

# 記憶つき相互作用の幾何における 再帰と妥当性

Recursion and Adequacy in  
Memoryful Geometry of Interaction

室屋 晃子 (蓮尾研)

# 概要

コンパイラ / 高位合成  
[Mackie '95] [Ghica '07]

暗黙的計算量理論  
[Dal Lago '06]

Gol  
[Girard '89]

memoryful Gol  
[Hoshino, M. & Hasuo '14]

memoryful Gol  
with recursion



# 背景 : Goltz

- Geometry of Interaction (相互作用の幾何)
- 表示的 (要素還元的) 意味論
  - 線形論理に対して [Girard '89]
  - 関数型プログラムに対して
- 操作的な性質
  - 計算の経過を表現

# 背景 : GOL

- Geometry of Interaction (相互作用の幾何)
- 表示的 (要素還元的) 意味論
  - 線形論理に対して [Girard '89]
  - 関数型プログラムに対して
- 操作的な性質
  - 計算の経過 :  
トークンマシン意味論  
[Mackie '95] [Danos & Regnier '99]

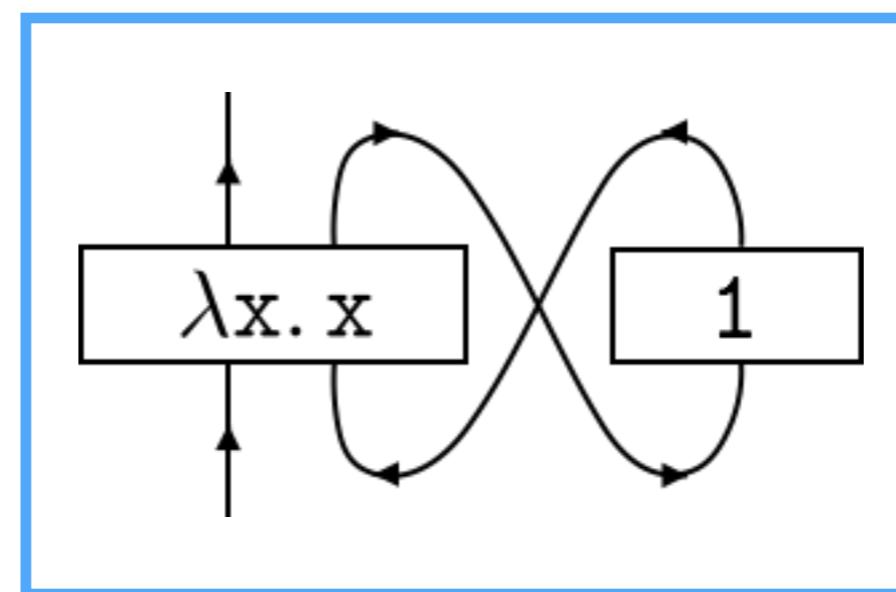
# 背景：トークンマシン意味論

関数型プログラム

$(\lambda x. x) 1$



トークンマシン



