad ap \cdot (k \times id) = f$$

Cálculo
de
Programas

Class T06

Universal property

$$\begin{tikzcd}
A \times B \to C \arrow[rr, bend left, "\overline{(\_)}"] \arrow[rr, phantom, "\cong"] & & A \to C^{B} \arrow[ll, bend left, "\widehat{(\_)}"]
\end{tikzcd}$$

$$k = \overline{f} \;\Leftrightarrow\; ap \cdot (k \times id) = f$$

Diagrams

$$\begin{tikzcd}
 & C \\
A \times B \arrow[ur, "f"'] &
\end{tikzcd}$$

Diagrams

$$\begin{tikzcd}
C^B & & C \\
A \arrow[u, "k = \overline{f}"] & A \times B \arrow[ur, "f"'] &
\end{tikzcd}$$

$\overline{f}$ abbreviates `curry` f

Diagrams

$$\begin{tikzcd}
C^B \\
A \arrow[u, "k = \overline{f}"]
\end{tikzcd}
\qquad
\begin{tikzcd}
C^B & C \\
A \times B \arrow[u, "k"] \arrow[ur, "f"']
\end{tikzcd}$$

$\overline{f}$ abbreviates `curry` f

Diagrams

$$\begin{tikzcd} C^B \\ A \arrow[u, "k = \overline{f}"] \end{tikzcd}$$

$$\begin{tikzcd} C^B & & B & C \\ A \arrow[u, "k"] & \times & B \arrow[u, "id"'] \arrow[ur, "f"'] & \end{tikzcd}$$

$\overline{f}$ abbreviates `curry` f

Universal property

$$\begin{tikzcd}
C^B & C^B \times B & C \\
A \arrow[u, "k = \overline{f}"] & A \times B \arrow[u, "k \times id"] \arrow[ur, "f"'] &
\end{tikzcd}$$

Universal property

$$\begin{tikzcd} C^B & & C^B \times B \arrow[r, "ap"] & C \\ A \arrow[u, "k = \overline{f}"] & & A \times B \arrow[u, "k \times id"] \arrow[ur, "f"'] & \end{tikzcd}$$

$$k = \overline{f} \;\Leftrightarrow\; ap \cdot (k \times id) = f$$

Reflexion $(k := id)$

$$\begin{tikzcd}
C^B \\
C^B \arrow[u, "id = \overline{f}"]
\end{tikzcd}
\qquad
\begin{tikzcd}
C^B \times B \arrow[r, "ap"] & C \\
C^B \times B \arrow[u, "id \times id" description] \arrow[ur, "f"']
\end{tikzcd}$$

$$id = \overline{f} \Leftrightarrow ap = f$$

Fusion

$$\begin{tikzcd}
C^B \\
A \arrow[u, "\overline{f}"]
\end{tikzcd}
\qquad
\begin{tikzcd}
C^B \times B \arrow[r, "ap"] & C \\
A \times B \arrow[u, "\overline{f} \times id"] \arrow[ur, "f"']
\end{tikzcd}$$

Fusion

$$\begin{tikzcd}
C^B \\
A \arrow[u, "\overline{f}"] \\
D \arrow[u, "g"]
\end{tikzcd}
\qquad
\begin{tikzcd}
C^B \times B \arrow[r, "ap"] & C \\
A \times B \arrow[u, "\overline{f} \times id"] \arrow[ur, "f"'] &
\end{tikzcd}$$

Fusion

$$\begin{tikzcd}
C^B & C^B \times B \arrow[r, "ap"] & C \\
A \arrow[u, "\overline{f}"] & A \times B \arrow[u, "\overline{f} \times id"] \arrow[ur, "f"'] & \\
D \arrow[u, "g"] & D \times B \arrow[u, "g \times id"] &
\end{tikzcd}$$

Fusion

$$\begin{tikzcd}
C^B & & C^B \times B \arrow[r, "ap"] & C \\
A \arrow[u, "\overline{f}"] & & A \times B \arrow[u, "\overline{f} \times id"] \arrow[ur, "f"] & \\
D \arrow[u, "g"] & & D \times B \arrow[u, "g \times id"] \arrow[uur, "f \cdot (g \times id)"'] &
\end{tikzcd}$$

Fusion

$$\begin{tikzcd}
C^B \\
\\
D \arrow[uu, "\overline{f} \cdot g"]
\end{tikzcd}$$

$$\begin{tikzcd}
C^B \times B \arrow[r, "ap"] & C \\
A \times B \arrow[u, "\overline{f} \times id"] \arrow[ur, "f"] & \\
D \times B \arrow[u, "g \times id"] \arrow[uur, "f \cdot (g \times id)"'] &
\end{tikzcd}$$

Fusion

$$\begin{tikzcd}
C^B & & C^B \times B \arrow[r, "ap"] & C \\
 & & A \times B \arrow[u, "\overline{f} \times id"] \arrow[ur, "f"] & \\
D \arrow[uu, "\overline{f \cdot (g \times id)}"] & & D \times B \arrow[u, "g \times id"] \arrow[uur, "f \cdot (g \times id)"'] &
\end{tikzcd}$$

$$\overline{f} \cdot g = \overline{f \cdot (g \times id)}$$

Curry | uncurry

$$\begin{tikzcd} f \arrow[r, mapsto, "\texttt{curry}"] & k \end{tikzcd}$$

$$\begin{tikzcd} A \times B \to C \arrow[rr, bend left, "\texttt{curry}"] & \cong & A \to C^{B} \arrow[ll, bend left, "\texttt{uncurry}"] \end{tikzcd}$$

$$\begin{tikzcd} f & k \arrow[l, mapsto, "\texttt{uncurry}"'] \end{tikzcd}$$

Curry | uncurry

$$\begin{tikzcd} f \arrow[rr, mapsto, "\texttt{curry}"] & & k \end{tikzcd}$$

$$\begin{tikzcd} A \times B \to C \arrow[rr, bend left, "\texttt{curry}"] & \cong & A \to C^B \arrow[ll, bend left, "\texttt{uncurry}"] \end{tikzcd}$$

$$\begin{tikzcd} f & & k \arrow[ll, mapsto, "\texttt{uncurry}"'] \end{tikzcd}$$

Abbreviations:

$$\texttt{curry}\ f = \overline{f}$$

Curry | uncurry

$$\begin{tikzcd}
f \arrow[r, mapsto, "\texttt{curry}"] & k
\end{tikzcd}$$

$$\begin{tikzcd}
A \times B \to C \arrow[rr, bend left, "\texttt{curry}"] & \cong & A \to C^B \arrow[ll, bend left, "\texttt{uncurry}"]
\end{tikzcd}$$

$$\begin{tikzcd}
f & k \arrow[l, mapsto, "\texttt{uncurry}"']
\end{tikzcd}$$

Abbreviations:

$$\texttt{curry}\ f = \overline{f}$$

$$\texttt{uncurry}\ f = \widehat{f}$$

Curry | uncurry

$$\begin{tikzcd} f \arrow[rrr, mapsto, "\mathtt{curry}"] & & & k \end{tikzcd}$$

$$\begin{tikzcd} A \times B \to C \arrow[rr, bend left, "\mathtt{curry}"] \arrow[rr, phantom, "\cong"] & & A \to C^B \arrow[ll, bend left, "\mathtt{uncurry}"] \end{tikzcd}$$

$$\begin{tikzcd} f & & & k \arrow[lll, mapsto, "\mathtt{uncurry}"'] \end{tikzcd}$$

Abbreviations:

$$\mathtt{curry}\ f = \overline{f}$$
$$\mathtt{uncurry}\ f = \widehat{f}$$

Isomorphisms:

$$f = g \Leftrightarrow \overline{f} = \overline{g}$$
$$k = h \Leftrightarrow \widehat{k} = \widehat{h}$$

Summary

$f \cdot g$ sequential

Summary

$f \cdot g$ sequential

$\langle f, g \rangle$

Summary

$f \cdot g$	sequential
$\langle f, g \rangle$, $f \times g$	parallel

Summary

$f \cdot g$ sequential

$\langle f, g \rangle$, $f \times g$ parallel

$[f, g]$

Summary

$f \cdot g$ sequential

$\langle f, g \rangle, \; f \times g$ parallel

$[f, g], \; f + g$

Summary

$f \cdot g$	sequential
$\langle f, g \rangle$, $f \times g$	parallel
$[f, g]$, $f + g$, $p \rightarrow f\ ,\ g$	alternation

Summary

$f \cdot g$	sequential
$\langle f, g \rangle$, $f \times g$	parallel
$[f, g]$, $f + g$, $p \rightarrow f\ ,\ g$	alternation
$\overline{f}$	

Summary

$f \cdot g$	sequential
$\langle f, g \rangle$, $f \times g$	parallel
$[f, g]$, $f + g$, $p \rightarrow f\ ,\ g$	alternation
$\overline{f}$	higher order

Summary

$f \cdot g$	sequential
$\langle f, g \rangle$, $f \times g$	parallel
$[f, g]$, $f + g$, $p \to f, g$	alternation
$\overline{f}$	higher order

Compositional programming 🙂

Summary

$A \times B$	records (“structs”)

Summary

$A \times B$	records ("structs")
$A + B$	variant records ("unions")

Summary

$A \times B$	records (“structs”)
$A + B$	variant records (“unions”)
$1 + A$	“pointers”

Summary

$A \times B$	records (“structs”)
$A + B$	variant records (“unions”)
$1 + A$	“pointers”
B^A	“arrays”, “streams”

Summary

$A \times B$	records ("structs")
$A + B$	variant records ("unions")
$1 + A$	"pointers"
B^A	"arrays", "streams"

data structuring ☺

Summary

$A \times B$	records ("structs")
$A + B$	variant records ("unions")
$1 + A$	"pointers"
B^A	"arrays", "streams"
	data structuring 🙂
$1 + A \times X^2$	polynomial structures (coming up soon)

Part II

Where do programs come from?

What is a program?

Where do programs come from?

The algorithms we know were born of what?

Where do programs come from?

The algorithms we know were born of what?

For instance, **quicksort**?

(See video 05b, t=3.25)

Where do programs come from?

$$\begin{cases} a \times 0 = 0 \\ a \times 1 = a \\ a \times (b + c) = a \times b + a \times c \end{cases}$$

Where do programs come from?

$$\begin{cases} a \times 0 = 0 \\ a \times 1 = a \\ a \times (b + 1) = a \times b + a \times 1 \end{cases}$$

Where do programs come from?

$$\begin{cases} a \times 0 = 0 \\ a \times 1 = a \\ a \times (b + 1) = a \times b + a \end{cases}$$

Where do programs come from?

$$\begin{cases} a \times 0 = 0 \\ a \times 1 = a \\ a \times (0 + 1) = a \times 0 + a \end{cases}$$

Where do programs come from?

$$\begin{cases} a \times 0 = 0 \\ a \times 1 = a \\ a \times 1 = a \times 0 + a \end{cases}$$

Where do programs come from?

$$\begin{cases} a \times 0 = 0 \\ a \times 1 = a \\ a \times 1 = 0 + a \end{cases}$$

Where do programs come from?

$$\begin{cases} a \times 0 = 0 \\ a \times 1 = a \\ a \times 1 = a \end{cases}$$

Where do programs come from?

$$\begin{cases} a \times 0 = 0 \\ a \times 1 = a \\ a \times (b + 1) = a \times b + a \end{cases}$$

Where do programs come from?

$$\begin{cases} a \times 0 = 0 \\ a \times (b + 1) = a \times b + a \end{cases}$$

What **is** a program?

$$\begin{cases} a \times 0 = 0 \\ a \times (b + 1) = a + a \times b \end{cases}$$

What is a program?

$$\begin{cases} (a\times)\ 0 = 0 \\ \\ (a\times)\ (b+1) = a + (a\times)\ b \end{cases}$$

What **is** a program?

$$\begin{cases} (a\times)\ 0 = 0 \\ \\ (a\times)\ (b+1) = (a+)\ ((a\times)\ b) \end{cases}$$

What is a program?

$$\begin{cases} (a\times)\ (\underline{0}\ x) = \underline{0}\ x \\ \\ (a\times)\ (b+1) = (a+)\ ((a\times)\ b) \end{cases}$$

What **is** a program?

$$\begin{cases} (a\times)\ (\underline{0}\ x) = \underline{0}\ x \\ \\ (a\times)\ (succ\ b) = (a+)\ ((a\times)\ b) \end{cases}$$

What **is** a program?

$$\begin{cases} ((a\times)\cdot\underline{0})\ x = \underline{0}\ x \\ \\ (a\times)\ (succ\ b) = (a+)\ ((a\times)\ b) \end{cases}$$

What **is** a program?

$$\begin{cases} ((a\times)\cdot\underline{0})\ x = \underline{0}\ x \\[2em] ((a\times)\cdot succ)\ b = (a+)\ ((a\times)\ b) \end{cases}$$

What **is** a program?

$$\begin{cases} ((a\times)\cdot\underline{0})\ x = \underline{0}\ x \\ \\ ((a\times)\cdot \mathit{succ})\ b = ((a+)\cdot(a\times))\ b \end{cases}$$

What **is** a program?

$$\begin{cases} (a\times)\cdot\underline{0} = \underline{0} \\ \\ ((a\times)\cdot succ)\ b = ((a+)\cdot(a\times))\ b \end{cases}$$

What **is** a program?

$$\begin{cases} (a\times)\cdot\underline{0} = \underline{0} \\ \\ (a\times)\cdot \mathit{succ} = (a+)\cdot(a\times) \end{cases}$$

What **is** a program?

$$[(a\times)\cdot\underline{0}, (a\times)\cdot \mathit{succ}] = [\underline{0}, (a+)\cdot(a\times)]$$

What **is** a program?

$$(a\times)\cdot[\underline{0},\mathit{succ}]=[\underline{0},(a+)\cdot(a\times)]$$

What is a program?

$$(a\times)\cdot[\underline{0}, succ] = [\underline{0}\cdot id, (a+)\cdot(a\times)]$$

What **is** a program?

$$(a\times)\cdot[\underline{0}, \mathit{succ}] = [\underline{0}, (a+)]\cdot(\mathit{id} + (a\times))$$

$$(a\times)\cdot[\underline{0}, succ] = [\underline{0}, (a+)]\cdot(id + (a\times))$$

$$\begin{tikzcd}
? \arrow[d, "(a\times)"'] \\
?
\end{tikzcd}$$

$$(a\times) \cdot [\underline{0}, succ] = [\underline{0}, (a+)] \cdot (id + (a\times))$$

$$\begin{tikzcd}
? \arrow[d, "(a\times)"'] & ? \arrow[l, "{[\underline{0}, succ]}"'] \\
?
\end{tikzcd}$$

$$(a\times)\cdot[\underline{0}, succ] = [\underline{0}, (a+)]\cdot(id + (a\times))$$

$$\begin{tikzcd}
? \arrow[d, "(a\times)"'] & ? \arrow[l, "{[\underline{0}, succ]}"'] \arrow[d, "id + (a\times)"] \\
? & ?
\end{tikzcd}$$

$$(a\times) \cdot [\underline{0}, succ] = [\underline{0}, (a+)] \cdot (id + (a\times))$$

$$\begin{tikzcd}
? \arrow[d, "(a\times)"'] & ? \arrow[l, "{[\underline{0}, succ]}"'] \arrow[d, "id + (a\times)"] \\
? & ? \arrow[l, "{[\underline{0}, (a+)]}"]
\end{tikzcd}$$

A program is a “rectangle”

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[d, "(a\times)"'] & ? \arrow[l, "{[\underline{0}, succ]}"'] \arrow[d, "id + (a\times)"] \\
\mathbb{N}_0 & ? \arrow[l, "{[\underline{0}, (a+)]}"]
\end{tikzcd}$$

A program is a “rectangle”

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[d, "(a\times)"'] & ? + ? \arrow[l, "{[\underline{0}, \mathit{succ}]}"'] \arrow[d, "\mathit{id} + (a\times)"] \\
\mathbb{N}_0 & ? + ? \arrow[l, "{[\underline{0}, (a+)]}"]
\end{tikzcd}$$

A program is a “rectangle”

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[d, "(a\times)"'] & ? + \mathbb{N}_0 \arrow[l, "{[\underline{0}, succ]}"'] \arrow[d, "id + (a\times)"] \\
\mathbb{N}_0 & ? + \mathbb{N}_0 \arrow[l, "{[\underline{0}, (a+)]}"]
\end{tikzcd}$$

A program is a “rectangle”

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[d, "(a\times)"'] & A + \mathbb{N}_0 \arrow[l, "{[\underline{0}, succ]}"'] \arrow[d, "id + (a\times)"] \\
\mathbb{N}_0 & A + \mathbb{N}_0 \arrow[l, "{[\underline{0}, (a+)]}"]
\end{tikzcd}$$

Does $[\underline{0}, succ]^\circ$ exist?

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[r, "{[\underline{0}, succ]^\circ}"] \arrow[d, "(a\times)"'] & A + \mathbb{N}_0 \arrow[d, "id + (a\times)"] \\
\mathbb{N}_0 & A + \mathbb{N}_0 \arrow[l, "{[\underline{0}, (a+)]}"]
\end{tikzcd}$$

Does $[\underline{0}, succ]^\circ$ exist?

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[d, "(a\times)"'] & A + \mathbb{N}_0 \arrow[l, "{[\underline{0}, succ]}"'] \arrow[d, "id + (a\times)"] \\
\mathbb{N}_0 & A + \mathbb{N}_0 \arrow[l, "{[\underline{0}, (a+)]}"]
\end{tikzcd}$$

Does $[\underline{0}, succ]^{\circ}$ exist?

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[r, "{[\underline{0}, succ]^{\circ}}"] \arrow[d, "(a\times)"'] & A + \mathbb{N}_0 \arrow[d, "id + (a\times)"] \\
\mathbb{N}_0 & A + \mathbb{N}_0 \arrow[l, "{[\underline{0}, (a+)]}"]
\end{tikzcd}$$

Does $[\underline{0}, succ]^\circ$ exist?

