

Games with new resources

Nikos Tzevelekos

Oxford University Computing Laboratory

What this talk is about

Generation of **new resources** (*names*)
is a pervasive feature in computation

We present games and related semantic
formalisms for new resources

Ingredients

Game Semantics

Nominal Sets

Fresh-Register Automata

Game Semantics

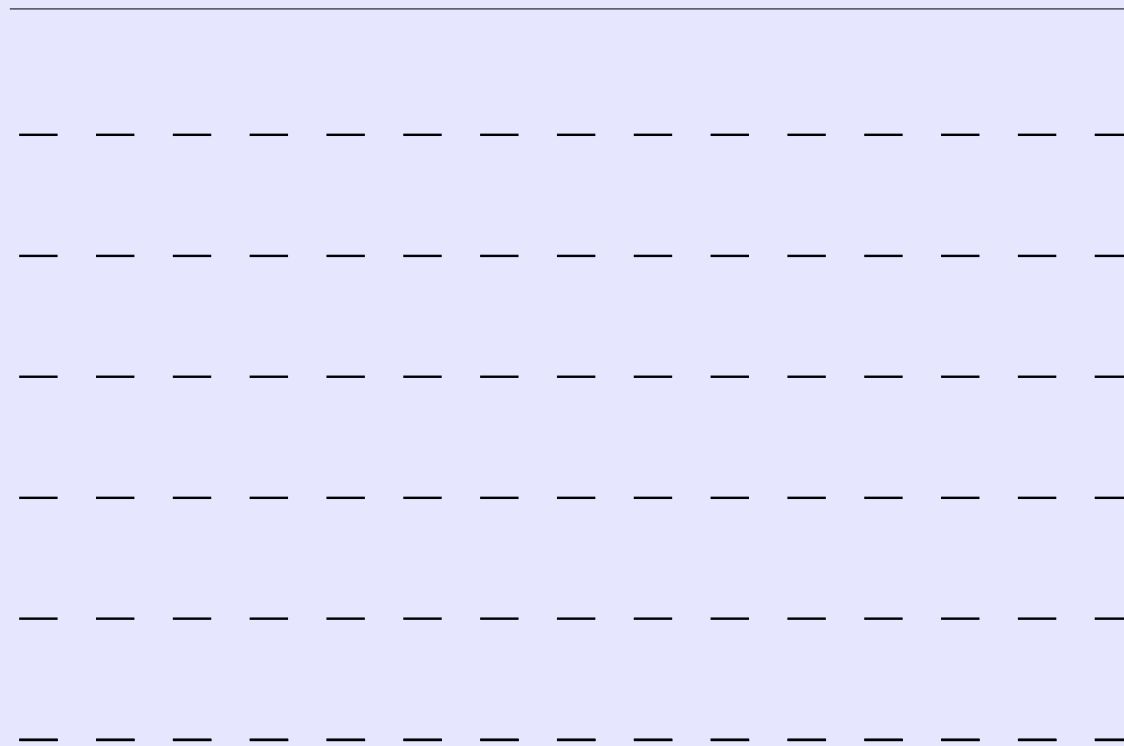
Game Semantics

- Computation is modelled as a 2-player game between:
 - *Opponent* (the environment)
 - *Proponent* (the program)

Example

$\vdash \lambda x. x+1 : \text{int} \rightarrow \text{int}$

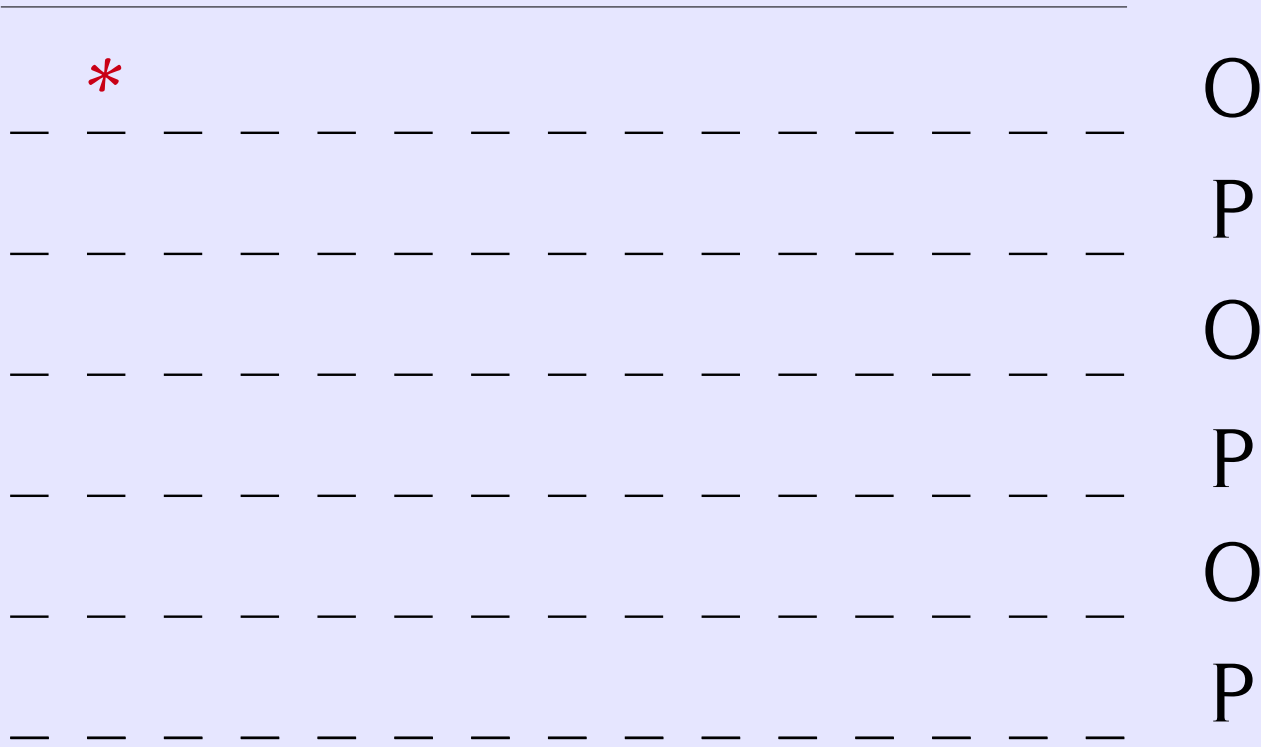
$1 \longrightarrow \text{Int} \rightarrow \text{Int}$



Example

$\vdash \lambda x. x+1 : \text{int} \rightarrow \text{int}$

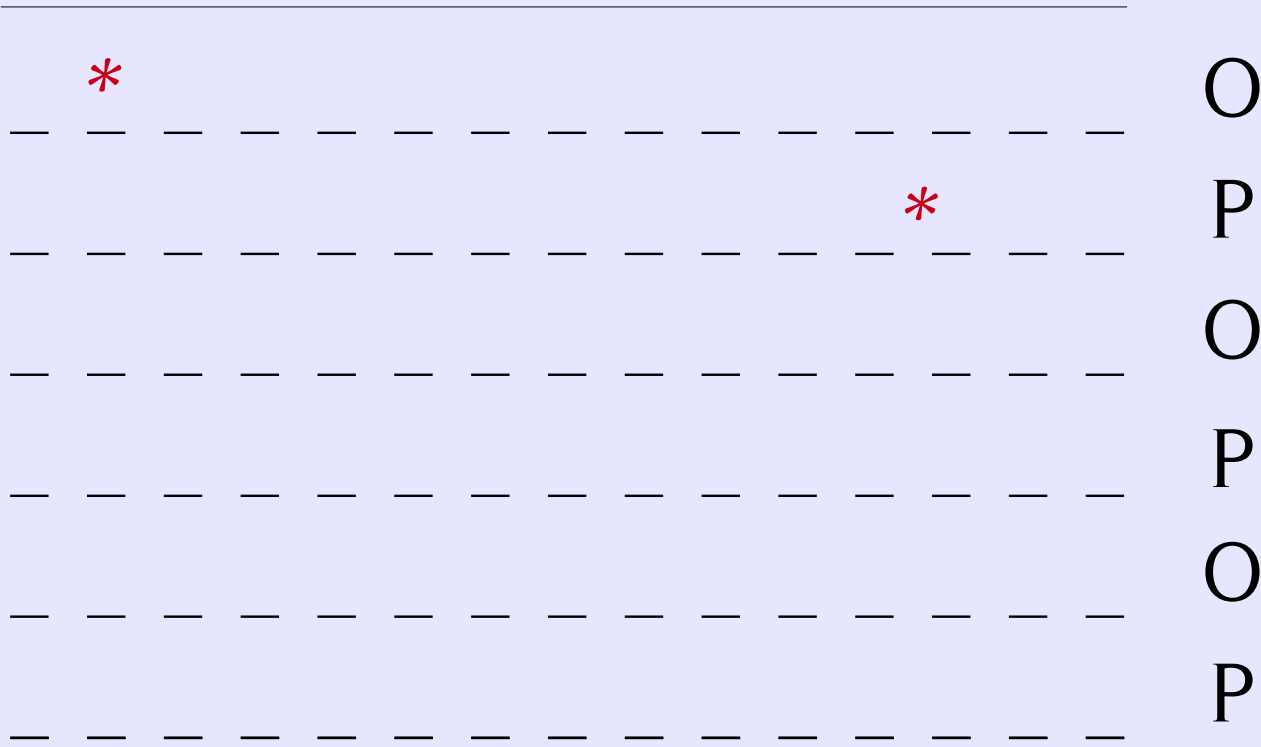
$1 \longrightarrow \text{Int} \rightarrow \text{Int}$



Example

$\vdash \lambda x. x+1 : \text{int} \rightarrow \text{int}$

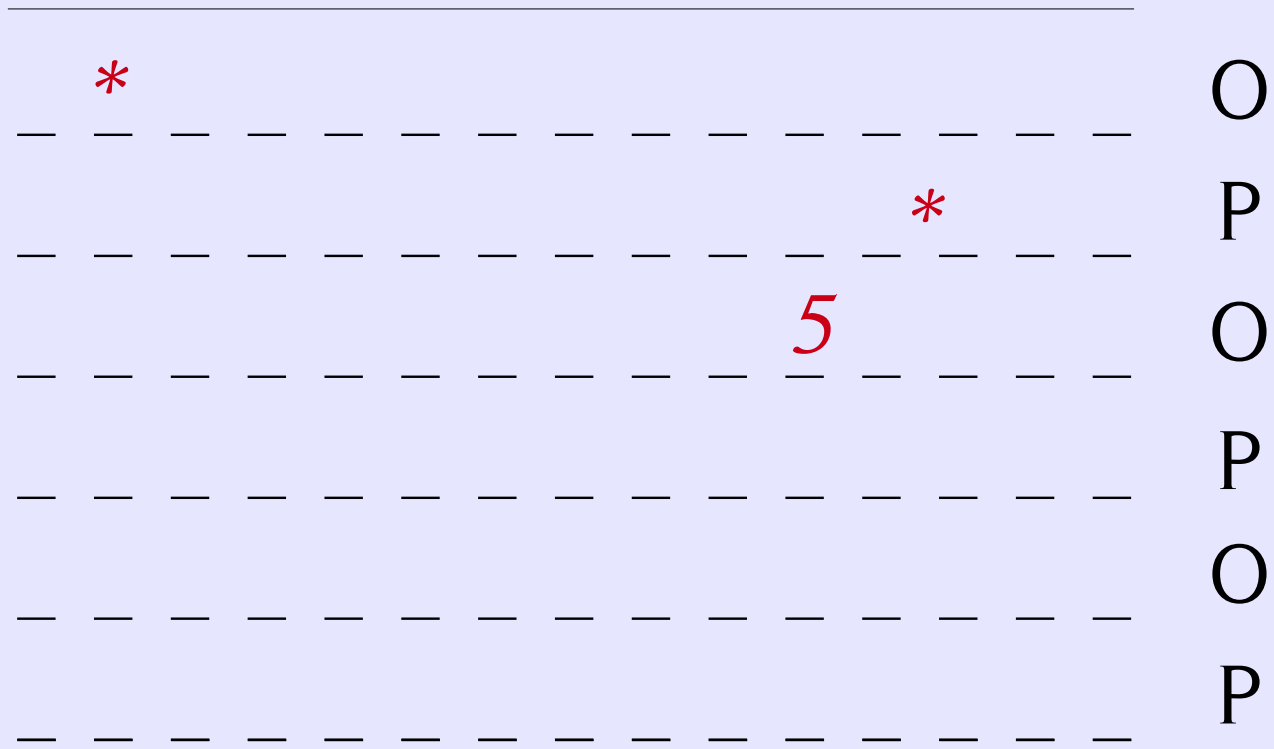
$1 \longrightarrow \text{Int} \rightarrow \text{Int}$



Example

$\vdash \lambda x. x+1 : \text{int} \rightarrow \text{int}$

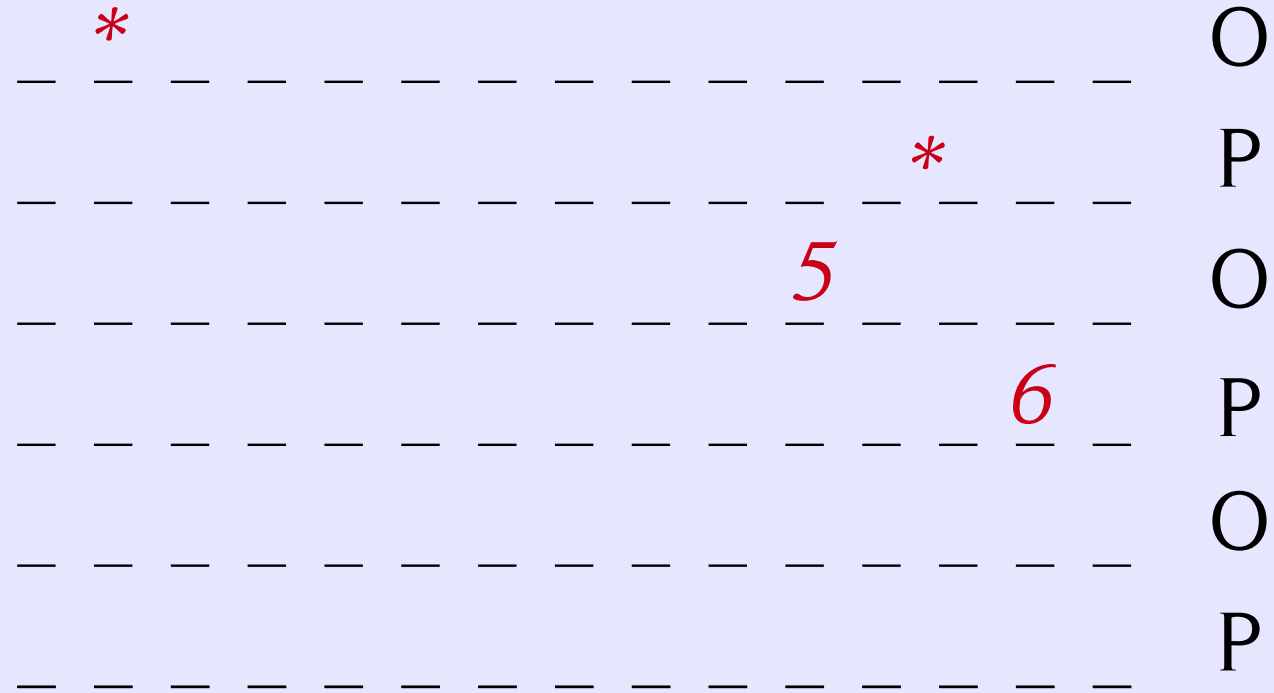
$1 \longrightarrow \text{Int} \rightarrow \text{Int}$



Example

$\vdash \lambda x. x+1 : \text{int} \rightarrow \text{int}$

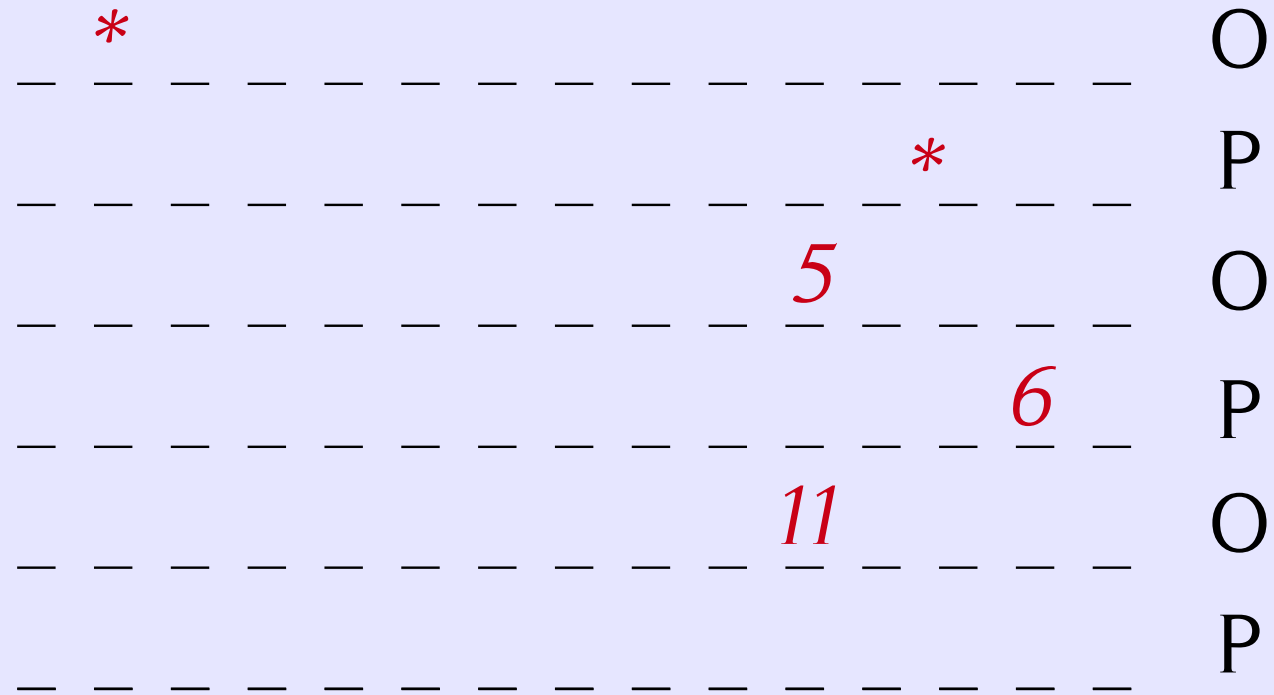
$1 \longrightarrow \text{Int} \rightarrow \text{Int}$



Example

$\vdash \lambda x. x+1 : \text{int} \rightarrow \text{int}$

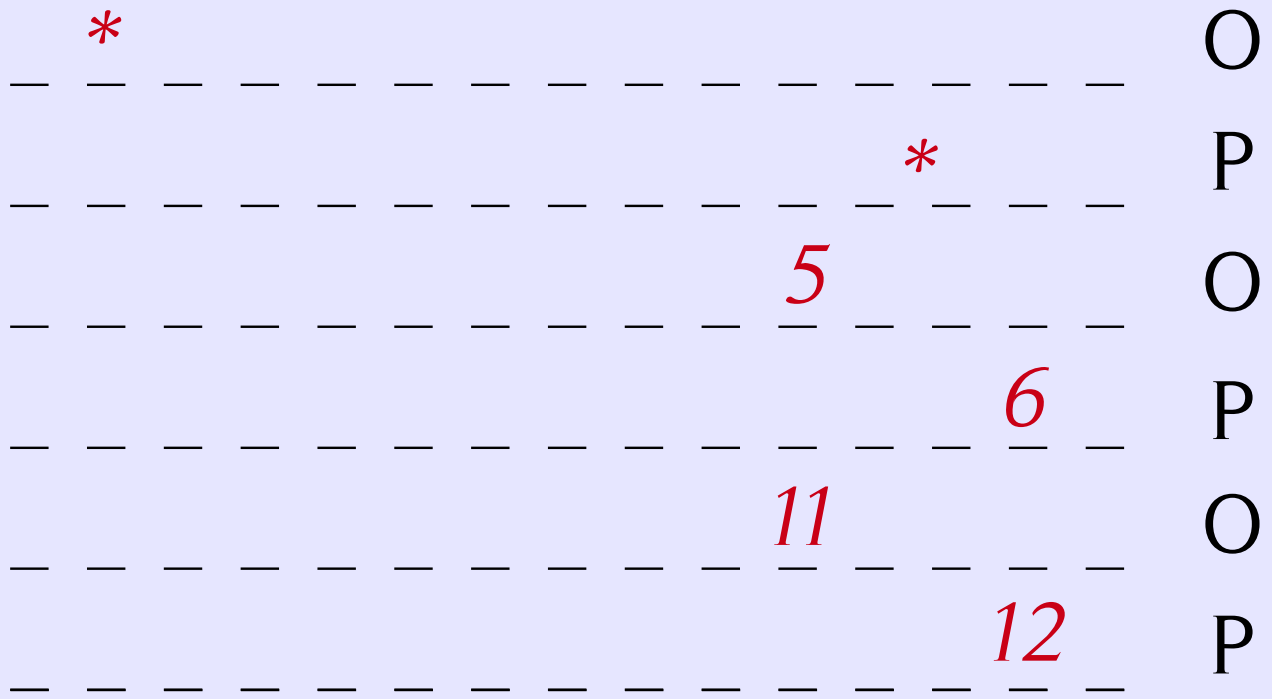
$1 \longrightarrow \text{Int} \rightarrow \text{Int}$



Example

$\vdash \lambda x. x+1 : \text{int} \rightarrow \text{int}$

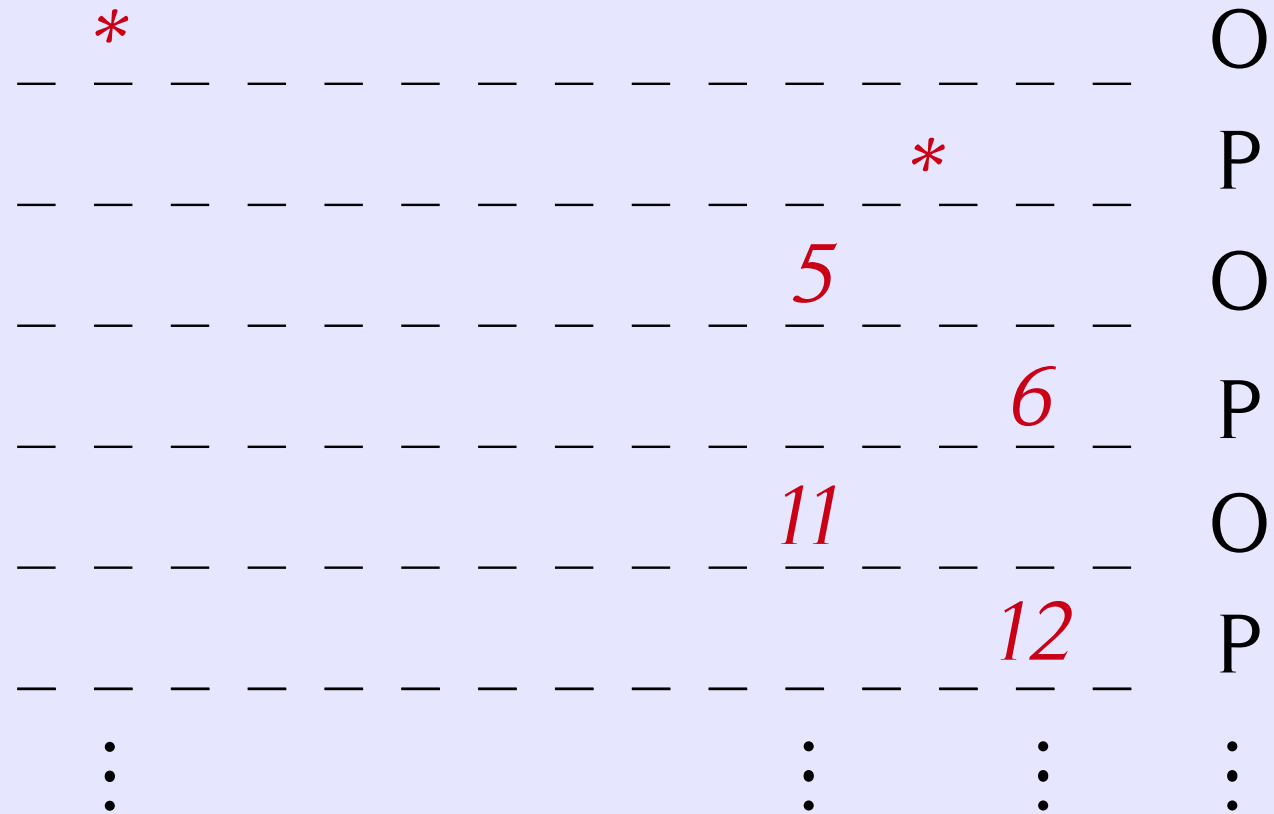
$1 \longrightarrow \text{Int} \rightarrow \text{Int}$



Example

$\vdash \lambda x. x+1 : \text{int} \rightarrow \text{int}$

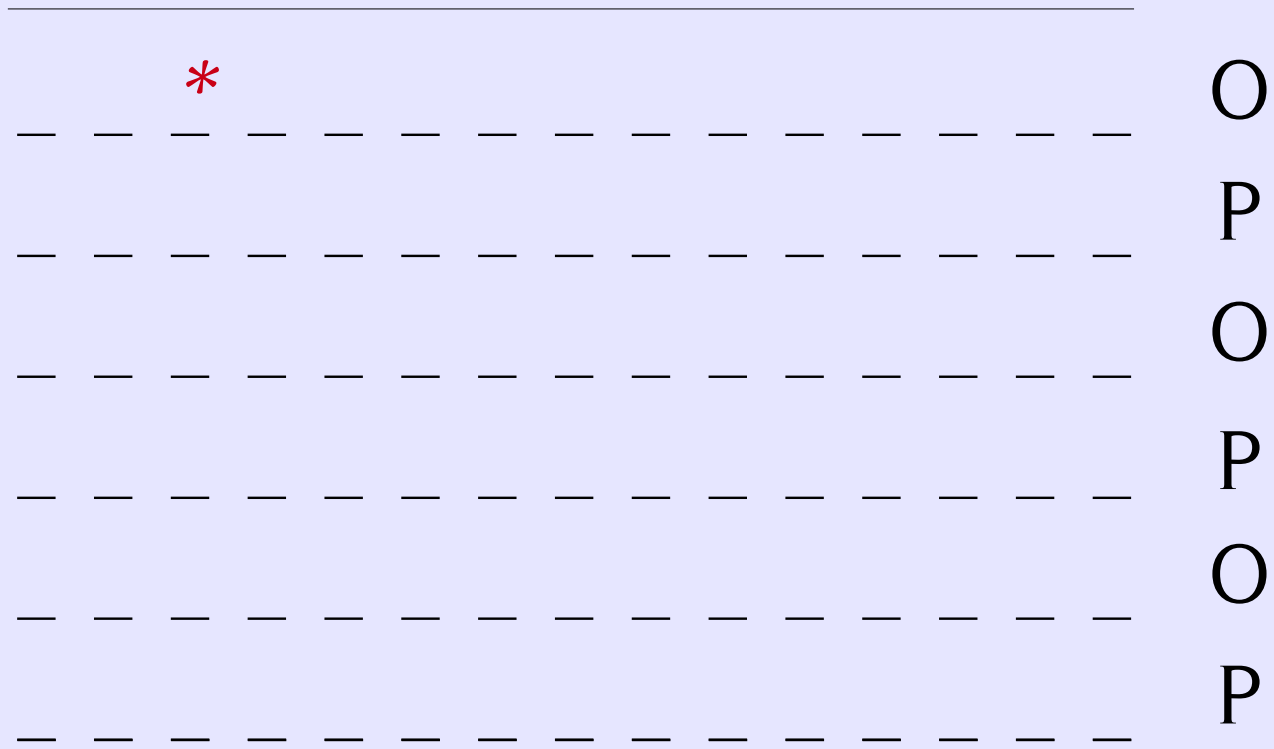
$1 \longrightarrow \text{Int} \rightarrow \text{Int}$



Example

$f : \text{int} \rightarrow \text{int} \vdash \lambda x. f(x)+1 : \text{int} \rightarrow \text{int}$

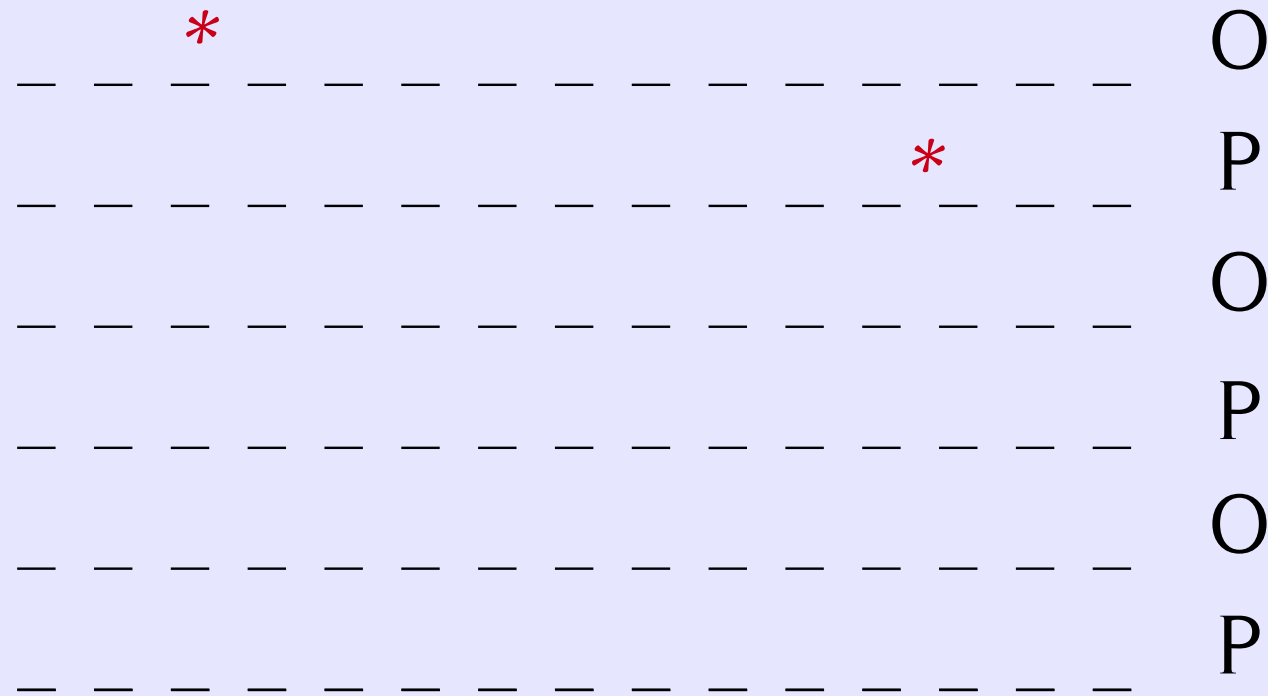
$\text{Int} \rightarrow \text{Int} \longrightarrow \text{Int} \rightarrow \text{Int}$



Example

$f : \text{int} \rightarrow \text{int} \vdash \lambda x. f(x)+1 : \text{int} \rightarrow \text{int}$

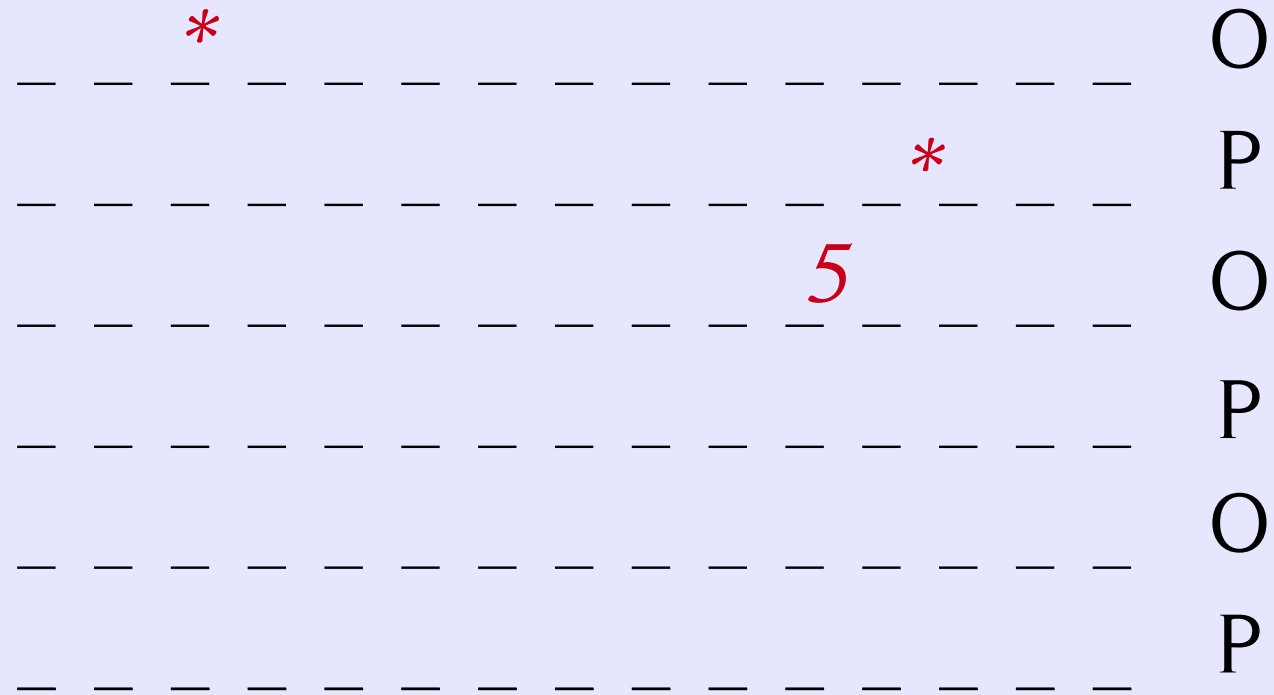
$\text{Int} \rightarrow \text{Int} \longrightarrow \text{Int} \rightarrow \text{Int}$