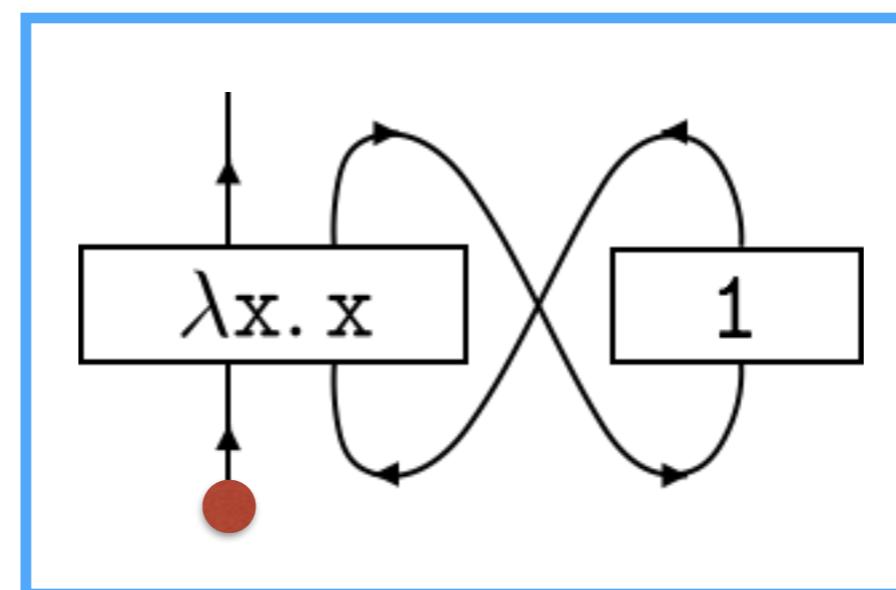
# 背景：トークンマシン意味論

関数型プログラム

$(\lambda x. x) 1$



トークンマシン



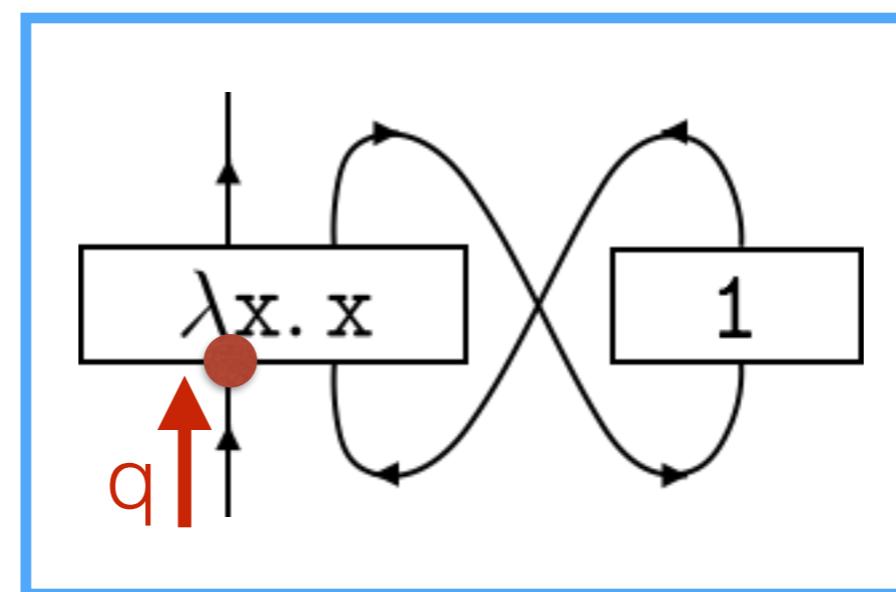
# 背景：トークンマシン意味論

関数型プログラム

$(\lambda x. x) 1$



トークンマシン



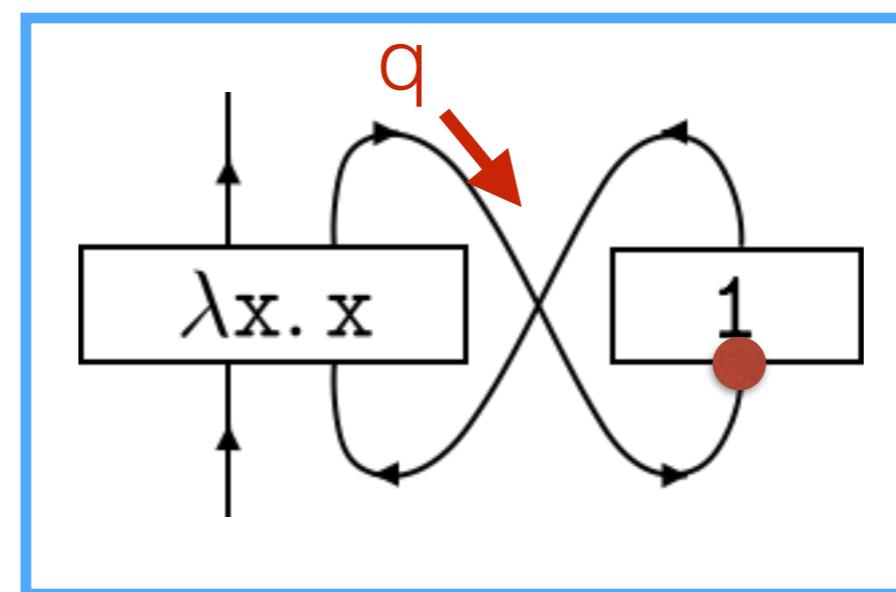
# 背景：トークンマシン意味論

関数型プログラム

$(\lambda x. x) 1$



トークンマシン



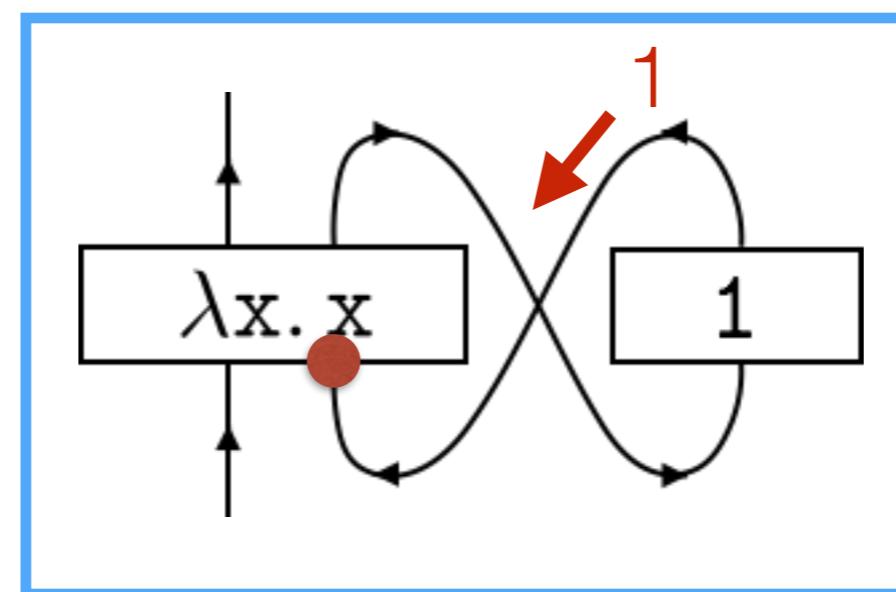
# 背景：トークンマシン意味論

関数型プログラム

$(\lambda x. x) 1$



トークンマシン



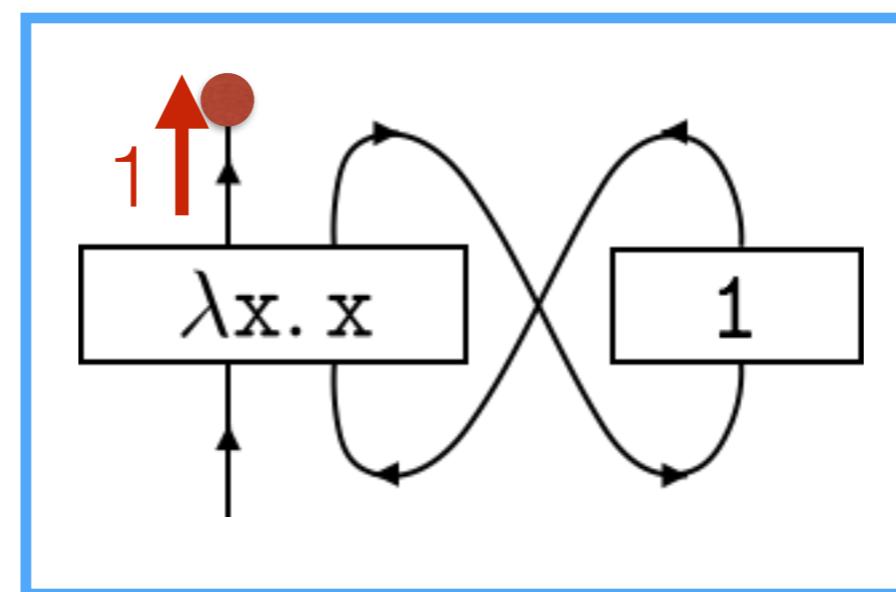
# 背景：トークンマシン意味論

関数型プログラム

$(\lambda x. x) 1$



トークンマシン



# 背景：トークンマシン意味論

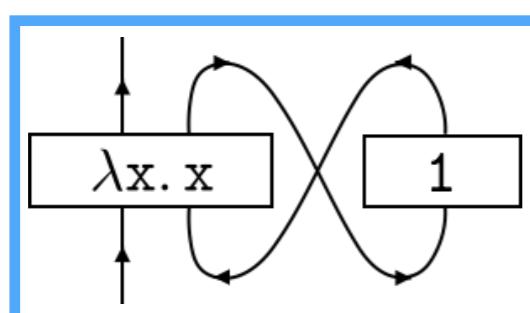
関数型プログラム

$(\lambda x. x) 1$

- コンパイル技術への応用 [Mackie '95] [Ghica '01]
- 暗黙的計算量理論への応用 [Dal Lago '06]
- 圈論による定式化 [Abramsky et al. '02]