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[r, "{[\underline{0}, succ]^\circ}"] \arrow[d, "(a\times)"'] & A + \mathbb{N}_0 \arrow[d, "id + (a\times)"] \\
\mathbb{N}_0 & A + \mathbb{N}_0 \arrow[l, "{[\underline{0}, (a+)]}"]
\end{tikzcd}$$

$$(a\times) = [\underline{0}, (a+)] \cdot (id + (a\times)) \cdot [\underline{0}, succ]^\circ (?)$$

Does $[\underline{0}, \mathit{succ}]^\circ$ exist?

Conjecture: $\mathbf{out} : \mathbb{N}_0 \to A + \mathbb{N}_0$ exists and is $[\underline{0}, \mathit{succ}]^\circ$

$$\mathbf{out} \cdot [\underline{0}, \mathit{succ}] = \mathit{id}$$

Does $[\underline{0}, \mathit{succ}]^\circ$ exist?

Conjecture: $\mathbf{out} : \mathbb{N}_0 \to A + \mathbb{N}_0$ exists and is $[\underline{0}, \mathit{succ}]^\circ$

$$\begin{aligned} & \mathbf{out} \cdot [\underline{0}, \mathit{succ}] = \mathit{id} \\ \Leftrightarrow \quad & \{ \ \text{+-fusion} \ \} \\ & [\mathbf{out} \cdot \underline{0}, \mathbf{out} \cdot \mathit{succ}] = \mathit{id} \end{aligned}$$

Does $[\underline{0}, succ]^\circ$ exist?

Conjecture: $\mathbf{out} : \mathbb{N}_0 \rightarrow A + \mathbb{N}_0$ exists and is $[\underline{0}, succ]^\circ$

$$\begin{array}{ll}
& \mathbf{out} \cdot [\underline{0}, succ] = id \\
\Leftrightarrow & \qquad \{ \ +\text{-fusion} \ \} \\
& [\mathbf{out} \cdot \underline{0}, \mathbf{out} \cdot succ] = id \\
\Leftrightarrow & \qquad \{ \ \text{universal-}+ \ \} \\
& \left\{ \begin{array}{l} \mathbf{out} \cdot \underline{0} = i_1 \\ \mathbf{out} \cdot succ = i_2 \end{array} \right.
\end{array}$$

Does $[\underline{0}, succ]^{\circ}$ exist?

$$\begin{cases} \mathbf{out} \cdot \underline{0} = i_1 \\ \mathbf{out} \cdot succ = i_2 \end{cases}$$

Does $[\underline{0}, succ]^\circ$ exist?

$$\begin{cases} \mathbf{out} \cdot \underline{0} = i_1 \\ \mathbf{out} \cdot succ = i_2 \end{cases}$$

$$\Leftrightarrow \qquad \{ \text{ introduce variables } \}$$

$$\begin{cases} \mathbf{out}\ (\underline{0}\ x) = i_1\ x \\ \mathbf{out}\ (succ\ x) = i_2\ x \end{cases}$$

Does $[\underline{0}, succ]^\circ$ exist?

$$\left\{\begin{array}{l} \mathbf{out} \cdot \underline{0} = i_1 \\ \mathbf{out} \cdot succ = i_2 \end{array}\right.$$

$$\Leftrightarrow \qquad \{ \text{ introduce variables } \}$$

$$\left\{\begin{array}{l} \mathbf{out}\ (\underline{0}\ x) = i_1\ x \\ \mathbf{out}\ (succ\ x) = i_2\ x \end{array}\right.$$

$$\Leftrightarrow \qquad \{ \text{ simplificar } \}$$

$$\left\{\begin{array}{l} \mathbf{out}\ 0 = i_1\ x \\ \mathbf{out}\ (x + 1) = i_2\ x \end{array}\right.$$

Does $[\underline{0}, \mathit{succ}]^{\circ}$ exist?

$$\begin{cases} \mathbf{out} \cdot \underline{0} = i_1 \\ \mathbf{out} \cdot \mathit{succ} = i_2 \end{cases}$$

$$\Leftrightarrow \qquad \{ \text{ introduce variables } \}$$

$$\begin{cases} \mathbf{out}\ (\underline{0}\ x) = i_1\ x \\ \mathbf{out}\ (\mathit{succ}\ x) = i_2\ x \end{cases}$$

$$\Leftrightarrow \qquad \{ \text{ simplificar } \}$$

$$\begin{cases} \mathbf{out}\ 0 = i_1\ x \\ \mathbf{out}\ (x + 1) = i_2\ x \end{cases}$$

$$\begin{cases} \mathbf{out}\ 0 = i_1\ x \\ \mathbf{out}\ (x + 1) = i_2\ x \end{cases}$$

?

Problem

$$\left\{\begin{array}{l}\mathbf{out}\ 0 = i_1\ x \\ \mathbf{out}\ (x+1) = i_2\ x\end{array}\right.$$

$$\begin{tikzcd}[column sep=huge] \mathbb{N}_0 \arrow[r, bend left] \arrow[r, phantom, "?"] & A + \mathbb{N}_0 \arrow[l, bend left, "\mathbf{in} = {[\underline{0}, succ]}"] \end{tikzcd}$$

?

Solution

$$\begin{cases} \mathbf{out}\ 0 = i_1\ () \\ \mathbf{out}\ (x+1) = i_2\ x \end{cases}$$

🙂

$$\begin{tikzcd} \mathbb{N}_0 \arrow[r, bend left] \arrow[r, phantom, "\cong"] & 1 + \mathbb{N}_0 \arrow[l, bend left, "\mathbf{in}{=}{[\underline{0}, succ]}"] \end{tikzcd}$$

Solution

$$\begin{cases} \textbf{out}\ 0 = i_1\ () \\ \textbf{out}\ (x+1) = i_2\ x \end{cases}$$

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[r, bend left] \arrow[r, phantom, "\cong"] \arrow[d, "(a\times)"'] & 1+\mathbb{N}_0 \arrow[l, bend left, "{\textbf{in}=[\underline{0}, succ]}"] \arrow[d, "id+(a\times)"] \\
\mathbb{N}_0 & 1+\mathbb{N}_0 \arrow[l, bend left, "{[\underline{0},(a+)]}"]
\end{tikzcd}$$

Diagram

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[r, bend left, "\mathbf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "(a\times)"'] & 1+\mathbb{N}_0 \arrow[l, bend left, "\mathbf{in}"] \arrow[d, "id+(a\times)"] \\
\mathbb{N}_0 & 1+\mathbb{N}_0 \arrow[l, bend left, "{[\underline{0}, a+]}"]
\end{tikzcd}$$

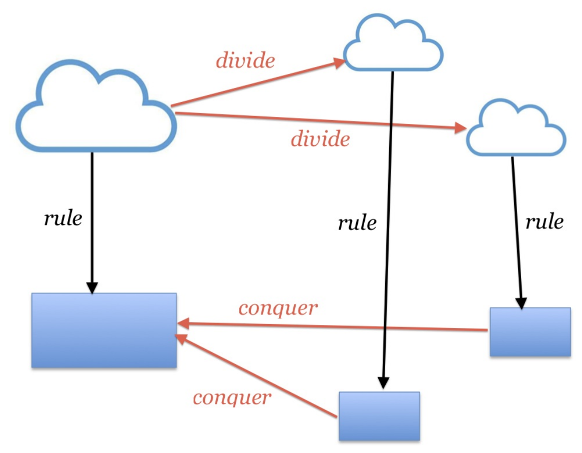
$$(a\times) = [\underline{0}, (a+)] \cdot (id + (a\times)) \cdot \mathbf{out}$$

Renaming

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[r, bend left, "\mathbf{out}"] \arrow[d, "k"'] \arrow[r, phantom, "\cong"] & 1 + \mathbb{N}_0 \arrow[l, bend left, "\mathbf{in}"] \arrow[d, "id+k"] \\
\mathbb{N}_0 & 1 + \mathbb{N}_0 \arrow[l, bend left, "{[\underline{0}, a+]}"]
\end{tikzcd}$$

$$(a\times) = k \textbf{ where } k = [\underline{0}, (a+)] \cdot (id + k) \cdot \mathbf{out}$$

Anticipation of what is to come...



Variations in diagram

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[r, bend left, "\mathsf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "k"'] & 1 + \mathbb{N}_0 \arrow[l, bend left, "\mathsf{in}"] \arrow[d, "id+k"] \\
\mathbb{N}_0 & 1 + \mathbb{N}_0 \arrow[l, bend left, "{[\underline{1},(a\times)]}"]
\end{tikzcd}$$

Variations in diagram

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[r, bend left, "\mathbf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "k"'] & 1 + \mathbb{N}_0 \arrow[l, bend left, "\mathbf{in}"] \arrow[d, "id+k"] \\
\mathbb{N}_0 & 1 + \mathbb{N}_0 \arrow[l, bend left, "{[\underline{1},(a\times)]}"]
\end{tikzcd}$$

$$a^b = k \textbf{ where } k = [\underline{1}, (a\times)] \cdot (id + k) \cdot \mathbf{out}$$

Generalizing the diagram

$$\begin{cases} a^{0} = 1 \\ a^{b+1} = a \times a^{b} \end{cases}$$

Generalizing the diagram

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[r, bend left, "\mathsf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "k\ g"'] & 1+\mathbb{N}_0 \arrow[l, bend left, "\mathsf{in}"] \arrow[d, "id+k\ g"] \\
A & 1+A \arrow[l, bend left, "g"]
\end{tikzcd}$$

$$k\ g = g \cdot (id + k\ g) \cdot \mathsf{out}$$

Generalizing the diagram

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[r, bend left, "\mathbf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "(\!| g |\!)"'] & 1+\mathbb{N}_0 \arrow[l, bend left, "\mathbf{in}"] \arrow[d, "id+(\!| g |\!)"] \\
A & 1+A \arrow[l, bend left, "g"]
\end{tikzcd}$$

$$(\!| g |\!) = g \cdot (id + (\!| g |\!)) \cdot \mathbf{out}$$

Catamorphism

$$(\!|\, g \,|\!)$$

$$\underbrace{\mathit{cata}}_{\mathit{downwards}} \ (\kappa\alpha\tau\alpha)$$

Catamorphism

$$(\!| g |\!)$$

$$\underbrace{\textit{cata}}_{\textit{downwards}} (\kappa\alpha\tau\alpha) + \underbrace{\textit{morphism}}_{\text{transformation}} \quad (\mu o \rho \phi \iota \sigma \mu o \zeta)$$

“Downwards transformation”

Definition

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[r, "\mathbf{out}"] \arrow[d, "(\!| g |\!)"'] & 1 + \mathbb{N}_0 \arrow[d, "id + (\!| g |\!)"] \\
A & 1 + A \arrow[l, "g"]
\end{tikzcd}$$

$$(\!| g |\!) = g \cdot (id + (\!| g |\!)) \cdot \mathbf{out}$$

Property

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[d, "\llparenthesis g \rrparenthesis"'] & 1 + \mathbb{N}_0 \arrow[l, "\mathbf{in}"'] \arrow[d, "id + \llparenthesis g \rrparenthesis"] \\
A & 1 + A \arrow[l, "g"]
\end{tikzcd}$$

$$\llparenthesis g \rrparenthesis \cdot \mathbf{in} = g \cdot (id + \llparenthesis g \rrparenthesis)$$

Property

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[d, "\llparenthesis g \rrparenthesis"'] & 1 + \mathbb{N}_0 \arrow[l, "\mathbf{in}"'] \arrow[d, "id + \llparenthesis g \rrparenthesis"] \\
A & 1 + A \arrow[l, "g"]
\end{tikzcd}$$

$$k = \llparenthesis g \rrparenthesis \Rightarrow k \cdot \mathbf{in} = g \cdot (id + k)$$

Uniqueness

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[d, "(\!| g |\!)"'] & 1 + \mathbb{N}_0 \arrow[l, "\mathbf{in}"'] \arrow[d, "id + (\!| g |\!)"] \\
A & 1 + A \arrow[l, "g"]
\end{tikzcd}$$

$$k = (\!| g |\!) \Leftarrow k \cdot \mathbf{in} = g \cdot (id + k)$$

Universal property

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[d, swap, "k"] & 1 + \mathbb{N}_0 \arrow[l, swap, "\mathbf{in}"] \arrow[d, "id + k"] \\
A & 1 + A \arrow[l, "g"]
\end{tikzcd}$$

$$k = (\!| g |\!) \quad \Leftrightarrow \quad k \cdot \mathbf{in} = g \cdot (id + k)$$

Universal-$\times$:

$$k = \langle f , g \rangle \;\Leftrightarrow\; \left\{ \begin{array}{l} \pi_1 \cdot k = f \\ \pi_2 \cdot k = g \end{array} \right.$$

Universal-$\times$:

$$k = \langle f, g \rangle \;\Leftrightarrow\; \begin{cases} \pi_1 \cdot k = f \\ \pi_2 \cdot k = g \end{cases}$$

Universal-$+$:

$$k = [f, g] \;\Leftrightarrow\; \begin{cases} k \cdot i_1 = f \\ k \cdot i_2 = g \end{cases}$$

Universal-$\times$:

$$k = \langle f, g \rangle \;\Leftrightarrow\; \begin{cases} \pi_1 \cdot k = f \\ \pi_2 \cdot k = g \end{cases}$$

Universal-$+$:

$$k = [f, g] \;\Leftrightarrow\; \begin{cases} k \cdot i_1 = f \\ k \cdot i_2 = g \end{cases}$$

Universal-expo:

$$k = \overline{f} \;\Leftrightarrow\; ap \cdot (k \times id) = f$$

Universal-$\times$:

$$k = \langle f, g \rangle \;\Leftrightarrow\; \begin{cases} \pi_1 \cdot k = f \\ \pi_2 \cdot k = g \end{cases}$$

Universal-$+$:

$$k = [f, g] \;\Leftrightarrow\; \begin{cases} k \cdot i_1 = f \\ k \cdot i_2 = g \end{cases}$$

Universal-expo:

$$k = \overline{f} \;\Leftrightarrow\; ap \cdot (k \times id) = f$$

Cata-Universal:

$$k = (\!| g |\!) \;\Leftrightarrow\; k \cdot \mathbf{in} = g \cdot (id + k)$$

Cálculo
de
Programas

Class T07

Cata-universal ($\mathbb{N}_0$)

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[d, "k"'] & 1 + \mathbb{N}_0 \arrow[l, "\mathsf{in}"'] \arrow[d, "id+k"] \\
A & 1 + A \arrow[l, "g"]
\end{tikzcd}$$

$$k = (\!| g |\!) \Leftrightarrow k \cdot \mathsf{in} = g \cdot (id + k)$$

Cata-reflexion

To be derived from

$$k = (\!| g |\!) \Leftrightarrow k \cdot \mathbf{in} = g \cdot (id + k)$$

Cata-reflexion ($k := id$)

$$id = (\!| g |\!) \Leftrightarrow id \cdot \mathbf{in} = g \cdot (id + id)$$

Cata-reflexion ($k := id$)

$$id = (\!|\, g \,|\!) \Leftrightarrow \mathbf{in} = g$$

Cata-reflexion

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[d, "id=(\!|\mathbf{in}|\!)"'] & 1+\mathbb{N}_0 \arrow[l, "\mathbf{in}=[\underline{0},succ]"'] \arrow[d, "id+(\!|\mathbf{in}|\!)"] \\
\mathbb{N}_0 & 1+\mathbb{N}_0 \arrow[l, "\mathbf{in}=[\underline{0},succ]"]
\end{tikzcd}$$

$$(\!|\mathbf{in}|\!) = id$$

Cata-cancellation

Another corollary of

$$k = (\!|\, g \,|\!) \Leftrightarrow k \cdot \mathbf{in} = g \cdot (id + k)$$

for $k := (\!|\, g \,|\!)$.