Example

$f : \text{int} \rightarrow \text{int} \vdash \lambda x. f(x)+1 : \text{int} \rightarrow \text{int}$

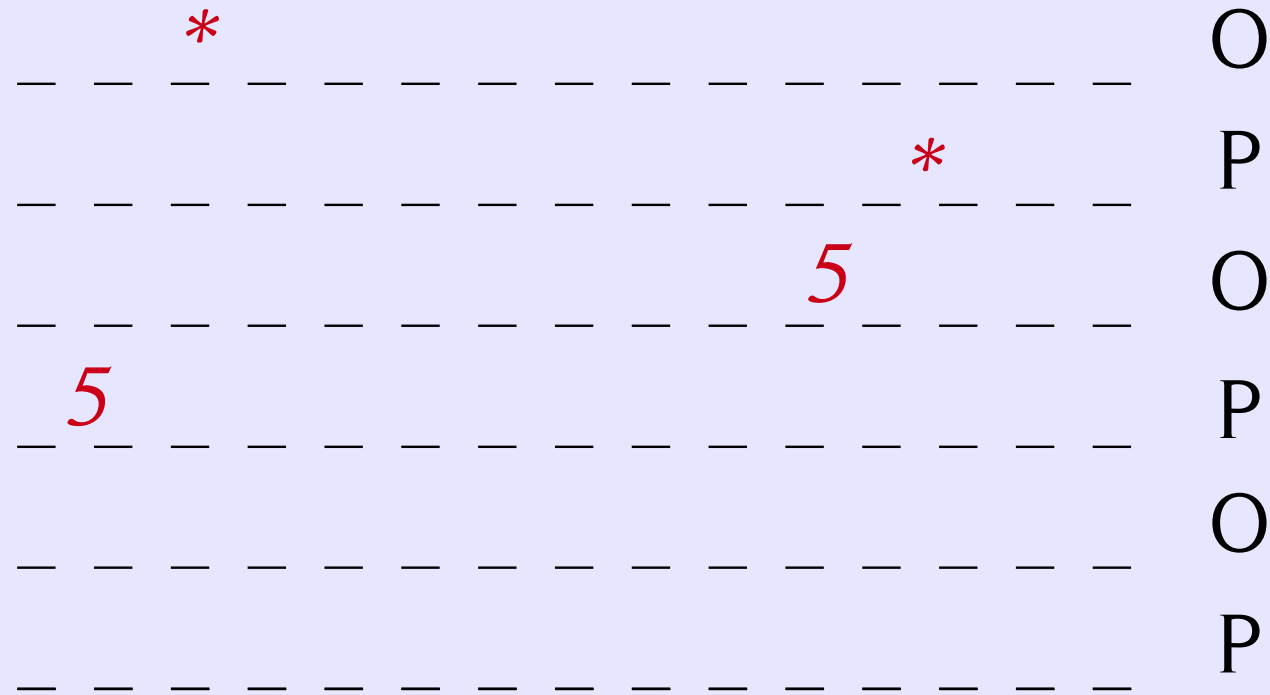
$\text{Int} \rightarrow \text{Int} \longrightarrow \text{Int} \rightarrow \text{Int}$



Example

$f : \text{int} \rightarrow \text{int} \vdash \lambda x. f(x)+1 : \text{int} \rightarrow \text{int}$

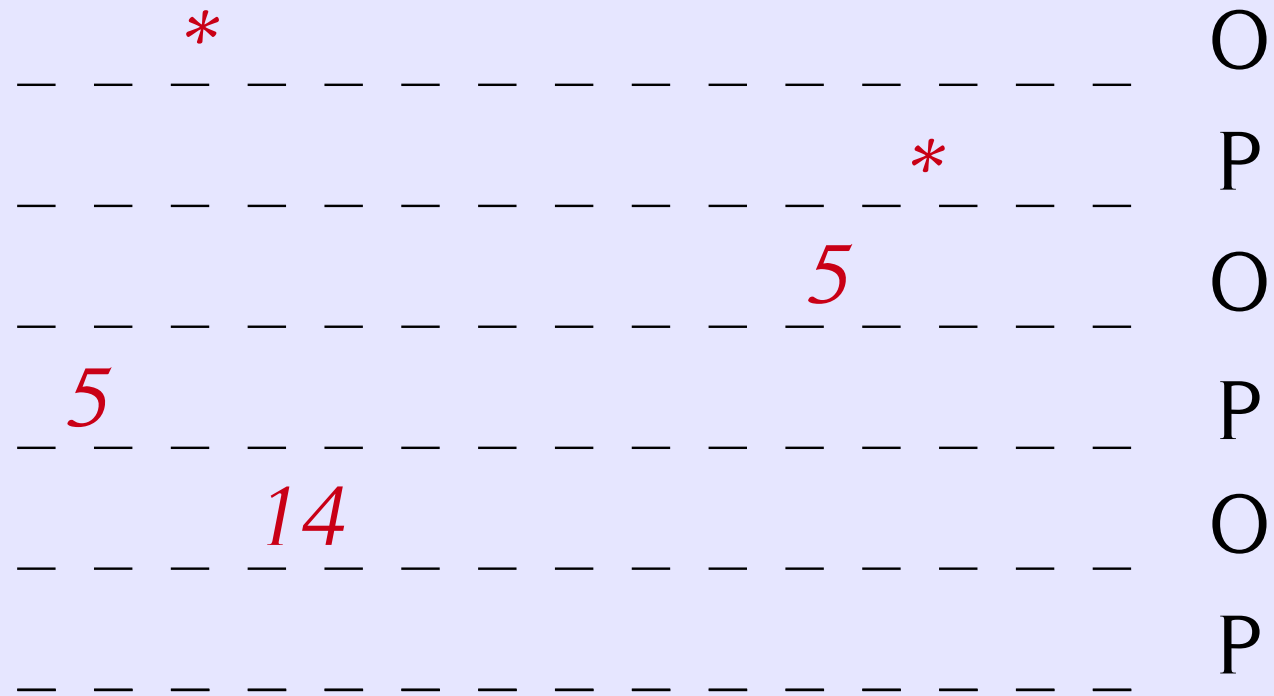
$\text{Int} \rightarrow \text{Int} \longrightarrow \text{Int} \rightarrow \text{Int}$



Example

$f : \text{int} \rightarrow \text{int} \vdash \lambda x. f(x)+1 : \text{int} \rightarrow \text{int}$

$\text{Int} \rightarrow \text{Int} \longrightarrow \text{Int} \rightarrow \text{Int}$



Example

$f : \text{int} \rightarrow \text{int} \vdash \lambda x. f(x)+1 : \text{int} \rightarrow \text{int}$

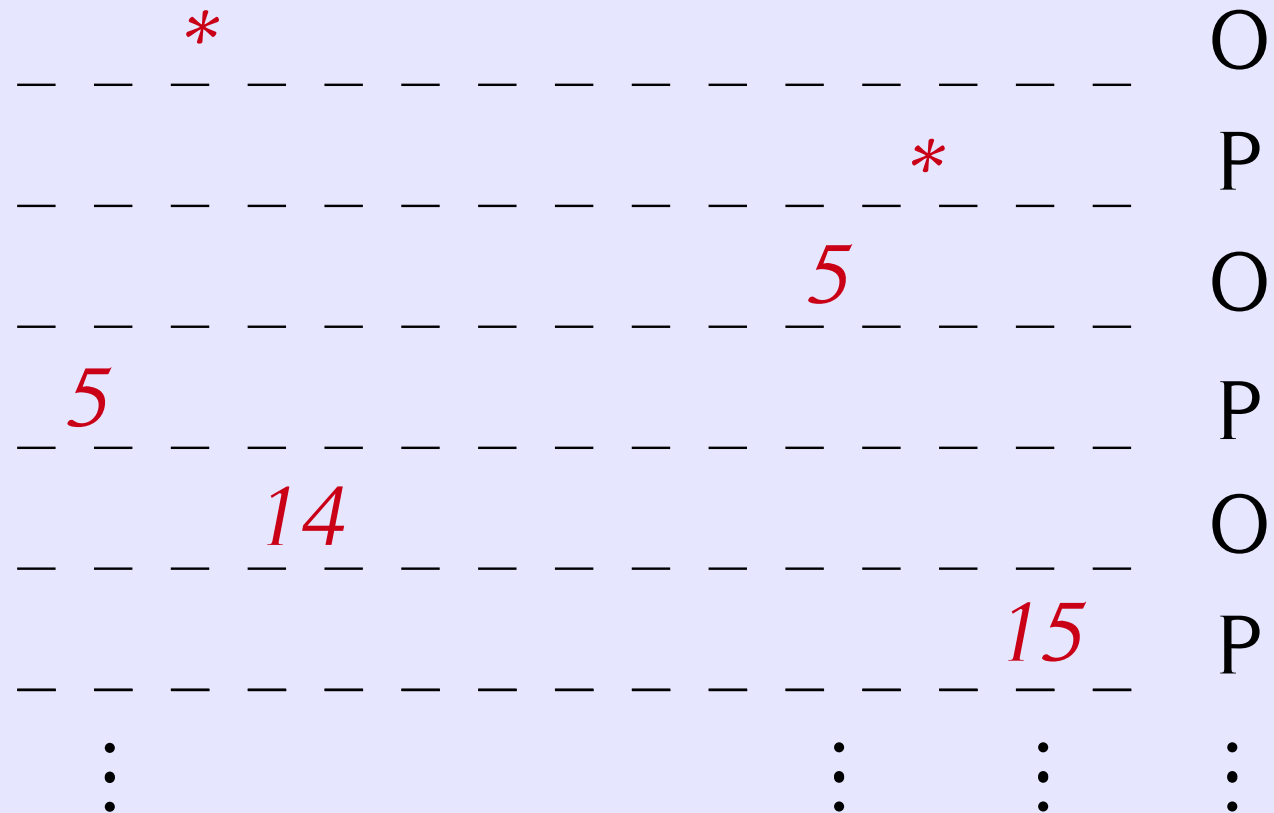
$\text{Int} \rightarrow \text{Int} \longrightarrow \text{Int} \rightarrow \text{Int}$



Example

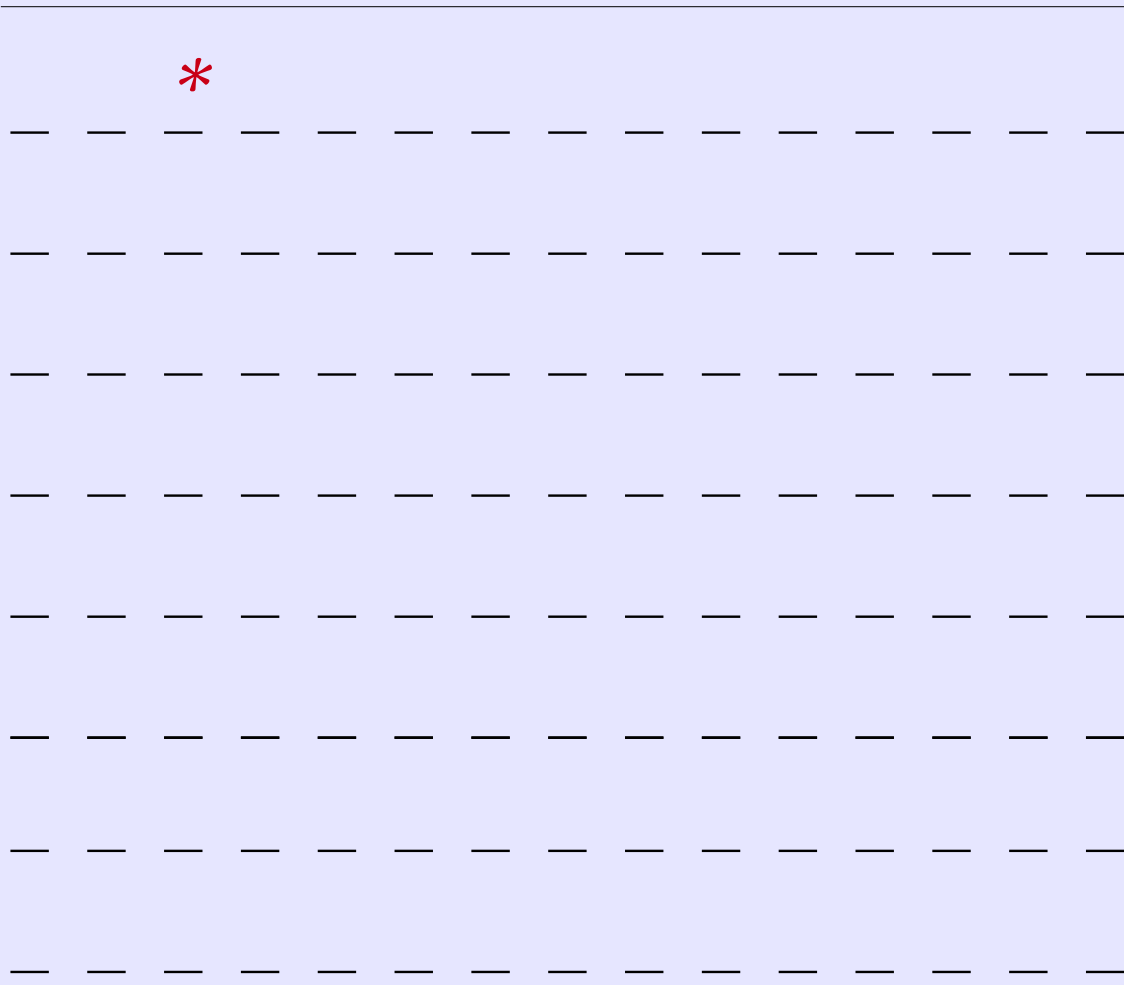
$f : \text{int} \rightarrow \text{int} \vdash \lambda x. f(x)+1 : \text{int} \rightarrow \text{int}$

$\text{Int} \rightarrow \text{Int} \longrightarrow \text{Int} \rightarrow \text{Int}$



Example

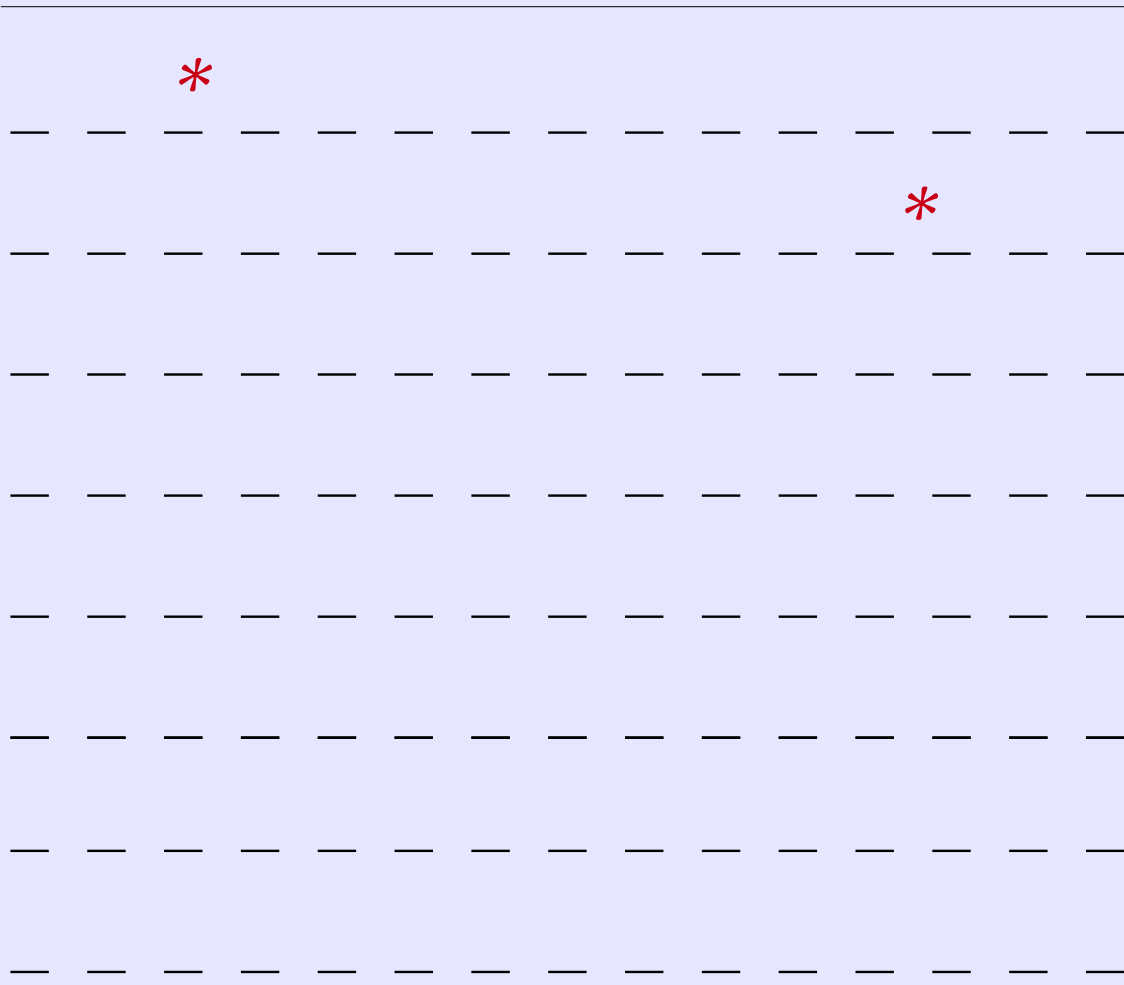
$Int \rightarrow Int \longrightarrow Int \rightarrow Int$



O
P
O
P
O
P
O
P

Example

$Int \rightarrow Int \longrightarrow Int \rightarrow Int$



O
P
O
P
O
P
O
P

Example

Int → *Int* —————▶ *Int* → *Int*



Example

$Int \rightarrow Int \longrightarrow Int \rightarrow Int$



Example

$Int \rightarrow Int \longrightarrow Int \rightarrow Int$



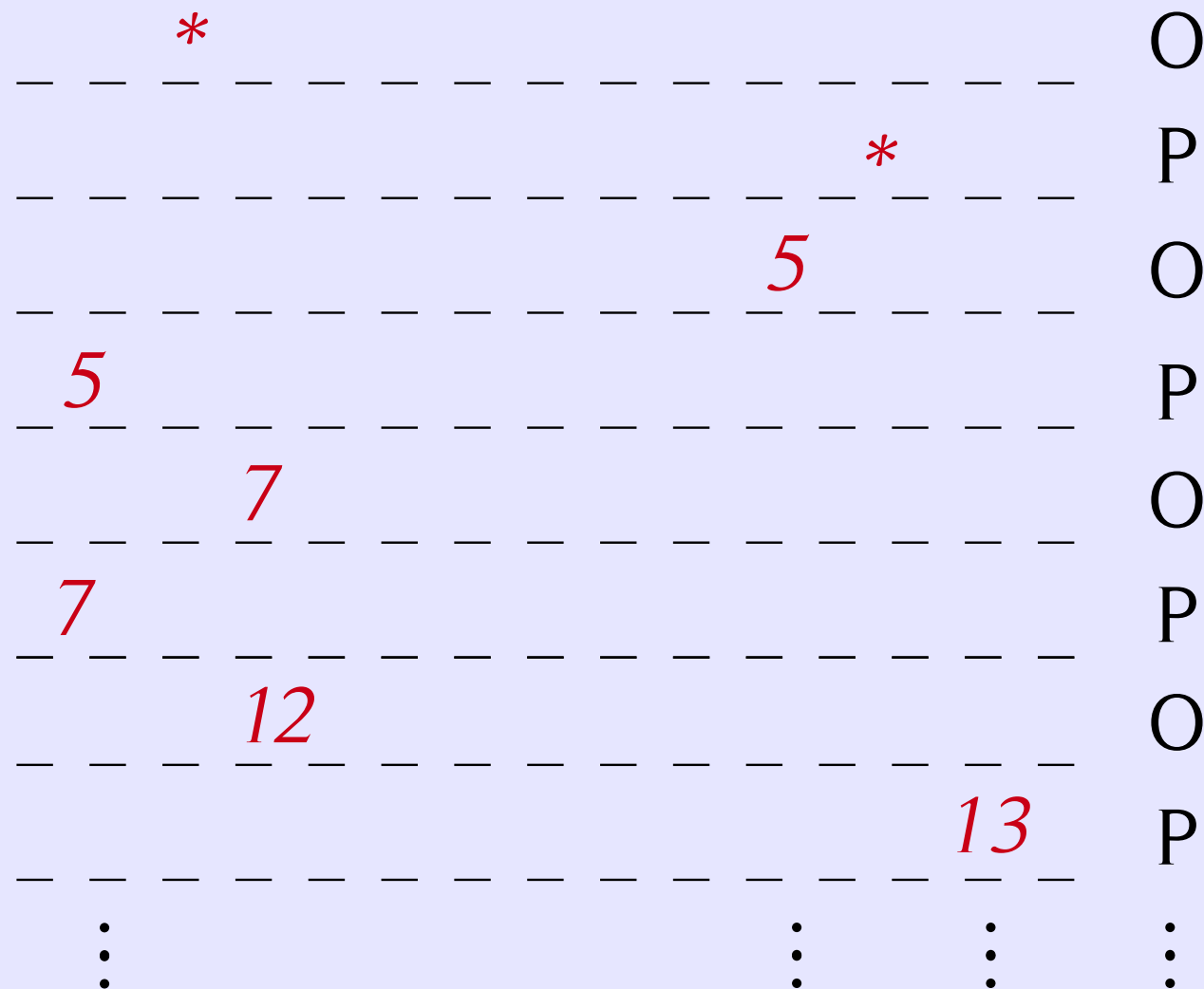
Example

$Int \rightarrow Int \longrightarrow Int \rightarrow Int$



$f : \text{int} \rightarrow \text{int} \vdash \lambda x. f(f(x))+1 : \text{int} \rightarrow \text{int}$

$\text{Int} \rightarrow \text{Int} \longrightarrow \text{Int} \rightarrow \text{Int}$



Game Semantics

- Computation is modelled as a 2-player game between:
 - *Opponent* (the environment)
 - *Proponent* (the program)
- Qualitative games

Game Semantics

- Computation is modelled as a 2-player game between:
 - *Opponent* (the environment)
 - *Proponent* (the program)
- Qualitative games
- Programs = *strategies* for Proponent

Composition

$\vdash \lambda x.x+1 : \text{int} \rightarrow \text{int}$

$f : \text{int} \rightarrow \text{int} \vdash \lambda x.f(x)+1 : \text{int} \rightarrow \text{int}$

Composition

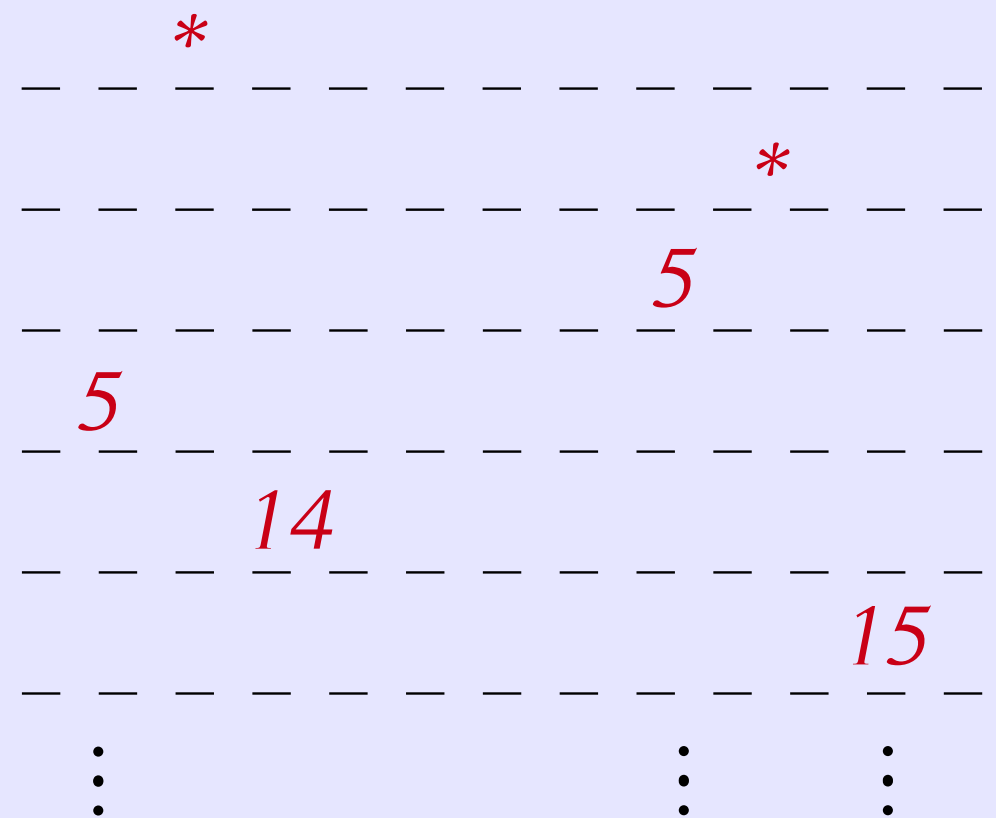
$\vdash \lambda x.x+1 : \text{int} \rightarrow \text{int}$

$f : \text{int} \rightarrow \text{int} \vdash \lambda x.f(x)+1 : \text{int} \rightarrow \text{int}$

Composition

$1 \longrightarrow \text{Int} \rightarrow \text{Int}$

$\text{Int} \rightarrow \text{Int} \longrightarrow \text{Int} \rightarrow \text{Int}$



$\vdash \lambda x.x+1 : \text{int} \rightarrow \text{int}$

$f : \text{int} \rightarrow \text{int} \vdash \lambda x.f(x)+1 : \text{int} \rightarrow \text{int}$

Composition

$1 \longrightarrow \text{Int} \rightarrow \text{Int}$

$\text{Int} \rightarrow \text{Int} \longrightarrow \text{Int} \rightarrow \text{Int}$

5

6

11

12

⋮ *⋮* *⋮*

5

5

14

15

⋮ *⋮* *⋮*

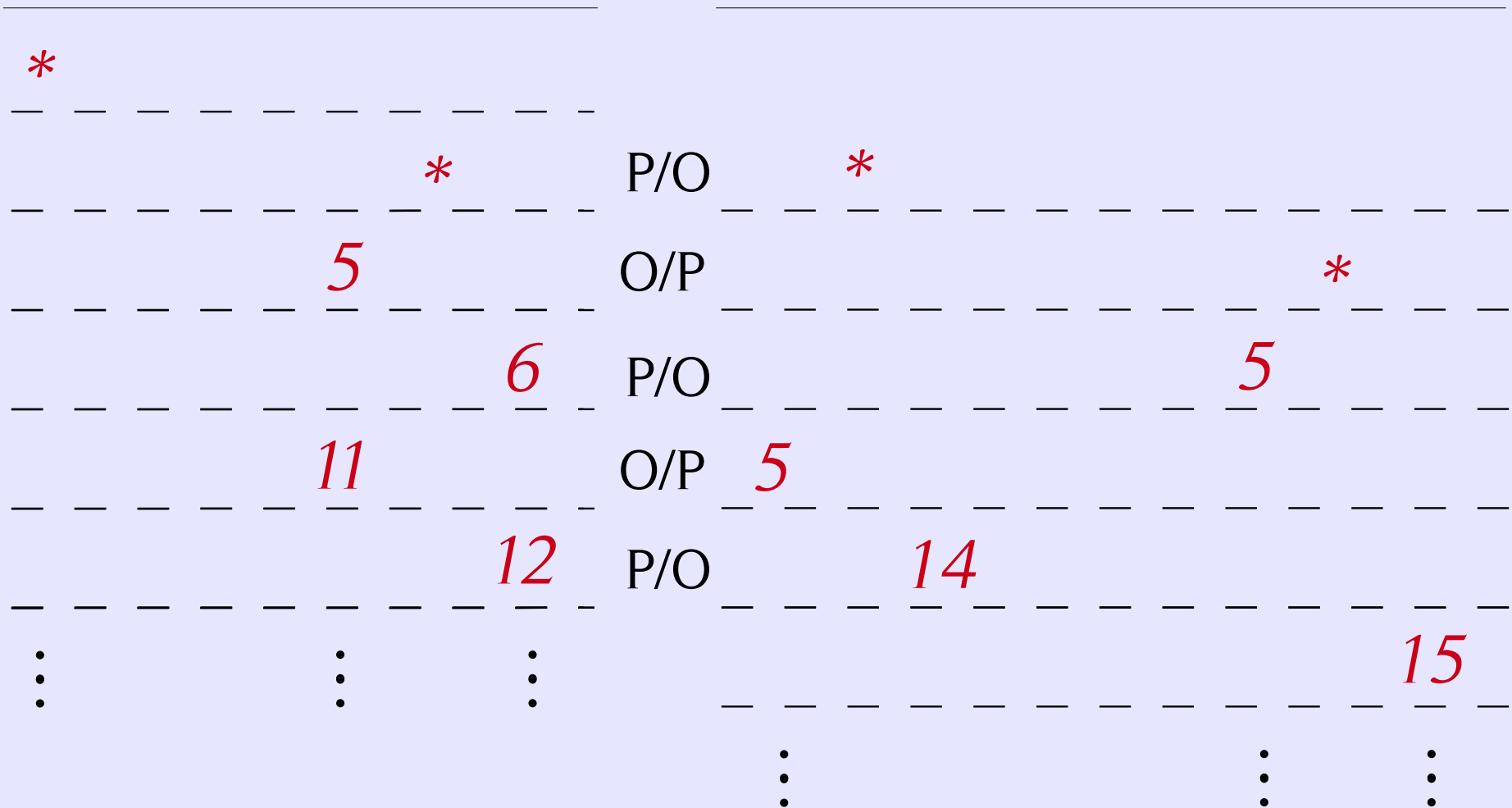
$\vdash \lambda x.x+1 : \text{int} \rightarrow \text{int}$

$f : \text{int} \rightarrow \text{int} \vdash \lambda x.f(x)+1 : \text{int} \rightarrow \text{int}$

Composition

$1 \longrightarrow \text{Int} \rightarrow \text{Int}$

$\text{Int} \rightarrow \text{Int} \longrightarrow \text{Int} \rightarrow \text{Int}$



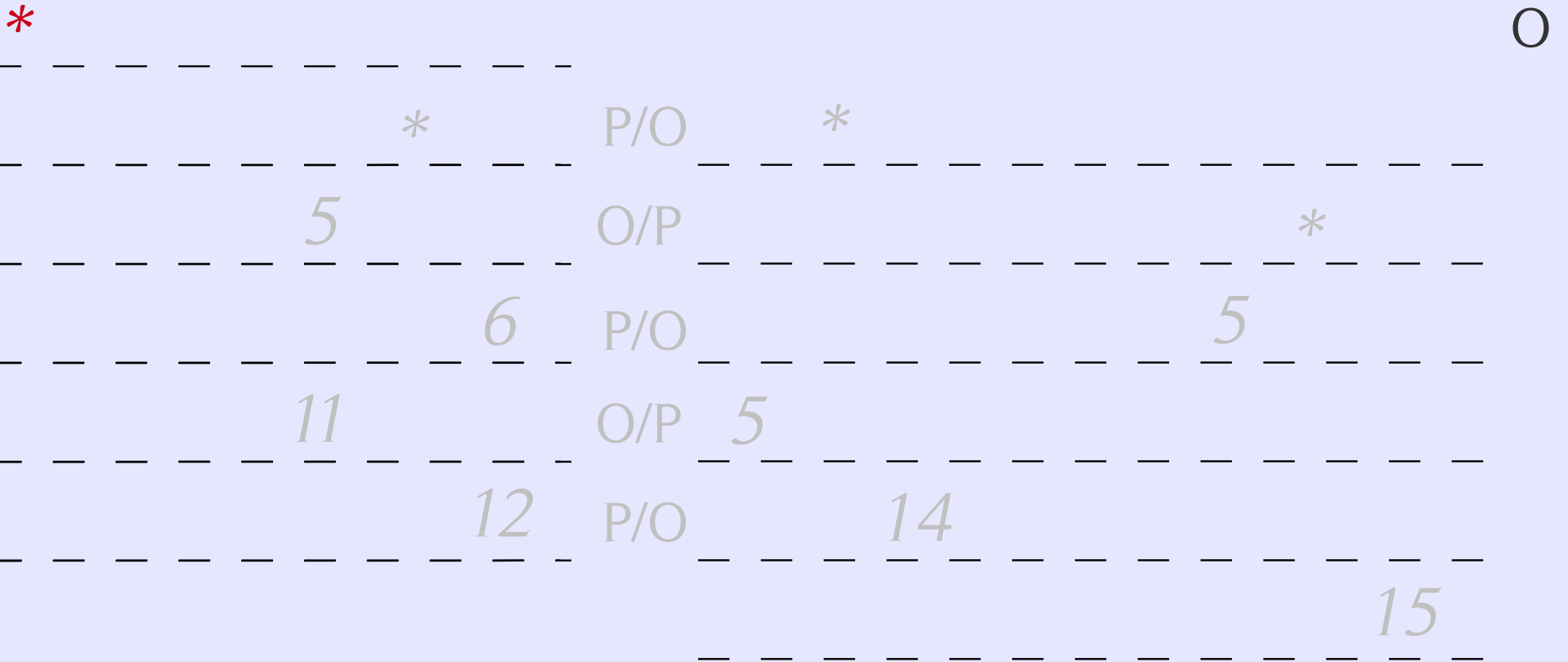
$\vdash \lambda x.x+1 : \text{int} \rightarrow \text{int}$