トークンマシン



# 概要

コンパイラ / 高位合成  
[Mackie '95] [Ghica '07]

暗黙的計算量理論  
[Dal Lago '06]

Gol  
[Girard '89]

memoryful Gol  
[Hoshino, M. & Hasuo '14]

memoryful Gol  
with recursion



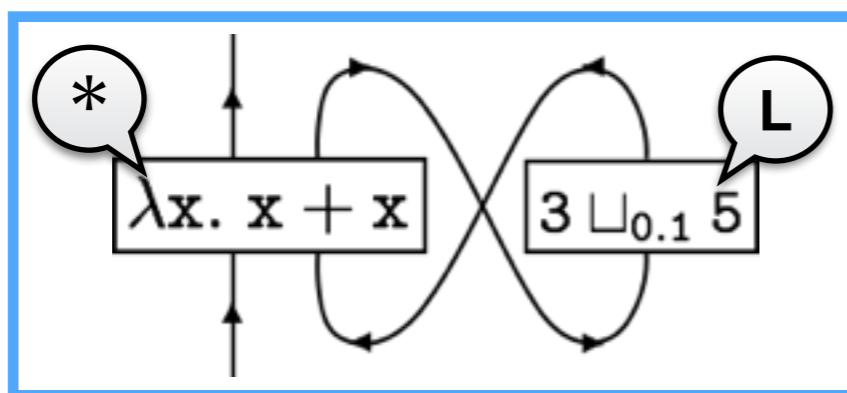
# 背景 : memoryful GoI [Hoshino, M. & Hasuo '14]

関数型プログラム + 計算副作用

$$(\lambda x. x + x) (3 \sqcup_{0.1} 5)$$



メモリつきトークンマシン



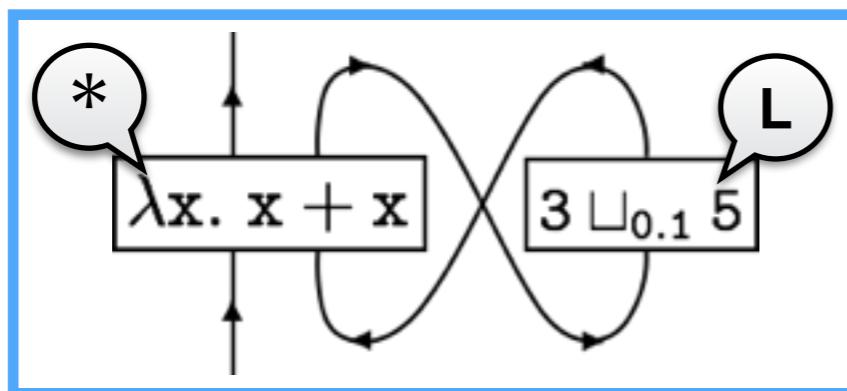
# 背景 : memoryful GOL [Hoshino, M. & Hasuo '14]

関数型プログラム + 計算副作用

$$(\lambda x. x + x) (3 \sqcup_{0.1} 5)$$



メモリつきトークンマシン



代数的副作用 [Plotkin & Power '01]

- 非決定的選択  $M \sqcup N$
- 確率的選択  $M \sqcup_p N$
- 大域変数の参照、更新

$\text{lookup}_l(M_{v_1}, \dots, M_{v_{|\text{Val}|}})$   
 $\text{update}_{l,v}(M)$

# 背景 : memoryful GoI [Hoshino, M. & Hasuo '14]

関数型プログラム + 計算副作用

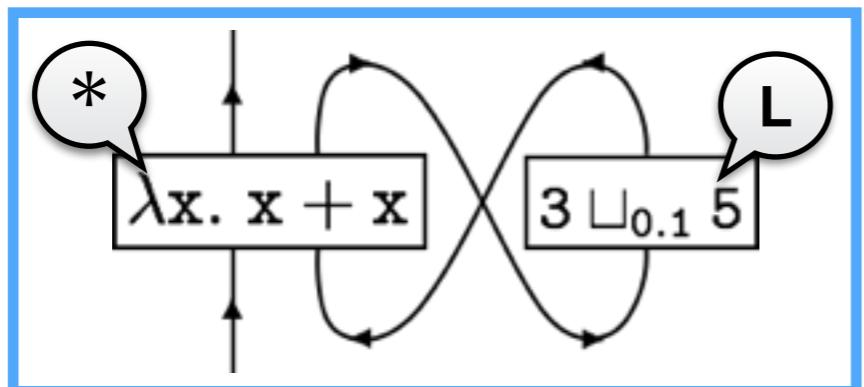
$$(\lambda x. x + x) (3 \sqcup_{0.1} 5)$$



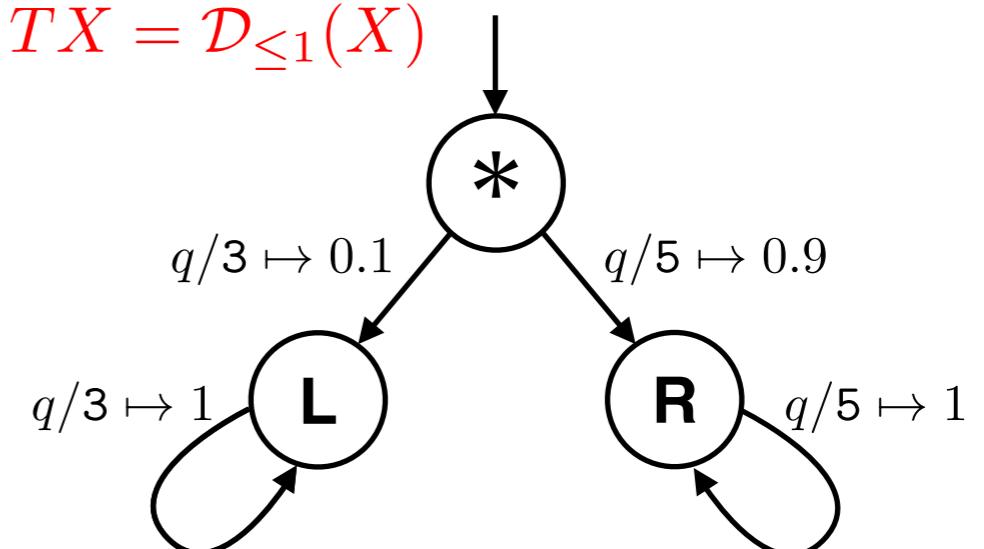
副作用つきミーリー・マシン

$$\left( \begin{array}{c} X, \\ c: X \times A \rightarrow T(X \times B), \\ x_0 \in X \end{array} \right) : A \rightarrow B$$