Cata-cancellation $(k := (\!| g |\!))$

$$(\!| g |\!) = (\!| g |\!) \Leftrightarrow (\!| g |\!) \cdot \mathbf{in} = g \cdot (id + (\!| g |\!))$$

Cata-cancellation

$$(\!|\, g \,|\!) \cdot \mathbf{in} = g \cdot (\mathit{id} + (\!|\, g \,|\!))$$

Cata-definition

$$(\!| g |\!) = g \cdot (id + (\!| g |\!)) \cdot \mathbf{out}$$

Cata-definition

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[r, "\mathbf{out}"] \arrow[d, "(\!| g |\!)"'] & 1 + \mathbb{N}_0 \arrow[d, "id + (\!| g |\!)"] \\
A & 1 + A \arrow[l, "g"]
\end{tikzcd}$$

$$(\!| g |\!) = g \cdot (id + (\!| g |\!)) \cdot \mathbf{out}$$

Cata-fusion

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[d, "\llparenthesis g \rrparenthesis"'] & & 1 + \mathbb{N}_0 \arrow[ll, "\mathbf{in} = {[\underline{0}, succ]}"'] \arrow[d, "id + \llparenthesis g \rrparenthesis"] \\
A \arrow[d, "f"'] & & 1 + A \arrow[ll, "g"] \arrow[d, "id + f"] \\
B & & 1 + B \arrow[ll, "h"]
\end{tikzcd}$$

Cata-fusion

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[d, "\llparenthesis\, g \,\rrparenthesis"'] \arrow[dd, bend right=50, "\llparenthesis\, h \,\rrparenthesis"'] & 1 + \mathbb{N}_0 \arrow[l, "\mathbf{in} = {[\underline{0}, succ]}"'] \arrow[d, "id + \llparenthesis\, g \,\rrparenthesis"] \\
A \arrow[d, "f"'] & 1 + A \arrow[l, "g"] \arrow[d, "id + f"] \\
B & 1 + B \arrow[l, "h"]
\end{tikzcd}$$

Cata-fusion

$$f \cdot (\!|\, g \,|\!) = (\!|\, h \,|\!)$$

Cata-fusion

$$
\begin{aligned}
& f \cdot (\!| g |\!) = (\!| h |\!) \\
\Leftrightarrow \quad & \{ \text{ Cata-universal for } k = f \cdot (\!| g |\!) \ \} \\
& f \cdot (\!| g |\!) \cdot \mathbf{in} = h \cdot (id + f \cdot (\!| g |\!))
\end{aligned}
$$

Cata-fusion

$$
\begin{array}{ll}
 & f \cdot (\!| g |\!) = (\!| h |\!) \\
\Leftrightarrow & \qquad \{ \text{ Cata-universal for } k = f \cdot (\!| g |\!) \ \} \\
 & f \cdot (\!| g |\!) \cdot \mathbf{in} = h \cdot (id + f \cdot (\!| g |\!)) \\
\Leftrightarrow & \qquad \{ \text{ Cata-cancellation } \} \\
 & f \cdot g \cdot (id + (\!| g |\!)) = h \cdot (id + f \cdot (\!| g |\!))
\end{array}
$$

Cata-fusion

$$f \cdot g \cdot (id + \llparenthesis\, g \,\rrparenthesis) = h \cdot (id \cdot id + f \cdot \llparenthesis\, g \,\rrparenthesis)$$

Cata-fusion

$$f \cdot g \cdot (id + (\!| g |\!)) = h \cdot (id \cdot id + f \cdot (\!| g |\!))$$

$$\Leftrightarrow \quad \{ \text{ functor-}+ \}$$

$$f \cdot g \cdot (id + (\!| g |\!)) = h \cdot (id + f) \cdot (id + (\!| g |\!))$$

Cata-fusion

$$f \cdot g \cdot (id + (\!| g |\!)) = h \cdot (id \cdot id + f \cdot (\!| g |\!))$$

$\Leftrightarrow$ { functor-$+$ }

$$f \cdot g \cdot (id + (\!| g |\!)) = h \cdot (id + f) \cdot (id + (\!| g |\!))$$

$\Leftarrow$ { Leibniz }

$$f \cdot g = h \cdot (id + f)$$

Cata-fusion

$$f \cdot g \cdot (id + (\!| g |\!)) = h \cdot (id \cdot id + f \cdot (\!| g |\!))$$

$$\Leftrightarrow \qquad \{ \text{ functor-}+ \}$$

$$f \cdot g \cdot (id + (\!| g |\!)) = h \cdot (id + f) \cdot (id + (\!| g |\!))$$

$$\Leftarrow \qquad \{ \text{ Leibniz } \}$$

$$f \cdot g = h \cdot (id + f)$$

Summing up:

$$f \cdot (\!| g |\!) = (\!| h |\!) \quad \Leftarrow \quad f \cdot g = h \cdot (id + f)$$

Cata-fusion

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[d, "\llparenthesis g \rrparenthesis"'] \arrow[dd, bend right=40, "\llparenthesis h \rrparenthesis"'] & & 1 + \mathbb{N}_0 \arrow[ll, "\mathbf{in}=[\underline{0}, succ]"'] \arrow[d, "id+\llparenthesis g \rrparenthesis"] \\
A \arrow[d, "f"'] & & 1 + A \arrow[ll, "g"] \arrow[d, "id+f"] \\
B & & 1 + B \arrow[ll, "h"]
\end{tikzcd}$$

What **is** a program?

What **is** a program?

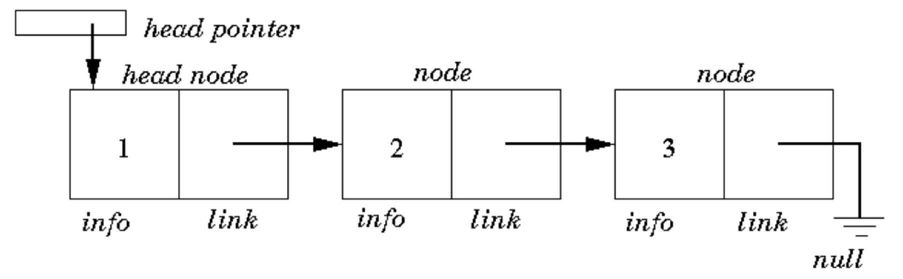
Programs are “rectangles” 🙂

What is a program?

Programs are “rectangles” 🙂

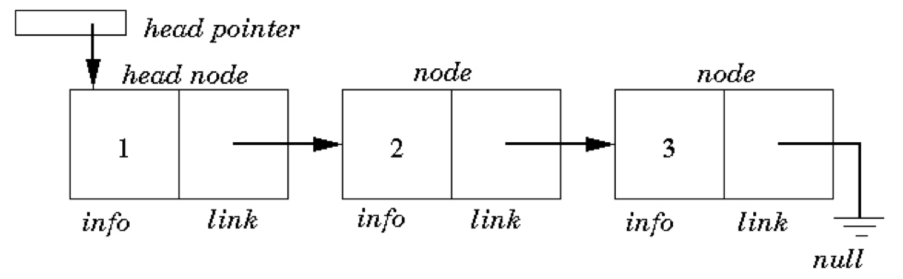
But there is a long way to the general case...

Linked list



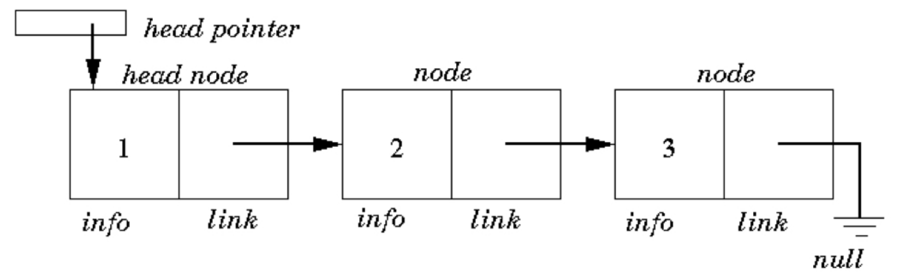
head pointer
head node
node
node
1
2
3
info
link
info
link
info
link
null

Linked list


head pointer
head node
node
node
1
2
3
info
link
info
link
info
link
null


A
B
$A \times B$

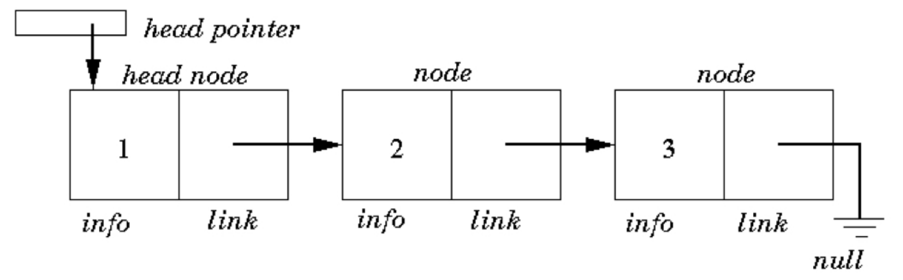
Linked list



$A \times B$

$1 + A$

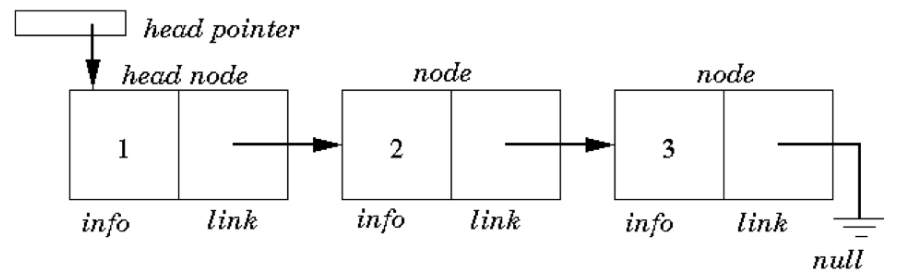
Linked list



"pointer"

$L = 1 + N$

Linked list



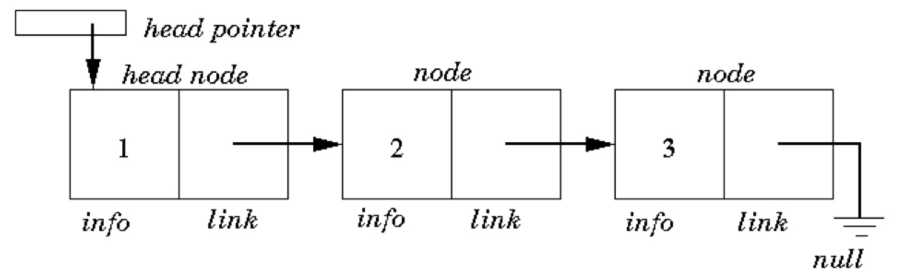
“pointer”

$$L = 1 + N$$

“node”

$$N = \mathbb{N}_0 \times L$$

Linked list

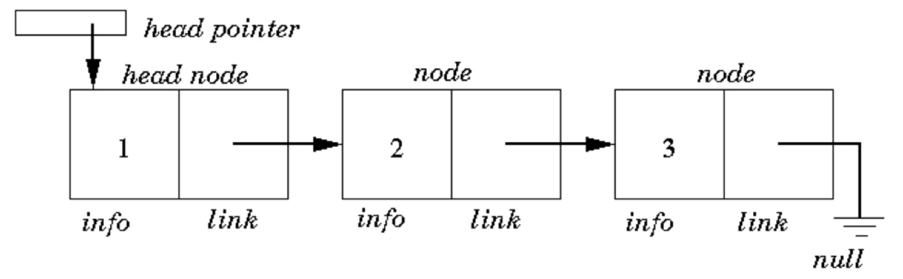


Substitution

$$\begin{cases} L = 1 + N \\ N = \mathbb{N}_0 \times L \end{cases}$$

$$L = 1 + \mathbb{N}_0 \times L$$

Linked list



In fact

$$L \cong 1 + \mathbb{N}_0 \times L$$

Linked list

$$L \cong 1 + \mathbb{N}_0 \times L$$

Linked list

$$L \cong 1 + \mathbb{N}_0 \times L$$

$$\begin{tikzcd} L \arrow[r, bend left, "\textbf{out}"] \arrow[r, phantom, "\cong"] & 1 + \mathbb{N}_0 \times L \arrow[l, bend left, "\textbf{in}"] \end{tikzcd}$$

Linked list

$$L \cong 1 + \mathbb{N}_0 \times L$$

$$\begin{tikzcd} L \arrow[r, bend left, "\mathbf{out}"] \arrow[r, phantom, "\cong"] & 1 + \mathbb{N}_0 \times L \arrow[l, bend left, "\mathbf{in}"] \end{tikzcd}$$

$$\mathbf{in} = [\mathit{nil}, \mathit{cons}]$$

Linked list

$$L \;\cong\; 1 + \mathbb{N}_0 \times L$$

$$\begin{tikzcd}
L \arrow[r, bend left, "\mathbf{out}"] \arrow[r, phantom, "\cong"] & 1 + \mathbb{N}_0 \times L \arrow[l, bend left, "\mathbf{in}"]
\end{tikzcd}$$

$$\mathbf{in} = [\mathit{nil}, \mathit{cons}] \qquad \begin{cases} \mathit{nil}\ \_ = [\,] \\ \mathit{cons}\ (a, l) = a : l \end{cases}$$

Linked list

$$L \cong 1 + \mathbb{N}_0 \times L$$

$$\begin{tikzcd} L \arrow[r, bend left, "\mathbf{out}"] \arrow[r, phantom, "\cong"] & 1 + \mathbb{N}_0 \times L \arrow[l, bend left, "\mathbf{in}"] \end{tikzcd}$$

$$\mathbf{in} = [nil, cons] \qquad \begin{cases} nil\ \_ = [\,] \\ cons\ (a, l) = a : l \end{cases}$$

$$\mathbf{out} \cdot \mathbf{in} = id$$

Linked list

$$\begin{tikzcd} L \arrow[r, bend left, "\mathbf{out}"] \arrow[r, phantom, "\cong"] & 1 + \mathbb{N}_0 \times L \arrow[l, bend left, "\mathbf{in}"] \end{tikzcd}$$

$$\begin{cases} \mathbf{out} \cdot \mathbf{in} = id \\ \mathbf{in} = [nil, cons] \end{cases}$$

Linked list

$$\begin{tikzcd}
L \arrow[rr, bend left, "\mathbf{out}"] \arrow[rr, phantom, "\cong"] & & 1 + \mathbb{N}_0 \times L \arrow[ll, bend left, "\mathbf{in}"]
\end{tikzcd}$$

$$\left\{ \begin{array}{l} \mathbf{out} \cdot \mathbf{in} = id \\ \mathbf{in} = [nil, cons] \end{array} \right. \Rightarrow$$

Linked list

$$\begin{tikzcd} L \arrow[r, bend left, "\mathbf{out}"] \arrow[r, phantom, "\cong"] & 1 + \mathbb{N}_0 \times L \arrow[l, bend left, "\mathbf{in}"] \end{tikzcd}$$

$$\left\{\begin{array}{l} \mathbf{out} \cdot \mathbf{in} = id \\ \mathbf{in} = [nil, cons] \end{array}\right. \Rightarrow \left\{\begin{array}{l} \mathbf{out}\ [\,] = i_1\ () \\ \mathbf{out}\ (a : x) = i_2\ (a, x) \end{array}\right.$$

Length of a list

Recall list **concatenation** $x +\!\!+\, y$

Length of a list

Recall list **concatenation** $x \mathbin{+\!\!+} y$ such that

$$[\,] \mathbin{+\!\!+} x = x$$

Length of a list

Recall list **concatenation** $x +\!\!+ y$ such that

$$[\,] +\!\!+ x = x$$
$$[a] +\!\!+ x = a : x$$

Length of a list

Recall list **concatenation** $x +\!\!+\, y$ such that

$$[\,] +\!\!+\, x = x$$
$$[a] +\!\!+\, x = a : x$$

Properties:

Length of a list

Recall list **concatenation** $x +\!\!+\, y$ such that

$$[\,] +\!\!+\, x = x$$

$$[a] +\!\!+\, x = a : x$$

Properties:

$$\mathit{length}\ [\,] = 0$$

Length of a list

Recall list **concatenation** $x \mathbin{+\!\!+} y$ such that

$$[\,] \mathbin{+\!\!+} x = x$$
$$[a] \mathbin{+\!\!+} x = a : x$$

Properties:

$$length\ [\,] = 0$$
$$length\ [a] = 1$$

Length of a list

Recall list **concatenation** $x +\!\!+\, y$ such that

$$[\,] +\!\!+\, x = x$$
$$[a] +\!\!+\, x = a : x$$

Properties:

$$\mathit{length}\ [\,] = 0$$
$$\mathit{length}\ [a] = 1$$
$$\mathit{length}\ (x +\!\!+\, y) = \mathit{length}\ x + \mathit{length}\ y$$

Length of a list

$$length\ [\,] = 0$$

$$length\ [a] = 1$$

$$length\ ([a] \mathbin{+\!\!+} y) = length\ [a] + length\ y$$

Length of a list

$$\mathit{length}\ [\,] = 0$$

$$\mathit{length}\ [a] = 1$$

$$\mathit{length}\ ([a] +\!\!+\ y) = \mathit{length}\ [a] + \mathit{length}\ y$$

$$\mathit{length}\ (a : y) = 1 + \mathit{length}\ y$$

Length of a list

$$\mathit{length}\ [\,] = 0$$
$$\mathit{length}\ (a : y) = 1 + \mathit{length}\ y$$

Length of a list

$$\mathit{length}\ [\,] = 0$$
$$\mathit{length}\ (a : y) = 1 + \mathit{length}\ y$$

$$(\mathit{length} \cdot \mathit{nil})\ \_ = \underline{0}\ \_$$
$$(\mathit{length} \cdot \mathit{cons})\ (a, y) = 1 + \mathit{length}\ (\pi_2\ (a, y))$$

Length of a list

$$\mathit{length}\ [\,] = 0$$
$$\mathit{length}\ (a : y) = 1 + \mathit{length}\ y$$

$$(\mathit{length} \cdot \mathit{nil})\ \_ = \underline{0}\ \_$$
$$(\mathit{length} \cdot \mathit{cons})\ (a, y) = 1 + \mathit{length}\ (\pi_2\ (a, y))$$

$$\mathit{length} \cdot \mathit{nil} = \underline{0}$$
$$\mathit{length} \cdot \mathit{cons} = \mathit{succ} \cdot \mathit{length} \cdot \pi_2$$

Length of a list

$$\mathit{length} \cdot \mathit{nil} = \underline{0}$$
$$\mathit{length} \cdot \mathit{cons} = \mathit{succ} \cdot \mathit{length} \cdot \pi_2$$

Length of a list

$$length \cdot nil = \underline{0}$$
$$length \cdot cons = succ \cdot length \cdot \pi_2$$

$$length \cdot [nil, cons] = [\underline{0}, succ \cdot \pi_2] \cdot (id + id \times length)$$

Length of a list

$$length \cdot nil = \underline{0}$$

$$length \cdot cons = succ \cdot length \cdot \pi_2$$

$$length \cdot [nil\,, cons] = [\underline{0}, succ \cdot \pi_2] \cdot (id + id \times length)$$

$$\begin{tikzcd}
L \arrow[r, bend left, "\mathbf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "length"'] & 1 + \mathbb{N}_0 \times L \arrow[l, bend left, "\mathbf{in}"] \arrow[d, "id+id\times length"] \\
\mathbb{N}_0 & 1 + \mathbb{N}_0 \times \mathbb{N}_0 \arrow[l, "{[\underline{0}, succ\cdot\pi_2]}"]
\end{tikzcd}$$

$$\begin{cases} \mathbf{in} = [nil\,, cons] \\ \mathbf{out} = \mathbf{in}^{\circ} \end{cases}$$

List sum

$$\begin{tikzcd}
L \arrow[r, bend left, "\mathbf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "sum"'] & 1 + \mathbb{N}_0 \times L \arrow[l, bend left, "\mathbf{in}"] \arrow[d, "id + id \times sum"] \\
\mathbb{N}_0 & 1 + \mathbb{N}_0 \times \mathbb{N}_0 \arrow[l, bend left, "{[\underline{0}, add]}"]
\end{tikzcd}$$

$$\left\{ \begin{array}{l} \mathbf{in} = [nil, cons] \\ \mathbf{out} = \mathbf{in}^\circ \end{array} \right.$$

$$sum \cdot \mathbf{in} = [\underline{0}, add] \cdot (id + id \times sum)$$

Generalizing

$$\begin{tikzcd}
L \arrow[r, bend left, "\textbf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "k"'] & 1 + \mathbb{N}_0 \times L \arrow[l, bend left, "\textbf{in}"] \arrow[d, "id+id\times k"] \\
B & 1 + \mathbb{N}_0 \times B \arrow[l, bend left, "g"]
\end{tikzcd}$$