$f : \text{int} \rightarrow \text{int} \vdash \lambda x.f(x)+1 : \text{int} \rightarrow \text{int}$

Composition

$1 \longrightarrow \text{Int} \rightarrow \text{Int}$

$\text{Int} \rightarrow \text{Int} \longrightarrow \text{Int} \rightarrow \text{Int}$



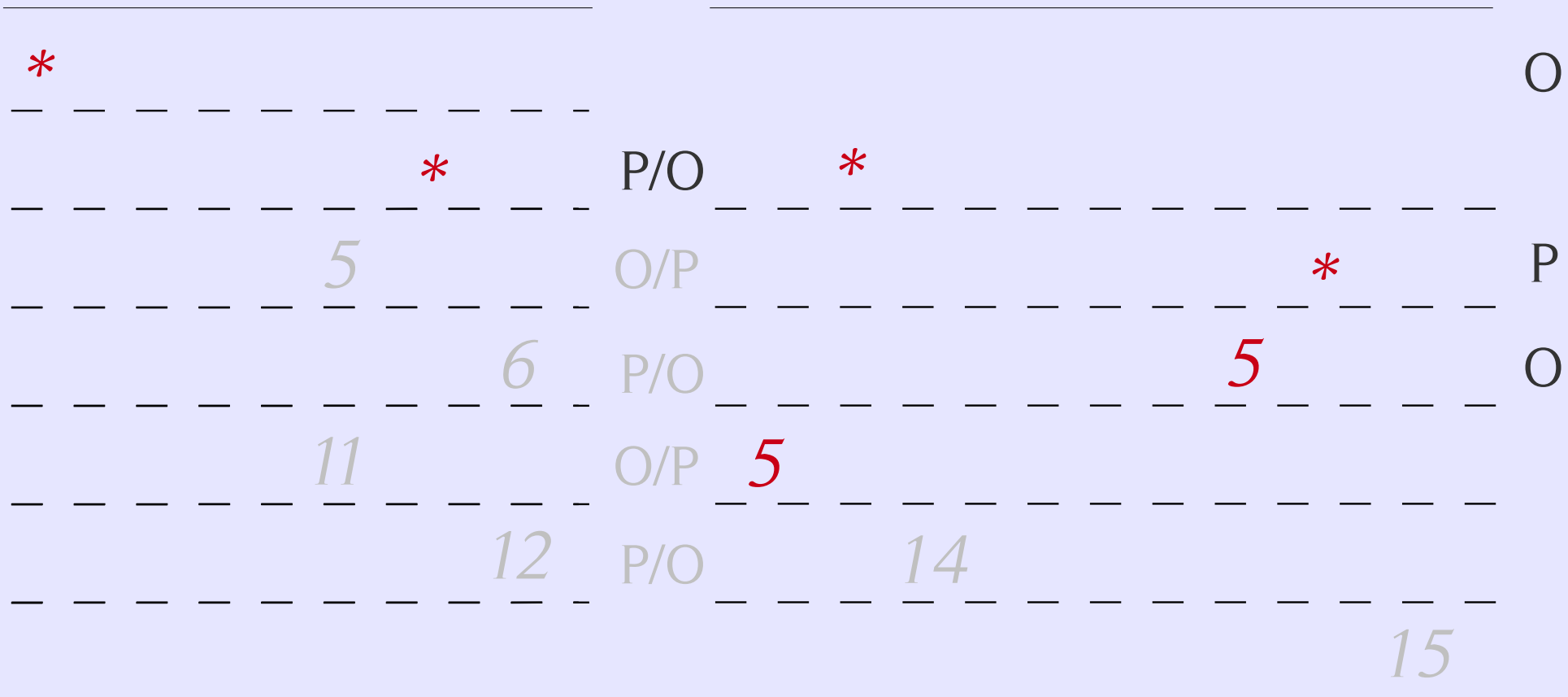
$\vdash \lambda x.x+1 : \text{int} \rightarrow \text{int}$

$f : \text{int} \rightarrow \text{int} \vdash \lambda x.f(x)+1 : \text{int} \rightarrow \text{int}$

Composition

$1 \longrightarrow \text{Int} \rightarrow \text{Int}$

$\text{Int} \rightarrow \text{Int} \longrightarrow \text{Int} \rightarrow \text{Int}$



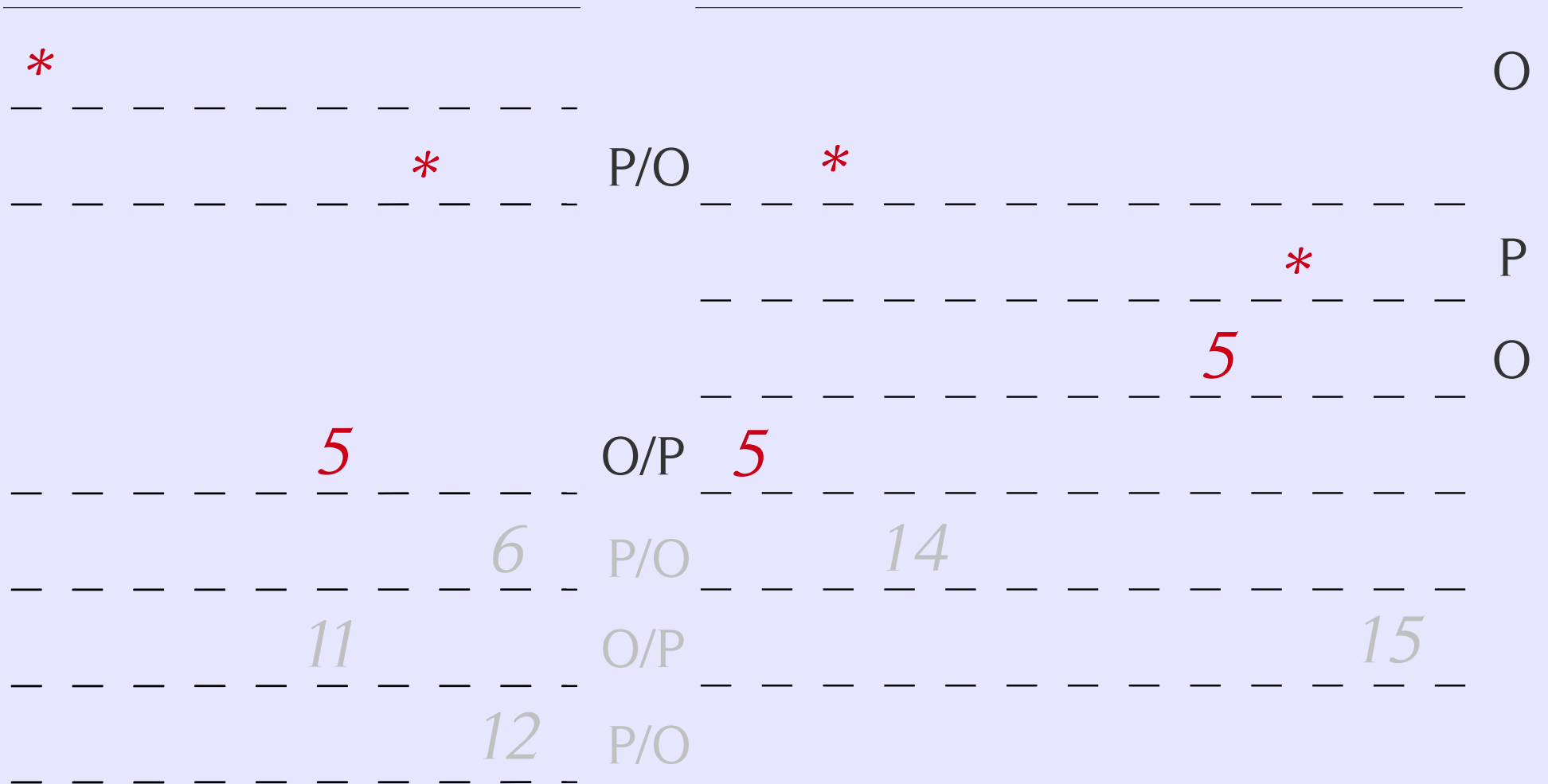
$\vdash \lambda x.x+1 : \text{int} \rightarrow \text{int}$

$f : \text{int} \rightarrow \text{int} \vdash \lambda x.f(x)+1 : \text{int} \rightarrow \text{int}$

Composition

$1 \longrightarrow \text{Int} \rightarrow \text{Int}$

$\text{Int} \rightarrow \text{Int} \longrightarrow \text{Int} \rightarrow \text{Int}$



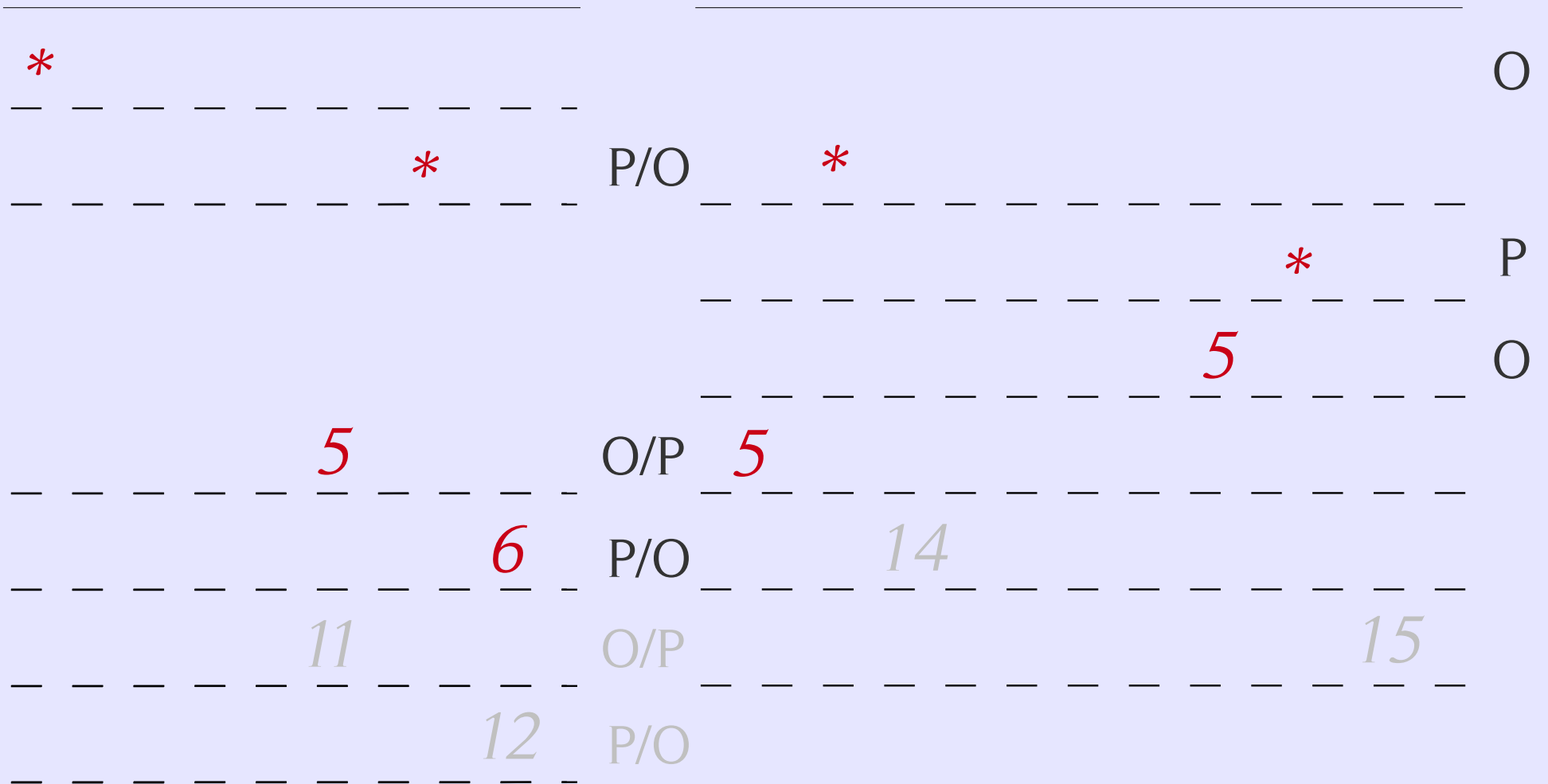
$\vdash \lambda x.x+1 : \text{int} \rightarrow \text{int}$

$f : \text{int} \rightarrow \text{int} \vdash \lambda x.f(x)+1 : \text{int} \rightarrow \text{int}$

Composition

$1 \longrightarrow \text{Int} \rightarrow \text{Int}$

$\text{Int} \rightarrow \text{Int} \longrightarrow \text{Int} \rightarrow \text{Int}$



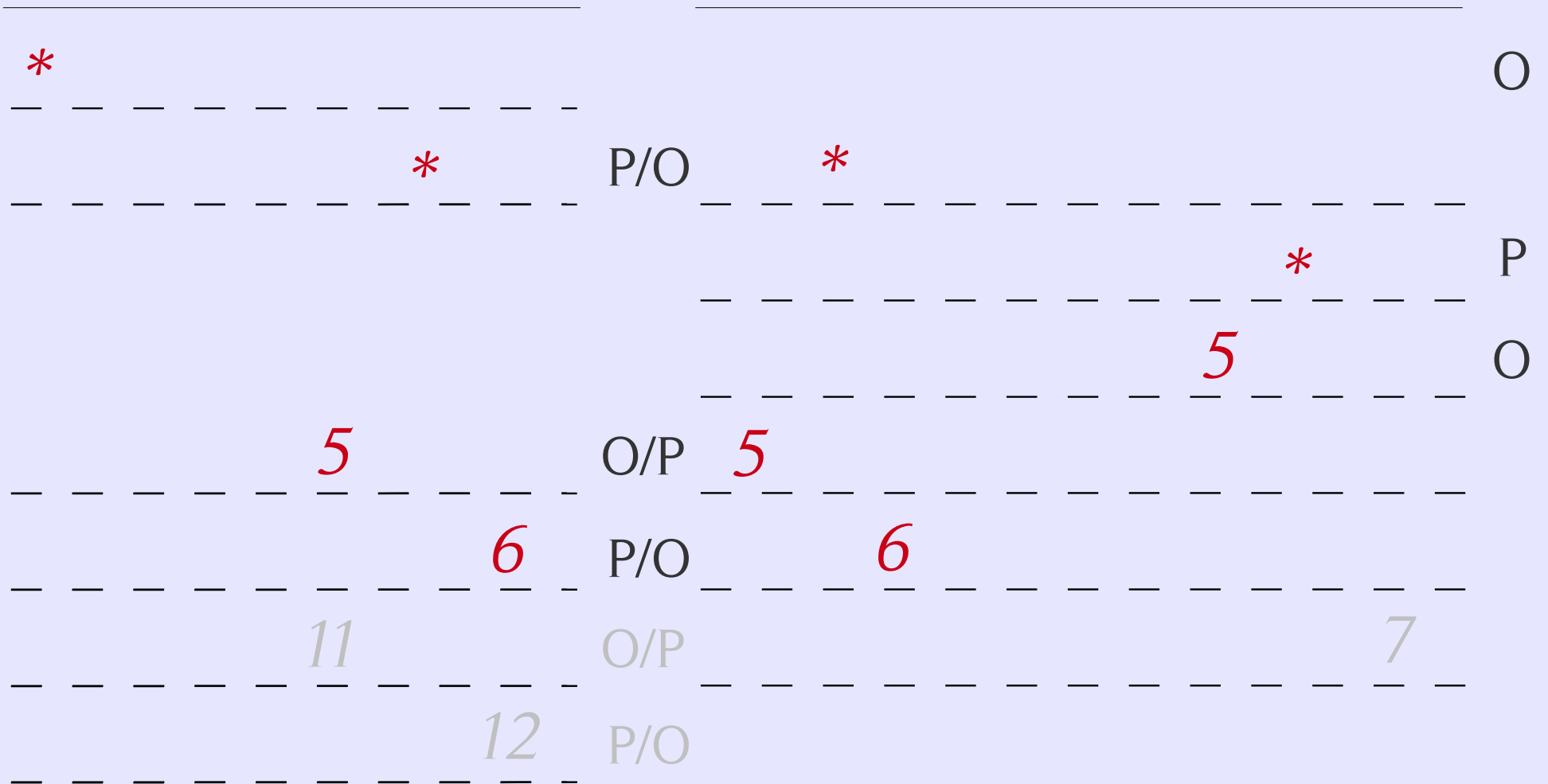
$\vdash \lambda x.x+1 : \text{int} \rightarrow \text{int}$

$f : \text{int} \rightarrow \text{int} \vdash \lambda x.f(x)+1 : \text{int} \rightarrow \text{int}$

Composition

$1 \longrightarrow \text{Int} \rightarrow \text{Int}$

$\text{Int} \rightarrow \text{Int} \longrightarrow \text{Int} \rightarrow \text{Int}$



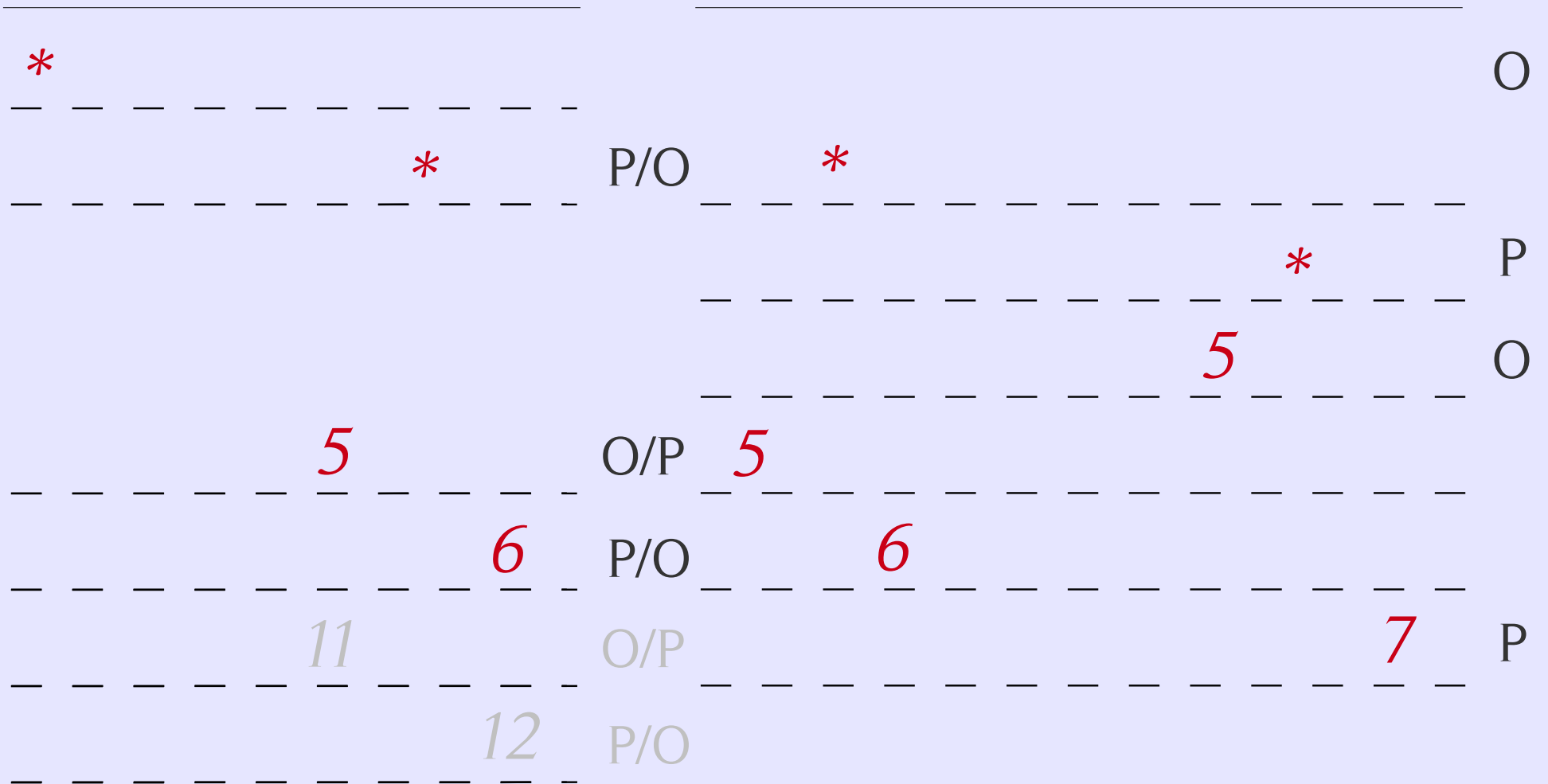
$\vdash \lambda x.x+1 : \text{int} \rightarrow \text{int}$

$f : \text{int} \rightarrow \text{int} \vdash \lambda x.f(x)+1 : \text{int} \rightarrow \text{int}$

Composition

$1 \longrightarrow \text{Int} \rightarrow \text{Int}$

$\text{Int} \rightarrow \text{Int} \longrightarrow \text{Int} \rightarrow \text{Int}$



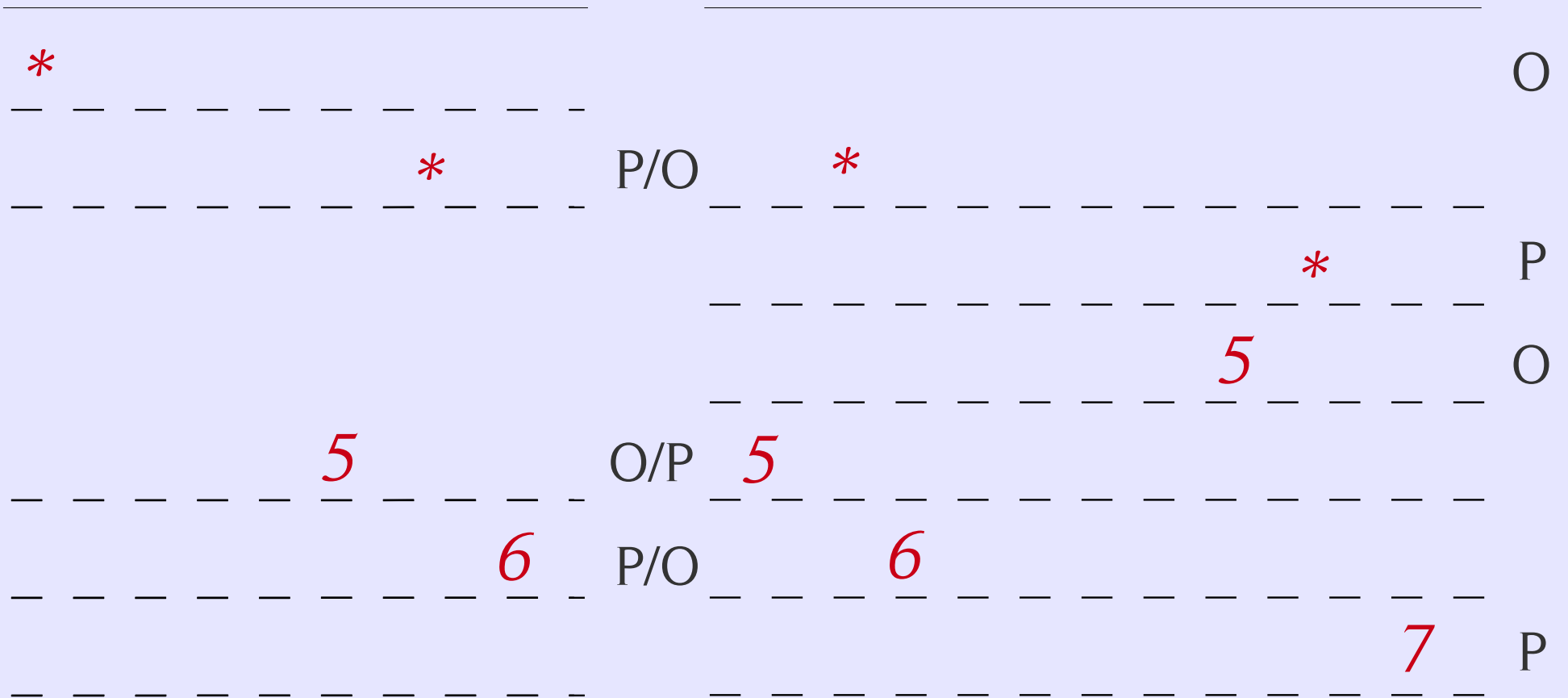
$\vdash \lambda x.x+1 : \text{int} \rightarrow \text{int}$

$f : \text{int} \rightarrow \text{int} \vdash \lambda x.f(x)+1 : \text{int} \rightarrow \text{int}$

Composition

$1 \longrightarrow \text{Int} \rightarrow \text{Int}$

$\text{Int} \rightarrow \text{Int} \longrightarrow \text{Int} \rightarrow \text{Int}$



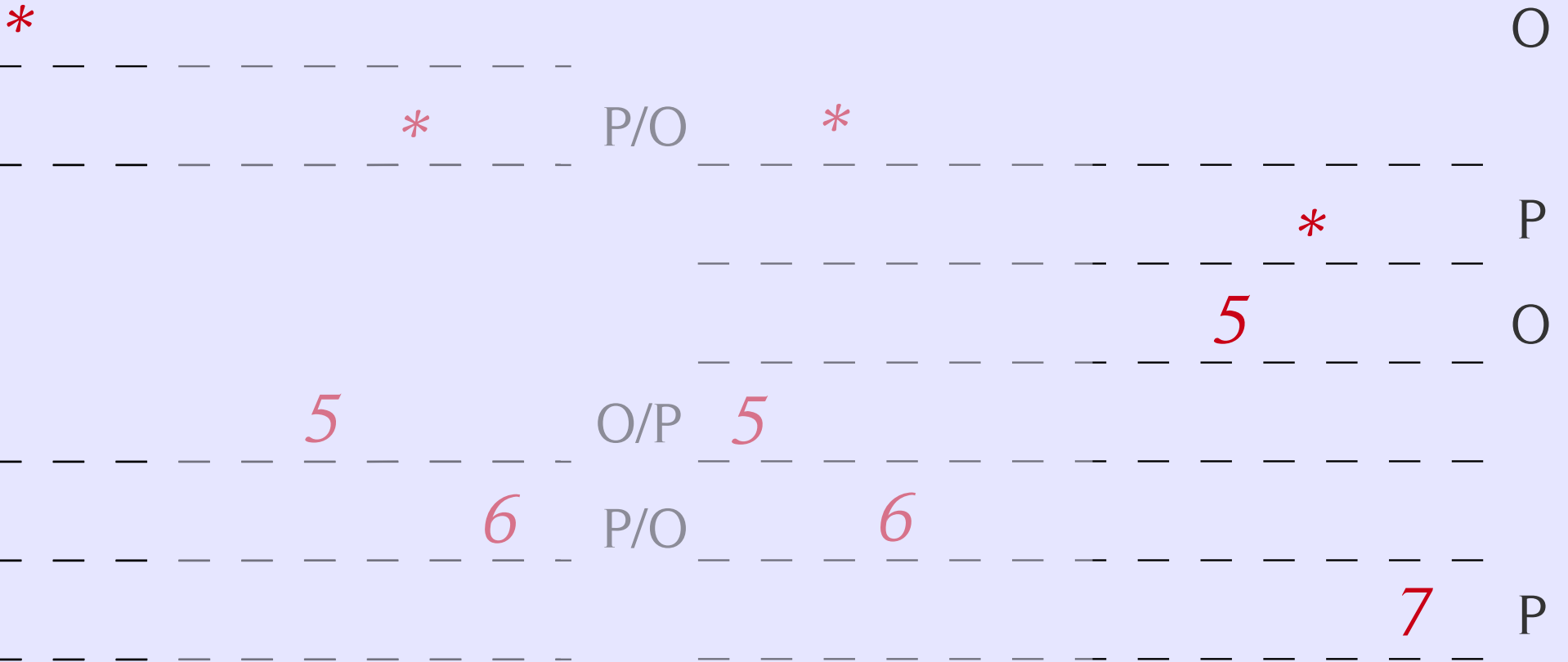
$\vdash \lambda x.x+1 : \text{int} \rightarrow \text{int}$

$f : \text{int} \rightarrow \text{int} \vdash \lambda x.f(x)+1 : \text{int} \rightarrow \text{int}$

Composition

$1 \longrightarrow \text{Int} \rightarrow \text{Int}$

$\text{Int} \rightarrow \text{Int} \longrightarrow \text{Int} \rightarrow \text{Int}$



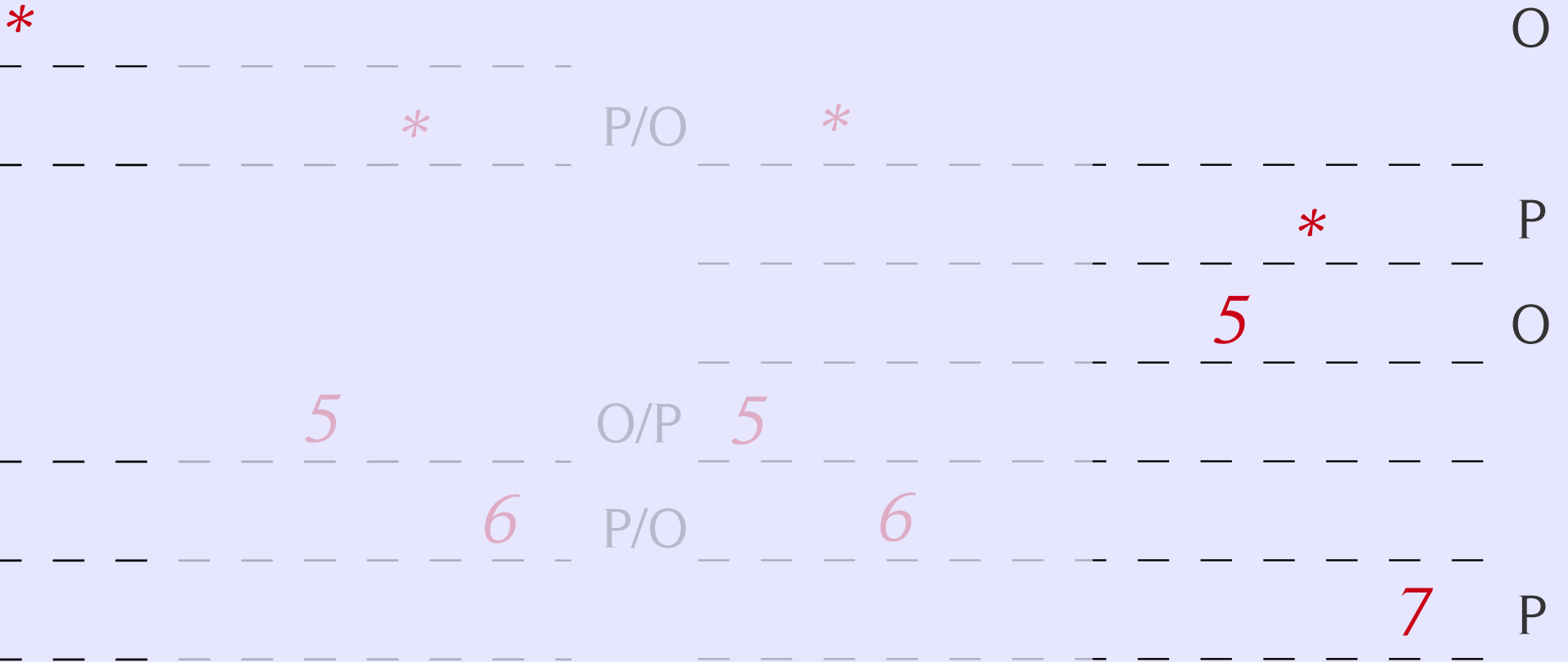
$\vdash \lambda x.x+1 : \text{int} \rightarrow \text{int}$

$f : \text{int} \rightarrow \text{int} \vdash \lambda x.f(x)+1 : \text{int} \rightarrow \text{int}$

Composition

$1 \longrightarrow \text{Int} \rightarrow \text{Int}$

$\text{Int} \rightarrow \text{Int} \longrightarrow \text{Int} \rightarrow \text{Int}$