メモリつきトークンマシン



$$TX = \mathcal{D}_{\leq 1}(X)$$



# 概要

コンパイラ / 高位合成  
[Mackie '95] [Ghica '07]

暗黙的計算量理論  
[Dal Lago '06]

Gol  
[Girard '89]

memoryful Gol  
[Hoshino, M. & Hasuo '14]

memoryful Gol  
with recursion



# 結果：memoryful GoI with recursion

関数型プログラム + 計算副作用 + 再帰



メモリつきトークンマシン

# 結果：memoryful GoI with recursion

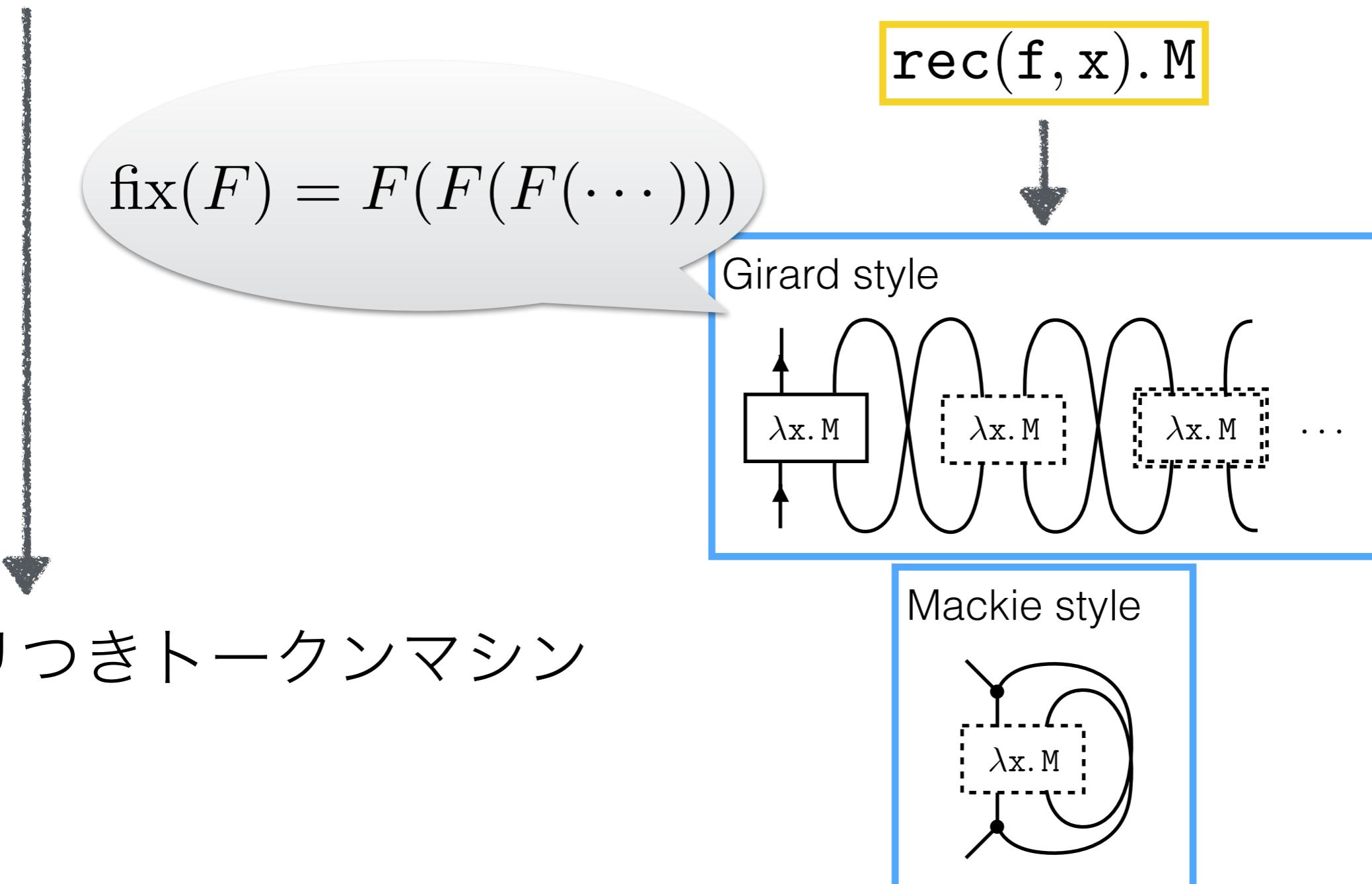
関数型プログラム + 計算副作用 + 再帰



メモリつきトーケンマシン

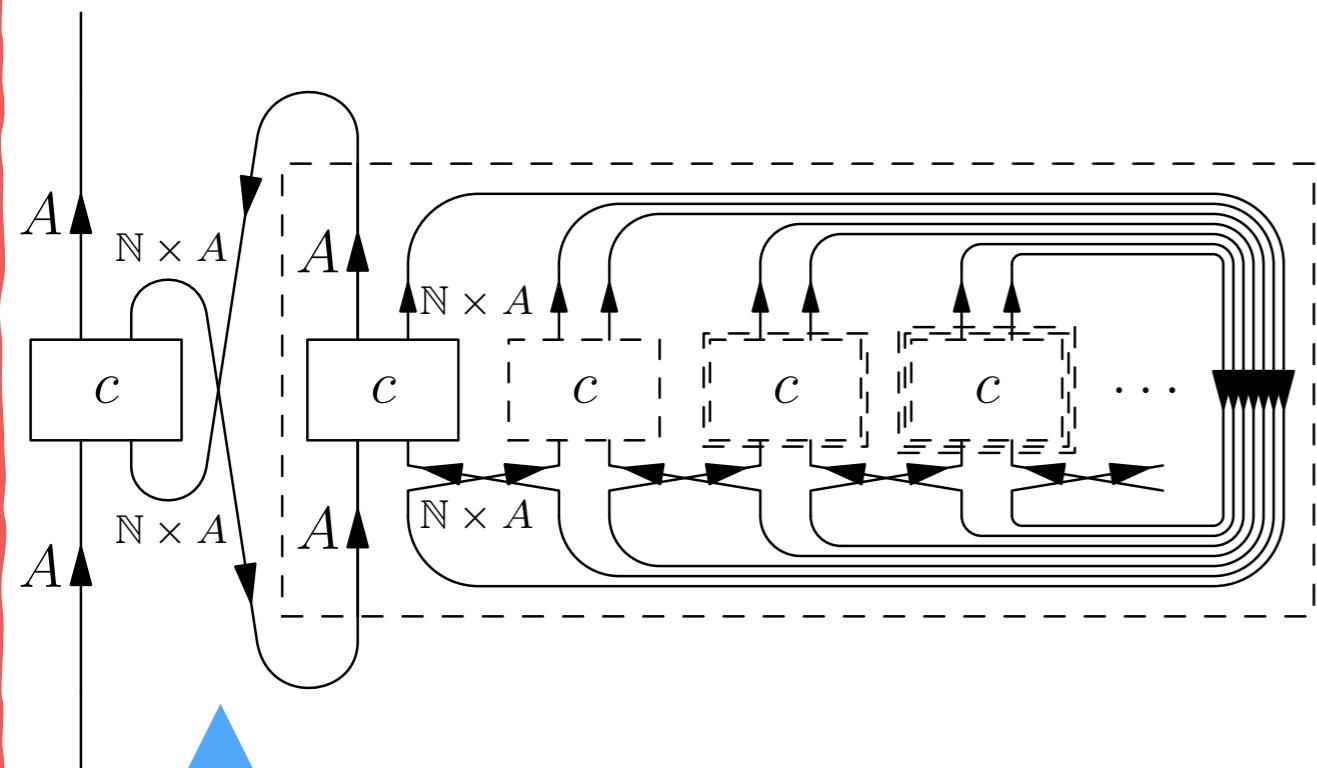
# 結果：memoryful GoI with recursion

関数型プログラム + 計算副作用 + 再帰

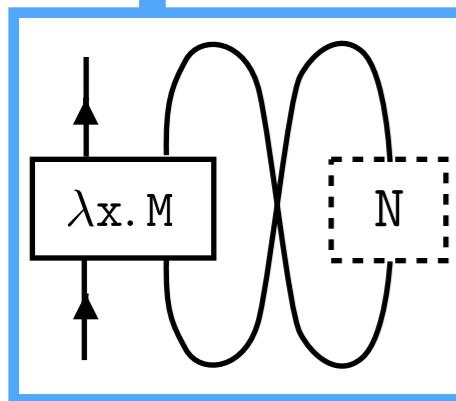
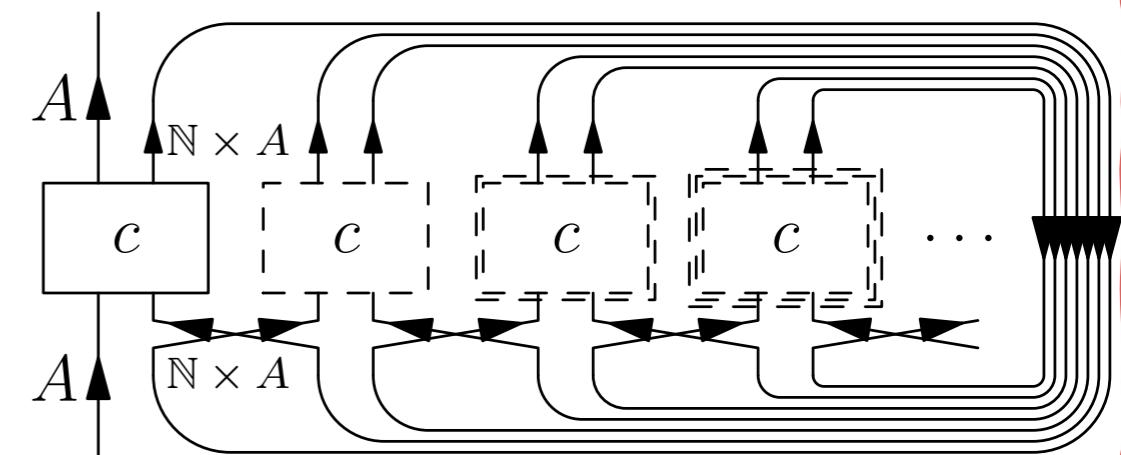


# 結果：memoryful Gol with recursion

Prop. (Girard style as **fixed point** wrt. “cross connection”)