Generalizing

$$\begin{tikzcd}
L \arrow[r, bend left, "\mathsf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "k"'] & 1 + \mathbb{N}_0 \times L \arrow[l, bend left, "\mathsf{in}"] \arrow[d, "id+id\times k"] \\
B & 1 + \mathbb{N}_0 \times B \arrow[l, bend left, "g"]
\end{tikzcd}$$

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[r, bend left, "\mathsf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "k"'] & 1 + \mathbb{N}_0 \arrow[l, bend left, "\mathsf{in}"] \arrow[d, "id+k"] \\
B & 1 + B \arrow[l, bend left, "g"]
\end{tikzcd}$$

Generalizing

$$\begin{tikzcd}
L \arrow[r, bend left, "\mathbf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "k"'] & 1 + \mathbb{N}_0 \times L \arrow[l, bend left, "\mathbf{in}"] \arrow[d, "id+id\times k"] \\
B & 1 + \mathbb{N}_0 \times B \arrow[l, bend left, "g"]
\end{tikzcd}$$

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[r, bend left, "\mathbf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "k"'] & 1 + \mathbb{N}_0 \arrow[l, bend left, "\mathbf{in}"] \arrow[d, "id+k"] \\
B & 1 + B \arrow[l, bend left, "g"]
\end{tikzcd}$$

$$k \cdot \mathbf{in} = g \cdot \underbrace{id + id \times k}_{\mathbf{F}\ k}$$

Generalizing

$$\begin{tikzcd}
L \arrow[r, bend left, "\mathbf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "k"'] & 1+\mathbb{N}_0\times L \arrow[l, bend left, "\mathbf{in}"] \arrow[d, "id+id\times k"] \\
B & 1+\mathbb{N}_0\times B \arrow[l, bend left, "g"]
\end{tikzcd}$$

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[r, bend left, "\mathbf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "k"'] & 1+\mathbb{N}_0 \arrow[l, bend left, "\mathbf{in}"] \arrow[d, "id+k"] \\
B & 1+B \arrow[l, bend left, "g"]
\end{tikzcd}$$

$$k \cdot \mathbf{in} = g \cdot \underbrace{id + id \times k}_{\mathbf{F}\ k}$$

$$k \cdot \mathbf{in} = g \cdot \underbrace{id + k}_{\mathbf{F}\ k}$$

Abstraction ($\mathbb{N}_0$)

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[r, bend left, "\mathsf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "k"'] & 1 + \mathbb{N}_0 \arrow[l, bend left, "\mathsf{in}"] \arrow[d, "id+k"] \\
B & 1 + B \arrow[l, bend left, "g"]
\end{tikzcd}$$

Abstraction ($\mathbb{N}_0$)

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[rr, bend left, "\mathsf{out}"] \arrow[rr, phantom, "\cong"] \arrow[d, "k"'] & & 1 + \mathbb{N}_0 \arrow[ll, bend left, "\mathsf{in}"] \arrow[d, "id+k"] \\
B & & 1 + B \arrow[ll, bend left, "g"]
\end{tikzcd}$$

$$\begin{tikzcd}
\mathbb{N}_0 \arrow[rr, bend left, "\mathsf{out}"] \arrow[rr, phantom, "\cong"] \arrow[d, "k"'] & & \mathsf{F}\ \mathbb{N}_0 \arrow[ll, bend left, "\mathsf{in}"] \arrow[d, "\mathsf{F}\ k"] \\
B & & \mathsf{F}\ B \arrow[ll, bend left, "g"]
\end{tikzcd}$$

$$\begin{cases} \mathsf{F}\ k = id + k \\ \mathsf{F}\ X = 1 + X \end{cases}$$

Abstraction (lists)

$$\begin{tikzcd}
L \arrow[r, bend left, "\mathsf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "k"'] & 1 + \mathbb{N}_0 \times L \arrow[l, bend left, "\mathsf{in}"] \arrow[d, "id + id \times k"] \\
B & 1 + \mathbb{N}_0 \times B \arrow[l, bend left, "g"]
\end{tikzcd}$$

Abstraction (lists)

$$\begin{tikzcd}
L \arrow[r, bend left, "\mathsf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "k"'] & 1+\mathbb{N}_0\times L \arrow[l, bend left, "\mathsf{in}"] \arrow[d, "id+id\times k"] \\
B & 1+\mathbb{N}_0\times B \arrow[l, bend left, "g"]
\end{tikzcd}$$

$$\begin{cases} \mathbf{F}\ k = id + id \times k \\ \mathbf{F}\ X = 1 + \mathbb{N}_0 \times X \end{cases}$$

Abstraction (lists)

$$\begin{tikzcd}
L \arrow[r, bend left, "\mathsf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "k"'] & 1+\mathbb{N}_0\times L \arrow[l, bend left, "\mathsf{in}"] \arrow[d, "id+id\times k"] \\
B & 1+\mathbb{N}_0\times B \arrow[l, bend left, "g"]
\end{tikzcd}$$

$$\begin{tikzcd}
L \arrow[r, bend left, "\mathsf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "k"'] & \mathsf{F}\ L \arrow[l, bend left, "\mathsf{in}"] \arrow[d, "\mathsf{F}\ k"] \\
B & \mathsf{F}\ B \arrow[l, bend left, "g"]
\end{tikzcd}$$

$$\left\{\begin{array}{l}
\mathsf{F}\ k = id + id\times k \\
\mathsf{F}\ X = 1+\mathbb{N}_0\times X
\end{array}\right.$$

Abstraction

$$\begin{tikzcd}
T \arrow[r, bend left, "\mathsf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "k"'] & \mathbf{F}\,T \arrow[l, bend left, "\mathsf{in}"] \arrow[d, "\mathbf{F}\,k"] \\
B & \mathbf{F}\,B \arrow[l, bend left, "g"]
\end{tikzcd}$$

Abstraction

$$\begin{tikzcd}
T \arrow[r, bend left, "\mathbf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "k"'] & \mathbf{F}\ T \arrow[l, bend left, "\mathbf{in}"] \arrow[d, "\mathbf{F}\ k"] \\
B & \mathbf{F}\ B \arrow[l, bend left, "g"]
\end{tikzcd}$$

$$k \cdot \mathbf{in} = g \cdot \mathbf{F}\ k$$

Abstraction

$$\begin{tikzcd}
T \arrow[r, bend left, "\mathbf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "k"'] & \mathbf{F}\ T \arrow[l, bend left, "\mathbf{in}"] \arrow[d, "\mathbf{F}\ k"] \\
B & \mathbf{F}\ B \arrow[l, bend left, "g"]
\end{tikzcd}$$

$$k = (\!| g |\!)$$

$$k \cdot \mathbf{in} = g \cdot \mathbf{F}\ k$$

Abstraction

$$\begin{tikzcd}
T \arrow[r, bend left, "\mathbf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "k"'] & \mathbf{F}\ T \arrow[l, bend left, "\mathbf{in}"] \arrow[d, "\mathbf{F}\ k"] \\
B & \mathbf{F}\ B \arrow[l, bend left, "g"]
\end{tikzcd}$$

$$k = \llparenthesis\, g \,\rrparenthesis$$

$$k \cdot \mathbf{in} = g \cdot \mathbf{F}\ k$$

Read $k = \llparenthesis\, g \,\rrparenthesis$ as before: k is the *catamorphism* of g.

Catamorphism

$$\begin{tikzcd}
T \arrow[r, bend left, "\mathbf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "(\!| g |\!)"'] & \mathbf{F}\ T \arrow[l, bend left, "\mathbf{in}"] \arrow[d, "\mathbf{F}\ (\!| g |\!)"] \\
B & \mathbf{F}\ B \arrow[l, bend left, "g"]
\end{tikzcd}$$

Catamorphism

$$\begin{tikzcd}
T \arrow[rr, bend left, "\mathbf{out}"] \arrow[rr, phantom, "\cong"] \arrow[d, "\llparenthesis g \rrparenthesis"'] & & \mathbf{F}\ T \arrow[ll, bend left, "\mathbf{in}"] \arrow[d, "\mathbf{F}\ \llparenthesis g \rrparenthesis"] \\
B & & \mathbf{F}\ B \arrow[ll, bend left, "g"]
\end{tikzcd}$$

Universal property:

$$k = \llparenthesis g \rrparenthesis \quad \Leftrightarrow \quad k \cdot \mathbf{in} = g \cdot \mathbf{F}\ k$$

Summary

$$\text{Lists: } \begin{cases} T = L \\ \mathbf{in} = [nil, cons] \\ \begin{cases} \mathbf{F}\ X = 1 + \mathbb{N}_0 \times X \\ \mathbf{F}\ f = id + id \times f \end{cases} \end{cases}$$

Summary

$$\text{Lists:} \begin{cases} T = L \\ \mathbf{in} = [nil, cons] \\ \begin{cases} \mathbf{F}\ X = 1 + \mathbb{N}_0 \times X \\ \mathbf{F}\ f = id + id \times f \end{cases} \end{cases}$$

$$\mathbf{foldr}\ f\ \ i = (\!|\,[\underline{i}, \widehat{f}\,]\,|\!)$$

Summary

$$\text{Lists:}\ \left\{\begin{array}{l} T = L \\ \mathbf{in} = [\mathit{nil}, \mathit{cons}] \\ \left\{\begin{array}{l} \mathbf{F}\ X = 1 + \mathbb{N}_0 \times X \\ \mathbf{F}\ f = \mathit{id} + \mathit{id} \times f \end{array}\right. \end{array}\right.$$

$$\mathbf{foldr}\ f\ i = \llparenthesis [\underline{i}, \widehat{f}] \rrparenthesis$$

$$\text{Natural numbers:}\ \left\{\begin{array}{l} T = \mathbb{N}_0 \\ \mathbf{in} = [\underline{0}, \mathit{succ}] \\ \left\{\begin{array}{l} \mathbf{F}\ X = 1 + X \\ \mathbf{F}\ f = \mathit{id} + f \end{array}\right. \end{array}\right.$$

Summary

$$\text{Lists:}\ \begin{cases} T = L \\ \mathbf{in} = [\mathit{nil}, \mathit{cons}] \\ \begin{cases} \mathbf{F}\ X = 1 + \mathbb{N}_0 \times X \\ \mathbf{F}\ f = \mathit{id} + \mathit{id} \times f \end{cases} \end{cases}$$

$$\mathbf{foldr}\ f\ i = (\!|\,[\underline{i}, \widehat{f}]\,|\!)$$

$$\text{Natural numbers:}\ \begin{cases} T = \mathbb{N}_0 \\ \mathbf{in} = [\underline{0}, \mathit{succ}] \\ \begin{cases} \mathbf{F}\ X = 1 + X \\ \mathbf{F}\ f = \mathit{id} + f \end{cases} \end{cases}$$

$$\mathbf{for}\ b\ i = (\!|\,[\underline{i}, b]\,|\!)$$

Examples already seen

Natural numbers:

$$(a\times) = \llparenthesis\, [\underline{0}, (a+)] \,\rrparenthesis$$

$$a^{b} = \llparenthesis\, [\underline{1}, (a\times)] \,\rrparenthesis\ b$$

Lists:

$$\mathit{sum} = \llparenthesis\, [\underline{0}, \mathit{add}] \,\rrparenthesis$$

$$\mathit{length} = \llparenthesis\, [\underline{0}, \mathit{succ} \cdot \pi_2] \,\rrparenthesis$$

Cata-cancellation

In

$$k = (\!| g |\!) \;\Leftrightarrow\; k \cdot \mathbf{in} = g \cdot \mathbf{F}\ k$$

make $k := (\!| g |\!)$:

Cata-cancellation

In

$$k = (\!|\, g \,|\!) \;\Leftrightarrow\; k \cdot \mathbf{in} = g \cdot \mathsf{F}\; k$$

make $k := (\!|\, g \,|\!)$:

$$(\!|\, g \,|\!) = (\!|\, g \,|\!) \;\Leftrightarrow\; (\!|\, g \,|\!) \cdot \mathbf{in} = g \cdot \mathsf{F}\; (\!|\, g \,|\!)$$

Cata-cancellation

In

$$k = (\!|\, g \,|\!) \;\Leftrightarrow\; k \cdot \mathbf{in} = g \cdot \mathbf{F}\ k$$

make $k := (\!|\, g \,|\!)$:

$$
\begin{array}{ll}
 & (\!|\, g \,|\!) = (\!|\, g \,|\!) \;\Leftrightarrow\; (\!|\, g \,|\!) \cdot \mathbf{in} = g \cdot \mathbf{F}\ (\!|\, g \,|\!) \\
\Leftrightarrow & \qquad \{\ x = x \Leftrightarrow \mathit{true}\ \} \\
 & \mathit{true} \;\Leftrightarrow\; (\!|\, g \,|\!) \cdot \mathbf{in} = g \cdot \mathbf{F}\ (\!|\, g \,|\!)
\end{array}
$$

Cata-cancellation

In

$$k = (\!| g |\!) \;\Leftrightarrow\; k \cdot \mathbf{in} = g \cdot \mathbf{F}\; k$$

make $k := (\!| g |\!)$:

$$(\!| g |\!) = (\!| g |\!) \;\Leftrightarrow\; (\!| g |\!) \cdot \mathbf{in} = g \cdot \mathbf{F}\; (\!| g |\!)$$

$$\Leftrightarrow \quad \{\; x = x \Leftrightarrow \mathit{true} \;\}$$

$$\mathit{true} \;\Leftrightarrow\; (\!| g |\!) \cdot \mathbf{in} = g \cdot \mathbf{F}\; (\!| g |\!)$$

$$\Leftrightarrow \quad \{\; \text{trivial} \;\}$$

$$(\!| g |\!) \cdot \mathbf{in} = g \cdot \mathbf{F}\; (\!| g |\!)$$

Cata-cancellation

$$\llparenthesis\, g \,\rrparenthesis \cdot \mathsf{in} = g \cdot \mathsf{F}\ \llparenthesis\, g \,\rrparenthesis$$

$$\Leftrightarrow \qquad \{ \text{ isomorphism } \mathsf{in} \,/\, \mathsf{out} \ \}$$

$$\llparenthesis\, g \,\rrparenthesis = g \cdot \mathsf{F}\ \llparenthesis\, g \,\rrparenthesis \cdot \mathsf{out}$$

Cata-cancellation

$$\llparenthesis\, g \,\rrparenthesis \cdot \mathbf{in} = g \cdot \mathbf{F}\,\llparenthesis\, g \,\rrparenthesis$$

$$\Leftrightarrow \qquad \{\ \text{isomorphism } \mathbf{in} \,/\, \mathbf{out}\ \}$$

$$\llparenthesis\, g \,\rrparenthesis = g \cdot \mathbf{F}\,\llparenthesis\, g \,\rrparenthesis \cdot \mathbf{out}$$

$$\begin{tikzcd}
T \arrow[r, bend left, "\mathbf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "\llparenthesis\, g \,\rrparenthesis"'] & \mathbf{F}\,T \arrow[l, bend left, "\mathbf{in}"] \arrow[d, "\mathbf{F}\,\llparenthesis\, g \,\rrparenthesis"] \\
B & \mathbf{F}\,B \arrow[l, bend left, "g"]
\end{tikzcd}$$

Cata-cancellation

$$(\!| g |\!) \cdot \mathsf{in} = g \cdot \mathsf{F}\,(\!| g |\!)$$

$$\Leftrightarrow \qquad \{ \text{ isomorphism } \mathsf{in} \,/\, \mathsf{out} \, \}$$

$$(\!| g |\!) = g \cdot \mathsf{F}\,(\!| g |\!) \cdot \mathsf{out}$$

$$\begin{tikzcd}
T \arrow[r, bend left, "\mathsf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "(\!| g |\!)"'] & \mathsf{F}\,T \arrow[l, bend left, "\mathsf{in}"] \arrow[d, "\mathsf{F}\,(\!| g |\!)"] \\
B & \mathsf{F}\,B \arrow[l, "g"]
\end{tikzcd}$$

$$(\!| g |\!) = g \cdot \mathsf{F}\,(\!| g |\!) \cdot \mathsf{out}$$

executable definition

Functors

What does $\mathbf{F}$ — in $k \cdot \mathbf{in} = g \cdot \mathbf{F}\ k$ — mean, exactly?

$$\begin{tikzcd}
A \arrow[r, dotted, no head] \arrow[d, "f"'] & \mathbf{F}\ A \arrow[d, "\mathbf{F}\ f"] \\
B \arrow[r, dotted, no head] & \mathbf{F}\ B
\end{tikzcd}$$

Functors

What does $\mathbf{F}$ — in $k \cdot \mathbf{in} = g \cdot \mathbf{F}\ k$ — mean, exactly?

$$\begin{tikzcd}
A \arrow[r, dotted, no head] \arrow[d, "f"'] & \mathbf{F}\ A \arrow[d, "\mathbf{F}\ f"] \\
B \arrow[r, dotted, no head] & \mathbf{F}\ B
\end{tikzcd}$$

Recall **natural** (free) **properties**

Functors

What does $\mathbf{F}$ — $\mathbf{in}\ k \cdot \mathbf{in} = g \cdot \mathbf{F}\ k$ — mean, exactly?

$$\begin{tikzcd}
A \arrow[r, dotted, no head] \arrow[d, "f"'] & \mathbf{F}\ A \arrow[d, "\mathbf{F}\ f"] \\
B \arrow[r, dotted, no head] & \mathbf{F}\ B
\end{tikzcd}$$

For instance:

$$\mathbf{F}\ X = 1 + \mathbb{N}_0 \times X$$

Recall **natural** (free) **properties**

Functors

What does $\mathbf{F}$ — in $k \cdot \mathbf{in} = g \cdot \mathbf{F}\ k$ — mean, exactly?

$$\begin{tikzcd}
A \arrow[r, dotted, no head] \arrow[d, "f"'] & \mathbf{F}\ A \arrow[d, "\mathbf{F}\ f"] \\
B \arrow[r, dotted, no head] & \mathbf{F}\ B
\end{tikzcd}$$

Recall **natural** (free) **properties**

For instance:

$$\mathbf{F}\ X = 1 + \mathbb{N}_0 \times X$$

Substitute uppercase (X) by lowercase f ($X \to f$):