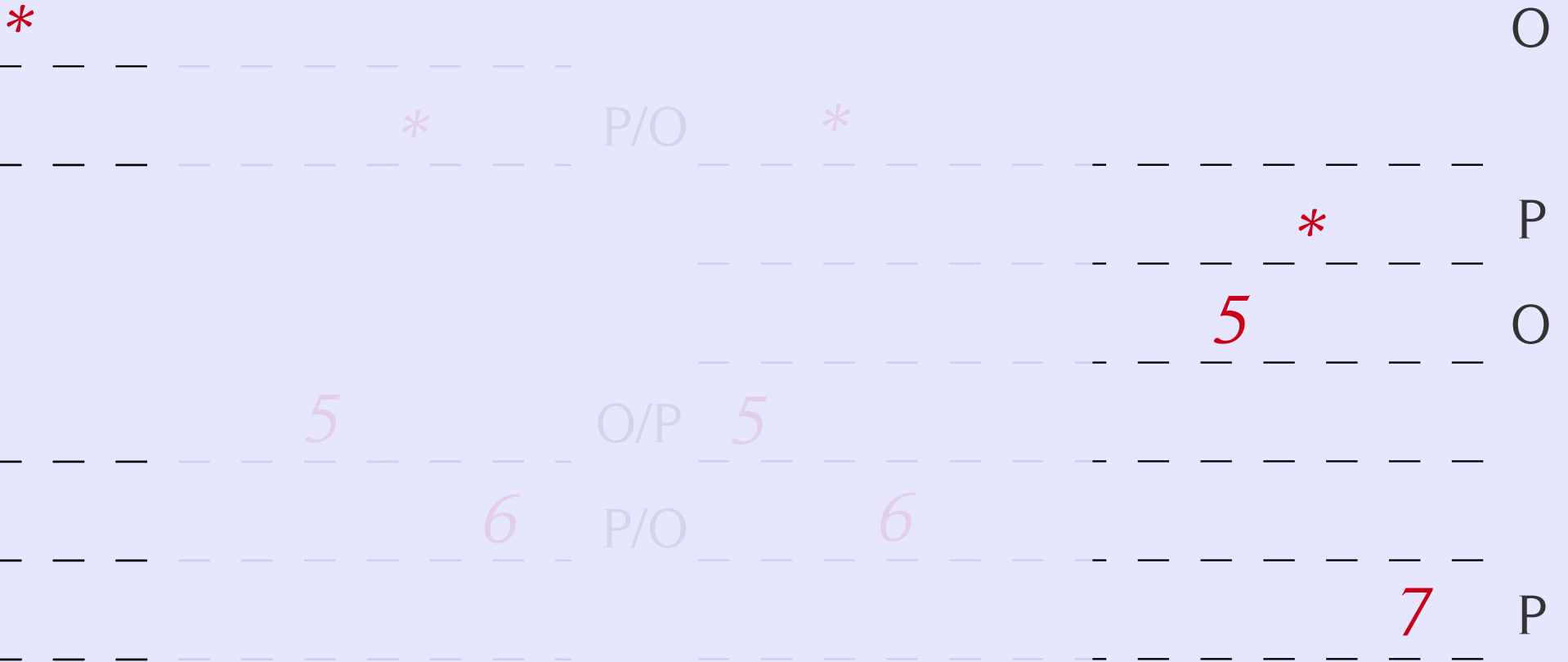
$\vdash \lambda x.x+1 : \text{int} \rightarrow \text{int}$

$f : \text{int} \rightarrow \text{int} \vdash \lambda x.f(x)+1 : \text{int} \rightarrow \text{int}$

Composition

$1 \longrightarrow \text{Int} \rightarrow \text{Int}$

$\text{Int} \rightarrow \text{Int} \longrightarrow \text{Int} \rightarrow \text{Int}$



$\vdash \lambda x.x+1 : \text{int} \rightarrow \text{int}$

$f : \text{int} \rightarrow \text{int} \vdash \lambda x.f(x)+1 : \text{int} \rightarrow \text{int}$

Composition

1 \longrightarrow *Int* \rightarrow *Int*

*** O

*** P

5 O

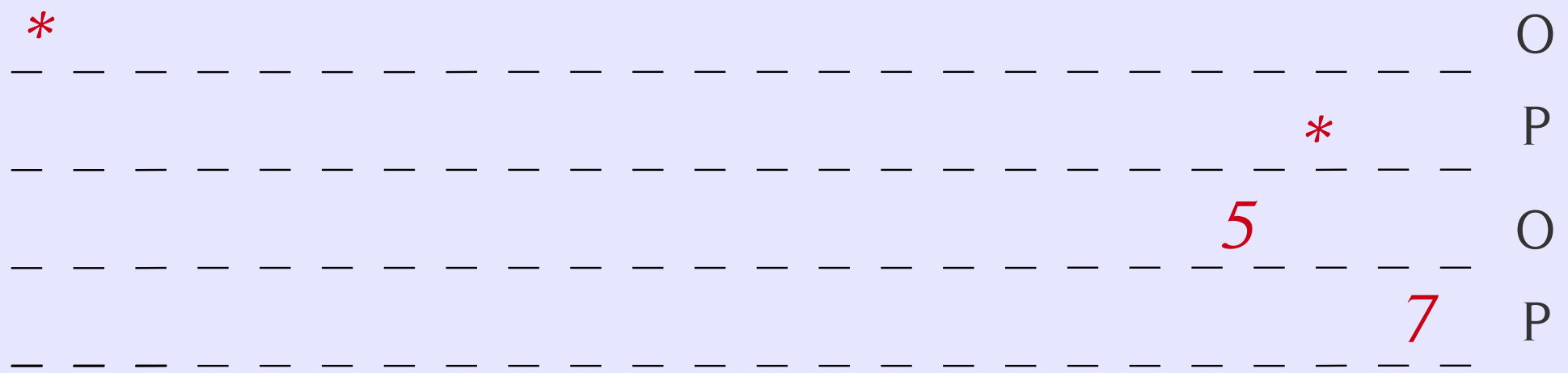
7 P

$\vdash \lambda x. x+1 : \text{int} \rightarrow \text{int}$

$f : \text{int} \rightarrow \text{int} \vdash \lambda x. f(x)+1 : \text{int} \rightarrow \text{int}$

Composition

1 \longrightarrow $\text{Int} \rightarrow \text{Int}$



$\vdash \lambda x. x+1 : \text{int} \rightarrow \text{int}$

; $\vdash f : \text{int} \rightarrow \text{int} \vdash \lambda x. f(x)+1 : \text{int} \rightarrow \text{int}$

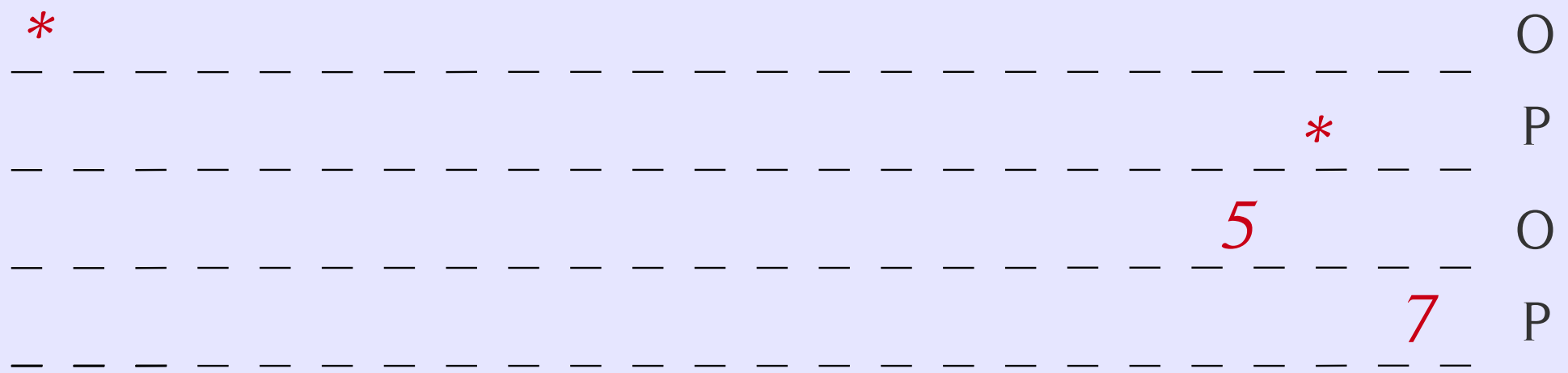
$= \vdash \lambda x. x+2 : \text{int} \rightarrow \text{int}$

$\vdash \lambda x. x+1 : \text{int} \rightarrow \text{int}$

$f : \text{int} \rightarrow \text{int} \vdash \lambda x. f(x)+1 : \text{int} \rightarrow \text{int}$

Composition

$1 \longrightarrow \text{Int} \rightarrow \text{Int}$



$\vdash \lambda x. x+1 : \text{int} \rightarrow \text{int} ; f : \text{int} \rightarrow \text{int} \vdash \lambda x. f(x)+1 : \text{int} \rightarrow \text{int}$

$= \vdash \lambda x. x+2 : \text{int} \rightarrow \text{int}$

$$A \xrightarrow{\sigma} B \xrightarrow{\tau} C = A \xrightarrow{\sigma; \tau} C$$

Game Semantics

- Computation is modelled as a 2-player game between:
 - *Opponent* (the environment)
 - *Proponent* (the program)
- Qualitative games
- Programs = *strategies* for Proponent
- Families (i.e. *categories*) of games

Quick storyline

Quick storyline

Introduced in early 90's

Full abstraction for PCF

- Two teams in the UK; one in Siegen
- Roots in Mathematical Logic

Quick storyline

Introduced in early 90's

- Two teams in the UK; one in Siegen
- Roots in Mathematical Logic

First stage (1993-2004)

- Models for various programming features
- Program analysis

Quick storyline

Introduced in early 90's

- Two teams in the UK; one in Siegen
- Roots in Mathematical Logic

First stage (1993-2004)

- Models for various programming features
- Program analysis

} dynamic behaviours?

Quick storyline

Introduced in early 90's

- Two teams in the UK; one in Siegen
- Roots in Mathematical Logic

First stage (1993-2004)

- Models for various programming features
- Program analysis

} dynamic behaviours?

Nominal game semantics (2004-)

Quick storyline

Introduced in early 90's

- Two teams in the UK; one in Siegen
- Roots in Mathematical Logic

First stage (1993-2004)

- Models for various programming features
- Program analysis

} dynamic behaviours?

Nominal game semantics (2004-)

- Models of realistic programs (2006-)
- Algorithmic games (2011-)

Computation with new resources

RedML = simply-typed lambda-calculus
+ integer variables (*names*)

Computation with new resources

RedML = simply-typed lambda-calculus
+ integer variables (*names*)

$\text{let } x = \text{newvar}(0) \text{ in } \lambda y. (x == y) \cong \lambda y. 0 : \text{intvar} \rightarrow \text{int}$

Computation with new resources

RedML = simply-typed lambda-calculus
+ integer variables (*names*)

$\text{let } x = \text{newvar}(0) \text{ in } \lambda y. (x == y) \quad \cong \quad \lambda y. 0 : \text{intvar} \rightarrow \text{int}$

$\lambda y. \text{newvar}(0) \quad \not\cong \quad \text{let } x = \text{newvar}(0) \text{ in } (\lambda y. x) : \text{com} \rightarrow \text{intvar}$

Computation with new resources

RedML = simply-typed lambda-calculus
+ integer variables (*names*)

$\text{let } x = \text{newvar}(0) \text{ in } \lambda y. (x == y) \cong \lambda y. 0 : \text{intvar} \rightarrow \text{int}$

$\lambda y. \text{newvar}(0) \not\cong \text{let } x = \text{newvar}(0) \text{ in } (\lambda y. x) : \text{com} \rightarrow \text{intvar}$

$f : \text{intvar} \rightarrow \text{int} \vdash \lambda y. \text{let } x = \text{newvar}(0) \text{ in } f(x)$
 $\cong \text{let } x = \text{newvar}(0) \text{ in } \lambda y. x := 0; f(x) : \text{com} \rightarrow \text{int}$

Computation with new resources

RedML = simply-typed lambda-calculus
+ integer variables (*names*)

$$\text{let } x = \text{newvar}(0) \text{ in } \lambda y. (x == y) \quad \cong \quad \lambda y. 0 : \text{intvar} \rightarrow \text{int}$$
$$\lambda y. \text{newvar}(0) \quad \not\cong \quad \text{let } x = \text{newvar}(0) \text{ in } (\lambda y. x) : \text{com} \rightarrow \text{intvar}$$
$$\begin{aligned} f : \text{intvar} \rightarrow \text{int} \vdash \lambda y. \text{let } x = \text{newvar}(0) \text{ in } f(x) \\ \cong \quad \text{let } x = \text{newvar}(0) \text{ in } \lambda y. x := 0; f(x) : \text{com} \rightarrow \text{int} \end{aligned}$$
$$\begin{aligned} f : \text{intvar} \rightarrow \text{com} \vdash \text{let } x = \text{newvar}(0) \text{ in let } y = \text{ref } (0) \text{ in } f(x); (y := !x); y \\ \cong \quad \text{let } x = \text{newvar}(0) \text{ in } f(x); x : \text{intvar} \end{aligned}$$

Nominal techniques

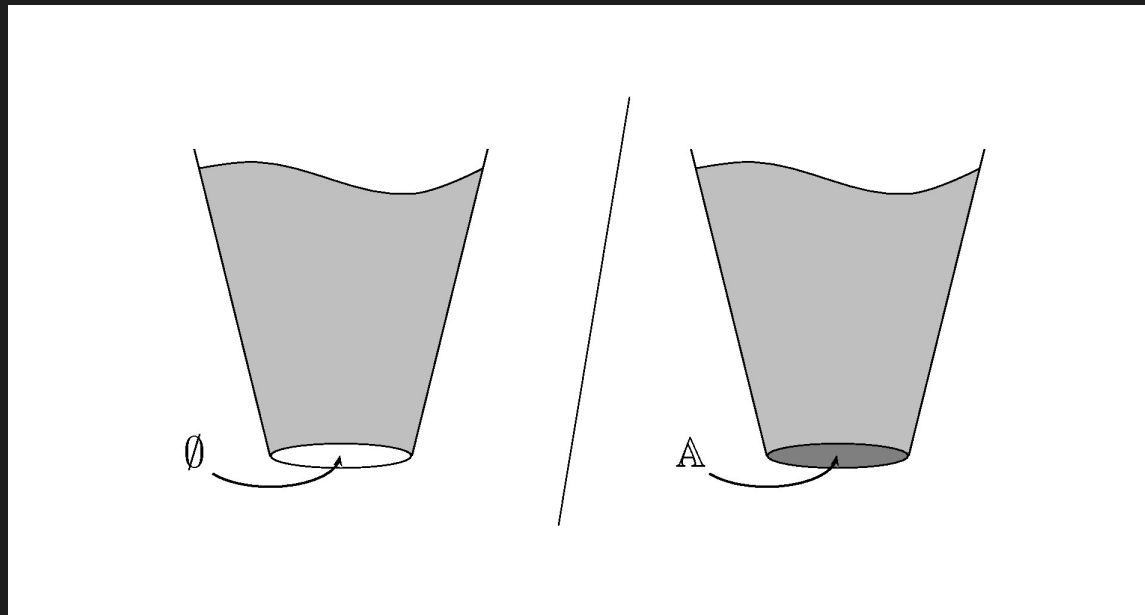
- A foundational theory for doing maths with **names** and **name-binding**

Nominal techniques

- A foundational theory for doing maths with **names** and **name-binding**
- Two presentations:
 - FM-set theory [Gabbay&Pitts 99, Gabbay01]
 - Nominal sets [Pitts01]

FM-sets

- Fraenkel-Mostowski model of ZFA:
 - Basis: emptyset + inf. set \mathcal{A} of **atoms**
 - Constructions have **finite support**



Nominal sets

- Countably infinite set \mathcal{N} of **names**

Nominal sets

- Countably infinite set \mathcal{N} of **names**
- A nominal set X consists of:
 - a carrier set X
 - an action $\bullet : \text{PERM}(\mathcal{N}) \times X \longrightarrow X$
e.g. $(n\ m) \bullet nmn = mnm$
 - all elements of X have **finite support**

Nominal sets

- Countably infinite set \mathcal{N} of **names**
- A nominal set X consists of:
 - a carrier set X
 - an action $\bullet : \text{PERM}(\mathcal{N}) \times X \longrightarrow X$
e.g. $(n\ m) \bullet nmn = mnm$
 - all elements of X have **finite support**
- **NOM**: nominal sets and equivariant functions

Nominal Logic

Example axioms (note sorts should match):

$$\forall a. \phi(\vec{x}) \iff \exists a. a \# \vec{x} \wedge \phi(\vec{x}) \quad (\text{Q})$$

$$(a \ a') \cdot (b \ b') \cdot x = ((a \ a') \cdot b \ (a \ a') \cdot b') \cdot (a \ a') \cdot x \quad (\text{E1})$$

$$b \# x \implies (a \ a') \cdot b \# (a \ a') \cdot x \quad (\text{E2})$$

$$a \# x \wedge a' \# x \implies (a \ a') \cdot x = x \quad (\text{F1})$$

$$a \cdot x = a' \cdot x' \iff (a = a' \vee a' \# x) \wedge x' = (a \ a') \cdot x \quad (\text{A1})$$

Nominal Logic

Example axioms (note sorts should match):

$$\underline{\forall a. \phi(\vec{x})} \iff \exists a. \underline{a \# \vec{x}} \wedge \phi(\vec{x}) \quad (\text{Q})$$

$$\underline{(a \ a')} \cdot (b \ b') \cdot x = ((a \ a') \cdot b \ (a \ a') \cdot b') \cdot (a \ a') \cdot x \quad (\text{E1})$$

$$b \# x \implies (a \ a') \cdot b \# (a \ a') \cdot x \quad (\text{E2})$$

$$a \# x \wedge a' \# x \implies (a \ a') \cdot x = x \quad (\text{F1})$$

$$\underline{a \cdot x} = a' \cdot x' \iff (a = a' \vee a' \# x) \wedge x' = (a \ a') \cdot x \quad (\text{A1})$$

Nominal Logic

Example axioms (note sorts should match):

$$\forall a. \phi(\vec{x}) \iff \exists a. a\#\vec{x} \wedge \phi(\vec{x}) \quad (\text{Q})$$

$$(a\ a') \cdot (b\ b') \cdot x = ((a\ a') \cdot b\ (a\ a') \cdot b') \cdot (a\ a') \cdot x \quad (\text{E1})$$

$$b\#x \implies (a\ a') \cdot b\#(a\ a') \cdot x \quad (\text{E2})$$

$$a\#x \wedge a'\#x \implies (a\ a') \cdot x = x \quad (\text{F1})$$

$$a.x = a'.x' \iff (a = a' \vee a'\#x) \wedge x' = (a\ a') \cdot x \quad (\text{A1})$$

NL gives us a strong handle on names. For example:

- $\phi(\vec{x}) \iff \phi((a\ a') \cdot \vec{x})$
- $(\exists a. a\#\vec{x} \wedge \phi(\vec{x})) \iff (\forall a. a\#\vec{x} \implies \phi(\vec{x}))$
- $b\#a.x \iff b = a \vee b\#x$
- $a.x = a'.x' \iff \forall b. (a\ b) \cdot x = (a'\ b) \cdot x'$

Nominal games

Nominal games

$\vdash \lambda x. \text{newvar}(0) : \text{com} \rightarrow \text{intvar}$

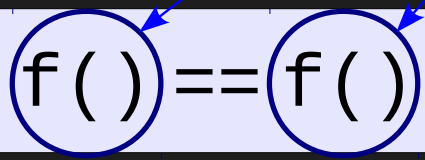
$\text{let } f = [_] \text{ in } \{ \text{f}() == \text{f}() \}$

Nominal games

names

$\vdash \lambda x. \text{newvar}(0) : \text{com} \rightarrow \text{intvar}$

$\text{let } f = [_] \text{ in } \{ f() == f() \}$



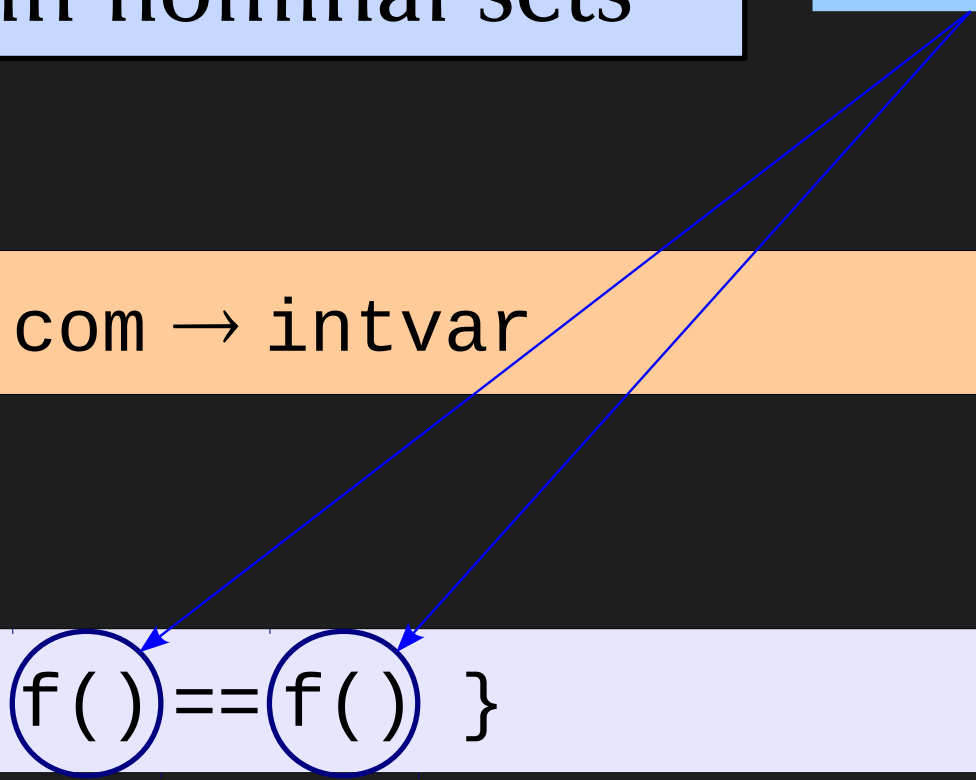
Nominal games

Games in nominal sets

names

$\vdash \lambda x. \text{newvar}(0) : \text{com} \rightarrow \text{intvar}$

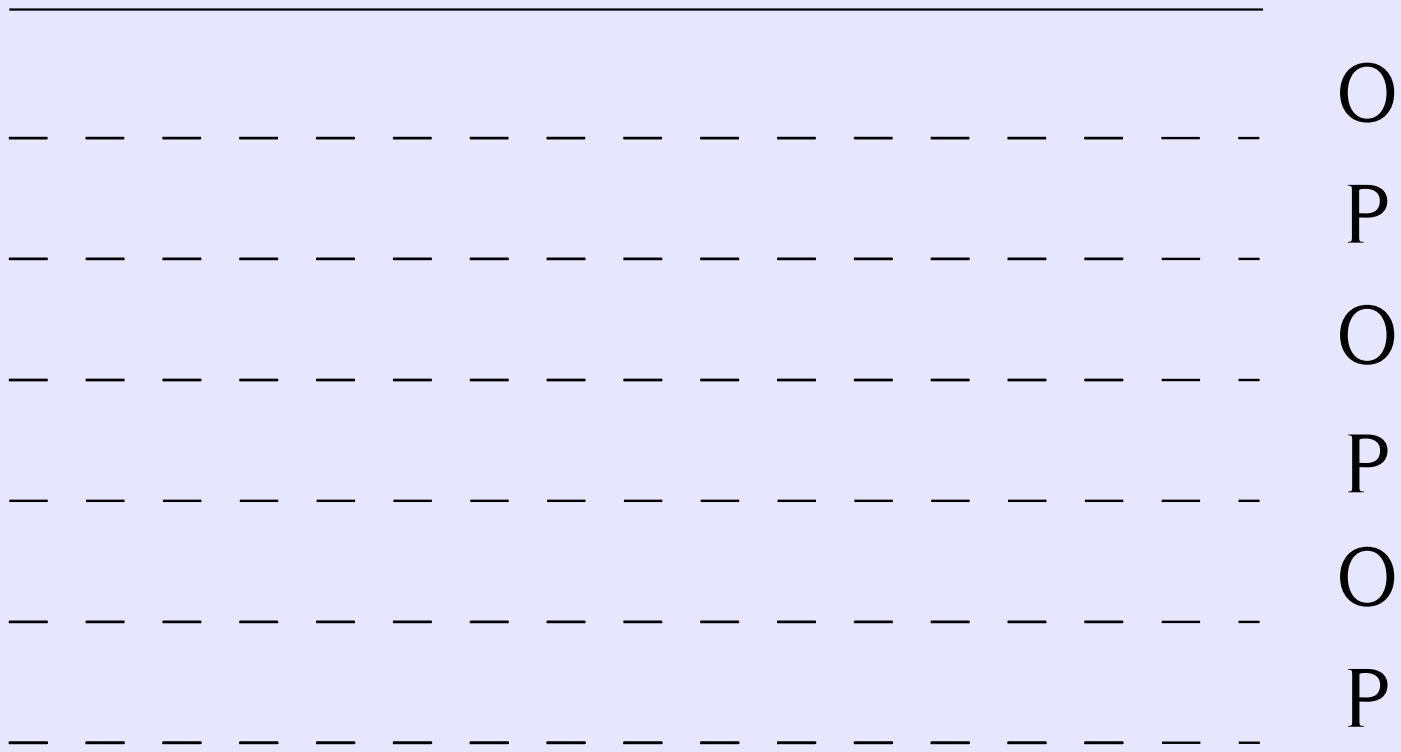
let $f = [_]$ in $\{ f() == f() \}$



Examples

$\vdash \lambda x. \text{newvar}(0) : \text{com} \rightarrow \text{intvar}$

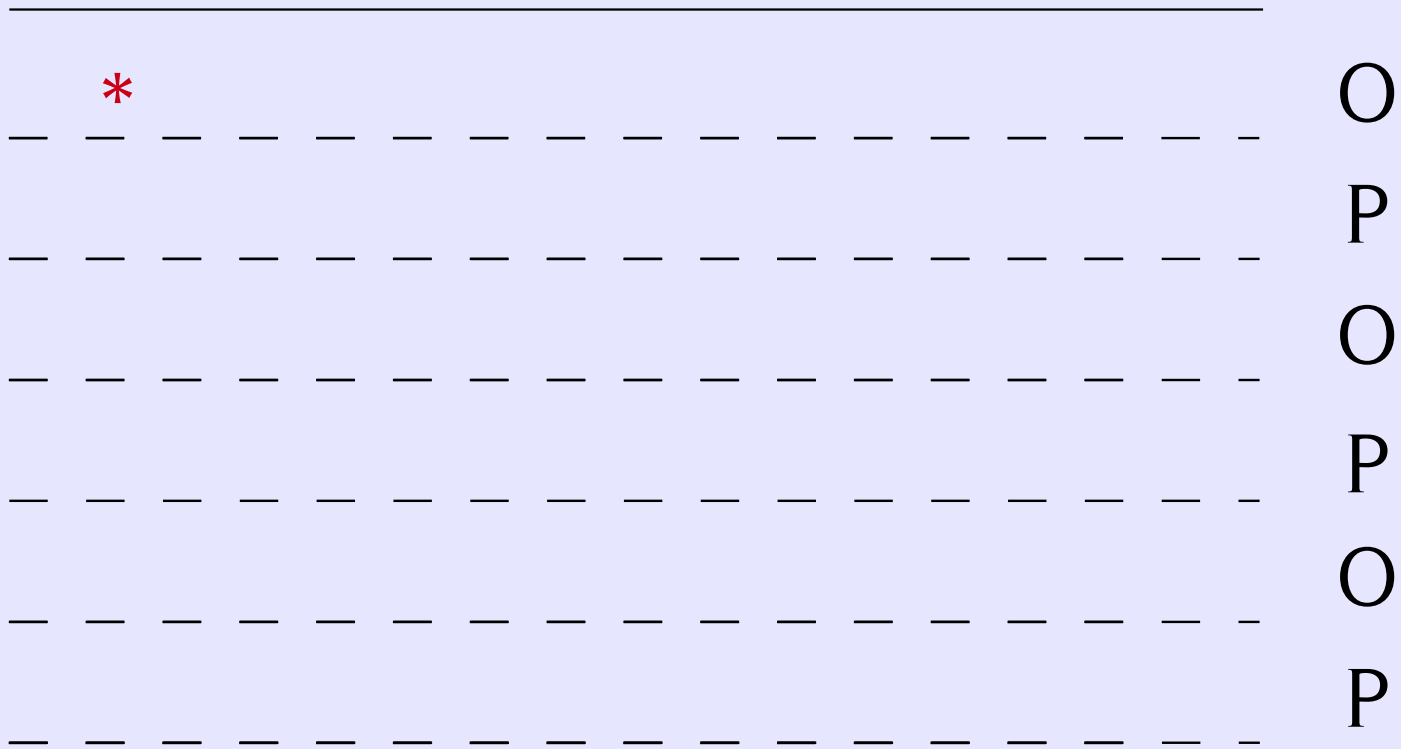
$1 \longrightarrow 1 \rightarrow \text{Var}$



Examples

$\vdash \lambda x.\text{newvar}(0) : \text{com} \rightarrow \text{intvar}$

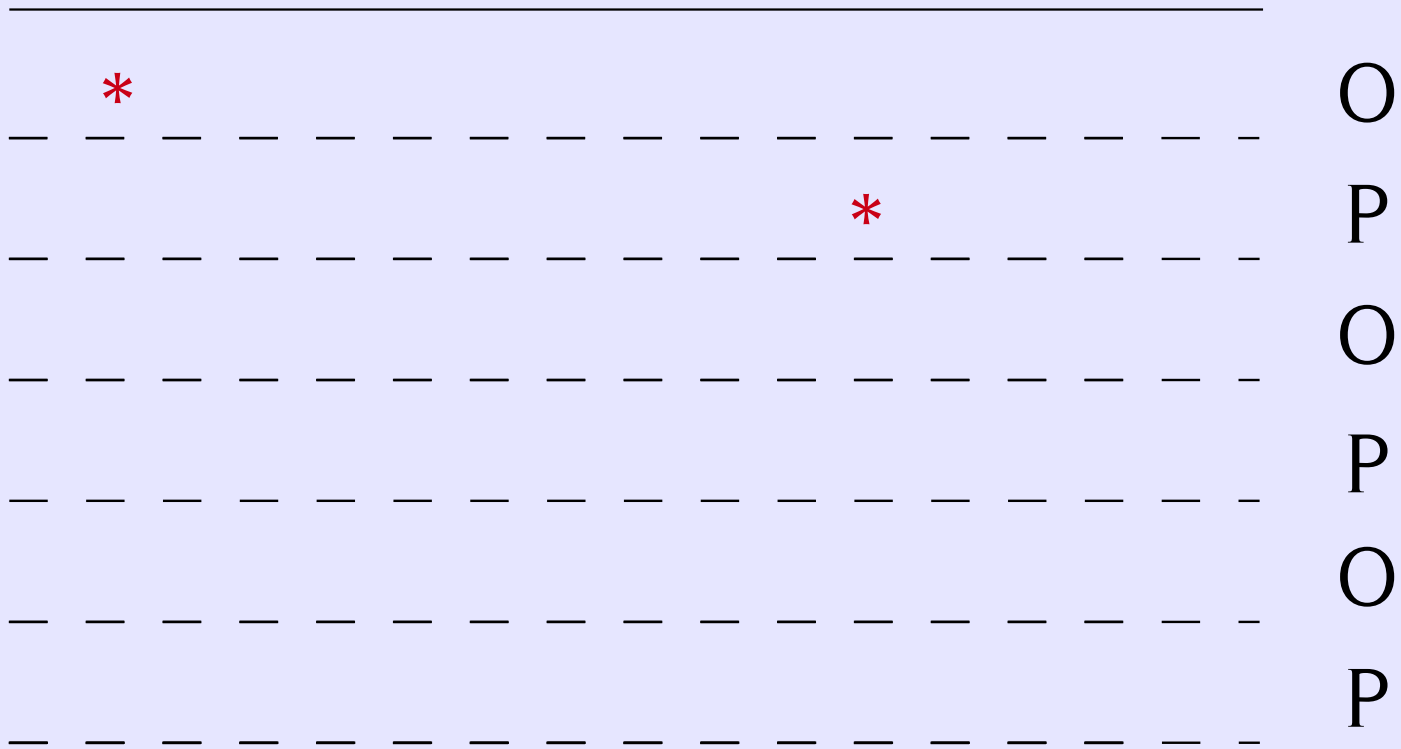
$1 \longrightarrow 1 \rightarrow \text{Var}$