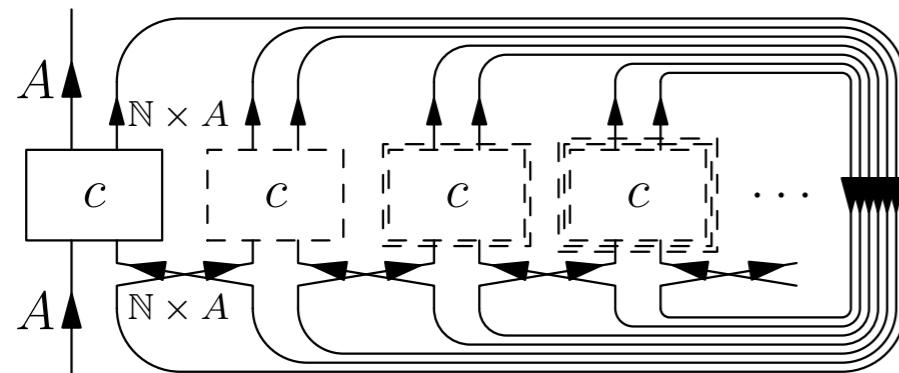


$\approx$

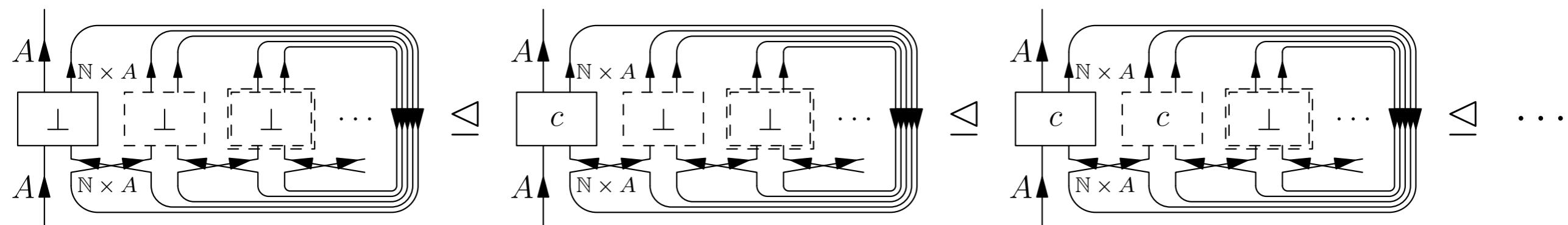


# 結果：memoryful Gol with recursion

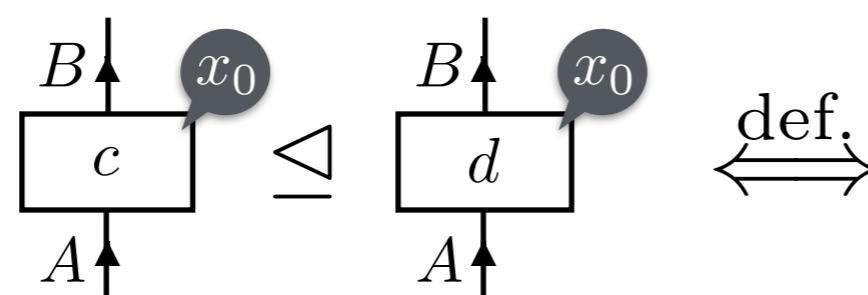
**Prop.** (Girard style as **supremum** of finite approximations)