Functors

What does $\mathbf{F}$ — in $k \cdot \mathbf{in} = g \cdot \mathbf{F}\ k$ — mean, exactly?

$$\begin{tikzcd}
A \arrow[r, dotted, no head] \arrow[d, "f"'] & \mathbf{F}\ A \arrow[d, "\mathbf{F}\ f"] \\
B \arrow[r, dotted, no head] & \mathbf{F}\ B
\end{tikzcd}$$

Recall **natural** (free) **properties**

For instance:

$$\mathbf{F}\ X = 1 + \mathbb{N}_0 \times X$$

Substitute uppercase (X) by lowercase f ($X \to f$):

$$\mathbf{F}\ f = id + id \times f$$

Properties

$$\begin{aligned} \mathbf{F}\ id &= id \\ \mathbf{F}\ (g \cdot f) &= (\mathbf{F}\ g) \cdot (\mathbf{F}\ f) \end{aligned}$$

$$\mathbf{F}\ id = id$$

$$\begin{tikzcd}
A \arrow[r, dotted, no head] \arrow[d, "f"'] & \mathbf{F}\ A \arrow[d, "\mathbf{F}\ f"] \\
B \arrow[r, dotted, no head] & \mathbf{F}\ B
\end{tikzcd}$$

$$\mathbf{F}\ id = id$$

$$\begin{tikzcd}
A \arrow[r, dash, dotted] \arrow[d, "id"'] & \mathbf{F}\ A \arrow[d, "\mathbf{F}\ id"] \\
A \arrow[r, dash, dotted] & \mathbf{F}\ A
\end{tikzcd}$$

$\mathbf{F}\ \mathit{id} = \mathit{id}$

$$\begin{tikzcd}
A \arrow[r, dotted, no head] \arrow[d, "\mathit{id}"'] & \mathbf{F}\ A \arrow[d, "\mathbf{F}\ \mathit{id}"] \arrow[d, bend left=70, "\mathit{id}"] \\
A \arrow[r, dotted, no head] & \mathbf{F}\ A
\end{tikzcd}$$

$$\mathbf{F}\,(g \cdot f) = (\mathbf{F}\,g) \cdot (\mathbf{F}\,f)$$

$$\begin{tikzcd}
A \arrow[r, dotted, no head] \arrow[d, "f"'] & \mathbf{F}\,A \arrow[d, "\mathbf{F}\,f"] \\
B \arrow[r, dotted, no head] \arrow[d, "g"'] & \mathbf{F}\,B \arrow[d, "\mathbf{F}\,g"] \\
C \arrow[r, dotted, no head] & \mathbf{F}\,C
\end{tikzcd}$$

$$\mathbf{F}\ (g \cdot f) = (\mathbf{F}\ g) \cdot (\mathbf{F}\ f)$$

$$\begin{tikzcd}
A \arrow[r, dotted, no head] \arrow[d, "f"'] \arrow[dd, bend right=60, "g \cdot f"'] & \mathbf{F}\ A \arrow[d, "\mathbf{F}\ f"] \\
B \arrow[r, dotted, no head] \arrow[d, "g"'] & \mathbf{F}\ B \arrow[d, "\mathbf{F}\ g"] \\
C \arrow[r, dotted, no head] & \mathbf{F}\ C
\end{tikzcd}$$

$$\mathbf{F}\,(g \cdot f) = (\mathbf{F}\,g) \cdot (\mathbf{F}\,f)$$

$$\begin{tikzcd}
A \arrow[r, dotted, no head] \arrow[d, "f"'] \arrow[dd, bend right=60, "g\cdot f"'] & \mathbf{F}\,A \arrow[d, "\mathbf{F}\,f"] \arrow[dd, bend left=50, "\mathbf{F}\,g\cdot\mathbf{F}\,f"] \arrow[dd, bend left=90, "\mathbf{F}\,(g\cdot f)"] \\
B \arrow[r, dotted, no head] \arrow[d, "g"'] & \mathbf{F}\,B \arrow[d, "\mathbf{F}\,g"] \\
C \arrow[r, dotted, no head] & \mathbf{F}\,C
\end{tikzcd}$$

Well known example

List functor:

$$\mathbf{F}\ f = f^{\star}$$

Well known example

List functor:

$$\mathbf{F}\ f = f^{\star} \quad \textbf{where} \quad f^{\star}\ x = [f\ a \mid a \leftarrow x] = \mathsf{map}\ f\ x$$

$$\begin{tikzcd}
A \arrow[r, dotted, no head] \arrow[d, "f"'] & A^{\star} \arrow[d, "f^{\star}"] \\
B \arrow[r, dotted, no head] & B^{\star}
\end{tikzcd}$$

In fact

$$id^{\star}\ x = [a \mid a \leftarrow x] = x$$

In fact

$$id^{\star}\ x = [a \mid a \leftarrow x] = x$$

$$(g^{\star} \cdot f^{\star})\ x$$

In fact

$$id^{\star}\ x = [a \mid a \leftarrow x] = x$$

$$(g^{\star} \cdot f^{\star})\ x = [g\ b \mid b \leftarrow [f\ a \mid a \leftarrow x]]$$

In fact

$$id^{\star}\ x = [a \mid a \leftarrow x] = x$$

$$(g^{\star} \cdot f^{\star})\ x = [g\ b \mid b \leftarrow [f\ a \mid a \leftarrow x]] = (g \cdot f)^{\star}\ x$$

Properties

Recall

$$\begin{aligned} \mathsf{F}\ \mathit{id} &= \mathit{id} \\ \mathsf{F}\ (g \cdot f) &= (\mathsf{F}\ g) \cdot (\mathsf{F}\ f) \end{aligned}$$

These properties are essential for reasoning about **catamorphisms**.

See example next.

Cata-fusion

$$\begin{tikzcd}
T \arrow[d, "\llparenthesis g \rrparenthesis"'] \arrow[dd, bend right=60, "\llparenthesis h \rrparenthesis"'] & \mathbf{F}\ T \arrow[l, "\mathsf{in}"'] \arrow[d, "\mathbf{F}\ \llparenthesis g \rrparenthesis"] \\
A \arrow[d, "f"'] & \mathbf{F}\ A \arrow[l, "g"] \arrow[d, "\mathbf{F}\ f"] \\
B & \mathbf{F}\ B \arrow[l, "h"]
\end{tikzcd}$$

Cata-fusion

$$f \cdot (\!| g |\!) = (\!| h |\!)$$

Cata-fusion

$$
\begin{array}{ll}
 & f \cdot (\!| g |\!) = (\!| h |\!) \\
\Leftrightarrow & \quad \{ \text{ Cata-universal for } k = f \cdot (\!| g |\!) \ \} \\
 & f \cdot (\!| g |\!) \cdot \mathbf{in} = h \cdot \mathsf{F}\,(f \cdot (\!| g |\!))
\end{array}
$$

Cata-fusion

$$f \cdot (\!| g |\!) = (\!| h |\!)$$

$\Leftrightarrow$ { Cata-universal for $k = f \cdot (\!| g |\!)$ }

$$f \cdot (\!| g |\!) \cdot \mathbf{in} = h \cdot \mathbf{F}\,(f \cdot (\!| g |\!))$$

$\Leftrightarrow$ { Cata-cancellation }

$$f \cdot g \cdot \mathbf{F}\,(\!| g |\!) = h \cdot \mathbf{F}\,(f \cdot (\!| g |\!))$$

Cata-fusion

$$f \cdot g \cdot \mathbf{F}\,(\!|\,g\,|\!) = h \cdot \mathbf{F}\,(f \cdot (\!|\,g\,|\!))$$

Cata-fusion

$$
\begin{aligned}
& f \cdot g \cdot \mathbf{F}\,(\!|\, g \,|\!) = h \cdot \mathbf{F}\,(f \cdot (\!|\, g \,|\!)) \\
\Leftrightarrow \quad & \{ \text{ functor-}\mathbf{F}\ (1) \ \} \\
& f \cdot g \cdot \mathbf{F}\,(\!|\, g \,|\!) = h \cdot \mathbf{F}\, f \cdot \mathbf{F}\,(\!|\, g \,|\!)
\end{aligned}
$$

Cata-fusion

$$f \cdot g \cdot \mathbf{F}\,(\!|\, g \,|\!) = h \cdot \mathbf{F}\,(f \cdot (\!|\, g \,|\!))$$

$\Leftrightarrow$ { functor-$\mathbf{F}$ (1) }

$$f \cdot g \cdot \mathbf{F}\,(\!|\, g \,|\!) = h \cdot \mathbf{F}\,f \cdot \mathbf{F}\,(\!|\, g \,|\!)$$

$\Leftarrow$ { Leibniz }

$$f \cdot g = h \cdot (\mathbf{F}\,f)$$

Cata-fusion

$$
\begin{array}{ll}
 & f \cdot g \cdot \mathsf{F}\, (\!| g |\!) = h \cdot \mathsf{F}\, (f \cdot (\!| g |\!)) \\
\Leftrightarrow & \qquad \{ \text{ functor-}\mathsf{F}\text{ (1) } \} \\
 & f \cdot g \cdot \mathsf{F}\, (\!| g |\!) = h \cdot \mathsf{F}\, f \cdot \mathsf{F}\, (\!| g |\!) \\
\Leftarrow & \qquad \{ \text{ Leibniz } \} \\
 & f \cdot g = h \cdot (\mathsf{F}\, f)
\end{array}
$$

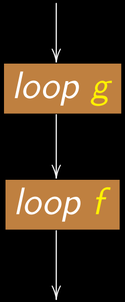
Summing up:

$$f \cdot (\!| g |\!) = (\!| h |\!) \quad \Leftarrow \quad f \cdot g = h \cdot (\mathsf{F}\, f)$$

Cálculo de Programas

Class T08

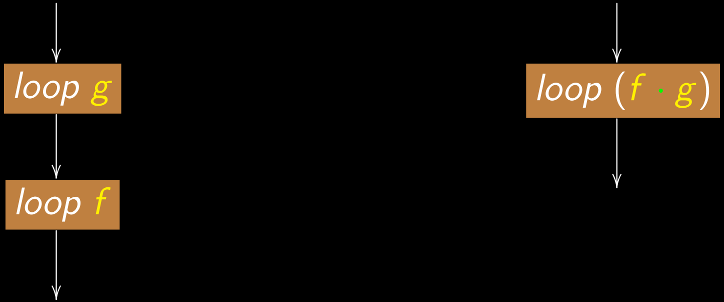
Fusion in programming — ”sequential”



loop g

loop f

Fusion in programming — ”sequential”

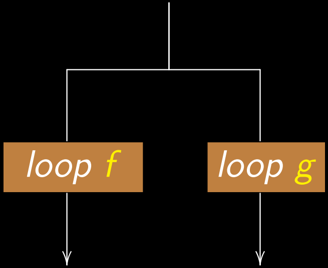


loop g

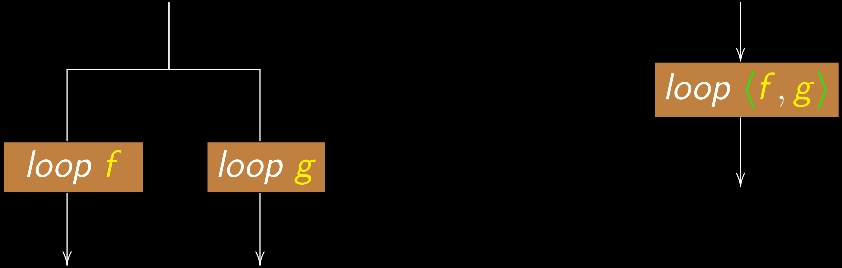
loop f

loop $(f \cdot g)$

Fusion in programming — ”parallel”



Fusion in programming — ”parallel”



The pattern $\langle f, g \rangle = (\!| h |\!)$

Catas that yield pairs:

$$\begin{tikzcd}
T \arrow[r, bend left, "\mathsf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "(\!| g |\!)"'] & \mathbf{F}\ T \arrow[l, bend left, "\mathsf{in}"] \arrow[d, "\mathbf{F}\ (\!| g |\!)"] \\
A \times B & \mathbf{F}\ (A \times B) \arrow[l, "g"]
\end{tikzcd}$$

Catas that yield pairs

$$\begin{tikzcd}
T \arrow[r, bend left, "\mathsf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "\langle f,h\rangle = (\!| g |\!)"'] & \mathbf{F}\ T \arrow[l, bend left, "\mathsf{in}"] \arrow[d, "\mathbf{F}\ \langle f,h\rangle"] \\
A \times B & \mathbf{F}\ (A \times B) \arrow[l, "g"]
\end{tikzcd}$$

Mutual recursion

$$\begin{tikzcd}
T \arrow[r, bend left, "\mathsf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "\langle f,h\rangle=(\!|\,\langle k,m\rangle\,|\!)"'] & \mathbf{F}\ T \arrow[l, bend left, "\mathsf{in}"] \arrow[d, "\mathbf{F}\ \langle f,h\rangle"] \\
A\times B & \mathbf{F}\ (A\times B) \arrow[l, "g=\langle k,m\rangle"]
\end{tikzcd}$$

Mutual recursion

$$\begin{tikzcd}
T \arrow[r, bend left, "\mathbf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "\langle f,h\rangle=\llparenthesis\langle k,m\rangle\rrparenthesis"'] & \mathbf{F}\ T \arrow[l, bend left, "\mathbf{in}"] \arrow[d, "\mathbf{F}\ \langle f,h\rangle"] \\
A \times B & \mathbf{F}\ (A \times B) \arrow[l, bend left, "g=\langle k,m\rangle"]
\end{tikzcd}$$

$$\langle f, h\rangle = \llparenthesis \langle k, m\rangle \rrparenthesis \Leftrightarrow \begin{cases} f \cdot \mathbf{in} = k \cdot \mathbf{F}\ \langle f, h\rangle \\ h \cdot \mathbf{in} = m \cdot \mathbf{F}\ \langle f, h\rangle \end{cases}$$

Mutual recursion

$$
\begin{array}{ll}
 & \langle f, h \rangle = (\!| \langle k, m \rangle |\!) \\
\Leftrightarrow & \quad \{ \text{ Cata-universal } \} \\
 & \langle f, h \rangle \cdot \mathsf{in} = \langle k, m \rangle \cdot \mathsf{F}\ \langle f, h \rangle \\
\Leftrightarrow & \quad \{ \times\text{-fusion } \} \\
 & \langle f \cdot \mathsf{in}, h \cdot \mathsf{in} \rangle = \langle k \cdot \mathsf{F}\ \langle f, h \rangle, m \cdot \mathsf{F}\ \langle f, h \rangle \rangle \\
\Leftrightarrow & \quad \{ \text{ Eq-}\times \} \\
 & \left\{ \begin{array}{l} f \cdot \mathsf{in} = k \cdot \mathsf{F}\ \langle f, h \rangle \\ h \cdot \mathsf{in} = m \cdot \mathsf{F}\ \langle f, h \rangle \end{array} \right.
\end{array}
$$

Mutual recursion — case $T = \mathbb{N}_0$

$$\langle f, g \rangle = (\!|\, \langle h, k \rangle \,|\!)$$

$$\Leftrightarrow \qquad \{ \text{ mutual recursion with } \mathbf{in} = [\underline{0}, succ], \ \mathsf{F}\, f = id + f \text{ etc } \}$$

$$\left\{ \begin{array}{l} f \cdot [\underline{0}, succ] = h \cdot (id + \langle f, g \rangle) \\ g \cdot [\underline{0}, succ] = k \cdot (id + \langle f, g \rangle) \end{array} \right.$$

$$\Leftrightarrow \qquad \{ \text{ let } h = [h_1, h_2] \text{ and } k = [k_1, k_2] \ \}$$

$$\left\{ \begin{array}{l} f \cdot [\underline{0}, succ] = [h_1, h_2] \cdot (id + \langle f, g \rangle) \\ g \cdot [\underline{0}, succ] = [k_1, k_2] \cdot (id + \langle f, g \rangle) \end{array} \right.$$

Mutual recursion — case $T = \mathbb{N}_0$

Moving on:

$$
\begin{array}{ll}
 & \langle f , g \rangle = (\!| \langle [h_1 , h_2], [k_1 , k_2] \rangle |\!) \\
\Leftrightarrow & \qquad \{ \text{ coproduct laws } \} \\
 & \left\{ \begin{array}{l} \left\{ \begin{array}{l} f \cdot \underline{0} = h_1 \\ f \cdot succ = h_2 \cdot \langle f , g \rangle \end{array} \right. \\ \left\{ \begin{array}{l} g \cdot \underline{0} = k_1 \\ g \cdot succ = k_2 \cdot \langle f , g \rangle \end{array} \right. \end{array} \right.
\end{array}
$$

Mutual recursion — case $T = \mathbb{N}_0$

Finally, add variables to righthand side, with $h_1 = \underline{a}$ and $k_1 = \underline{b}$:

$$\langle f, g \rangle = (\!|\, \langle [\underline{a}, h_2], [\underline{b}, k_2] \rangle \,|\!) \quad \Leftrightarrow \quad \left\{ \begin{array}{l} \left\{ \begin{array}{l} f\ 0 = a \\ f\ (n+1) = h_2\ (f\ n, g\ n) \end{array} \right. \\ \left\{ \begin{array}{l} g\ 0 = b \\ g\ (n+1) = k_2\ (f\ n, g\ n) \end{array} \right. \end{array} \right.$$

Example — Fibonacci

Clearly, the following function is not a catamorphism of $\mathbb{N}_0$:

$$
\begin{aligned}
\mathit{fib}\ 0 &= 1 \\
\mathit{fib}\ 1 &= 1 \\
\mathit{fib}\ (n+2) &= \mathit{fib}\ (n+1) + \mathit{fib}\ n
\end{aligned}
$$

But it can be transformed into one using mutual recursion.