Examples

$\vdash \lambda x.\text{newvar}(0) : \text{com} \rightarrow \text{intvar}$

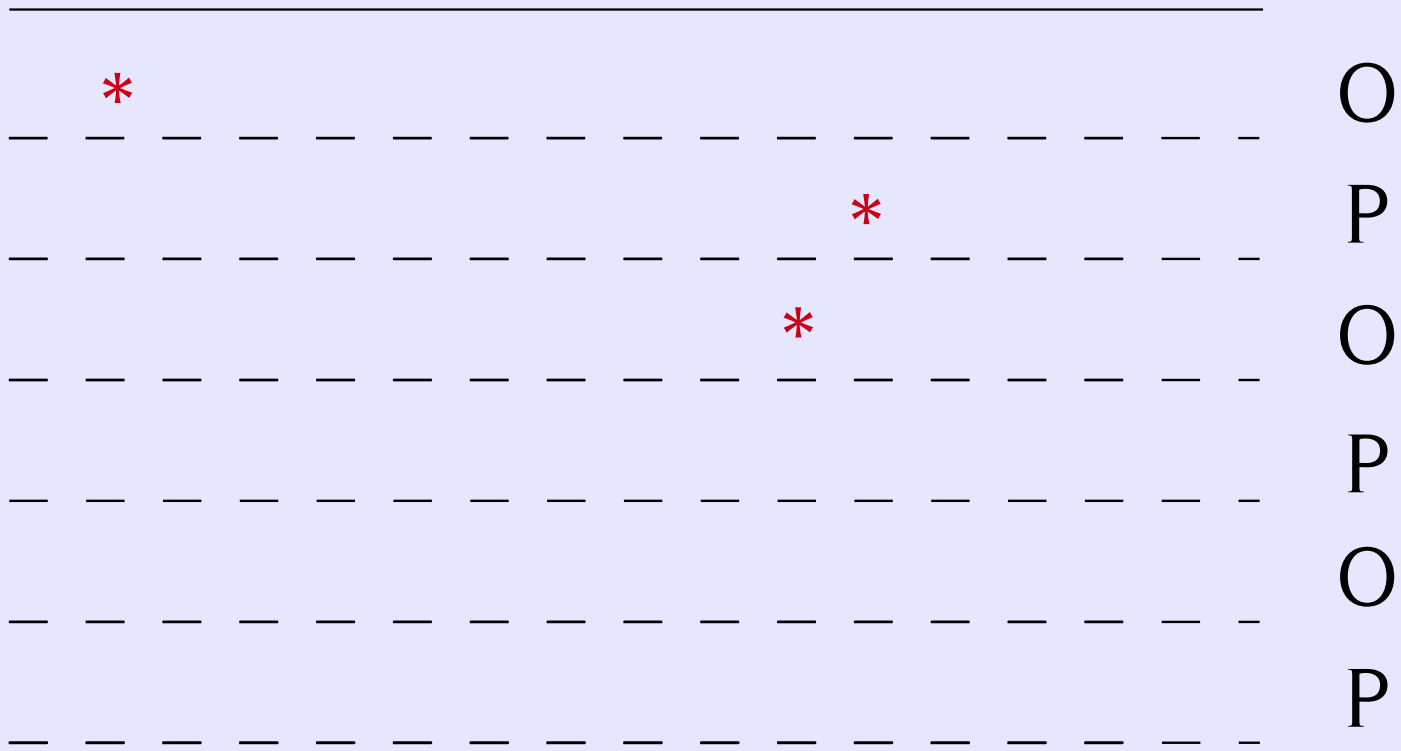
$1 \longrightarrow 1 \rightarrow \text{Var}$



Examples

$\vdash \lambda x.\text{newvar}(0) : \text{com} \rightarrow \text{intvar}$

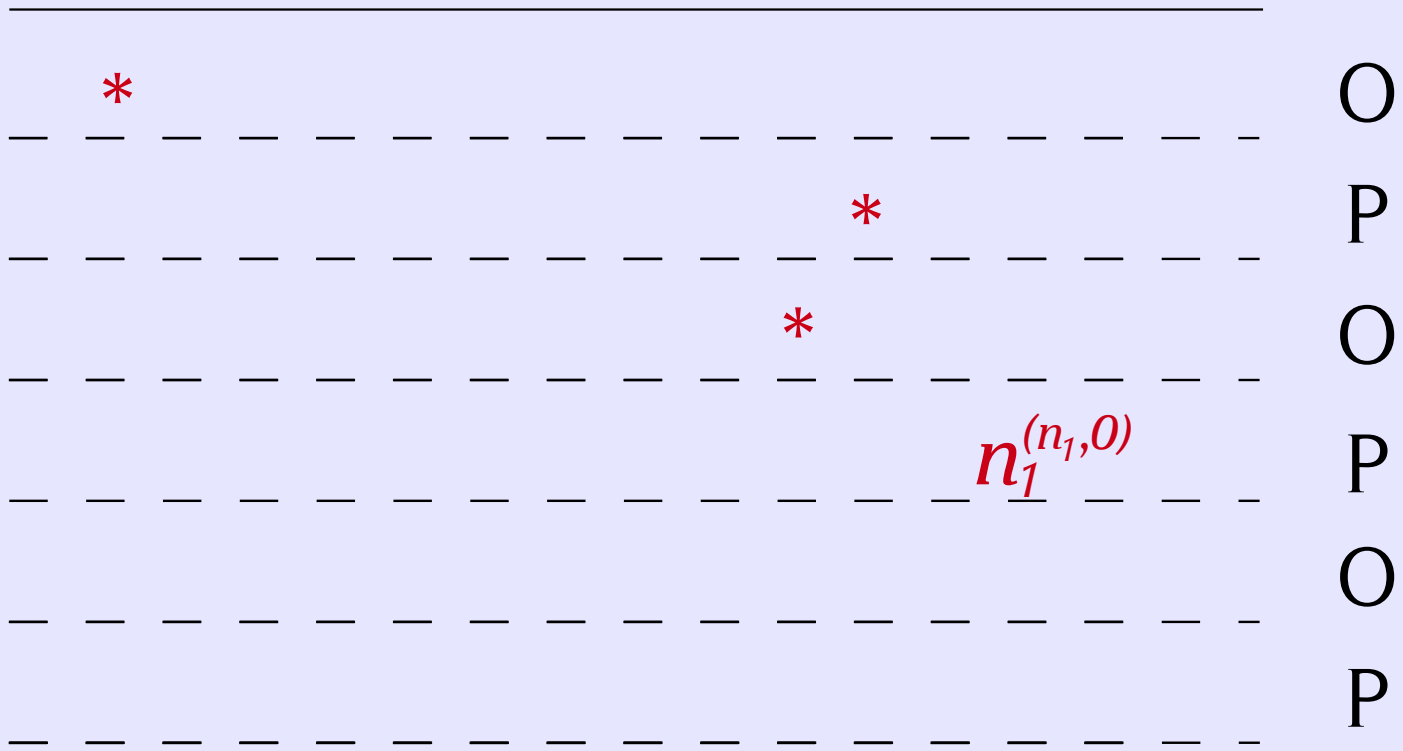
$1 \longrightarrow 1 \rightarrow \text{Var}$



Examples

$\vdash \lambda x.\text{newvar}(0) : \text{com} \rightarrow \text{intvar}$

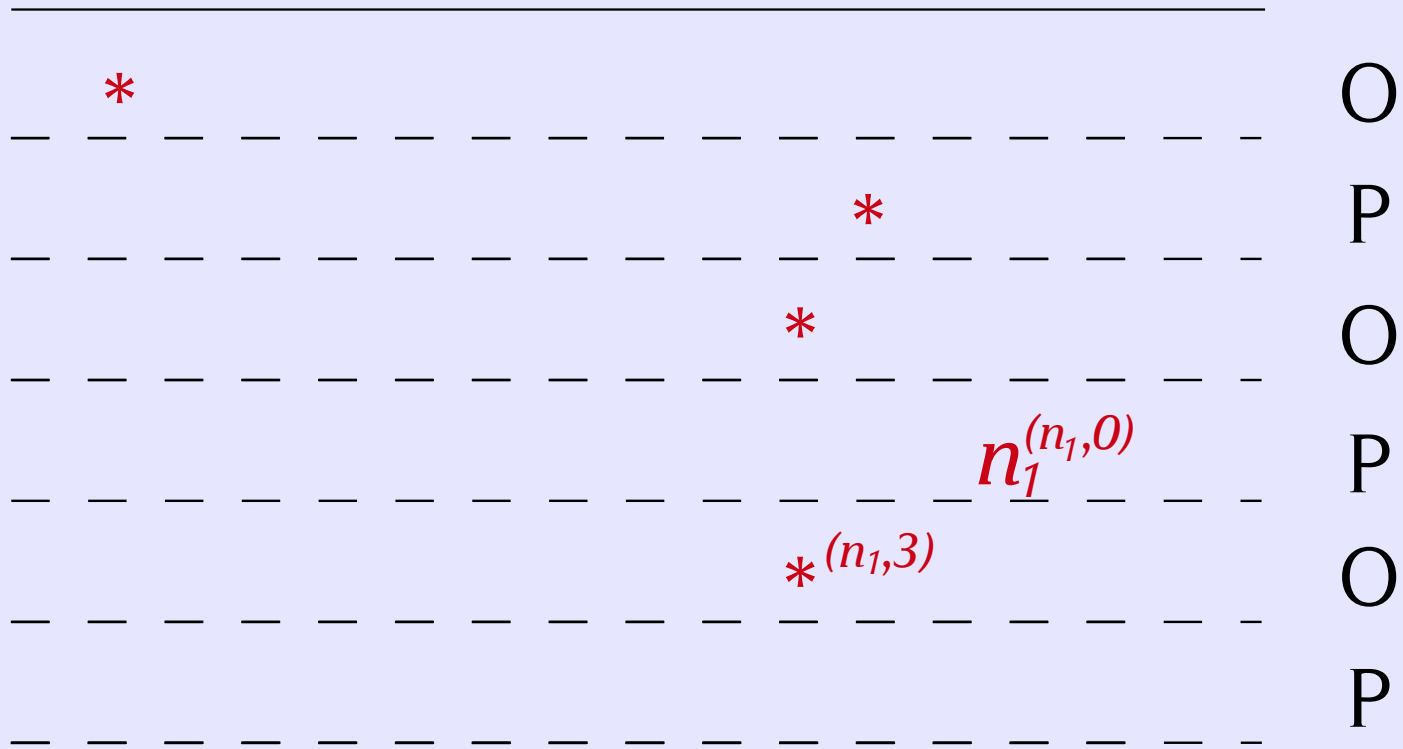
$1 \longrightarrow 1 \rightarrow \text{Var}$



Examples

$\vdash \lambda x. \text{newvar}(0) : \text{com} \rightarrow \text{intvar}$

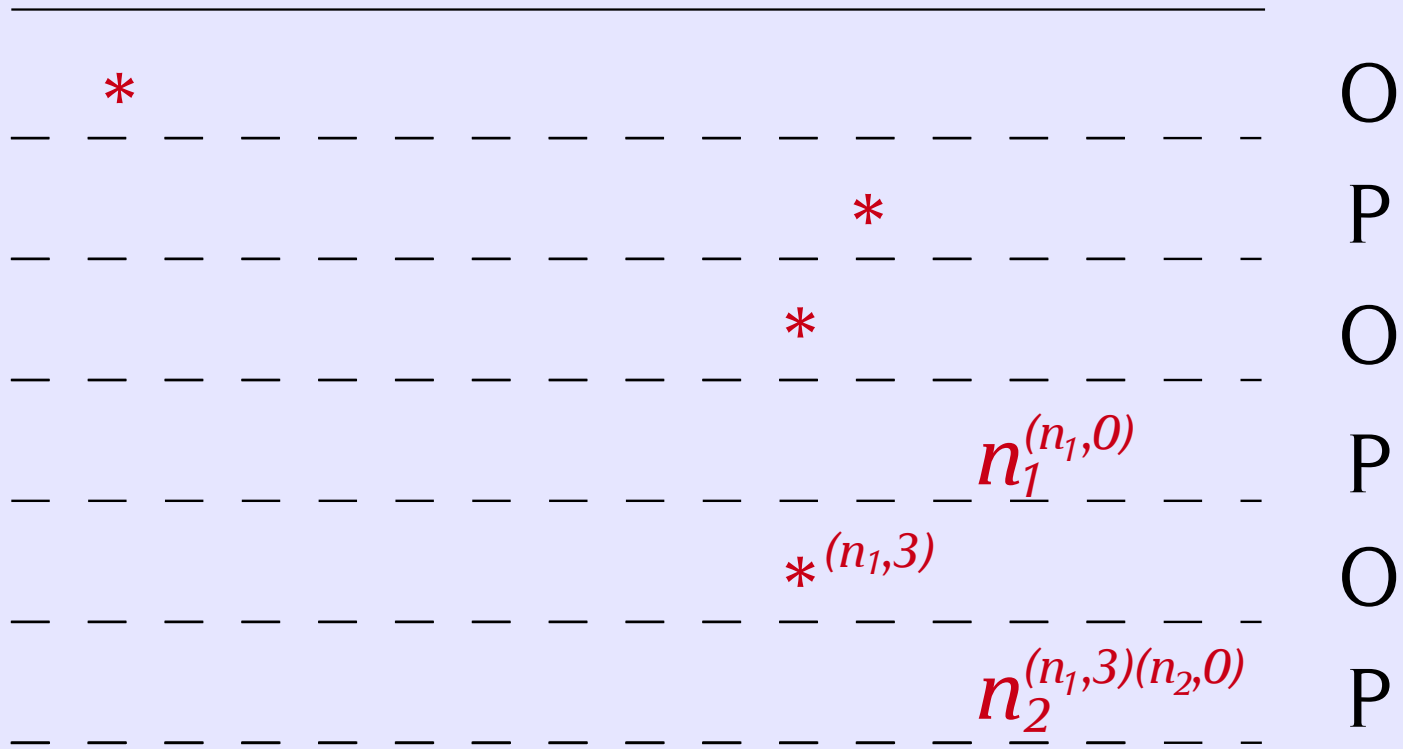
$1 \longrightarrow 1 \rightarrow \text{Var}$



Examples

$\vdash \lambda x. \text{newvar}(0) : \text{com} \rightarrow \text{intvar}$

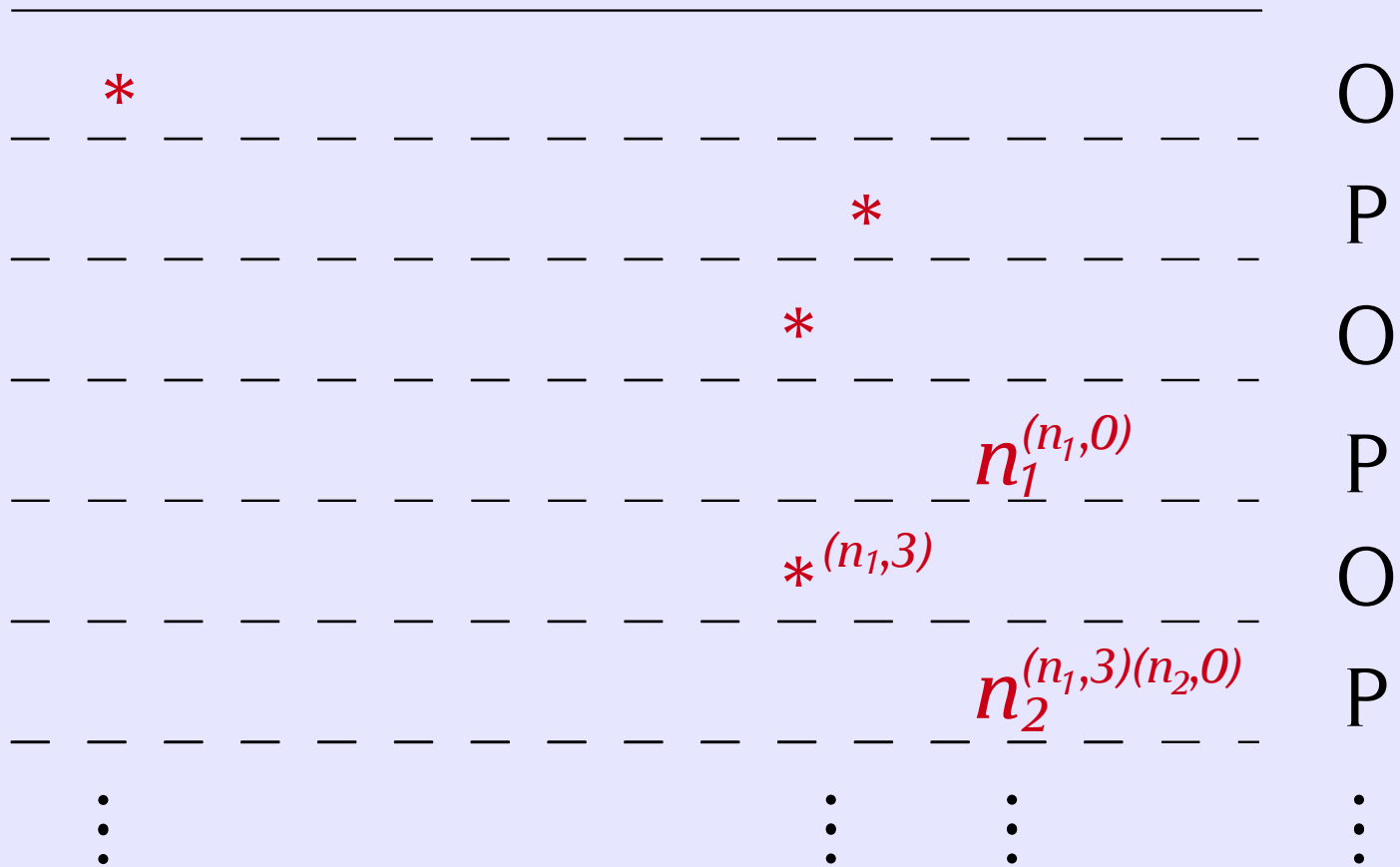
$1 \longrightarrow 1 \rightarrow \text{Var}$



Examples

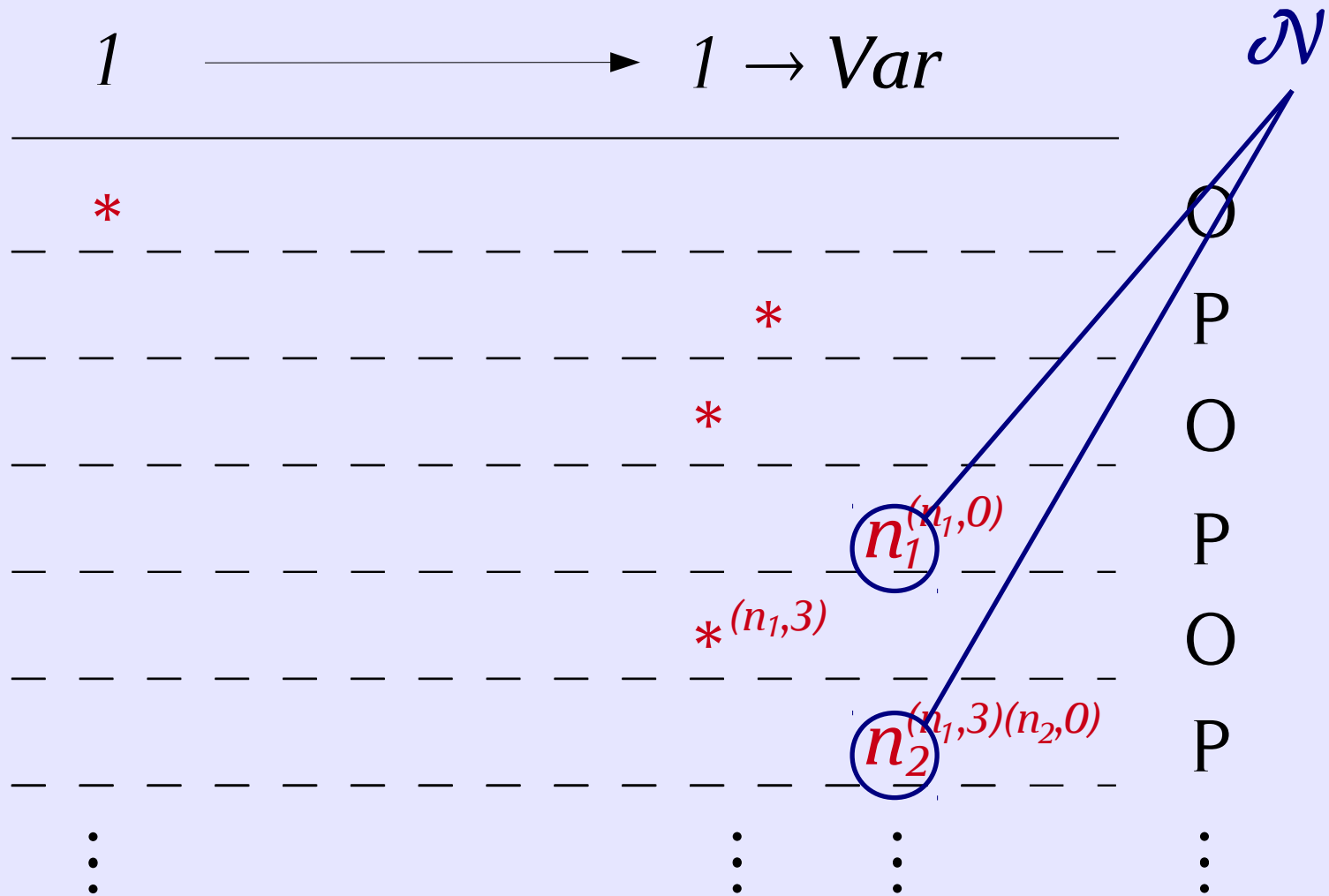
$\vdash \lambda x.\text{newvar}(0) : \text{com} \rightarrow \text{intvar}$

$1 \longrightarrow 1 \rightarrow \text{Var}$



Examples

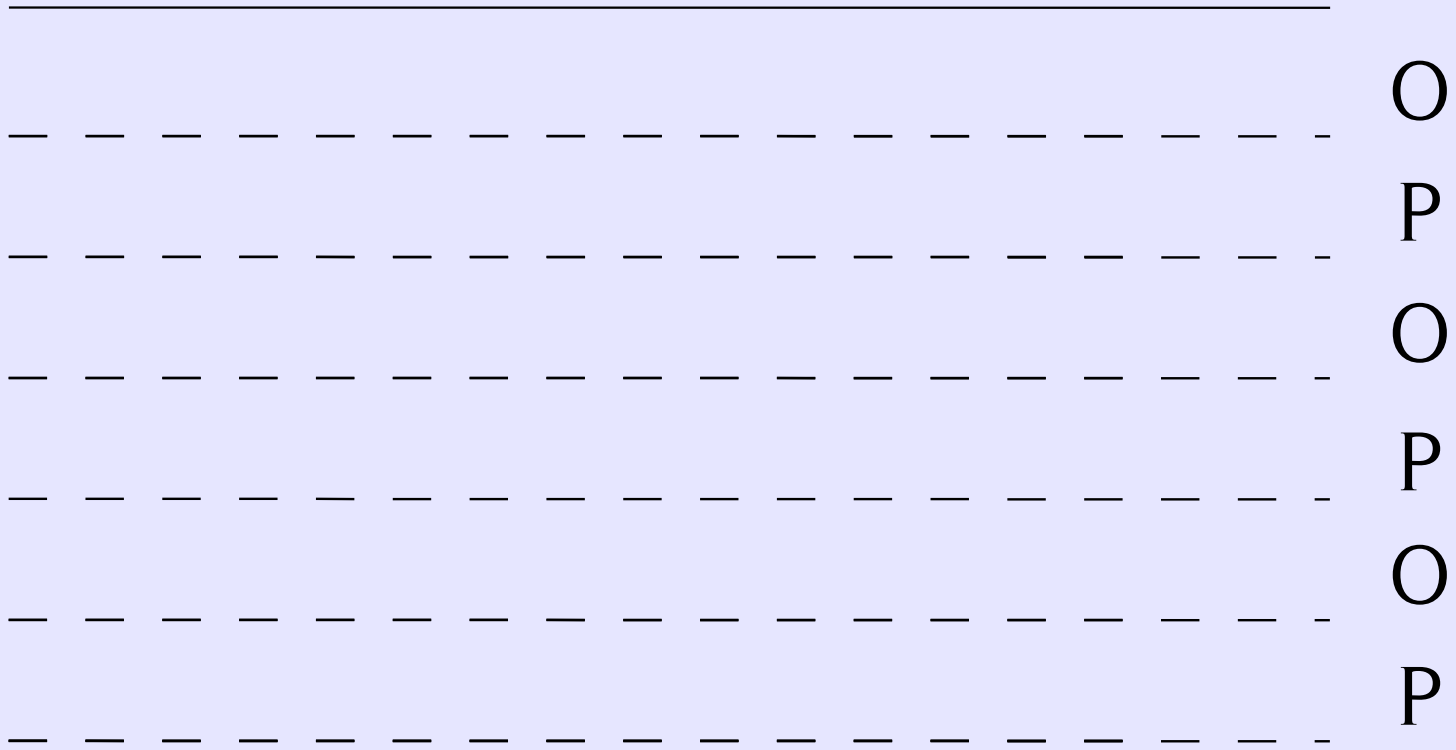
$\vdash \lambda x. \text{newvar}(0) : \text{com} \rightarrow \text{intvar}$



Examples

$x : \text{intvar} \vdash \lambda y. (x == y) : \text{intvar} \rightarrow \text{int}$

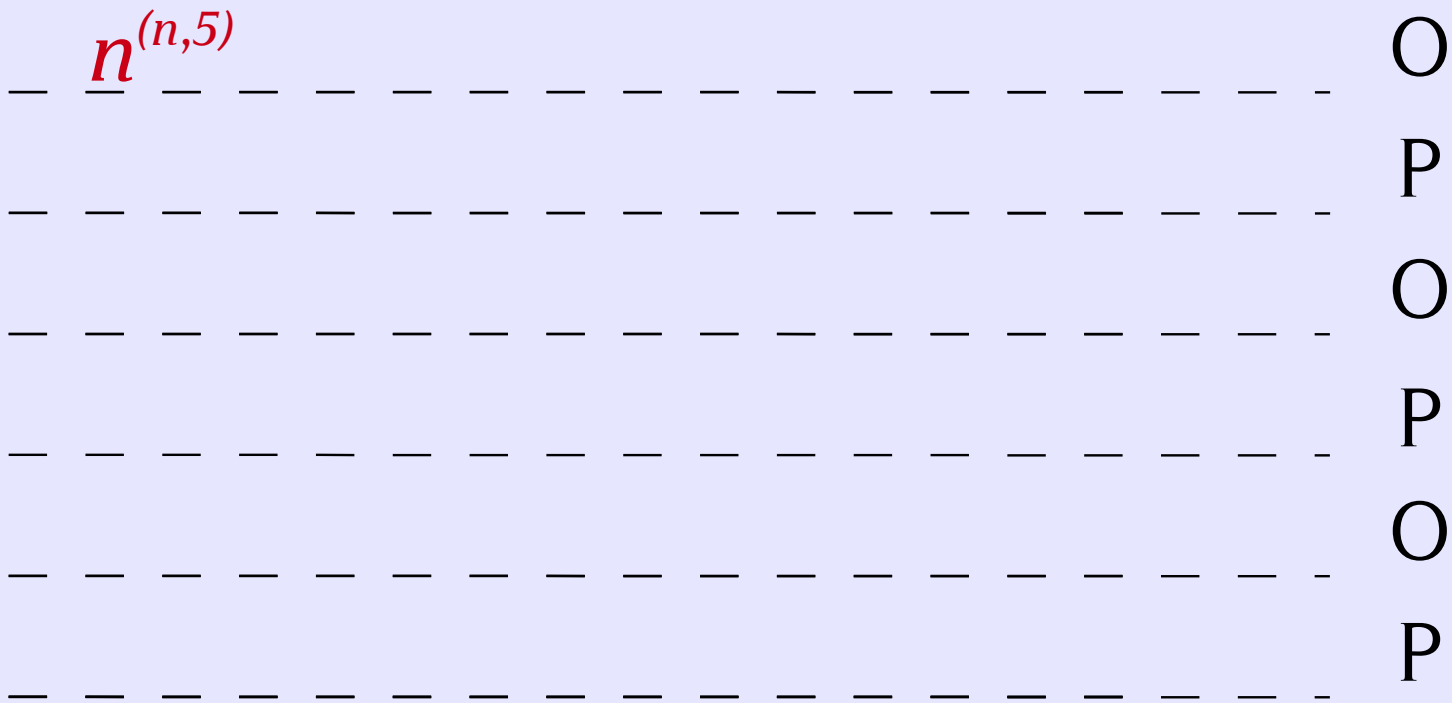
$\text{Var} \longrightarrow \text{Var} \rightarrow \text{Int}$



Examples

$x : \text{intvar} \vdash \lambda y. (x == y) : \text{intvar} \rightarrow \text{int}$

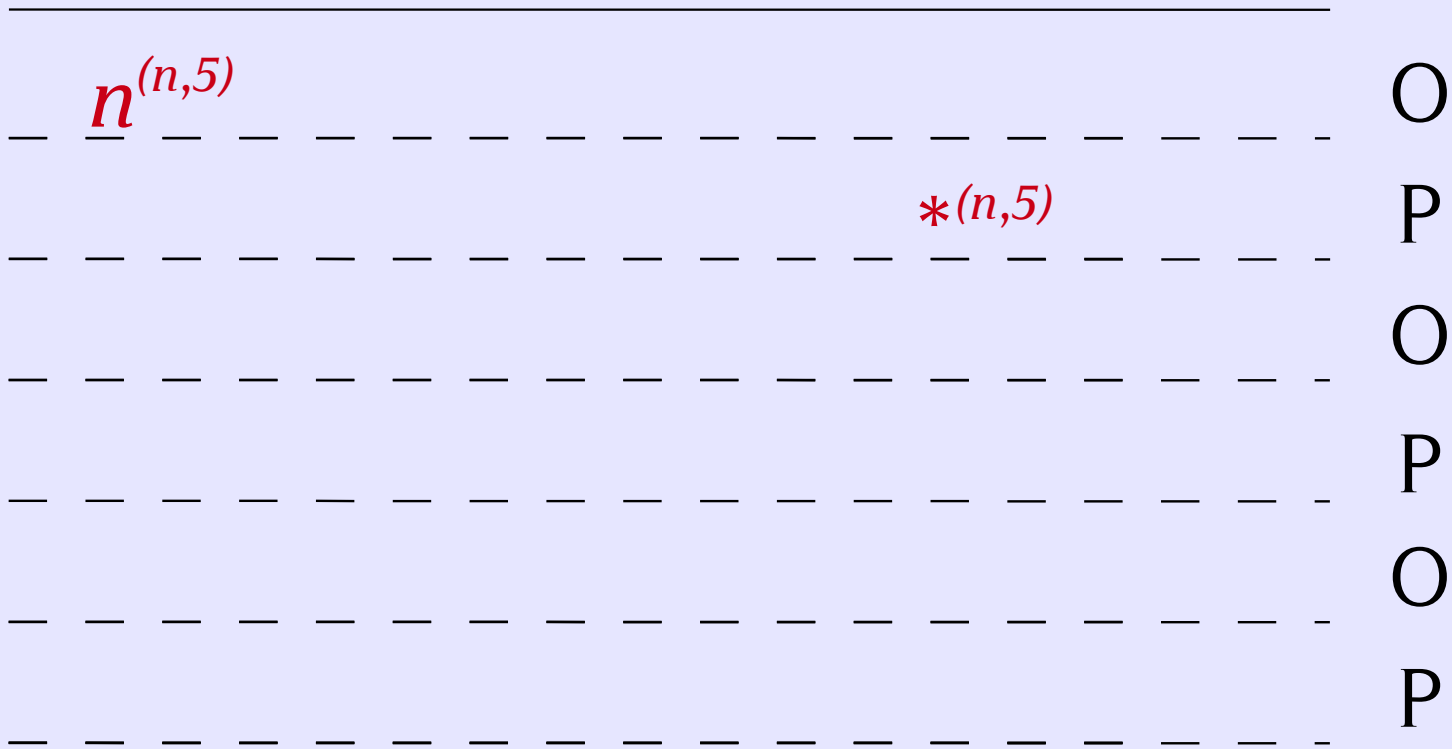
$Var \longrightarrow Var \rightarrow Int$



Examples

$x : \text{intvar} \vdash \lambda y. (x == y) : \text{intvar} \rightarrow \text{int}$

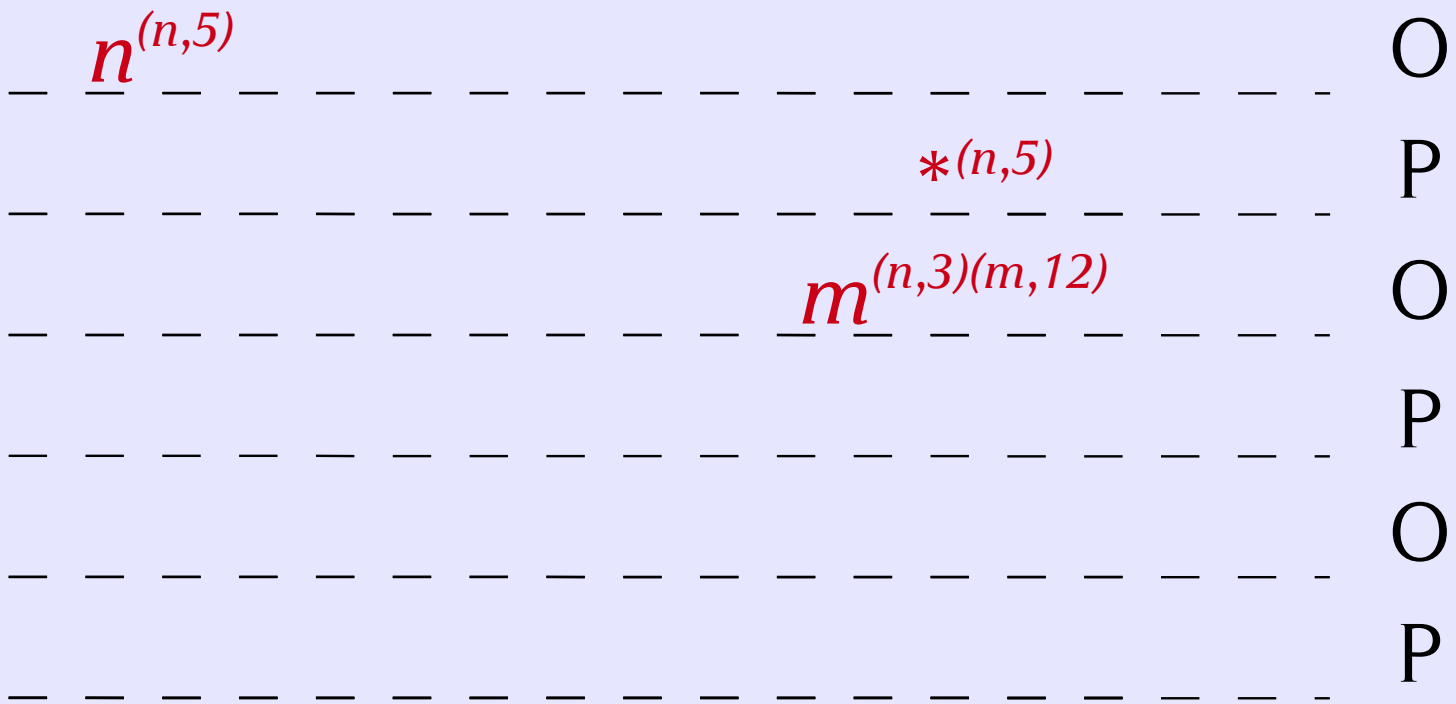
$Var \longrightarrow Var \rightarrow Int$