is the supremum of the  $\omega$ -chain



where



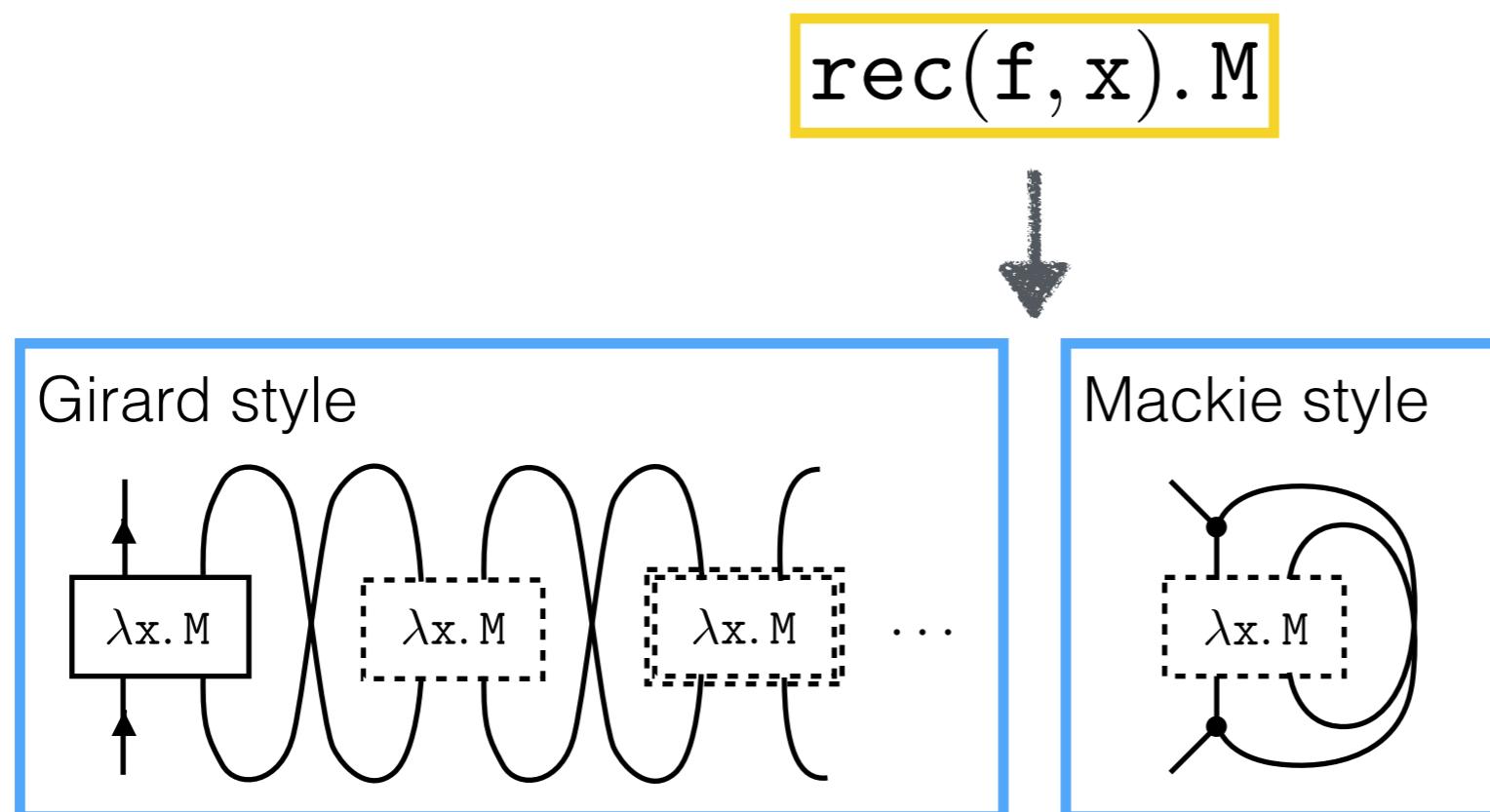
$$T(X \times B)$$

$$c \left( \sqsubseteq \right) d$$

# 結果：memoryful Gol with recursion

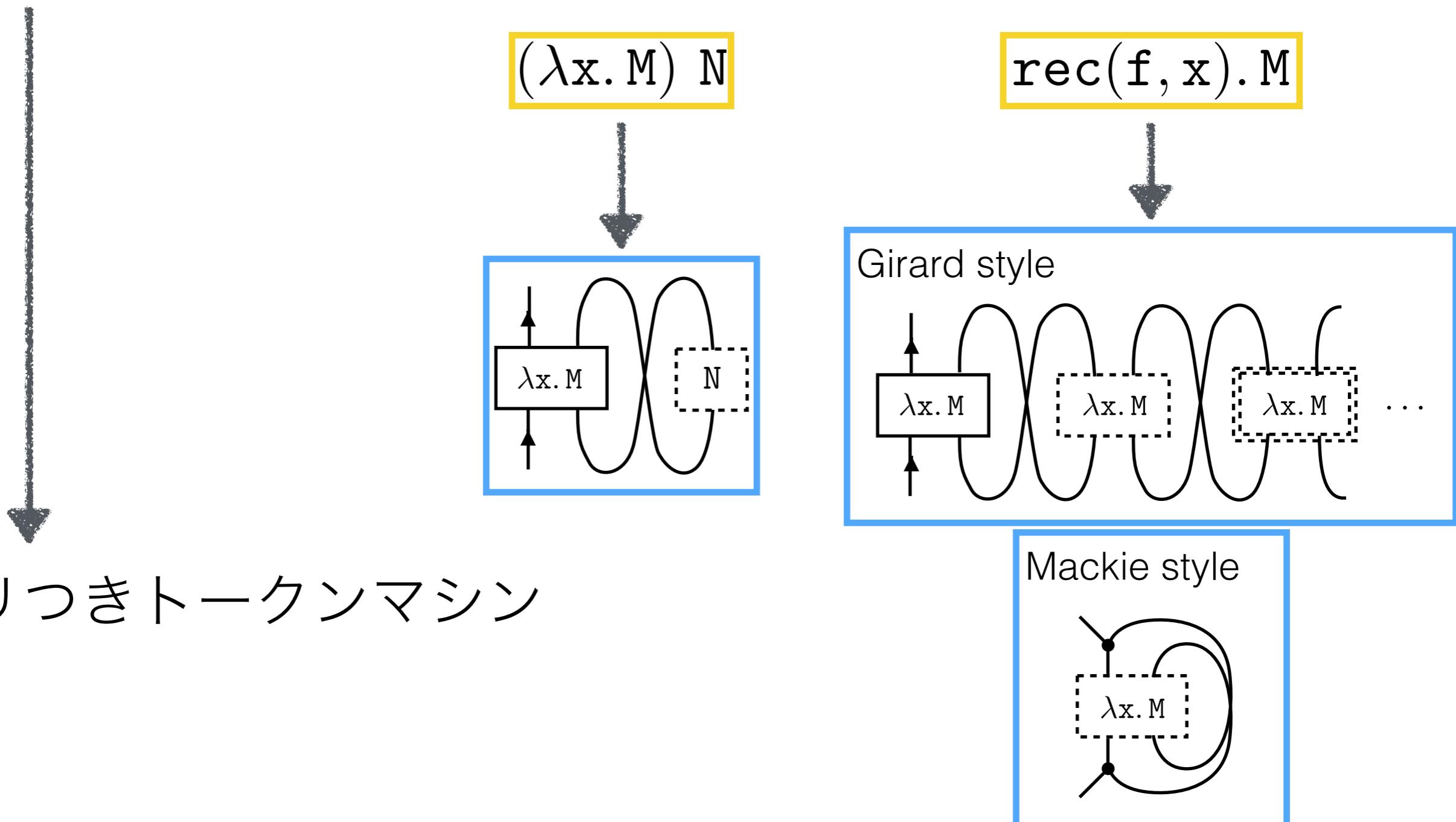
**Thm. (coincidence) of Girard style & Mackie style)**

$$(\mathbb{M})_{\text{Girard}} \simeq (\mathbb{M})_{\text{Mackie}}$$