Example — Fibonacci

Let us define the auxiliary function:

$$f\ n = \mathit{fib}\ (n+1)$$

Then:

$$\begin{aligned} &f\ 0 = \mathit{fib}\ 1 = 1 \\ &f\ (n+1) = \mathit{fib}\ (n+2) = f\ n + \mathit{fib}\ n \end{aligned}$$

Example — Fibonacci

Concerning *fib*:

$$\begin{aligned} fib\ 0 &= 1 \\ fib\ (n+1) &= f\ \ n \end{aligned}$$

Summing up, we have the following mutual recursion:

$$\left\{ \begin{array}{l} \left\{ \begin{array}{l} f\ \ 0 = 1 \\ f\ \ (n+1) = f\ \ n + fib\ n \end{array} \right. \\ \left\{ \begin{array}{l} fib\ 0 = 1 \\ fib\ (n+1) = f\ \ n \end{array} \right. \end{array} \right.$$

Example — Fibonacci

By applying the mutual recursion law for $T = \mathbb{N}_0$:

$$\left\{ \begin{array}{l} \left\{ \begin{array}{l} f\ 0 = a \\ f\ (n+1) = h_2\ (f\ \ n, \mathit{fib}\ n) \end{array} \right. \\ \left\{ \begin{array}{l} \mathit{fib}\ 0 = b \\ \mathit{fib}\ (n+1) = k_2\ (f\ \ n, \mathit{fib}\ n) \end{array} \right. \end{array} \right. \quad \Leftrightarrow \quad \langle f, \mathit{fib} \rangle = (\!|\, \langle [\underline{a}, h_2], [\underline{b}, k_2] \rangle \,|\!)$$

Example — Fibonacci

Solutions:

$$\begin{aligned} a &= 1 \\ b &= 1 \\ h_2 &= add \\ k_2 &= \pi_1 \end{aligned}$$

Then:

$$\langle f, fib \rangle = \llparenthesis \, \langle [\underline{1}, add], [\underline{1}, \pi_1] \rangle \, \rrparenthesis$$

Fibonacci becomes a for-loop

Given $\langle f, \mathit{fib} \rangle$ (a $\mathbb{N}_0$ **catamorphism**), we can convert it into a **for-loop** according to the definition:

$$\mathbf{for}\ b\ i = \llparenthesis [\underline{i}, b] \rrparenthesis$$

Fibonacci becomes a for-loop

Given $\langle f, fib \rangle$ (a $\mathbb{N}_0$ **catamorphism**), we can convert it into a **for-loop** according to the definition:

$$\mathbf{for}\ b\ i = \llparenthesis\, [\underline{i}, b]\, \rrparenthesis$$

$$\begin{aligned}
& \langle f, fib \rangle = \llparenthesis\, \langle [\underline{1}, add], [\underline{1}, \pi_1] \rangle\, \rrparenthesis \\
\Leftrightarrow \quad & \{ \text{ exchange law } \} \\
& \langle f, fib \rangle = \llparenthesis\, [\langle \underline{1}, \underline{1} \rangle, \langle add, \pi_1 \rangle]\, \rrparenthesis \\
\Leftrightarrow \quad & \{ \text{ recall exercise sheet 3 } \} \\
& \langle f, fib \rangle = \llparenthesis\, [\underline{(1,1)}, \langle add, \pi_1 \rangle]\, \rrparenthesis \\
\Leftrightarrow \quad & \{ \text{ definition of } \mathbf{for}\ b\ i \ \} \\
& \langle f, fib \rangle = \mathbf{for}\ \langle add, \pi_1 \rangle\ (1,1)
\end{aligned}$$

Example — Fibonacci

In summary:

$$fib = \pi_2 \cdot (\textbf{for}\ \langle add, \pi_1 \rangle\ (1, 1))$$

Cf. the same in the syntax of **C**:

```
int fib(int n)
{
int x=1; int y=1; int i;
for (i=1;i<=n;i++) {int a=x; x=x+y; y=a;}
return y;
};
```

Example — factorial

$$
\begin{aligned}
fac\ 0 &= 1 \\
fac\ (n+1) &= \underbrace{(n+1)}_{f\ n} \times fac\ n
\end{aligned}
$$

Then:

$$
\langle f, fac \rangle = (\!|\, \langle [\underline{a}, h_2], [\underline{b}, k_2] \rangle \,|\!) \quad \Leftrightarrow \quad \left\{ \begin{array}{l} \left\{ \begin{array}{l} f\ 0 = a \\ f\ (n+1) = h_2\ (f\ n, fac\ n) \end{array} \right. \\ \left\{ \begin{array}{l} fac\ 0 = b \\ fac\ (n+1) = k_2\ (f\ n, fac\ n) \end{array} \right. \end{array} \right.
$$

Example — factorial

$$\langle f, fac \rangle = (\!|\, \langle [\underline{a}, h_2], [\underline{b}, k_2] \rangle \,|\!) \quad \Leftrightarrow \quad \left\{ \begin{array}{l} \left\{ \begin{array}{l} f\ 0 = a \\ f\ (n+1) = h_2\ (f\ n, fac\ n) \end{array} \right. \\ \left\{ \begin{array}{l} fac\ 0 = b \\ fac\ (n+1) = k_2\ (f\ n, fac\ n) \end{array} \right. \end{array} \right.$$

$$k_2 = mul$$
$$b = 1$$
$$f\ n = n + 1$$
$$a = 1$$

(Still need h_2...)

Example — factorial

$$\begin{aligned} & f\ (n+1) \\ = \ & (n+1)+1 \\ = \ & 1+(n+1) \\ = \ & 1+f\ n \\ = \ & 1+\pi_1\ (f\ n, fac\ n) \\ = \ & \underbrace{succ \cdot \pi_1}_{h_2}\ (f\ n, fac\ n) \end{aligned}$$

Then:

$$k_2 = mul$$

$$b = 1$$

$$h_2 = succ \cdot \pi_1$$

$$a = 1$$

Example — factorial becomes a for-loop

Finally:

$$
\begin{aligned}
& \langle f, \mathit{fac} \rangle \\
= {} & \llparenthesis\, \langle [\underline{1}, \mathit{succ} \cdot \pi_1], [\underline{1}, \mathit{mul}] \rangle \,\rrparenthesis \\
= {} & \llparenthesis\, [\underline{(1, 1)}, \langle \mathit{succ} \cdot \pi_1, \mathit{mul} \rangle] \,\rrparenthesis \\
= {} & \mathbf{for}\ \langle \mathit{succ} \cdot \pi_1, \mathit{mul} \rangle\ ((1, 1)) \\
= {} & \mathbf{for}\ g\ (1, 1)\ \mathbf{where}\ g\ (x, y) = (x + 1, x \times y)
\end{aligned}
$$

For-loops (in C)

In general, $k = \textbf{for}\ f\ i$ can be encoded in the syntax of **C** by writing:

```
int k(int n) {
  int r=i;
  int j;
  for (j=1;j<n+1;j++) {r=f(r);}
  return r;
};
```

Factorial as a for-loop (in C)

From

$$\mathit{fac} = \pi_2\ (\textbf{for}\ g\ (1, 1))\ \textbf{where}\ g\ (x, y) = (x + 1, x \times y)$$

we thus obtain:

```
int fac(int n) {
  int x=1; int y=1;
  int j;
  for (j=1;j<n+1;j++) {y=x*y;x=x+1;}
  return y;
};
```

Banana-split

Banana-split

$$(\!|\, \langle \_, \_ \rangle \,|\!)$$

Banana-split

$$\llparenthesis \langle \_, \_ \rangle \rrparenthesis$$

$$\langle \llparenthesis \_ \rrparenthesis, \llparenthesis \_ \rrparenthesis \rangle$$

Banana-split

$$\llparenthesis \langle \_, \_ \rangle \rrparenthesis$$

$$\langle \llparenthesis \_ \rrparenthesis, \llparenthesis \_ \rrparenthesis \rangle$$

$$\langle \llparenthesis i \rrparenthesis, \llparenthesis j \rrparenthesis \rangle$$

Banana-split

$$\langle (\!|\, i \,|\!), (\!|\, j \,|\!) \rangle$$

$$\begin{tikzcd}
 & & & & & \mathbf{F}\,T \arrow[llllld, "\mathsf{in}"'] \arrow[ddll, "\mathbf{F}(\!|\, i \,|\!)"'] \arrow[ddd, dotted, "f_6"'] \arrow[ddddr, "\mathbf{F}(\!|\, j \,|\!)"] & \\
 & T \arrow[ddl, "(\!|\, i \,|\!)"'] \arrow[ddd, dotted, "f_7"'] \arrow[ddddr, "(\!|\, j \,|\!)"] & & & & & \\
 & & & \mathbf{F}A \arrow[dlll, "i"'] & & & \\
A & & & & & \mathbf{F}(A \times B) \arrow[ull, "f_1"'] \arrow[lllll, dotted, "f_5"'] \arrow[dllll, dotted, "f_4"'] \arrow[ddlll, dotted, "f_3"] \arrow[dr, "f_2"'] & \\
 & A \times B \arrow[ul, "\pi_1"] \arrow[dr, "\pi_2"'] & & & & & \mathbf{F}B \arrow[dllll, "j"] \\
 & & B & & & &
\end{tikzcd}$$

Banana-split

$\langle (\!| i |\!), (\!| j |\!) \rangle$

$$\begin{tikzcd}
 & & & & \mathbf{F}T \arrow[dlll, "\mathsf{in}"'] \arrow[ddl, "\mathbf{F}(\!| i |\!)"'] \arrow[ddd, dotted, "f_6"'] \arrow[ddddr, "\mathbf{F}(\!| j |\!)"] & \\
 & T \arrow[ddl, "(\!| i |\!)"'] \arrow[ddd, dotted, "f_7"'] \arrow[ddddr, "(\!| j |\!)"] & & & & \\
 & & & \mathbf{F}A \arrow[dlll, "i"'] & & \\
A & & & & \mathbf{F}(A \times B) \arrow[ul, "\mathbf{F}\ \pi_1"'] \arrow[llll, dotted, "f_5"'] \arrow[dlll, dotted, "f_4"'] \arrow[ddll, dotted, "f_3"'] \arrow[dr, "\mathbf{F}\ \pi_2"'] & \\
 & A \times B \arrow[ul, "\pi_1"] \arrow[dr, "\pi_2"'] & & & & \mathbf{F}B \arrow[dlll, "j"] \\
 & & B & & &
\end{tikzcd}$$

Banana-split

$$\langle \llparenthesis\, i \,\rrparenthesis, \llparenthesis\, j \,\rrparenthesis \rangle$$

$$\begin{tikzcd}
 & & & \mathbf{F}\,T \arrow[dlll, "\mathsf{in}"'] \arrow[ddl, "\mathbf{F}\llparenthesis\, i \,\rrparenthesis"'] \arrow[ddd, dotted, "f_6"'] \arrow[ddddr, "\mathbf{F}\llparenthesis\, j \,\rrparenthesis"] & \\
 T \arrow[ddl, "\llparenthesis\, i \,\rrparenthesis"'] \arrow[ddd, dotted, "f_7"'] \arrow[ddddr, "\llparenthesis\, j \,\rrparenthesis"] & & & & \\
 & & \mathbf{F}A \arrow[dll, "i"'] & & \\
 A & & & \mathbf{F}(A \times B) \arrow[ul, "\mathbf{F}\ \pi_1"'] \arrow[lll, dotted, "i \cdot \mathbf{F}\ \pi_1"'] \arrow[dll, dotted, "f_4"'] \arrow[ddl, dotted, "j \cdot \mathbf{F}\ \pi_2"'] \arrow[dr, "\mathbf{F}\ \pi_2"'] & \\
 & A \times B \arrow[ul, "\pi_1"] \arrow[dr, "\pi_2"'] & & & \mathbf{F}B \arrow[dll, "j"] \\
 & & B & &
\end{tikzcd}$$

Banana-split

$$\langle \llparenthesis i \rrparenthesis , \llparenthesis j \rrparenthesis \rangle$$

$$\begin{tikzcd}
&&&& \mathbf{F}\,T \arrow[dlll, "\mathbf{in}"'] \arrow[ddl, "\mathbf{F}\llparenthesis i \rrparenthesis"'] \arrow[ddd, dotted, "f_6"'] \arrow[ddddr, "\mathbf{F}\llparenthesis j \rrparenthesis"] & \\
& T \arrow[ddl, "\llparenthesis i \rrparenthesis"'] \arrow[ddd, dotted, "f_7"'] \arrow[ddddr, "\llparenthesis j \rrparenthesis"] &&&& \\
&&& \mathbf{F}A \arrow[dlll, "i"'] && \\
A &&&& \mathbf{F}(A \times B) \arrow[ul, "\mathbf{F}\ \pi_1"'] \arrow[llll, dotted, "i \cdot \mathbf{F}\ \pi_1"'] \arrow[dlll, dotted, "\langle i \cdot \mathbf{F}\ \pi_1 , j \cdot \mathbf{F}\ \pi_2 \rangle"'] \arrow[ddll, dotted, "j \cdot \mathbf{F}\ \pi_2"] \arrow[dr, "\mathbf{F}\ \pi_2"'] & \\
& A \times B \arrow[ul, "\pi_1"] \arrow[dr, "\pi_2"'] &&&& \mathbf{F}B \arrow[dlll, "j"] \\
&& B &&&
\end{tikzcd}$$

Banana-split

$\langle \llparenthesis\, i \,\rrparenthesis, \llparenthesis\, j \,\rrparenthesis \rangle$

$$\begin{tikzcd}
 & & & & \mathbf{F}\,T \arrow[dlll, "\mathsf{in}"'] \arrow[ddll, "\mathbf{F}\llparenthesis\, i \,\rrparenthesis"'] \arrow[dddl, dotted, "\mathbf{F}\,\langle \llparenthesis\, i \,\rrparenthesis, \llparenthesis\, j \,\rrparenthesis \rangle"'] \arrow[dddd, "\mathbf{F}\llparenthesis\, j \,\rrparenthesis"] \\
 & T \arrow[ddl, "\llparenthesis\, i \,\rrparenthesis"'] \arrow[ddd, dotted, "\langle \llparenthesis\, i \,\rrparenthesis, \llparenthesis\, j \,\rrparenthesis \rangle"'] \arrow[ddddr, "\llparenthesis\, j \,\rrparenthesis"] & & & \\
 & & \mathbf{F}A \arrow[dll, "i"'] & & \\
A & & & \mathbf{F}(A \times B) \arrow[ul, "\mathbf{F}\ \pi_1"'] \arrow[lll, dotted, "i \cdot \mathbf{F}\ \pi_1"'] \arrow[dll, dotted, "\langle i \cdot \mathbf{F}\ \pi_1, j \cdot \mathbf{F}\ \pi_2 \rangle"'] \arrow[ddl, dotted, "j \cdot \mathbf{F}\ \pi_2"'] \arrow[dr, "\mathbf{F}\ \pi_2"] & \\
 & A \times B \arrow[ul, "\pi_1"] \arrow[dr, "\pi_2"'] & & & \mathbf{F}B \arrow[dll, "j"] \\
 & & B & &
\end{tikzcd}$$

Banana-split

$$\langle \llparenthesis\, i \,\rrparenthesis, \llparenthesis\, j \,\rrparenthesis \rangle$$

$$\begin{tikzcd}
 & & & & \mathsf{F}\,T \arrow[dlll, "\mathsf{in}"'] \arrow[ddl, "\mathsf{F} \llparenthesis\, i \,\rrparenthesis"'] \arrow[ddd, dotted, "\mathsf{F}\, \langle \llparenthesis\, i \,\rrparenthesis, \llparenthesis\, j \,\rrparenthesis \rangle"'] \arrow[ddddr, "\mathsf{F} \llparenthesis\, j \,\rrparenthesis"] & \\
 & T \arrow[ddl, "\llparenthesis\, i \,\rrparenthesis"'] \arrow[ddd, dotted, "\langle \llparenthesis\, i \,\rrparenthesis, \llparenthesis\, j \,\rrparenthesis \rangle"'] \arrow[ddddr, "\llparenthesis\, j \,\rrparenthesis"] & & & & \\
 & & & \mathsf{F}A \arrow[dlll, "i"'] & & \\
A & & & & \mathsf{F}(A \times B) \arrow[ul, "\mathsf{F}\ \pi_1"'] \arrow[llll, dotted, "i \cdot \mathsf{F}\ \pi_1"'] \arrow[dlll, dotted, "\langle i \cdot \mathsf{F}\ \pi_1, j \cdot \mathsf{F}\ \pi_2 \rangle"'] \arrow[ddll, dotted, "j \cdot \mathsf{F}\ \pi_2"] \arrow[dr, "\mathsf{F}\ \pi_2"] & \\
 & A \times B \arrow[ul, "\pi_1"] \arrow[dr, "\pi_2"'] & & & & \mathsf{F}B \arrow[dlll, "j"] \\
 & & B & & &
\end{tikzcd}$$

Banana-split

$$\begin{tikzcd}
 & & & & \mathbf{F}\,T \arrow[llld, "\mathsf{in}"'] \arrow[ldd, "\mathbf{F}\llparenthesis i \rrparenthesis"'] \arrow[ddd, dotted, "\mathbf{F}\,\langle\llparenthesis i \rrparenthesis,\llparenthesis j \rrparenthesis\rangle"'] \arrow[rdddd, "\mathbf{F}\llparenthesis j \rrparenthesis"] & \\
 & T \arrow[ldd, "\llparenthesis i \rrparenthesis"'] \arrow[ddd, dotted, "\langle\llparenthesis i \rrparenthesis,\llparenthesis j \rrparenthesis\rangle"'] \arrow[rdddd, "\llparenthesis j \rrparenthesis"] & & & & \\
 & & & \mathbf{F}\,A \arrow[llld, "i"'] & & \\
A & & & & \mathbf{F}(A \times B) \arrow[lu, "\mathbf{F}\ \pi_1"'] \arrow[llll, dotted, "i{\cdot}\mathbf{F}\ \pi_1"'] \arrow[llld, dotted, "\langle i{\cdot}\mathbf{F}\ \pi_1, j{\cdot}\mathbf{F}\ \pi_2\rangle"'] \arrow[lldd, dotted, "j{\cdot}\mathbf{F}\ \pi_2"] \arrow[rd, "\mathbf{F}\ \pi_2"] & \\
 & A \times B \arrow[lu, "\pi_1"] \arrow[rd, "\pi_2"'] & & & & \mathbf{F}\,B \arrow[llld, "j"] \\
 & & B & & &
\end{tikzcd}$$

$$\langle \llparenthesis\, i \,\rrparenthesis, \llparenthesis\, j \,\rrparenthesis \rangle = \llparenthesis\, \langle i \cdot \mathbf{F}\ \pi_1, j \cdot \mathbf{F}\ \pi_2 \rangle \,\rrparenthesis$$