Examples

$x : \text{intvar} \vdash \lambda y. (x == y) : \text{intvar} \rightarrow \text{int}$

$\text{Var} \longrightarrow \text{Var} \rightarrow \text{Int}$



Examples

$x : \text{intvar} \vdash \lambda y. (x == y) : \text{intvar} \rightarrow \text{int}$

$Var \longrightarrow Var \rightarrow Int$

| | | |
|-------------|-------------------|---|
| $n^{(n,5)}$ | | O |
| | $*(n,5)$ | P |
| | $m^{(n,3)(m,12)}$ | O |
| | $o^{(n,3)(m,12)}$ | P |
| | | O |
| | | P |

Examples

$x : \text{intvar} \vdash \lambda y. (x == y) : \text{intvar} \rightarrow \text{int}$

$Var \longrightarrow Var \rightarrow Int$

| | | |
|-------------|-------------------|---|
| $n^{(n,5)}$ | | O |
| | $*(n,5)$ | P |
| | $m^{(n,3)(m,12)}$ | O |
| | $o^{(n,3)(m,12)}$ | P |
| | $n^{(n,13)}$ | O |
| | | P |

Examples

$x : \text{intvar} \vdash \lambda y. (x == y) : \text{intvar} \rightarrow \text{int}$

$Var \longrightarrow Var \rightarrow Int$

| | | |
|-------------|-------------------|---|
| $n^{(n,5)}$ | | O |
| | $*(n,5)$ | P |
| | $m^{(n,3)(m,12)}$ | O |
| | $o^{(n,3)(m,12)}$ | P |
| | $n^{(n,13)}$ | O |
| | $1^{(n,13)}$ | P |

Examples

$x : \text{intvar} \vdash \lambda y. (x == y) : \text{intvar} \rightarrow \text{int}$

$Var \longrightarrow Var \rightarrow Int$

| | | | |
|-------------|----------|-------------------|----------|
| $n^{(n,5)}$ | | | O |
| | | $*(n,5)$ | P |
| | | $m^{(n,3)(m,12)}$ | O |
| | | $o^{(n,3)(m,12)}$ | P |
| | | $n^{(n,13)}$ | O |
| | | $1^{(n,13)}$ | P |
| \vdots | \vdots | \vdots | \vdots |

Games with new resources

Models of realistic programs



Games with new resources

Models of realistic programs

- General name-features [Tz.07, Tz.08]

Games with new resources

Models of realistic programs

- General name-features [Tz.07, Tz.08]

$$\llbracket A \rightarrow B \rrbracket = \llbracket A \rrbracket \Rightarrow (S \Rightarrow \llbracket B \rrbracket \otimes S)$$

$$S = \bigotimes_A (N_A \Rightarrow \llbracket A \rrbracket)$$

Games with new resources

Models of realistic programs

- General name-features [Tz.07, Tz.08]

$$\llbracket A \rightarrow B \rrbracket = \llbracket A \rrbracket \Rightarrow (S \Rightarrow \llbracket B \rrbracket \otimes S)$$

$$S = \bigotimes_A (N_A \Rightarrow \llbracket A \rrbracket)$$

$$M \cong N \iff \llbracket M \rrbracket \cong \llbracket N \rrbracket$$

Games with new resources

Models of realistic programs

- General name-features [Tz.07, Tz.08]
- Direct models [Laird 06, Laird 08, Murawski&Tz.09, Mur.&Tz.11]

Games with new resources

Models of realistic programs

- General name-features [Tz.07, Tz.08]
- Direct models [Laird 06, Laird 08, Murawski&Tz.09, Mur.&Tz.11]

$$M \cong N \iff \text{comp}(\llbracket M \rrbracket) = \text{comp}(\llbracket N \rrbracket)$$

Games with new resources

Models of realistic programs

- General name-features [Tz.07, Tz.08]
- Direct models [Laird 06, Laird 08, Murawski&Tz.09, Mur.&Tz.11]

Algorithmic games

$$M \cong N \iff \text{comp}(\llbracket M \rrbracket) = \text{comp}(\llbracket N \rrbracket)$$

Fresh-register automata [POPL'11]

$\lambda z . \text{newvar}(0) \mapsto \{ * * * n_1 * n_2 * n_3 \dots \mid n_i \text{'s distinct} \}$

Fresh-register automata [POPL'11]

$\lambda z . \text{newvar}(0) \mapsto \{ * * * n_1 * n_2 * n_3 \dots \mid n_i \text{'s distinct} \}$

Automata with names

- Infinite alphabet \mathcal{N}
- *Freshness* recognition

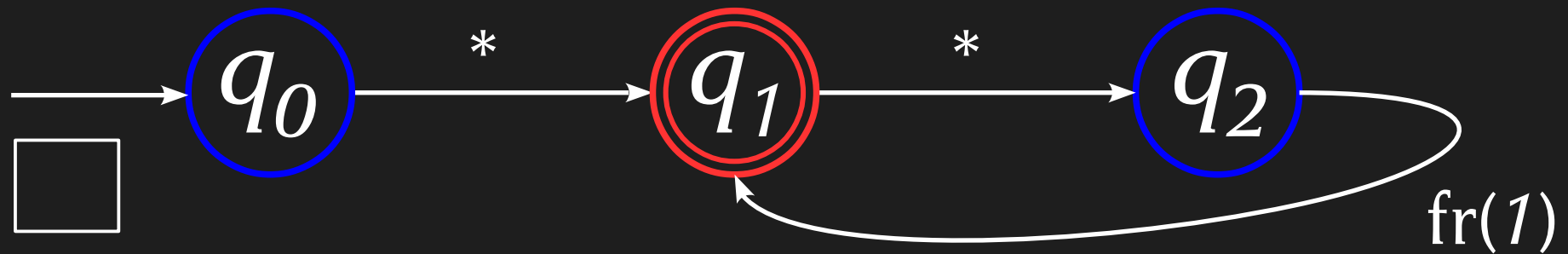
Fresh-register automata [POPL'11]

$\lambda z . \text{newvar}(0) \mapsto \{ * * * n_1 * n_2 * n_3 \dots \mid n_i \text{'s distinct} \}$

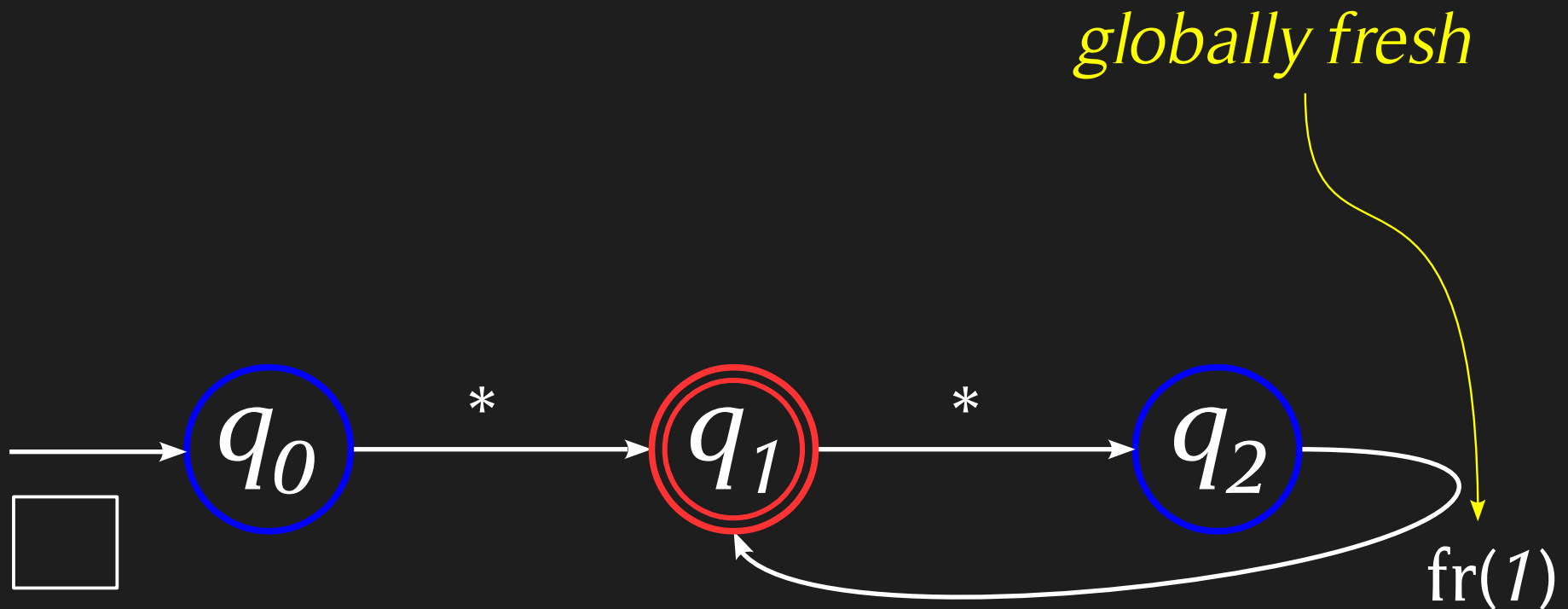
Automata with names

- Finite-state machines with registers
- Infinite alphabet \mathcal{N}
- *Freshness* recognition

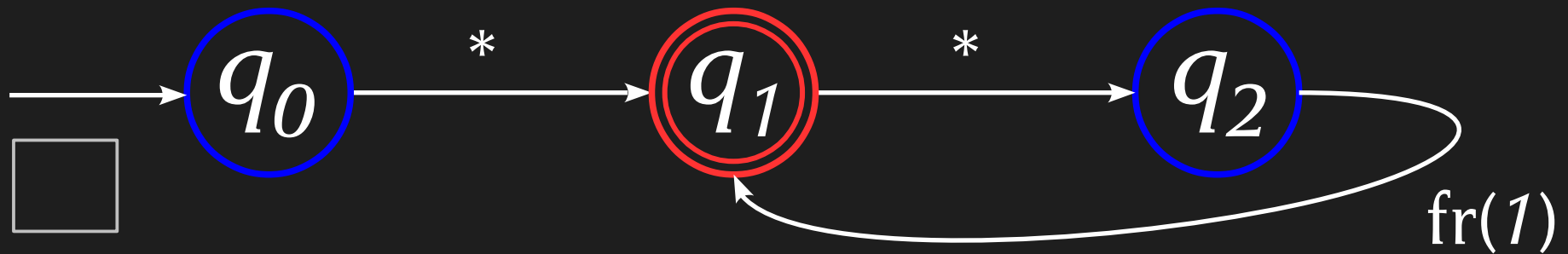
Fresh-register automata



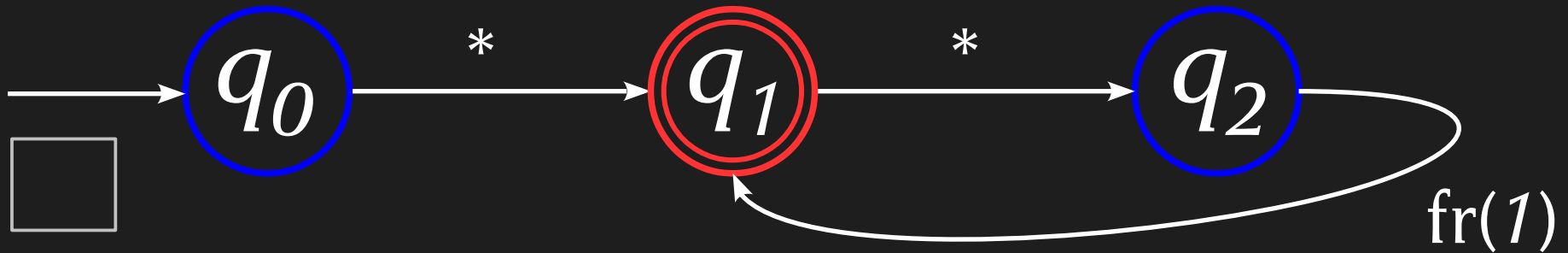
Fresh-register automata



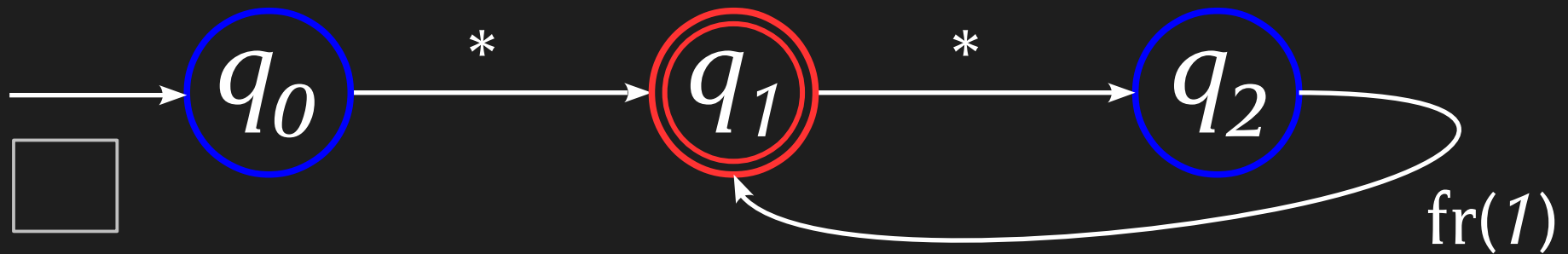
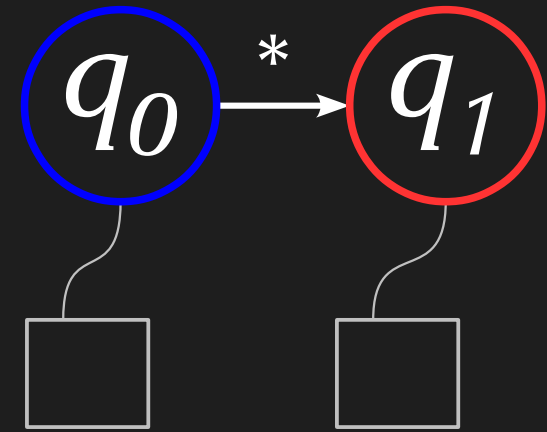
Fresh-register automata



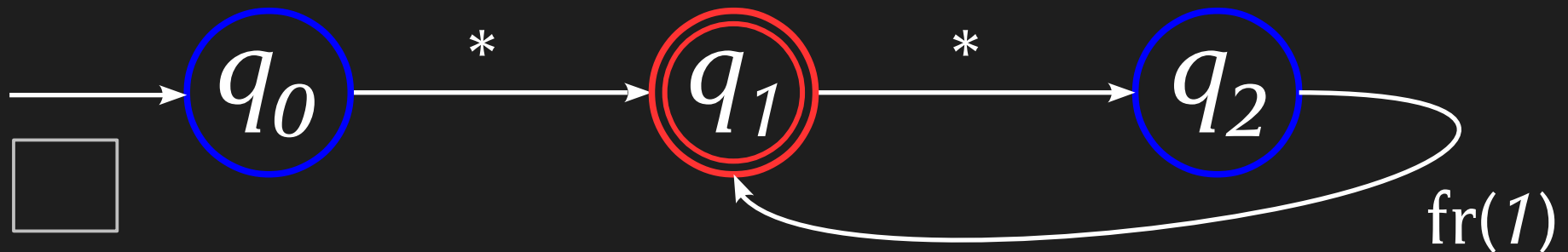
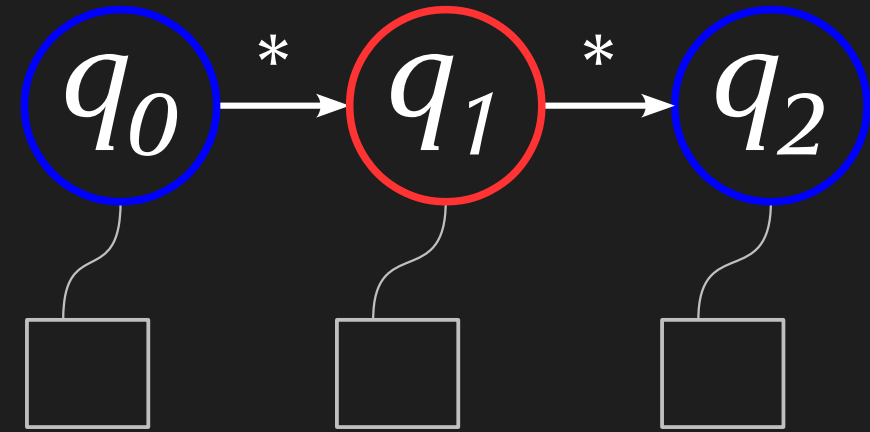
Fresh-register automata



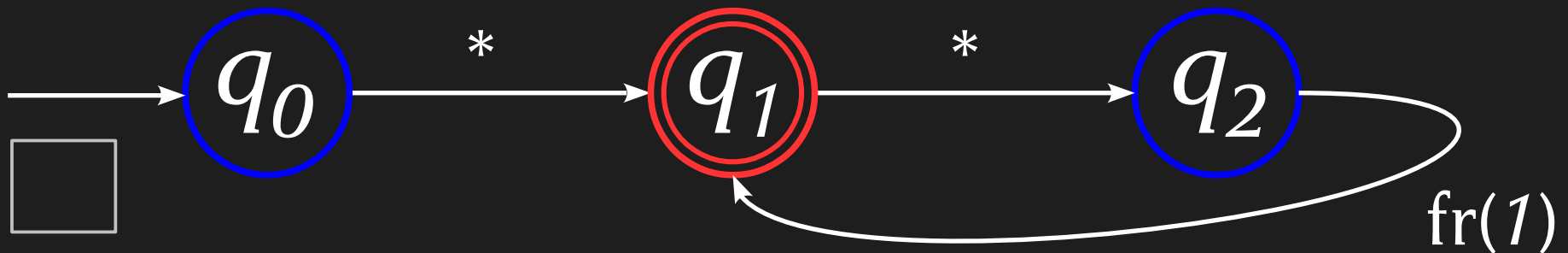
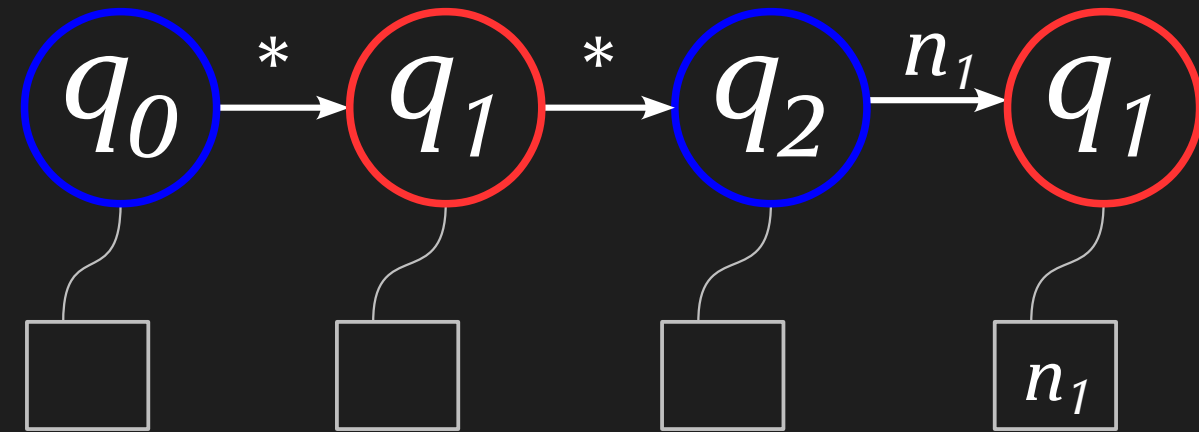
Fresh-register automata



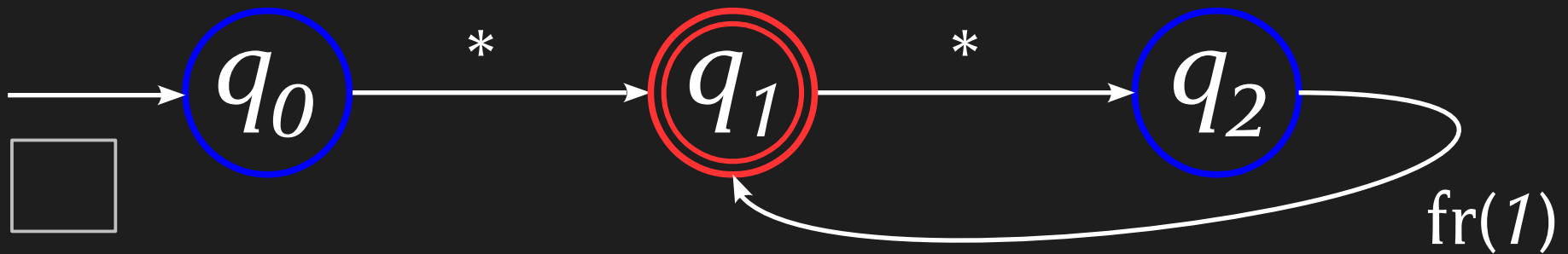
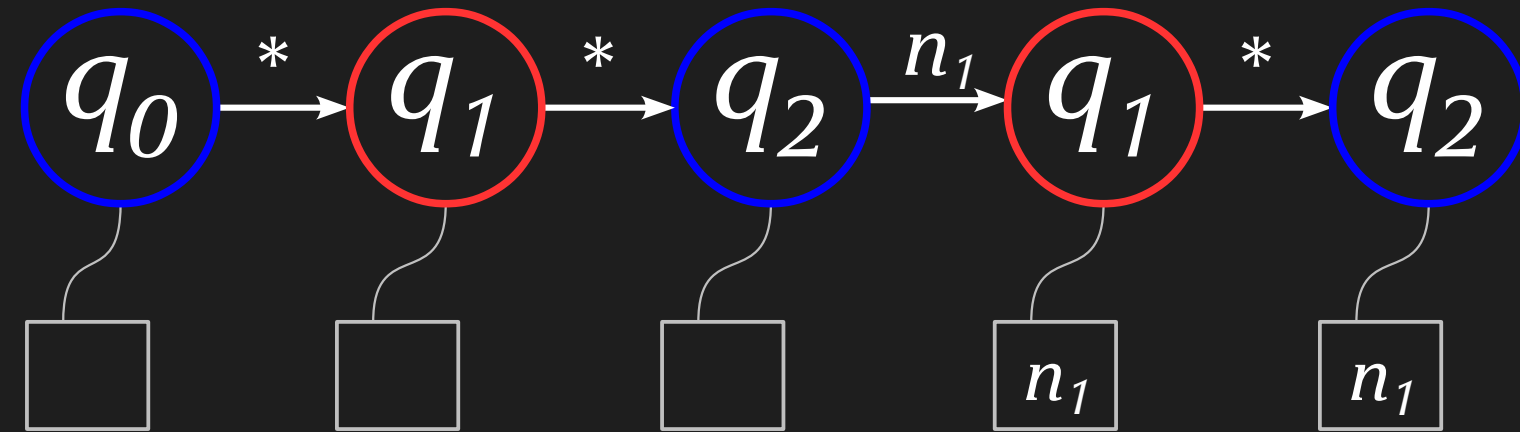
Fresh-register automata



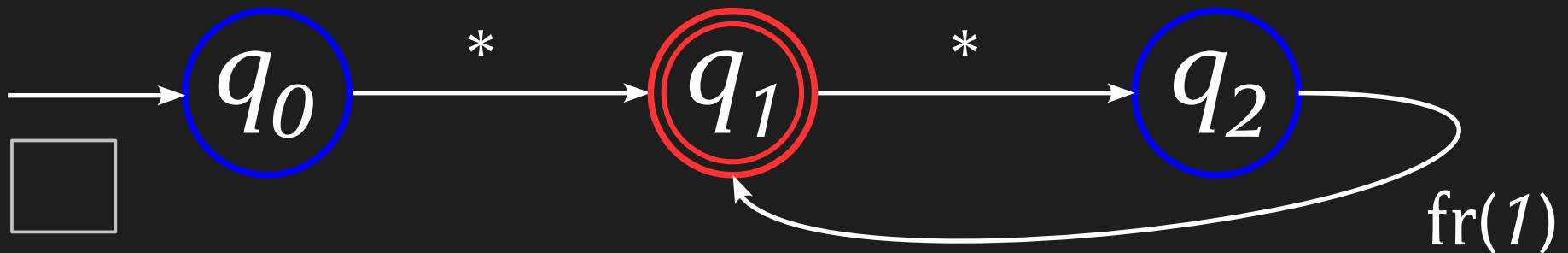
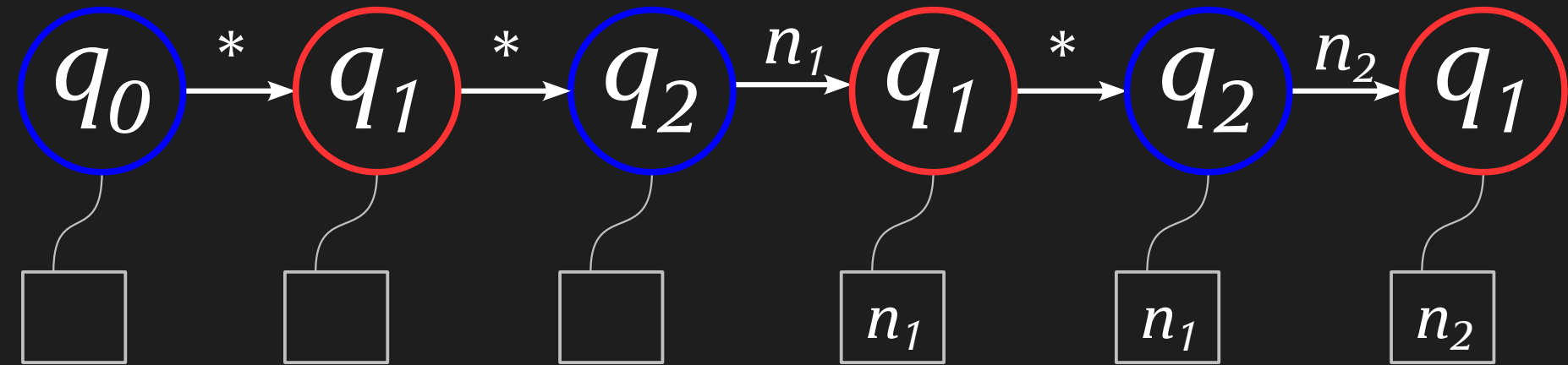
Fresh-register automata



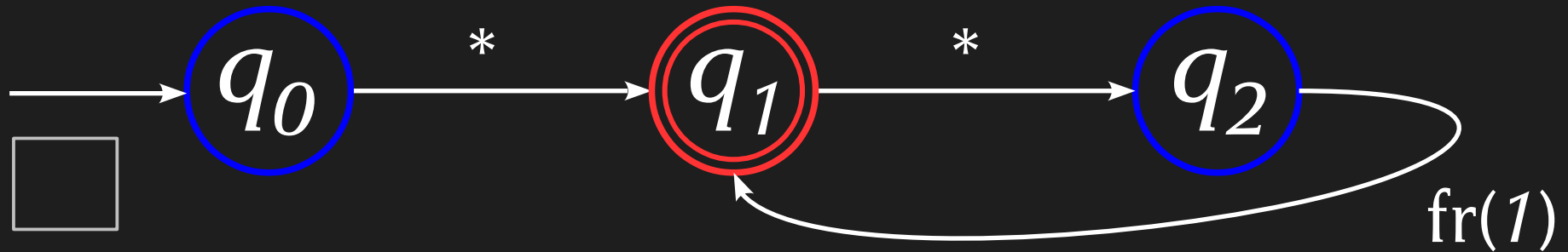
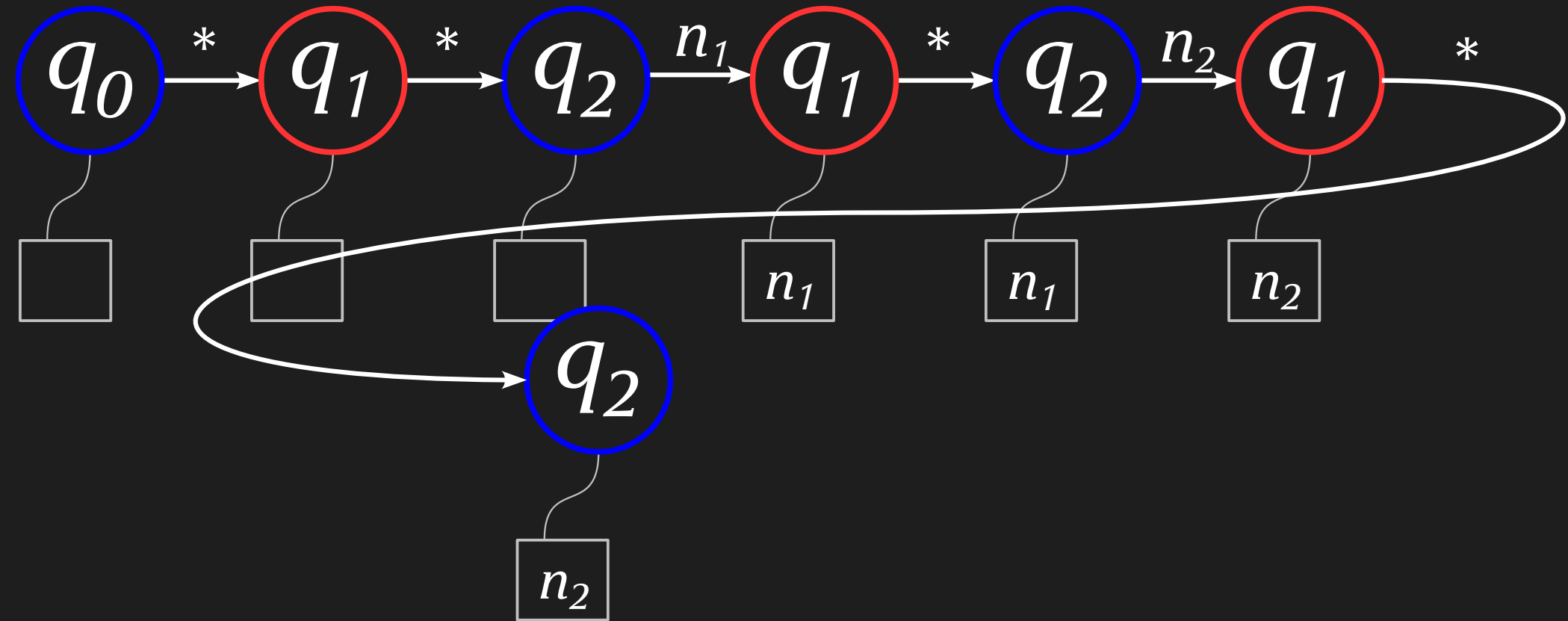
Fresh-register automata