# 結果：memoryful Gol with recursion

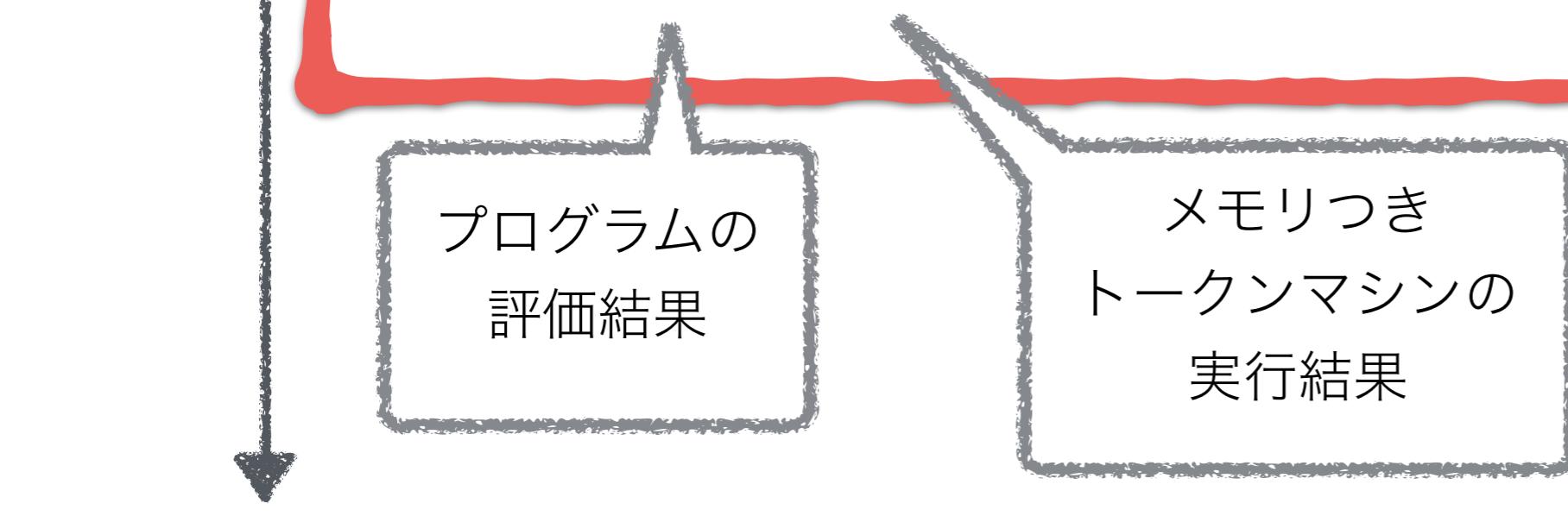
関数型プログラム + 計算副作用 + 再帰



# 結果：memoryful Go! with recursion

# 関数型プログラム + 計算副作用 + 再帰

**Theorem 6.4** (adequacy). Any closed term  $M$  of base type  $\text{nat}$  satisfies  $\llbracket M \rrbracket = (\llbracket M \rrbracket)^\dagger$ .