Example

Average of a (non empty) list:

$$average = (/) \cdot \langle sum, length \rangle$$

where

$$\begin{aligned} sum &= (\![[0, add]]\!) \\ length &= (\![[0, succ \cdot \pi_2]]\!) \end{aligned}$$

Example

$$average = (/) \cdot \underbrace{\langle sum, length \rangle}_{aux}$$

By banana-split:

$$\begin{aligned} aux &= (\!|\, \langle f_5, f_3 \rangle \,|\!) \ \textbf{where} \\ f_5 &= [0, add] \cdot (id + id \times \pi_1) \\ f_3 &= [0, succ \cdot \pi_2] \cdot (id + id \times \pi_2) \end{aligned}$$

Example

Finally, solve $aux = (\!|\, \langle f_5, f_3 \rangle \,|\!)$ and convert everything to pointwise notation:

$$
\begin{array}{l}
average\ l = x \mathbin{/} y\ \textbf{where} \\
\quad (x, y) = aux\ l \\
\quad aux\ [\,] = (0, 0) \\
\quad aux\ (a : l) = (a + x, y + 1)\ \textbf{where}\ (x, y) = aux\ l
\end{array}
$$

Example

Summing up,

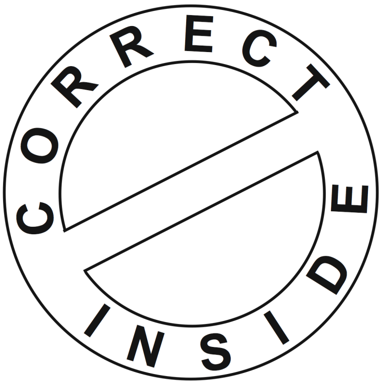
$$\begin{array}{l}
\mathit{average}\ l = x \mathbin{/} y\ \textbf{where} \\
\quad (x, y) = \mathit{aux}\ l \\
\quad \mathit{aux}\ [\,] = (0, 0) \\
\quad \mathit{aux}\ (a : l) = (a + x, y + 1)\ \textbf{where}\ (x, y) = \mathit{aux}\ l
\end{array}$$

twice as fast as

$$\begin{array}{l}
\mathit{average}\ l = (\mathit{sum}\ l) \mathbin{/} (\mathit{length}\ l)\ \textbf{where} \\
\quad \mathit{sum}\ [\,] = 0 \\
\quad \mathit{sum}\ (h : t) = h + \mathit{sum}\ t \\
\quad \mathit{length}\ [\,] = 0 \\
\quad \mathit{length}\ (h : t) = 1 + \mathit{length}\ t
\end{array}$$

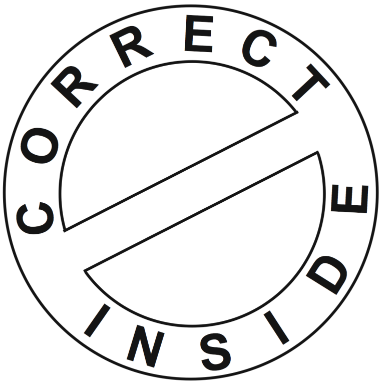
and **guaranteed** to be correct.

Program Calculation



Efficiency without sacrificing **correction**.

Program Calculation



Efficiency without sacrificing **correction**.

Who cares about an efficient program that gives wrong outputs?

Program Calculation

Efficiency should never sacrifice **correction**!

Cálculo
de
Programas

Class T09

Recap

Inductive type T and the functor $\mathbf{F}$ that determines its pattern of recursion:

$$\begin{tikzcd} T \arrow[r, bend left, "\mathbf{out}"] \arrow[r, phantom, "\cong"] & \mathbf{F}\ T \arrow[l, bend left, "\mathbf{in}"] \end{tikzcd}$$

Recap

Inductive type T and the functor $\mathbf{F}$ that determines its pattern of recursion:

$$\begin{tikzcd} T \arrow[r, bend left, "\mathbf{out}"] \arrow[r, phantom, "\cong"] & \mathbf{F}\ T \arrow[l, bend left, "\mathbf{in}"] \end{tikzcd}$$

Two examples so far: $T = \mathbb{N}_0$ and $T = A^{\star}$

Recap

Inductive type T and the functor $\mathbf{F}$ that determines its pattern of recursion:

$$\begin{tikzcd} T \arrow[r, bend left, "\mathbf{out}"] \arrow[r, phantom, "\cong"] & \mathbf{F}\ T \arrow[l, bend left, "\mathbf{in}"] \end{tikzcd}$$

Two examples so far: $T = \mathbb{N}_0$ and $T = A^\star$

Let us see some more.

Binary trees

Trees with data of type A in their nodes, for example

$$\texttt{data}\ \textbf{BTree} = \textbf{Empty} \mid \textbf{Node}\ (A, (\textbf{BTree}, \textbf{BTree}))$$

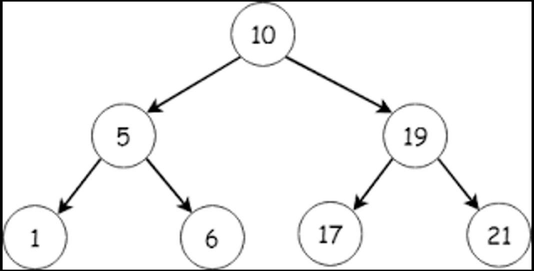
(A assumed pre-defined.)

Binary trees

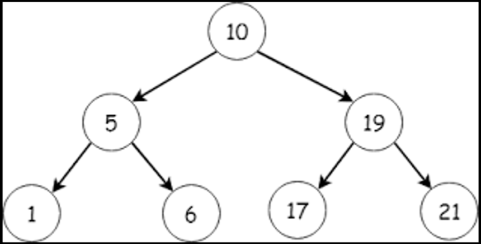
Trees with data of type A in their nodes, for example

$$\texttt{data}\ \mathbf{BTree} = \mathbf{Empty} \mid \mathbf{Node}\ (A, (\mathbf{BTree}, \mathbf{BTree}))$$

(A assumed pre-defined.) For instance

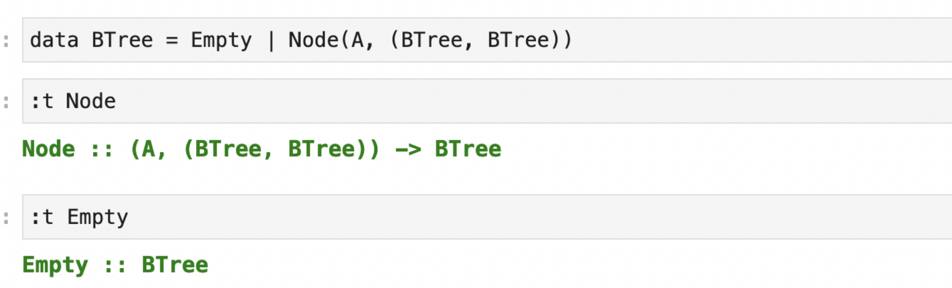


Binary trees



```
Node (10,
    (Node (5,
        (Node (1,
            (Empty,Empty)),
         Node (6,
            (Empty,Empty)))),
     Node (19,
         (Node (17,
            (Empty,Empty)),
          Node (21,
            (Empty,Empty))))))
```

Binary trees



```
data BTree = Empty | Node(A, (BTree, BTree))
```

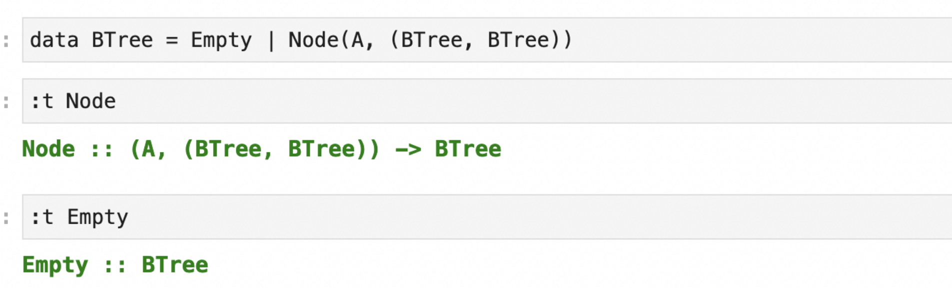
```
:t Node
```

```
Node :: (A, (BTree, BTree)) -> BTree
```

```
:t Empty
```

```
Empty :: BTree
```

Binary trees



```
data BTree = Empty | Node(A, (BTree, BTree))
```

```
:t Node
```

```
Node :: (A, (BTree, BTree)) -> BTree
```

```
:t Empty
```

```
Empty :: BTree
```

$$\mathsf{Node} : A \times (\mathsf{BTree} \times \mathsf{BTree}) \to \mathsf{BTree}$$

$$\underline{\mathsf{Empty}} : 1 \to \mathsf{BTree}$$

Binary trees

$$\mathsf{in} = [\underline{\mathsf{Empty}}, \mathsf{Node}]$$

Binary trees

$$\mathsf{in} = [\underline{\mathsf{Empty}}, \mathsf{Node}]$$

$$\begin{tikzcd} \mathsf{BTree} \arrow[r, bend left, "\mathsf{out}"] \arrow[r, phantom, "\cong"] & \underbrace{1 + A \times (\mathsf{BTree} \times \mathsf{BTree})}_{\mathsf{F\ BTree}} \arrow[l, bend left, "\mathsf{in}"] \end{tikzcd}$$

$$\mathtt{data}\ \mathsf{BTree} = \mathsf{Empty} \mid \mathsf{Node}\ (A, (\mathsf{BTree}, \mathsf{BTree}))$$

Binary trees

$$\begin{tikzcd}
\mathbf{BTree} \arrow[r, phantom, "\cong"] \arrow[r, bend left, "\mathbf{out}"] & \underbrace{1 + A \times (\mathbf{BTree} \times \mathbf{BTree})}_{\mathbf{F\ BTree}} \arrow[l, bend left, "\mathbf{in}"]
\end{tikzcd}$$

$$T = \mathbf{BTree}$$

$$\begin{cases} \mathbf{F}\ X = 1 + A \times (X \times X) \\ \mathbf{F}\ f = id + id \times (f \times f) \end{cases}$$

$$\mathbf{in} = [\underline{\mathbf{Empty}}, \mathbf{Node}]$$

Catamorphisms of binary trees

$$\begin{tikzcd}
\mathbf{BTree} \arrow[r, bend left, "\mathbf{out}"] \arrow[r, phantom, "\cong"] \arrow[d, "(\!| g |\!)"'] & \underbrace{1 + A \times (\mathbf{BTree} \times \mathbf{BTree})}_{\mathbf{F}\ \mathbf{BTree}} \arrow[l, bend left, "\mathbf{in}"] \arrow[d, "\underbrace{id + id \times (\!| g |\!) \times (\!| g |\!)}_{\mathbf{F}\ (\!| g |\!)}"] \\
B & \underbrace{1 + A \times (B \times B)}_{\mathbf{F}\ B} \arrow[l, "g"']
\end{tikzcd}$$

Catamorphisms of binary trees

In Haskell:

```
cataBTree g = g . (recBTree (cataBTree g)) . outBTree

outBTree Empty               = Left ()
outBTree (Node (a,(t1,t2))) = Right(a,(t1,t2))

recBTree f = id -|- id >< (f >< f)

inBTree = either (const Empty) Node
```

Catamorphisms of binary trees

Example: function

$$f : \textbf{BTree} \rightarrow \mathbb{N}_0$$

that computes the **size** of the tree (total number of nodes).

Catamorphisms of binary trees

Example: function

$$f : \mathbf{BTree} \to \mathbb{N}_0$$

that computes the **size** of the tree (total number of nodes).

It suffices to focus on “gene” g in

$$f = (\!| g |\!)$$

where

$$\mathbb{N}_0 \xleftarrow{g} 1 + A \times (\mathbb{N}_0 \times \mathbb{N}_0)$$

Catamorphisms of binary trees

$$\begin{tikzcd}
\mathbb{N}_0 & 1 + A \times (\mathbb{N}_0 \times \mathbb{N}_0) \arrow[l, "g"']
\end{tikzcd}$$

Catamorphisms of binary trees

$$\begin{tikzcd}
\mathbb{N}_0 & 1 + A \times (\mathbb{N}_0 \times \mathbb{N}_0) \arrow[l, "g = {[g_1\,, g_2]}"']
\end{tikzcd}$$

Catamorphisms of binary trees

empty
tree

$$\begin{tikzcd} \mathbb{N}_0 & 1 + A \times (\mathbb{N}_0 \times \mathbb{N}_0) \arrow[l, "{g = [g_1 , g_2]}"'] \end{tikzcd}$$

$$g_1 \; \_ = 0$$

Catamorphisms of binary trees

empty tree

$$\begin{tikzcd} \mathbb{N}_0 & 1 + A \times (\mathbb{N}_0 \times \mathbb{N}_0) \arrow[l, "{g = [g_1 , g_2]}"'] \end{tikzcd}$$

$$g_1 \ \_ = 0$$

total size

size of left sub-tree

size of right sub-tree

Catamorphisms of binary trees

empty
tree

$$\mathbb{N}_0 \xleftarrow{g = [g_1 , g_2]} 1 + A \times (\mathbb{N}_0 \times \mathbb{N}_0)$$

$$g_1 \ \_ = 0$$

$$g_2 \ (a, (l, r)) = 1 + l + r$$

total

size

size of
left
sub-tree

size of
right
sub-tree

Catamorphisms of binary trees

Summing up,

$$\begin{aligned} & f = (\![g_1, g_2]\!) \ \textbf{where} \\ & \quad g_1 \ \_ = 0 \\ & \quad g_2 \ (a, (l, r)) = 1 + l + r \end{aligned}$$

Catamorphisms of binary trees

Summing up,

$$
\begin{aligned}
&f = (\!|\, [g_1\,, g_2] \,|\!) \textbf{ where} \\
&\quad g_1 \;\_ = 0 \\
&\quad g_2\;(a, (l, r)) = 1 + l + r
\end{aligned}
$$

Haskell:

```
size = cataBTree(either g1 g2) where
   g1 _ = 0
   g2(a,(l,r)) = 1 + l + r
```

Inductive type repertoire (BTree)

(Assuming A predefined, for the moment).

Trees whose data of type A are stored in their **nodes**:

$$\mathbf{T} = \mathbf{BTree} \quad \text{where} \quad \left\{ \begin{array}{l} \mathsf{F}\ X = 1 + A \times X^2 \\ \mathsf{F}\ f = id + id \times f^2 \\ \mathbf{in} = [\underline{\mathbf{Empty}}, \mathbf{Node}] \end{array} \right.$$

Haskell:

```
data BTree a = Empty | Node(a, (BTree a, BTree a))
```

Inductive type repertoire (LTree)

Trees with data in their **leaves**:

$$\mathsf{T} = \mathsf{LTree} \quad \text{where} \quad \left\{ \begin{array}{l} \mathsf{F}\ X = A + X^2 \\ \mathsf{F}\ f = id + f^2 \\ \mathsf{in} = [\mathsf{Leaf}, \mathsf{Fork}] \end{array} \right.$$

Haskell:

```
data LTree a = Leaf a | Fork (LTree a, LTree a)
```

Inductive type repertoire (**FTree**)

(Assuming A and B predefined, for the moment).

Trees bearing data in **both** leaves and nodes:

$$\mathbf{T} = \mathbf{FTree} \quad \text{where} \quad \left\{ \begin{array}{l} \mathbf{F}\ X = B + A \times X^2 \\ \mathbf{F}\ f = id + id \times f^2 \\ \mathbf{in} = [\mathbf{Unit}, \mathbf{Comp}] \end{array} \right.$$

Haskell:

```
data FTree b a = Unit b | Comp (a,(FTree b a, FTree b
a))
```

Inductive type repertoire (Expr)

(Assume V and O predefined, for the moment).