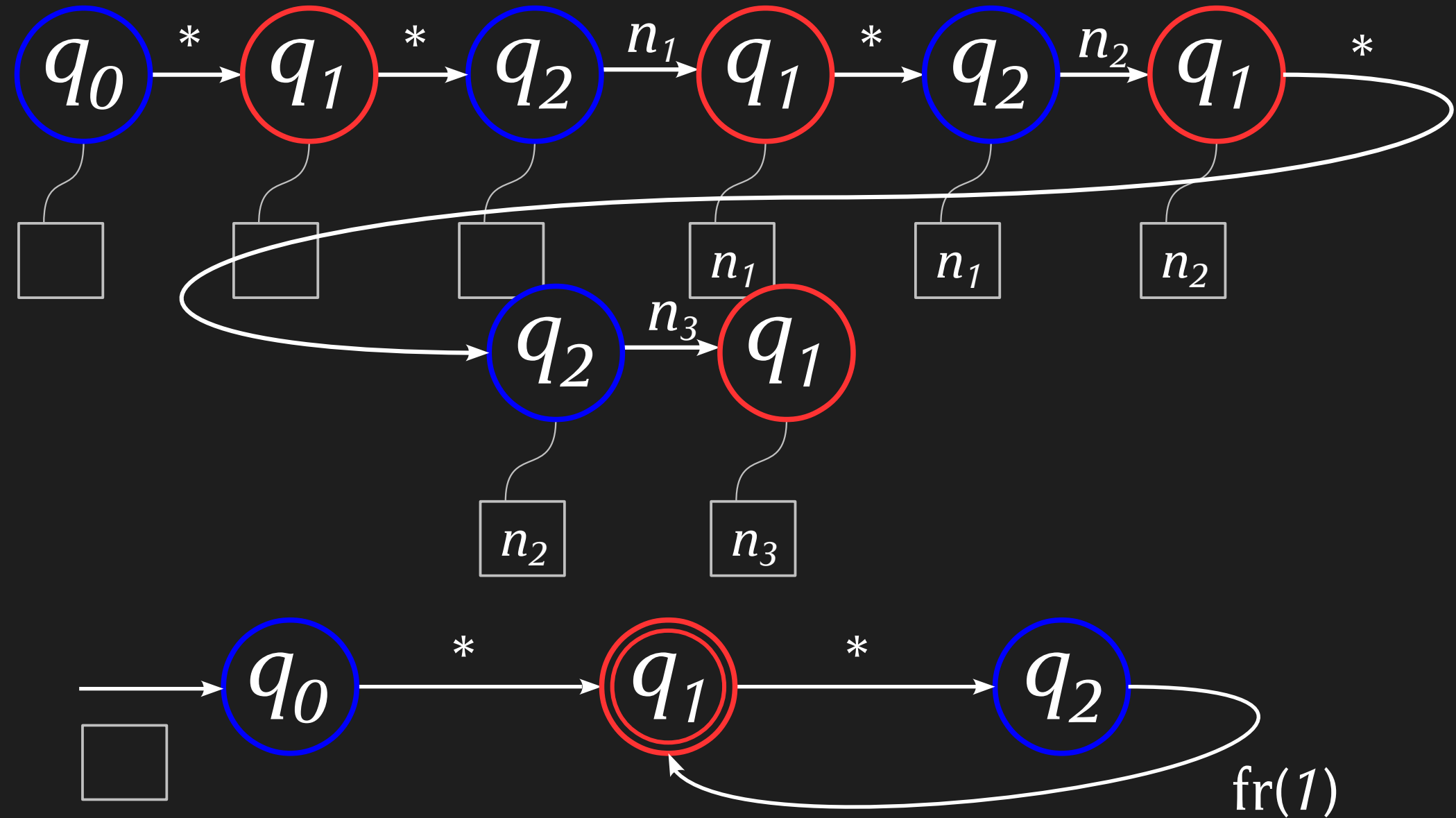
Fresh-register automata



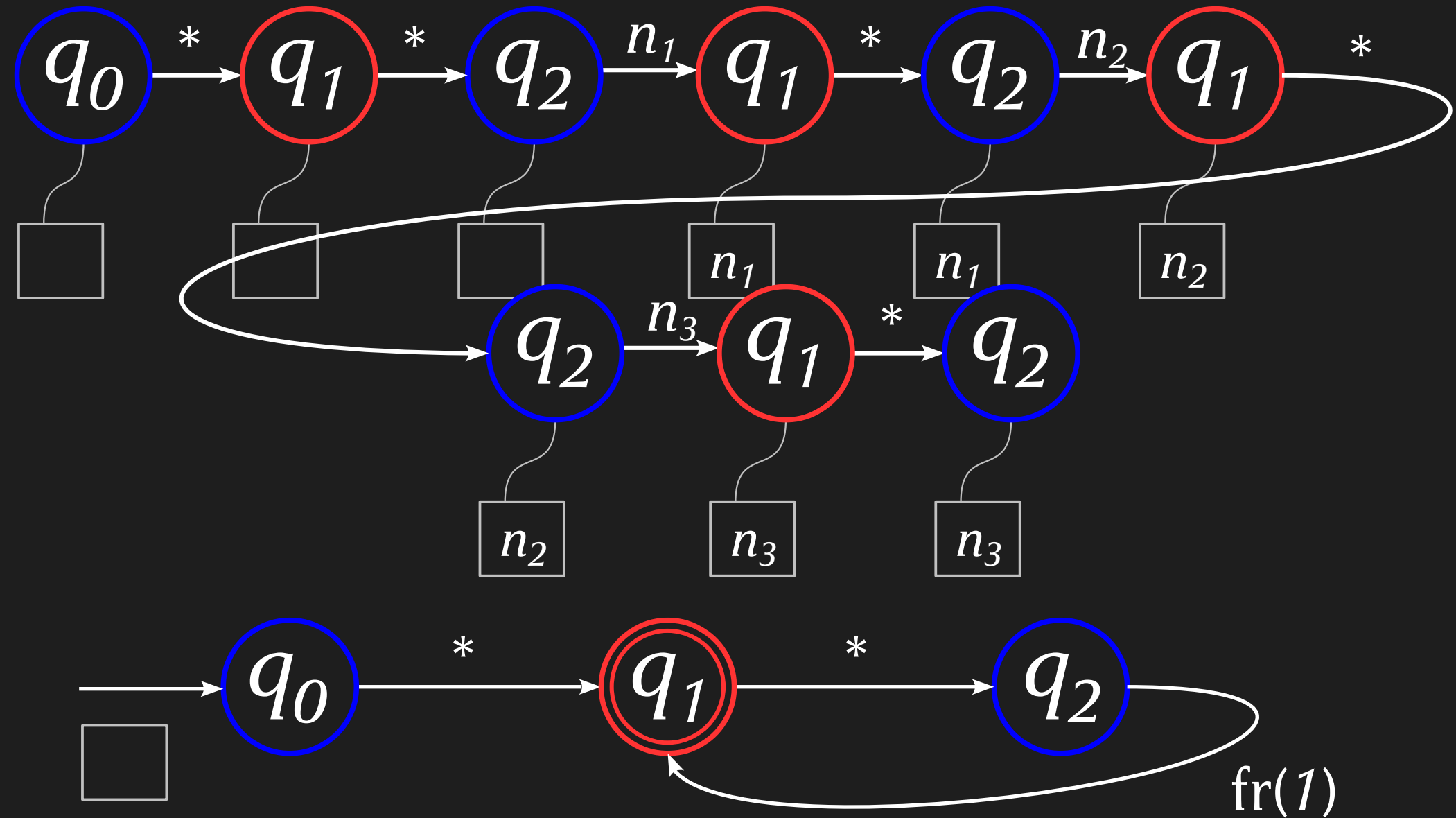
Fresh-register automata



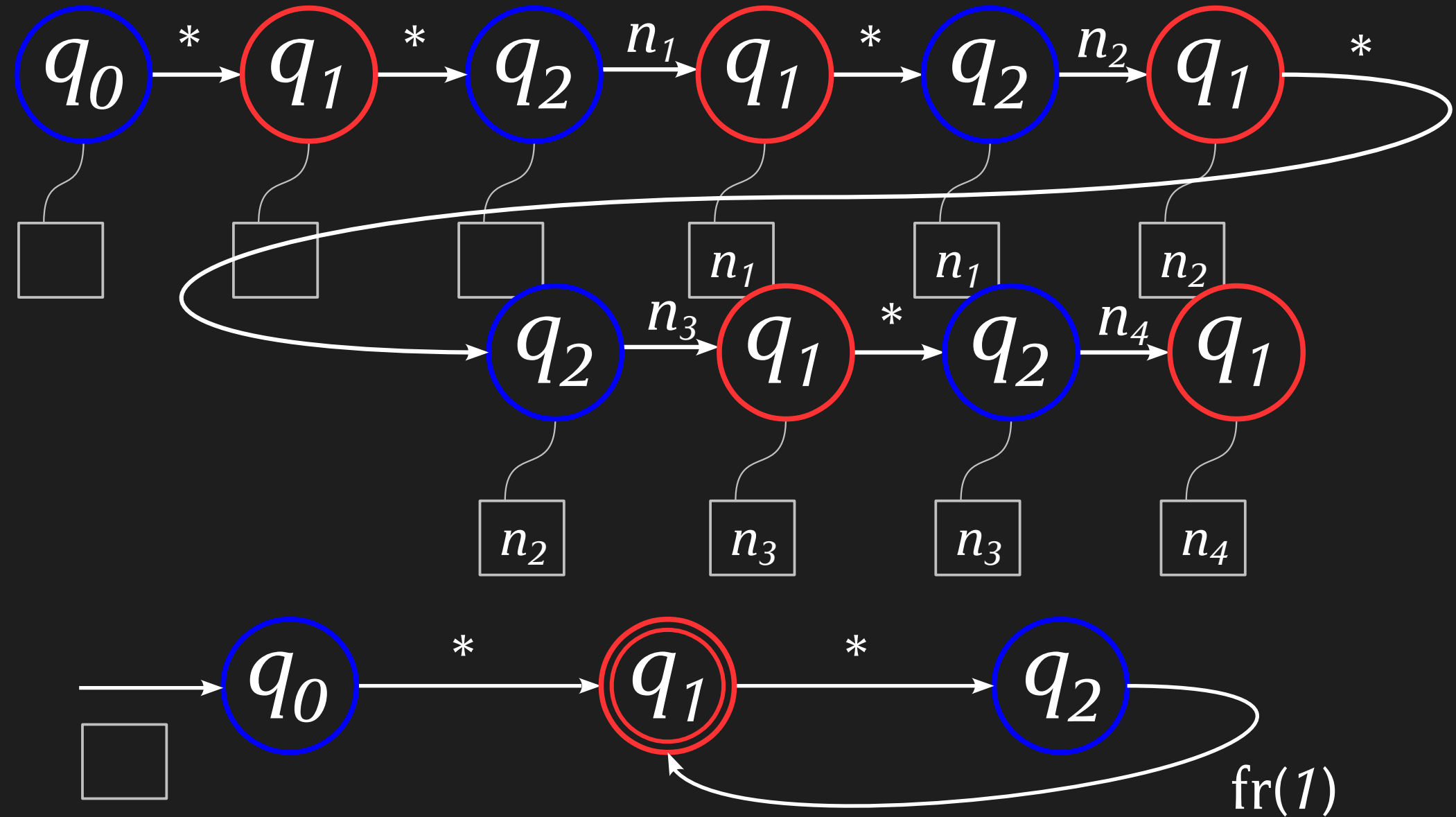
Fresh-register automata



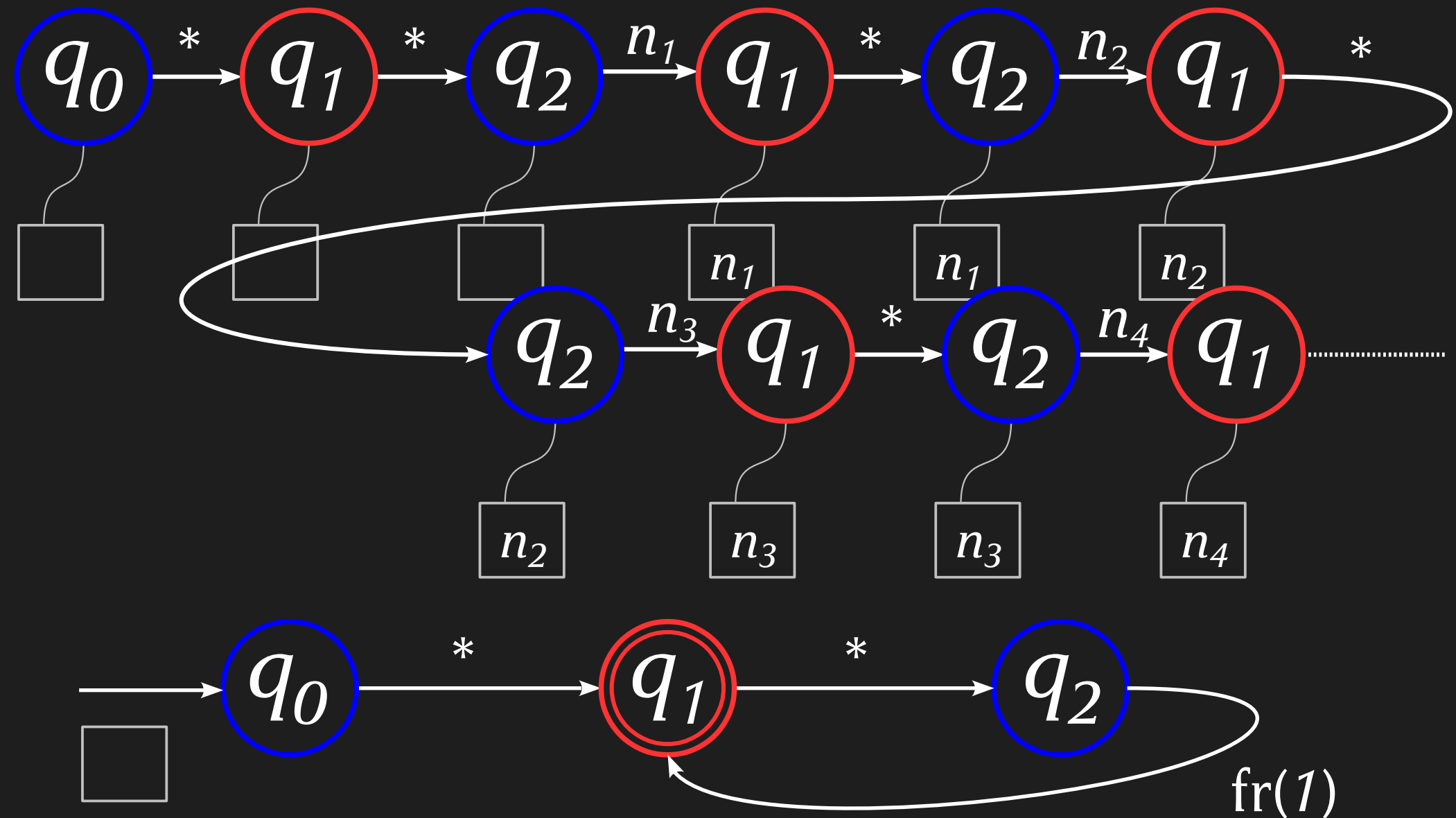
Fresh-register automata



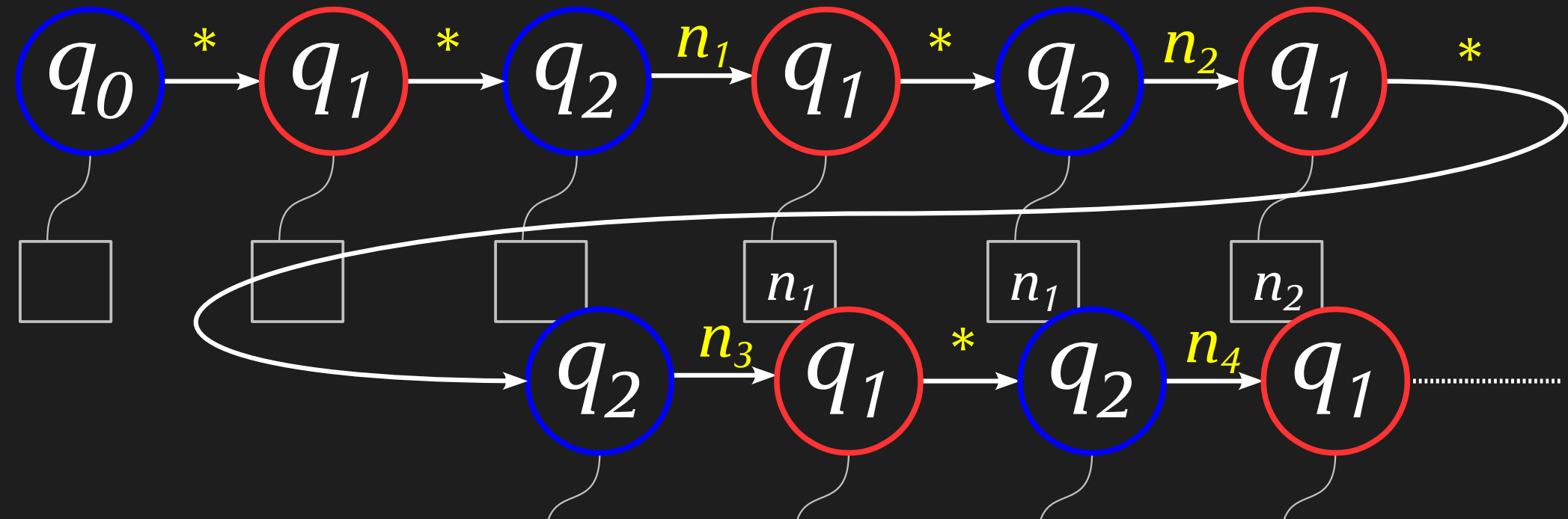
Fresh-register automata



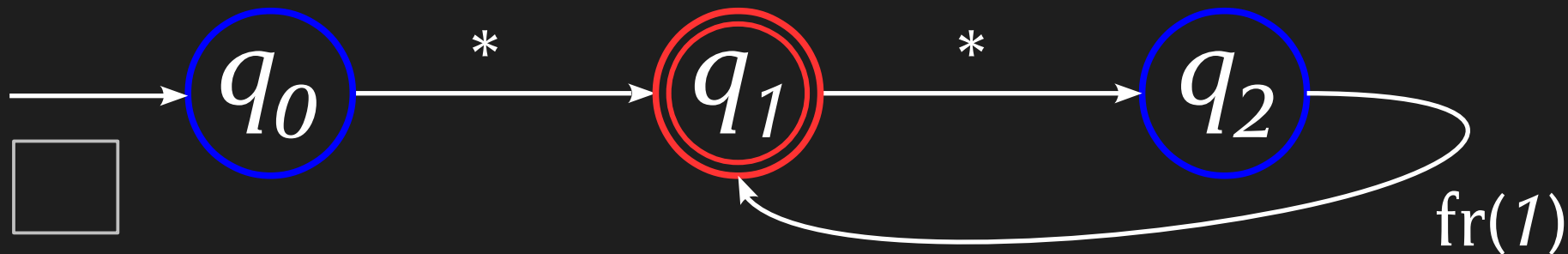
Fresh-register automata



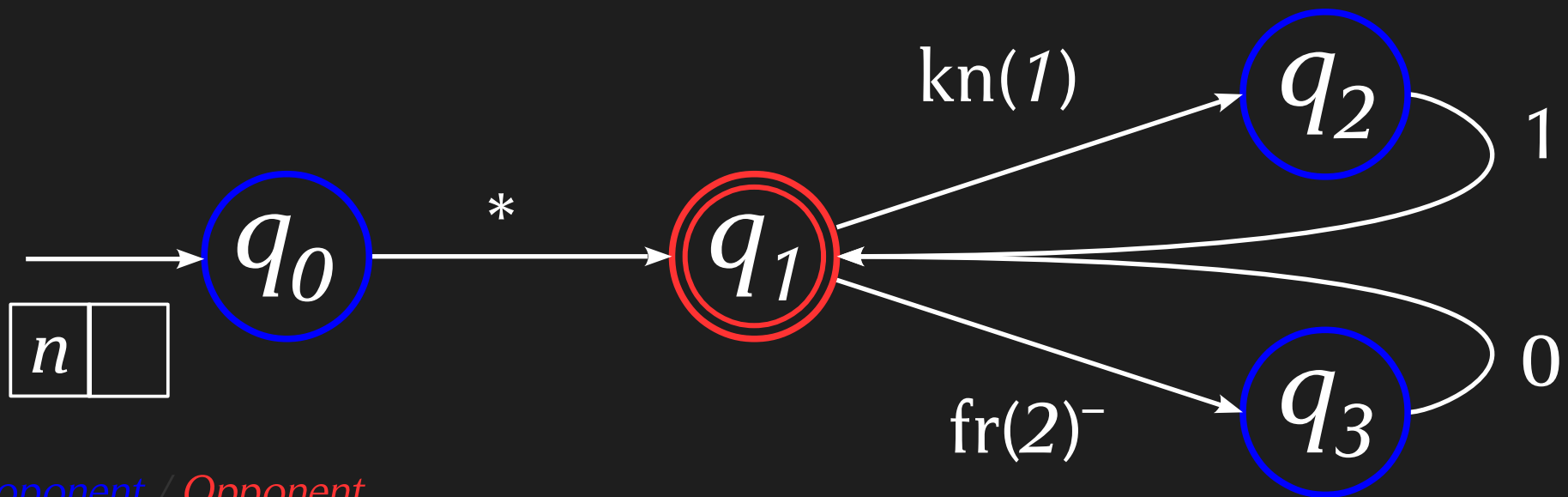
Fresh-register automata



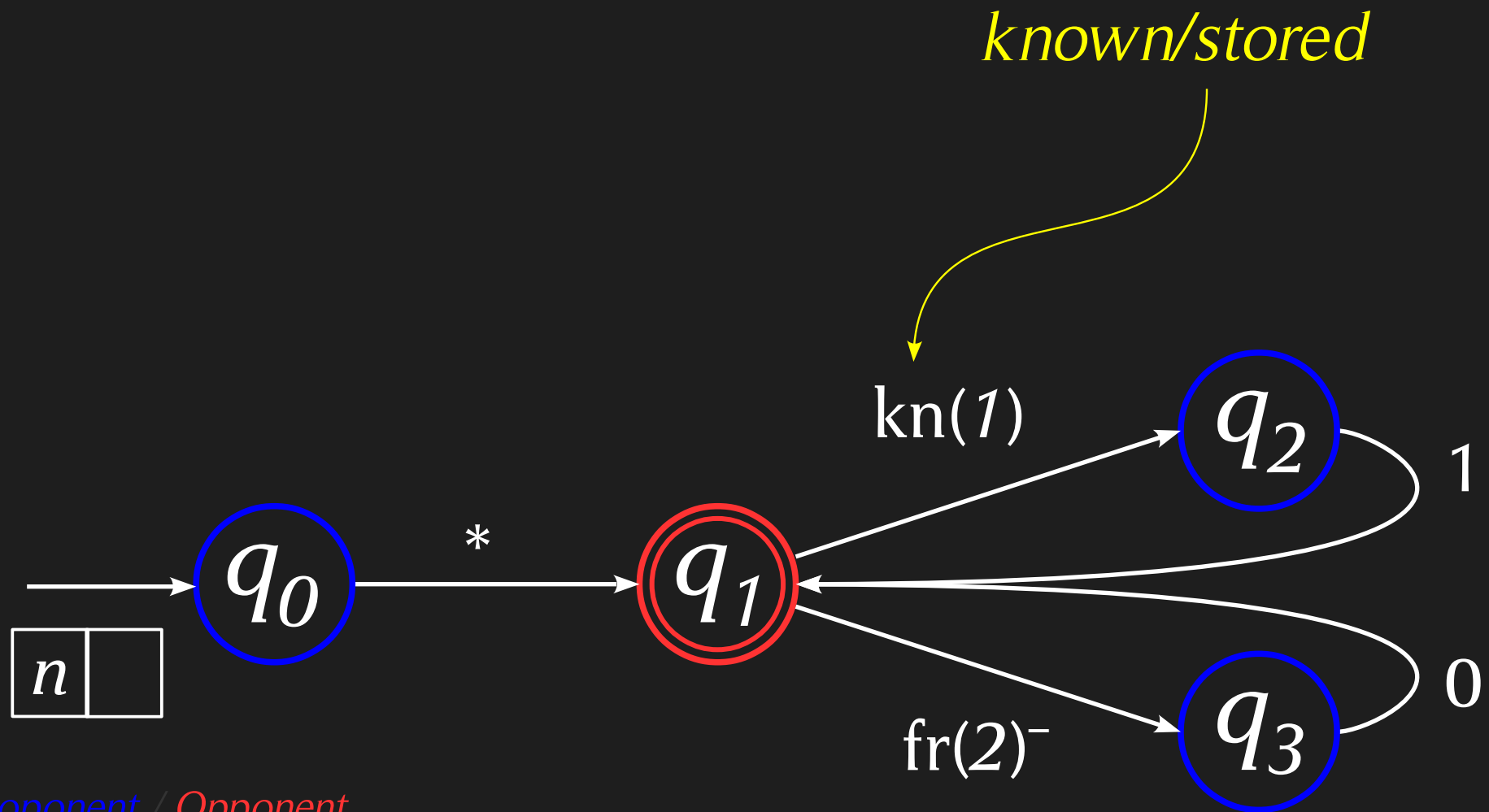
$\vdash \lambda z. \text{newvar}(0) : \text{com} \rightarrow \text{intvar}$



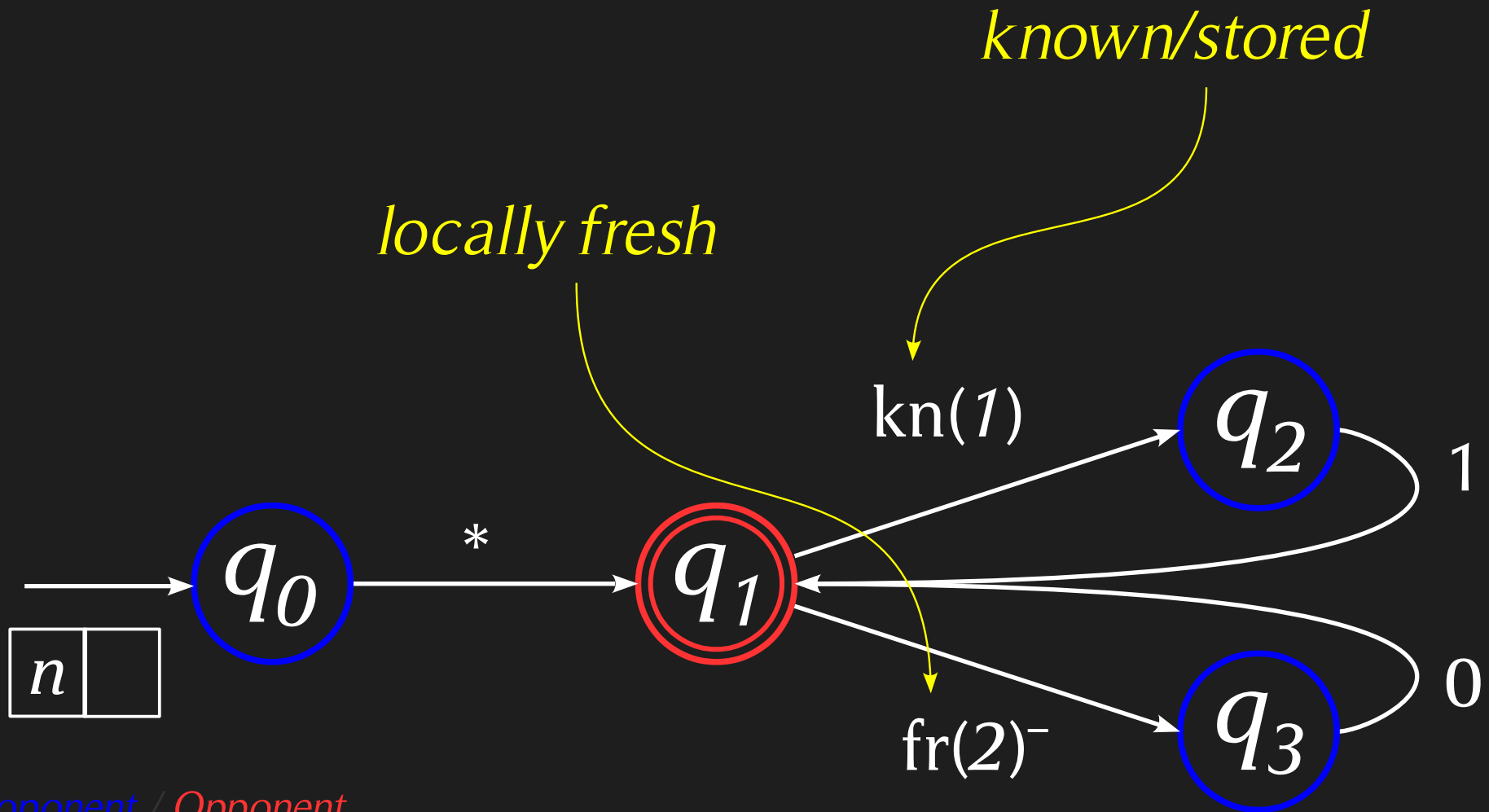
Fresh-register automata



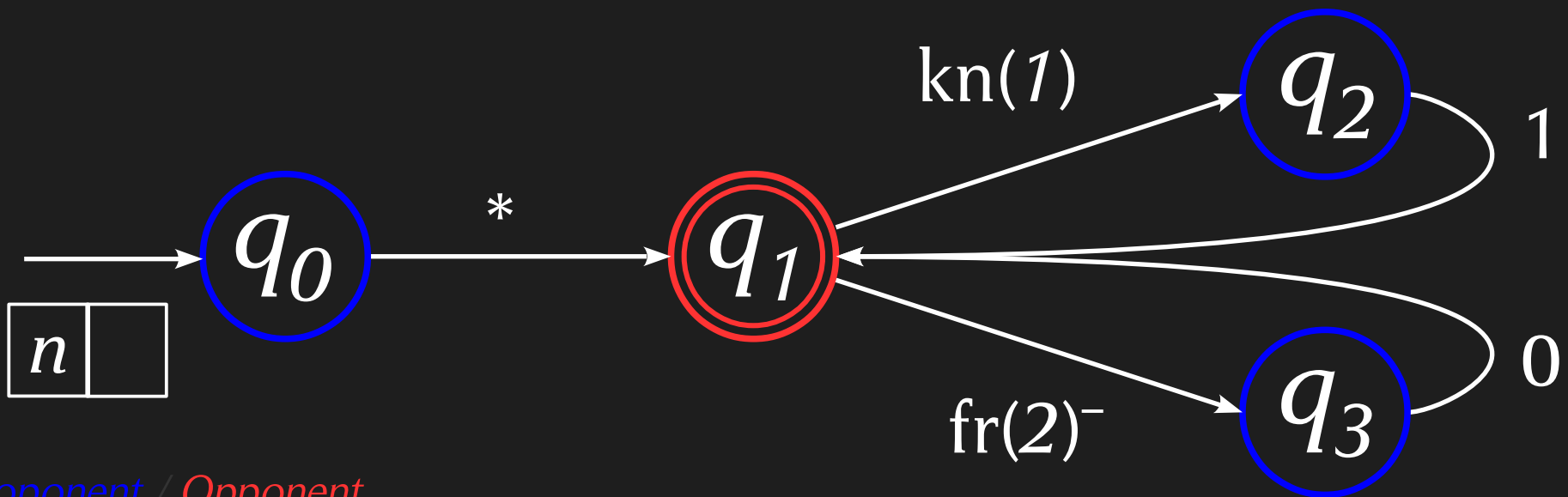
Fresh-register automata



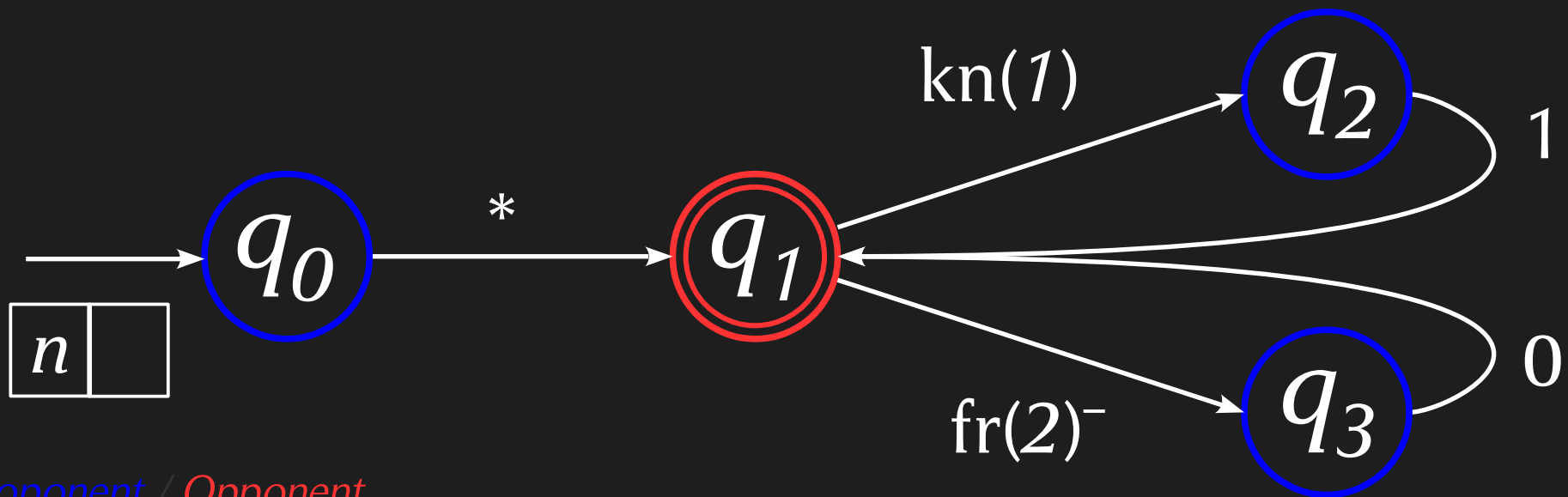
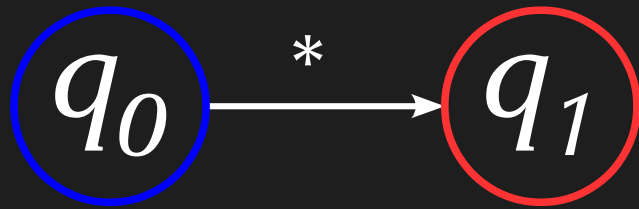
Fresh-register automata