Expression trees:

$$\mathbf{T} = \textbf{Expr} \quad \text{where} \quad \left\{ \begin{array}{l} \mathbf{F}\ X = V + O \times X^{\star} \\ \mathbf{F}\ f = id + id \times map\ f \\ \textbf{in} = [\textbf{Var}, \textbf{Term}] \end{array} \right.$$

Haskell:

```
data Expr v o = Var v | Term (o,[Expr v o])
```

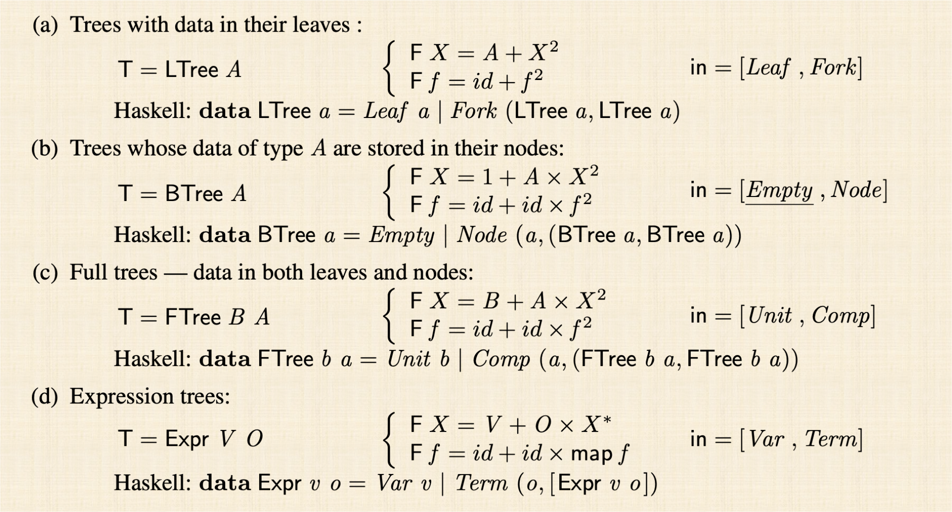
Inductive type repertoire (Expr)

```
<html>
  <head>
    <title>Title of the document</title>
  </head>
  <body>

    <h1>This is a heading</h1>
    <p>This is a paragraph.</p>

  </body>
</html>
```

Inductive type repertoire (Expr)

```
Term "html" [
   Term "head" [
     Term "title"
        [Var "Title of the document" ]
   ],
     Term "body" [
        Term "h1" [
           Var "This is a heading" ],
        Term "p" [
           Var "This is a paragraph." ]
     ]
]
```

Inductive types



(a) Trees with data in their leaves :

$$\mathsf{T} = \mathsf{LTree}\ A \qquad \begin{cases} \mathsf{F}\ X = A + X^2 \\ \mathsf{F}\ f = id + f^2 \end{cases} \qquad \mathsf{in} = [\mathit{Leaf}\ , \mathit{Fork}]$$

Haskell: $\textbf{data}\ \mathsf{LTree}\ a = \mathit{Leaf}\ \ a \mid \mathit{Fork}\ (\mathsf{LTree}\ a, \mathsf{LTree}\ a)$

(b) Trees whose data of type A are stored in their nodes:

$$\mathsf{T} = \mathsf{BTree}\ A \qquad \begin{cases} \mathsf{F}\ X = 1 + A \times X^2 \\ \mathsf{F}\ f = id + id \times f^2 \end{cases} \qquad \mathsf{in} = [\underline{\mathit{Empty}}\ , \mathit{Node}]$$

Haskell: $\textbf{data}\ \mathsf{BTree}\ a = \mathit{Empty} \mid \mathit{Node}\ (a, (\mathsf{BTree}\ a, \mathsf{BTree}\ a))$

(c) Full trees — data in both leaves and nodes:

$$\mathsf{T} = \mathsf{FTree}\ B\ \ A \qquad \begin{cases} \mathsf{F}\ X = B + A \times X^2 \\ \mathsf{F}\ f = id + id \times f^2 \end{cases} \qquad \mathsf{in} = [\mathit{Unit}\ , \mathit{Comp}]$$

Haskell: $\textbf{data}\ \mathsf{FTree}\ b\ \ a = \mathit{Unit}\ \ b \mid \mathit{Comp}\ (a, (\mathsf{FTree}\ b\ \ a, \mathsf{FTree}\ b\ \ a))$

(d) Expression trees:

$$\mathsf{T} = \mathsf{Expr}\ V\ \ O \qquad \begin{cases} \mathsf{F}\ X = V + O \times X^* \\ \mathsf{F}\ f = id + id \times \mathsf{map}\ f \end{cases} \qquad \mathsf{in} = [\mathit{Var}\ , \mathit{Term}]$$

Haskell: $\textbf{data}\ \mathsf{Expr}\ v\ \ o = \mathit{Var}\ \ v \mid \mathit{Term}\ (o, [\mathsf{Expr}\ v\ \ o])$

Inductive parametric types

Let us now show how recursive types become **parametric**.

Inductive parametric types

Let us now show how recursive types become **parametric**.

That is, how they become **functors**.

Inductive parametric types

Let us now show how recursive types become **parametric**.

That is, how they become **functors**.

It all amounts to **generalize** *map* (lists) to other recursive types.

Recall...

Functor **F**:

$$\begin{tikzcd}
A \arrow[r, dotted, no head] \arrow[d, "f"'] & \mathbf{F}\ A \arrow[d, "\mathbf{F}\ f"] \\
B \arrow[r, dotted, no head] & \mathbf{F}\ B
\end{tikzcd}$$

Recall...

Functor $\mathbf{F}$:

$$\begin{tikzcd}
A \arrow[r, dash, dotted] \arrow[d, "f"'] & \mathbf{F}\ A \arrow[d, "\mathbf{F}\ f"] \\
B \arrow[r, dash, dotted] & \mathbf{F}\ B
\end{tikzcd}$$

$$\begin{cases} \mathbf{F}\ id = id \\ \mathbf{F}\ (g \cdot f) = (\mathbf{F}\ g) \cdot (\mathbf{F}\ f) \end{cases}$$

$map\ f$

Functor $\mathbf{F}\ A = A^\star$:

$$\begin{tikzcd}
A \arrow[r, dotted, no head] \arrow[d, "f"'] & A^\star \arrow[d, "f^\star = map\ f"] \\
B \arrow[r, dotted, no head] & B^\star
\end{tikzcd}$$

$map\ f$

Functor $\mathbf{F}\ A = A^{\star}$:

$$\begin{tikzcd}
A \arrow[r, dash, dotted] \arrow[d, "f"'] & A^{\star} \arrow[d, "f^{\star} = map\ f"] \\
B \arrow[r, dash, dotted] & B^{\star}
\end{tikzcd}$$

$$\begin{cases} map\ id = id \\ map\ (g \cdot f) = (map\ g) \cdot (map\ f) \end{cases}$$

How do we **prove** this? How do we **generalize** this?

$map\ f$

From *specification*

$$map\ f\ [\,] = [\,]$$
$$map\ f\ [a] = [f\ a]$$
$$map\ f\ (x +\!\!+\ y) = map\ f\ x +\!\!+\ map\ f\ y$$

easy to obtain (as before):

$$map\ f\ [\,] = [\,]$$
$$map\ f\ (h : t) = f\ h : map\ f\ t$$

map f

The *implementation*

$$
\begin{aligned}
&map\ f\ [\,] = [\,]\\
&map\ f\ (h : t) = f\ h : map\ f\ t
\end{aligned}
$$

$map\ f$

The *implementation*

$$\begin{aligned} map\ f\ [\,] &= [\,] \\ map\ f\ (h : t) &= f\ h : map\ f\ t \end{aligned}$$

is catamorphism:

$$map\ f = (\![[nil, g_2]]\!)\ \textbf{where}$$
$$\quad g_2\ (h, x) = f\ h : x$$

$map\ f$

The *implementation*

$$\begin{aligned} &map\ f\ [\,] = [\,]\\ &map\ f\ (h : t) = f\ h : map\ f\ t \end{aligned}$$

is catamorphism:

$$\begin{aligned} &map\ f = (\!|[nil, g_2]|\!)\ \textbf{where}\\ &\quad g_2\ (h, x) = f\ h : x \end{aligned}$$

Pointfree equivalent: $map\ f = (\!|[nil, cons] \cdot (id + f \times id)|\!)$

$map\ f$

For instance, for $sq\ x = x^2$

$$map\ sq\ [1, 7, 8, 9, 3, 2] = [1, 49, 64, 81, 9, 4]$$

$map\ f$

For instance, for $sq\ x = x^2$

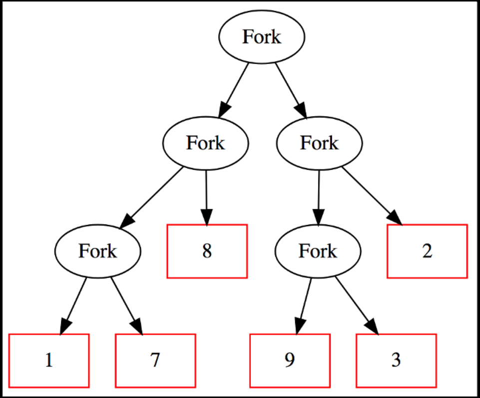
$$map\ sq\ [1, 7, 8, 9, 3, 2] = [1, 49, 64, 81, 9, 4]$$

But still we haven't proved

$$map\ id = id$$
$$map\ (f \cdot g) = (map\ f) \cdot (map\ g)$$

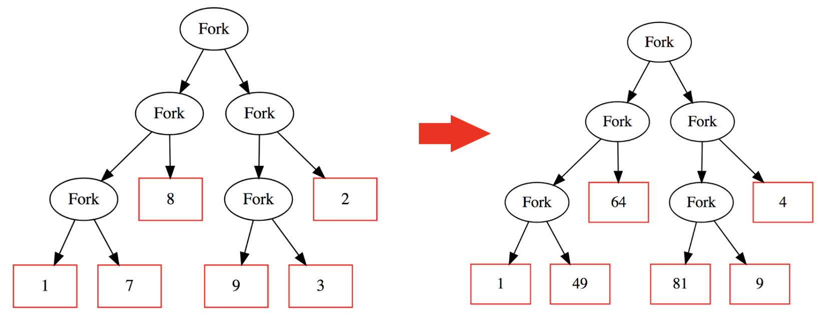
(Coming up soon.)

Inductive parametric types



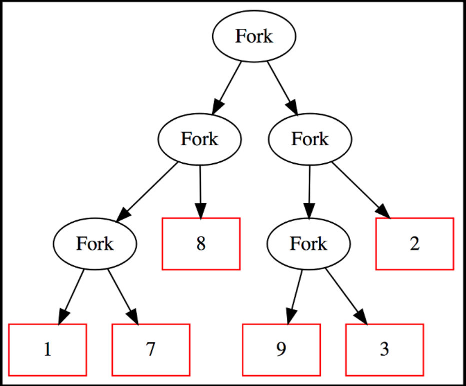
$\texttt{data}\ \mathbf{LTree} = \mathbf{Leaf}\ \mathit{Int} \mid \mathbf{Fork}\ (\mathbf{LTree}, \mathbf{LTree})$

Inductive parametric types



Inductive parametric types

$t =$ **Fork**
  (**Fork**
    (**Fork** (**Leaf** 1, **Leaf** 7),
      Leaf 8),
  Fork
      (**Fork** (**Leaf** 9, **Leaf** 3),
        Leaf 2))



Inductive parametric types

$$k\ (\textbf{Leaf}\ a) = \textbf{Leaf}\ a^2$$
$$k\ (\textbf{Fork}\ (l, r)) = \textbf{Fork}\ (k\ l, k\ r)$$

$$
\begin{aligned}
k\ t = {} & \textbf{Fork} \\
& \quad (\textbf{Fork} \\
& \qquad (\textbf{Fork}\ (\textbf{Leaf}\ 1, \textbf{Leaf}\ 49), \\
& \qquad \textbf{Leaf}\ 64), \\
& \quad \textbf{Fork} \\
& \qquad (\textbf{Fork}\ (\textbf{Leaf}\ 81, \textbf{Leaf}\ 9), \\
& \qquad \textbf{Leaf}\ 4))
\end{aligned}
$$

Inductive parametric types

$$k\ (\mathbf{Leaf}\ a) = \mathbf{Leaf}\ a^2$$
$$k\ (\mathbf{Fork}\ (l, r)) = \mathbf{Fork}\ (k\ l, k\ r)$$

Inductive parametric types

$$\begin{aligned} k\ (\mathbf{Leaf}\ a) &= \mathbf{Leaf}\ a^2 \\ k\ (\mathbf{Fork}\ (l, r)) &= \mathbf{Fork}\ (k\ l, k\ r) \end{aligned}$$

By analogy with *map f*:

$$\begin{aligned} k\ f\ (\mathbf{Leaf}\ a) &= \mathbf{Leaf}\ (f\ a) \\ k\ f\ (\mathbf{Fork}\ (l, r)) &= \mathbf{Fork}\ (k\ f\ l, k\ f\ r) \end{aligned}$$

Inductive parametric types

$$map\ id = id$$

$$map\ (f \cdot g) = (map\ f) \cdot (map\ g)$$

Inductive parametric types

$$map\ id = id$$
$$map\ (f \cdot g) = (map\ f) \cdot (map\ g)$$

$$k\ id = id$$
$$k\ (f \cdot g) = (k\ f) \cdot (k\ g)$$

?

Inductive parametric types

$$\begin{cases} k\ f\ (\textbf{Leaf}\ a) = \textbf{Leaf}\ (f\ \ a) \\ k\ f\ (\textbf{Fork}\ (l, r)) = \textbf{Fork}\ (k\ f\ l, k\ f\ r) \end{cases}$$

Inductive parametric types

$$\left\{\begin{array}{l} k\ f\ (\mathbf{Leaf}\ a) = \mathbf{Leaf}\ (f\ a) \\ k\ f\ (\mathbf{Fork}\ (l, r)) = \mathbf{Fork}\ (k\ f\ l, k\ f\ r) \end{array}\right.$$

$$\Leftrightarrow \qquad \{\ \text{go pointfree}\ \}$$

$$\left\{\begin{array}{l} k\ f \cdot \mathbf{Leaf} = \mathbf{Leaf} \cdot f \\ k\ f \cdot \mathbf{Fork} = \mathbf{Fork} \cdot (k\ f \times k\ f) \end{array}\right.$$

Inductive parametric types

$$
\left\{\begin{array}{l}
k\ f\ (\mathbf{Leaf}\ a) = \mathbf{Leaf}\ (f\ a) \\
k\ f\ (\mathbf{Fork}\ (l, r)) = \mathbf{Fork}\ (k\ f\ l, k\ f\ r)
\end{array}\right.
$$

$$\Leftrightarrow \qquad \{\ \text{go pointfree}\ \}$$

$$
\left\{\begin{array}{l}
k\ f \cdot \mathbf{Leaf} = \mathbf{Leaf} \cdot f \\
k\ f \cdot \mathbf{Fork} = \mathbf{Fork} \cdot (k\ f \times k\ f)
\end{array}\right.
$$

$$\Leftrightarrow \qquad \{\ +\text{-Eq}\ ;\ +\text{-fusion}\ ;\ +\text{-absorption}\ \}$$

Inductive parametric types

$$k\ f \cdot [\mathsf{Leaf}, \mathsf{Fork}] = [\mathsf{Leaf}, \mathsf{Fork}] \cdot (f + k\ f \times k\ f)$$

Inductive parametric types

$$
\begin{aligned}
& k\ f \cdot [\mathsf{Leaf}, \mathsf{Fork}] = [\mathsf{Leaf}, \mathsf{Fork}] \cdot (f + k\ f \times k\ f) \\
\Leftrightarrow \quad & \qquad \{\ [\mathsf{Leaf}, \mathsf{Fork}] = \mathsf{in}\ ;\ \text{functor-}+\ \} \\
& k\ f \cdot \mathsf{in} = \mathsf{in} \cdot (f + id) \cdot (id + k\ f \times k\ f)
\end{aligned}
$$

Inductive parametric types

$$\begin{tikzcd}
Int \arrow[d, "f"'] & \mathbf{LTree} \arrow[d, "k\ f = (\!|\, g\, |\!)"'] & & Int + \mathbf{LTree}^2 \arrow[lll, "\mathbf{in}"'] \arrow[d, "id + (k\ f)^2"] \\
Int & \mathbf{LTree} & & Int + \mathbf{LTree}^2 \arrow[lll, "g"] \arrow[dl, "f + id"] \\
& & Int + \mathbf{LTree}^2 \arrow[ul, "\mathbf{in}"] &
\end{tikzcd}$$

$$k\ f \cdot \mathbf{in} = \mathbf{in} \cdot (f + id) \cdot (id + k\ f \times k\ f)$$

Inductive parametric types

$$\begin{tikzcd}
Int \arrow[d, "f"'] & \mathbf{LTree} \arrow[d, "k\ f = \llparenthesis g \rrparenthesis"'] & & Int + \mathbf{LTree}^2 \arrow[ll, "\mathbf{in}"'] \arrow[d, "id + (k\ f)^2"] \\
Int & \mathbf{LTree} & & Int + \mathbf{LTree}^2 \arrow[ll, "g"] \arrow[dl, "f + id"] \\
 & & Int + \mathbf{LTree}^2 \arrow[ul, "\mathbf{in}"] &
\end{tikzcd}$$

$$k\ f \cdot \mathbf{in} = \mathbf{in} \cdot (f + id) \cdot (id + k\ f \times k\ f)$$

NB: $\mathbf{LTree}^2$ abbreviates $\mathbf{LTree} \times \mathbf{LTree}$

Inductive parametric types

$$\begin{tikzcd}
A \arrow[d, "f"'] & \mathbf{LTree} \arrow[d, "k\ f = (\!| g |\!)"'] & & A + \mathbf{LTree}^2 \arrow[ll, "\mathbf{in}"'] \arrow[d, "id + (k\ f)^2"] \\
B & \mathbf{LTree} & & A + \mathbf{LTree}^2 \arrow[ll, "g"] \arrow[dl, "f + id"] \\
& & B + \mathbf{LTree}^2 \arrow[ul, "\mathbf{in}"] &
\end{tikzcd}$$

Inductive parametric types

$$\begin{tikzcd}
A \arrow[d, "f"'] & \mathbf{LTree} \arrow[d, "k\ f = (\!| g |\!)"'] & & A + \mathbf{LTree}^2 \arrow[ll, "\mathbf{in}"'] \arrow[d, "id + (k\ f)^2"] \\
B & \mathbf{LTree} & & A + \mathbf{LTree}^2 \arrow[ll, "g"] \arrow[dl, "f + id"] \\
 & & B + \mathbf{LTree}^2 \arrow[ul, "\mathbf{in}"] &
\end{tikzcd}$$

!!

Inductive parametric types

$$\begin{tikzcd}
A \arrow[d, "f"'] & \mathbf{LTree}\ A \arrow[d, "k\ f = (\!|\, g\, |\!)"'] & & A + (\mathbf{LTree}\ A)^2 \arrow[ll, "\mathbf{in}_A"'] \arrow[d, "id + (k\ f)^2"] \\
B & \mathbf{LTree}\ B & & A + (\mathbf{LTree}\ B)^2 \arrow[ll, "g"] \arrow[dl, "f + id"] \\
 & & B + (\mathbf{LTree}\ B)^2 \arrow[ul, "\mathbf{in}_B"] &
\end{tikzcd}$$

Inductive parametric types

$$\begin{tikzcd}
A \arrow[d, "f"'] & \mathbf{LTree}\ A \arrow[d, "k\ f = (\!|\, g\, |\!)"'] & & A + (\mathbf{LTree}\ A)^2 \arrow[ll, "\mathbf{in}_A"'] \arrow[d, "id + (k\ f)^2"] \\
B & \mathbf{LTree}\ B & & A + (\mathbf{LTree}\ B)^2 \arrow[ll, "g"] \arrow[dl, "f + id"] \\
& & B + (\mathbf{LTree}\ B)^2 \arrow[ul, "\mathbf{in}_B"] &
\end{tikzcd}$$

$$k\ f = (\!|\, \mathbf{in} \cdot (f + id)\, |\!)$$

Follow-up: `cp2324s12.pdf`