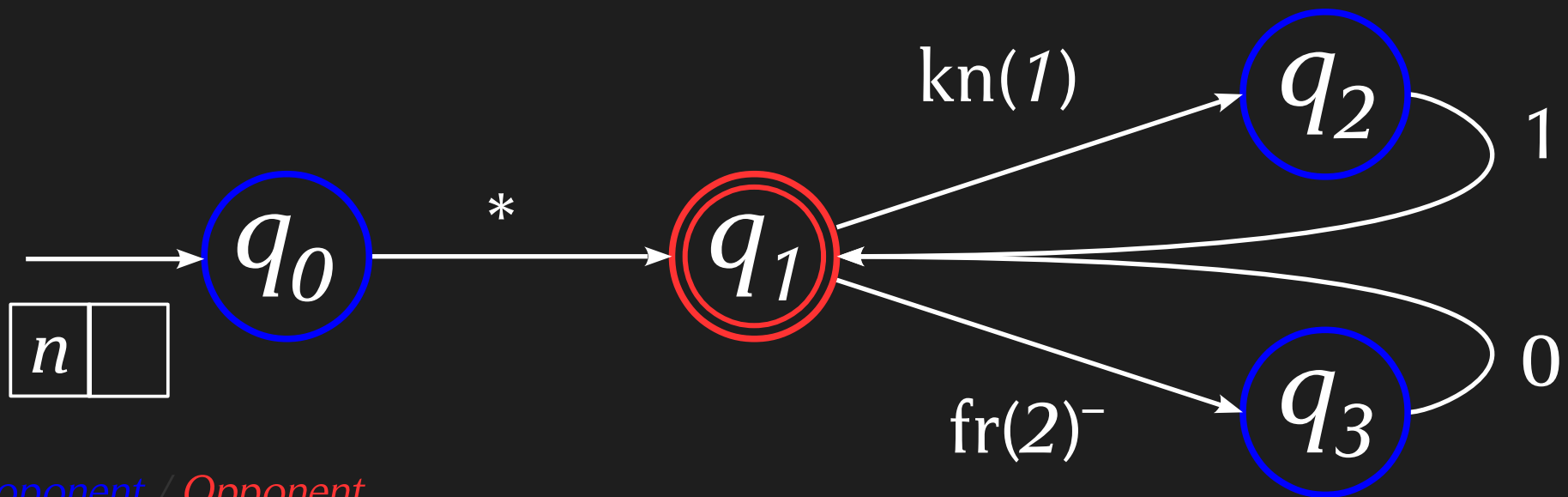
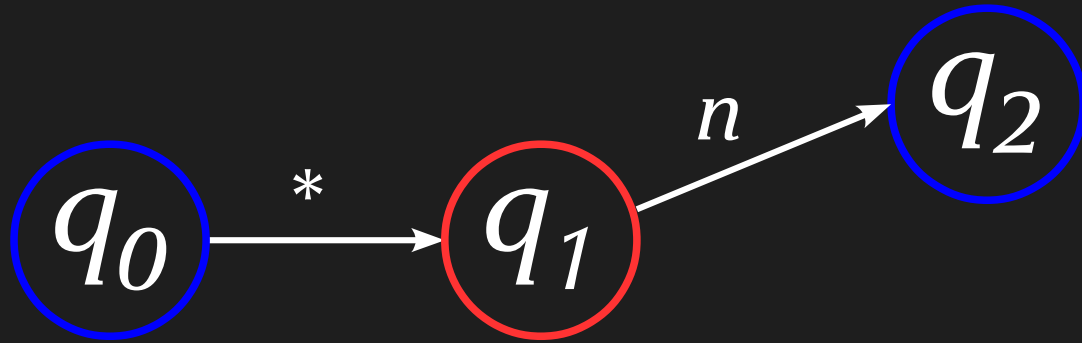
Fresh-register automata



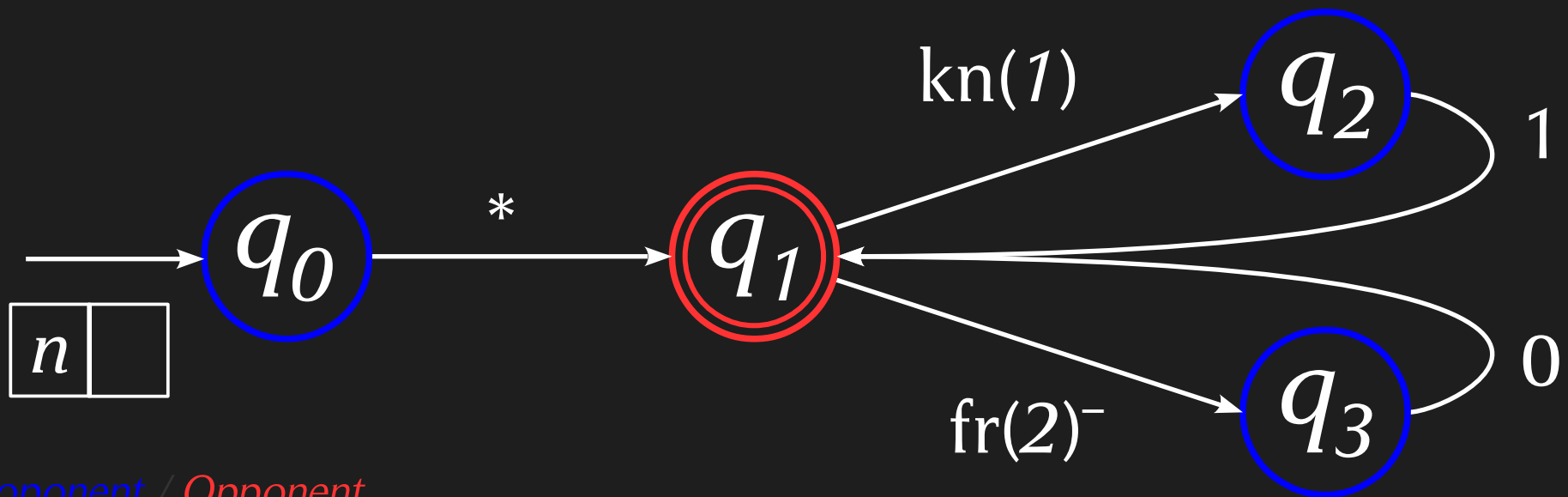
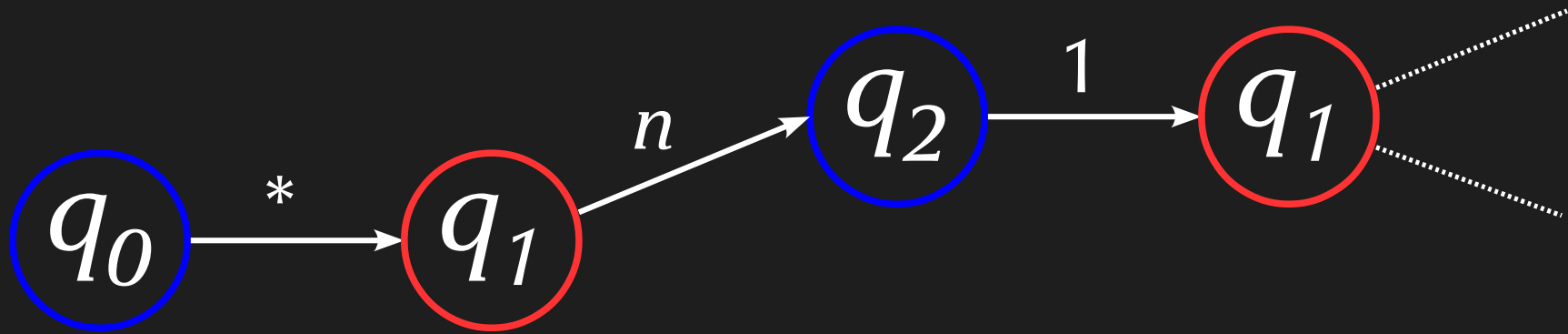
Fresh-register automata



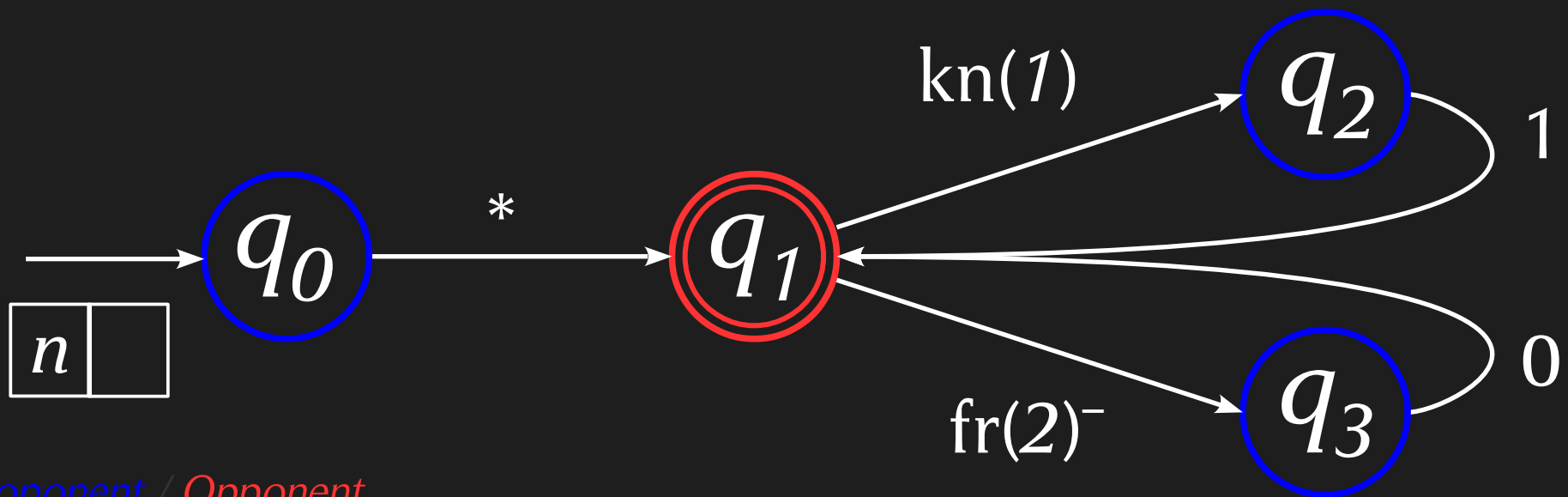
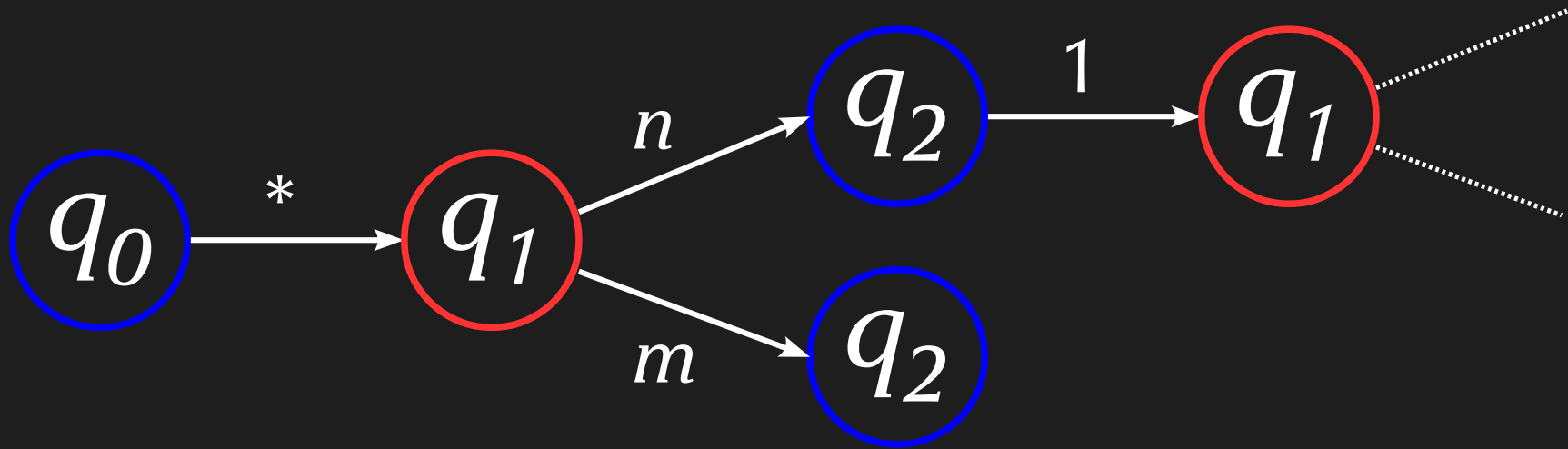
Fresh-register automata



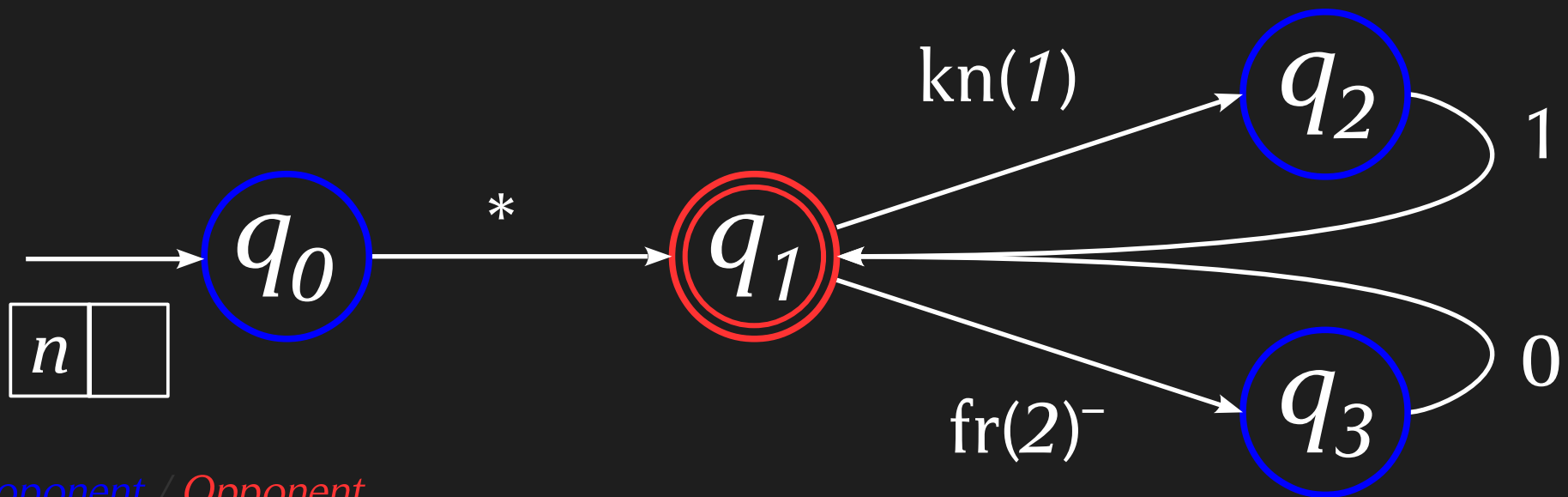
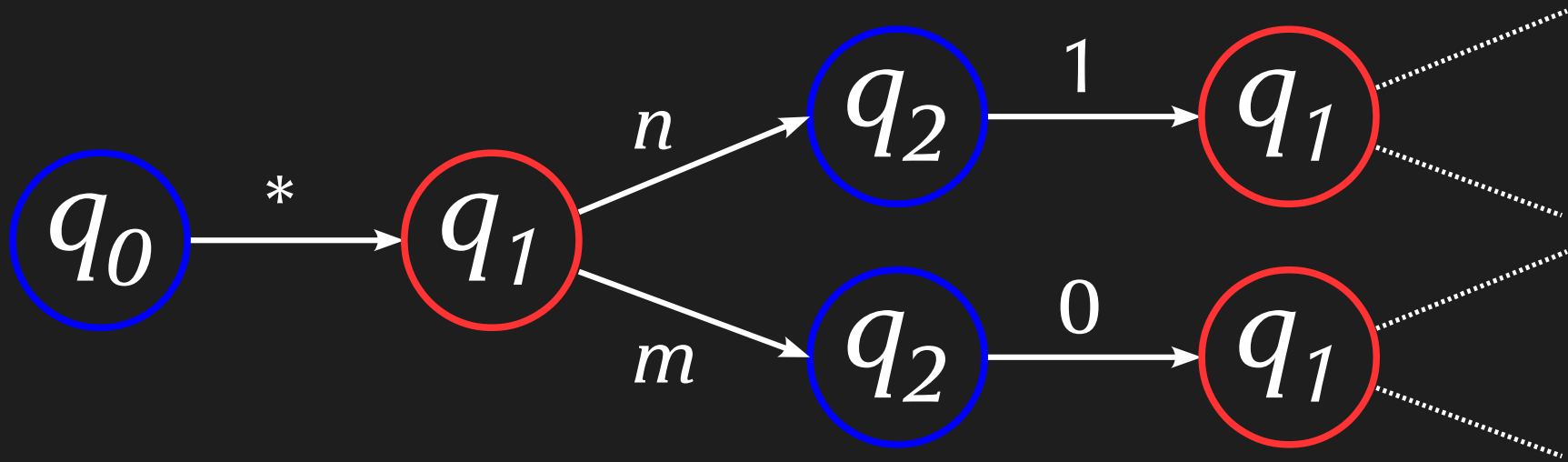
Fresh-register automata



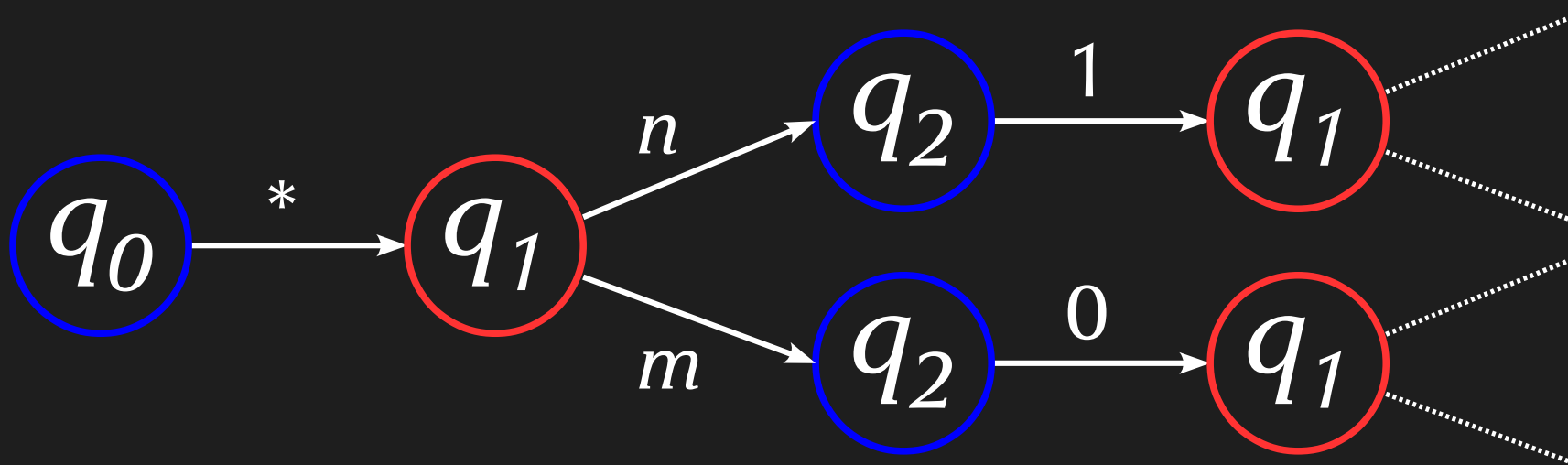
Fresh-register automata



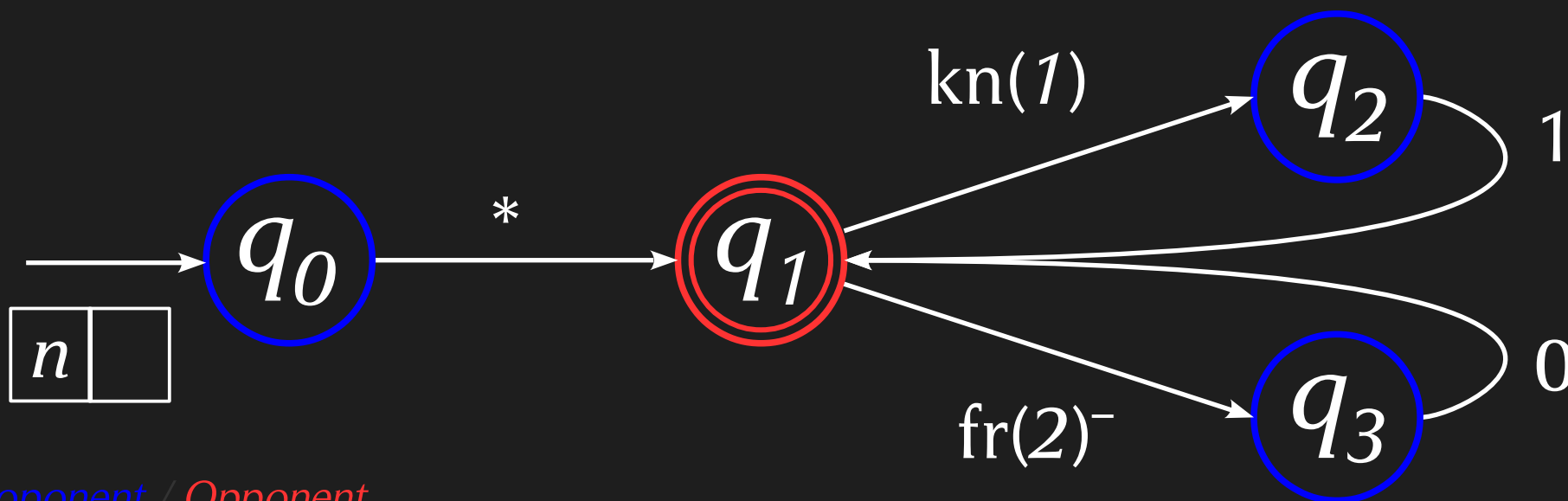
Fresh-register automata



Fresh-register automata



$x : \text{intvar} \vdash \lambda y. (x == y) : \text{intvar} \rightarrow \text{int}$



Games with new resources

Models of realistic programs (2007-)

- General name-features [Tz.07, Tz.08]
- Direct models [Laird 06, Laird 08, Murawski&Tz.09, Mur.&Tz.11]

Algorithmic games

- Fresh-register automata [Tz.11]

Games with new resources

Models of realistic programs (2007-)

- General name-features [Tz.07, Tz.08]
- Direct models [Laird 06, Laird 08, Murawski&Tz.09, Mur.&Tz.11]

Algorithmic games

- Fresh-register automata [Tz.11]
- Algorithmic model for mini-RedML [Mur.&Tz.11']

Games with new resources

Models of realistic programs (2007-)

- General name-features [Tz.07, Tz.08]
- Direct models [Laird 06, Laird 08, Murawski&Tz.09, Mur.&Tz.11]

Algorithmic games

- Fresh-register automata [Tz.11]
- Algorithmic model for mini-RedML [Mur.&Tz.11']

$$M \cong N \iff \mathcal{A}_M \sim \mathcal{A}_N$$

Games with new resources

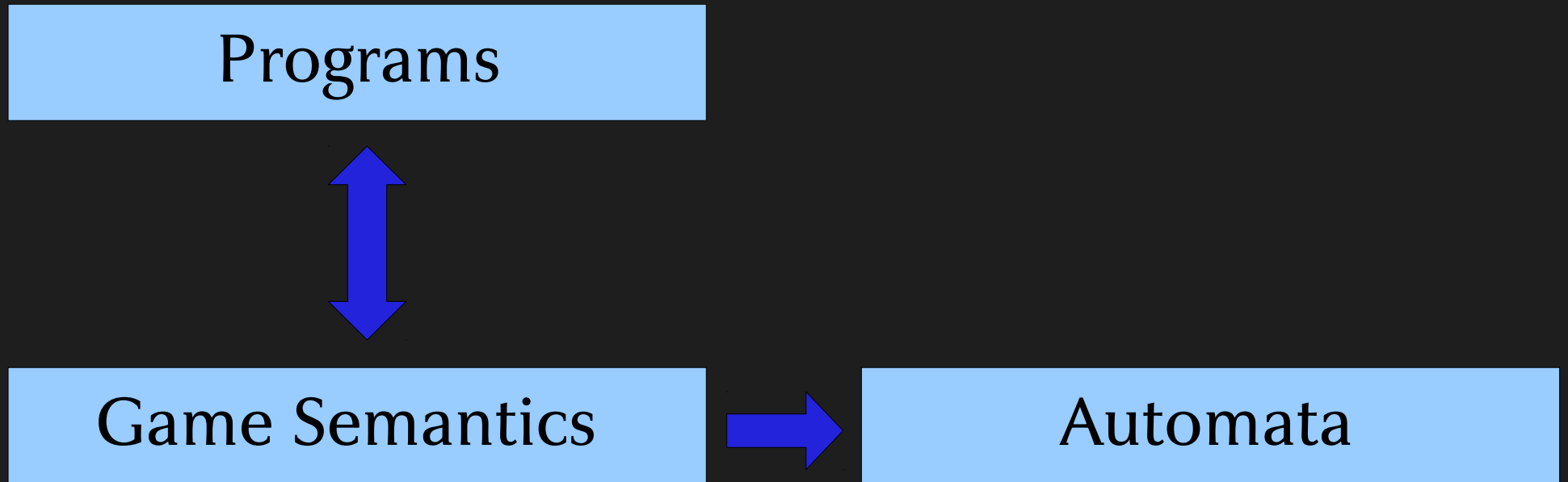
Models of realistic programs (2007-)

- General name-features [Tz.07, Tz.08]
- Direct models [Laird 06, Laird 08, Murawski&Tz.09, Mur.&Tz.11]

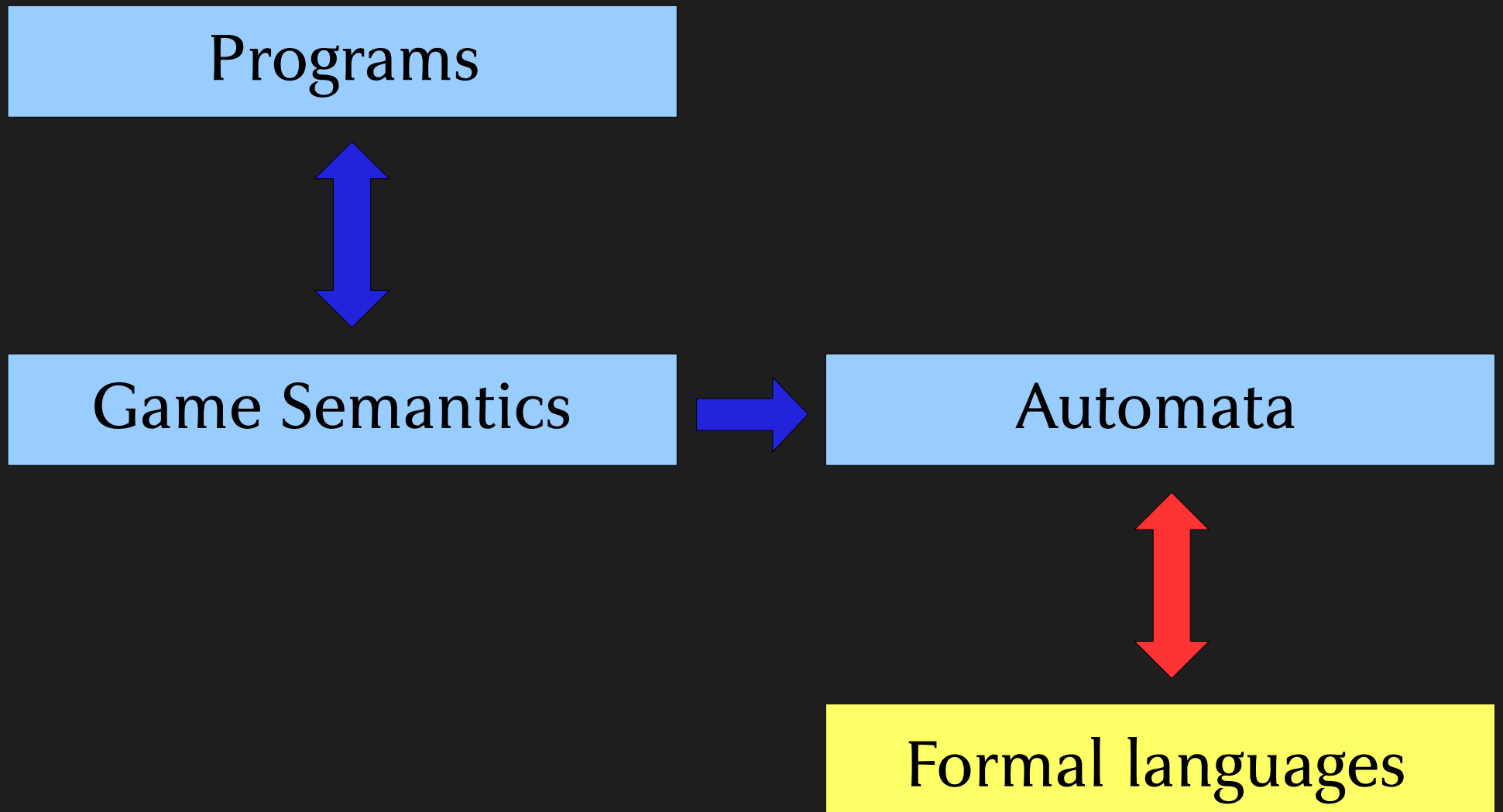
Algorithmic games

- Fresh-register automata [Tz.11]
- Algorithmic model for mini-RedML [Mur.&Tz.11']

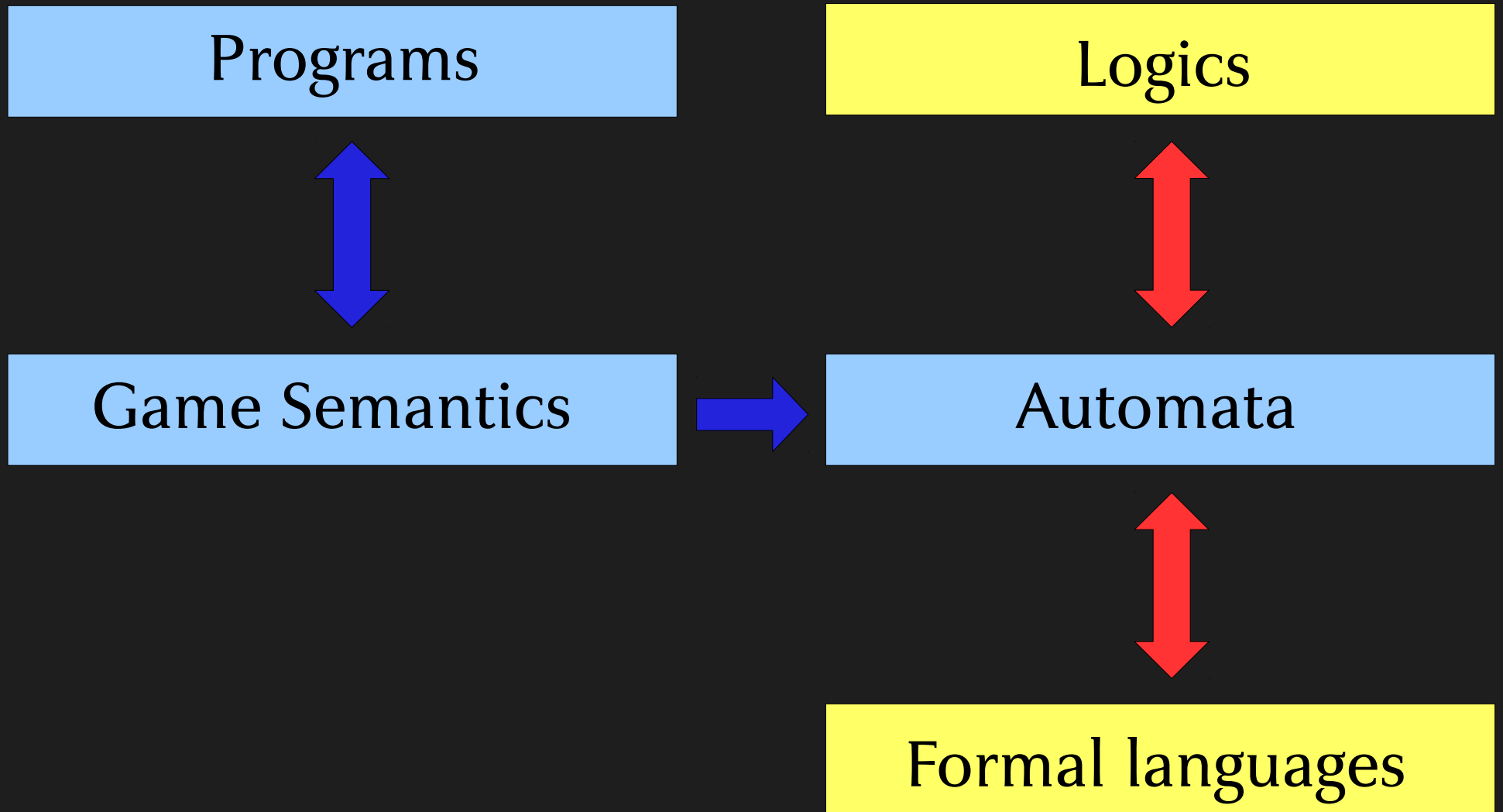
Reasoning about new resources



Reasoning about new resources



Reasoning about new resources



thank you!