# メモリつきトークンマシン

# 例：再帰的な確率プログラム

```
(rec(flipLoop, x).choose0.4(x, flipLoop(x + 1))) 0
```

コイントスの結果	プログラムの返り値	確率
H	0	0.4
TH	1	$0.4 * 0.6^2$
TTH	2	$0.4 * 0.6^3$
TTTH	3	$0.4 * 0.6^4$

# 例：再帰的な確率プログラム

```
(rec(flipLoop, x).choose0.4(x, flipLoop(x + 1))) 0
```

$$\text{プログラムの評価結果} = \begin{bmatrix} 0 & \mapsto & 0.4 \\ 1 & \mapsto & 0.4 \times 0.6 \\ 2 & \mapsto & 0.4 \times 0.6^2 \\ 3 & \mapsto & 0.4 \times 0.6^3 \\ & \vdots & \end{bmatrix} \in \mathcal{D}_{\leq 1}(\mathbb{N})$$

# 例：再帰的な確率プログラム

```
(rec(flipLoop, x).choose0.4(x, flipLoop(x + 1))) 0
```

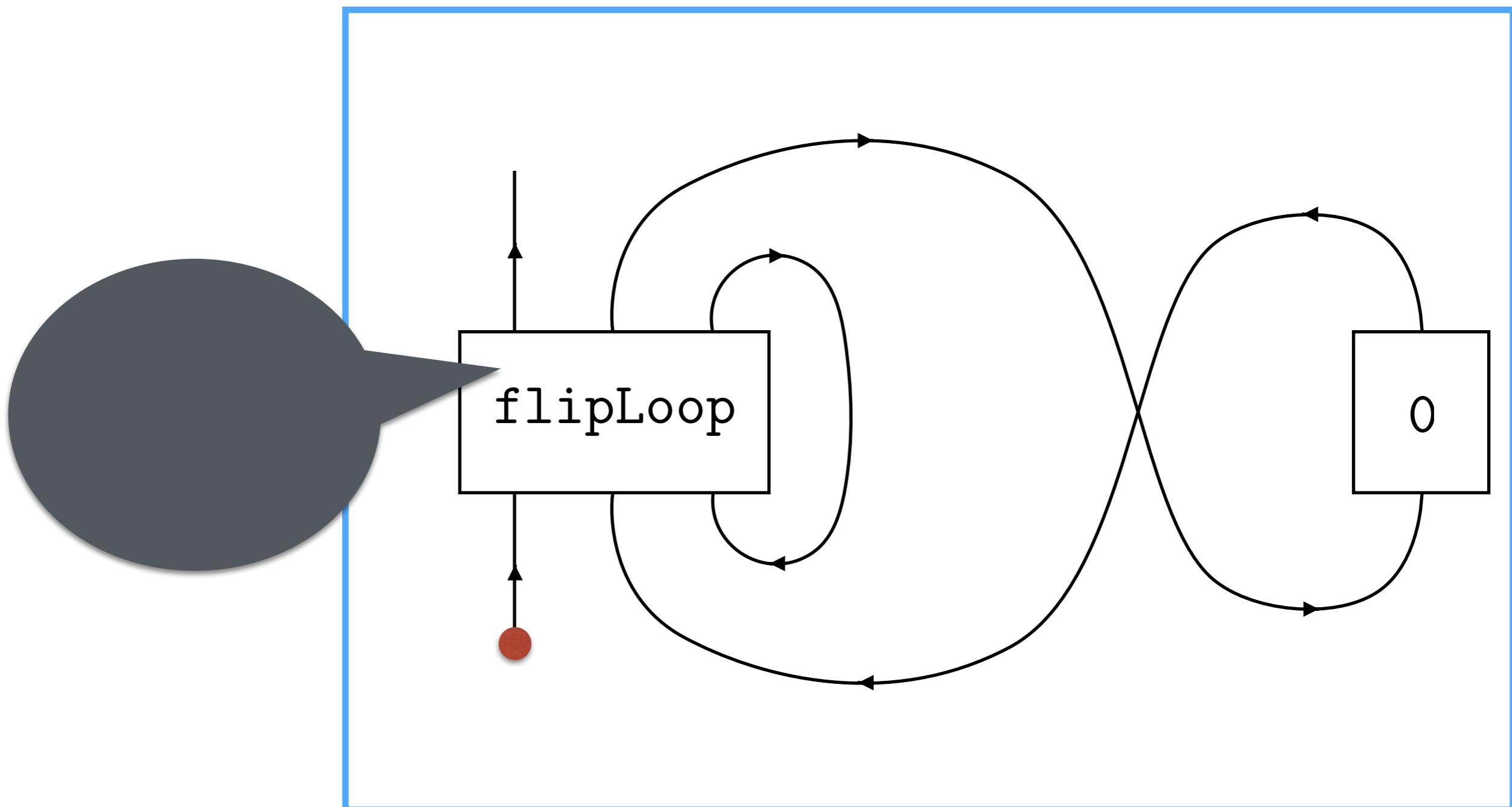
$$\text{プログラムの評価結果} = \begin{bmatrix} 0 & \mapsto & 0.4 \\ 1 & \mapsto & 0.4 \times 0.6 \\ 2 & \mapsto & 0.4 \times 0.6^2 \\ 3 & \mapsto & 0.4 \times 0.6^3 \\ & \vdots & \end{bmatrix} \in \mathcal{D}_{\leq 1}(\mathbb{N})$$

||

得られる  
メモリつきトークンマシンの  
実行結果

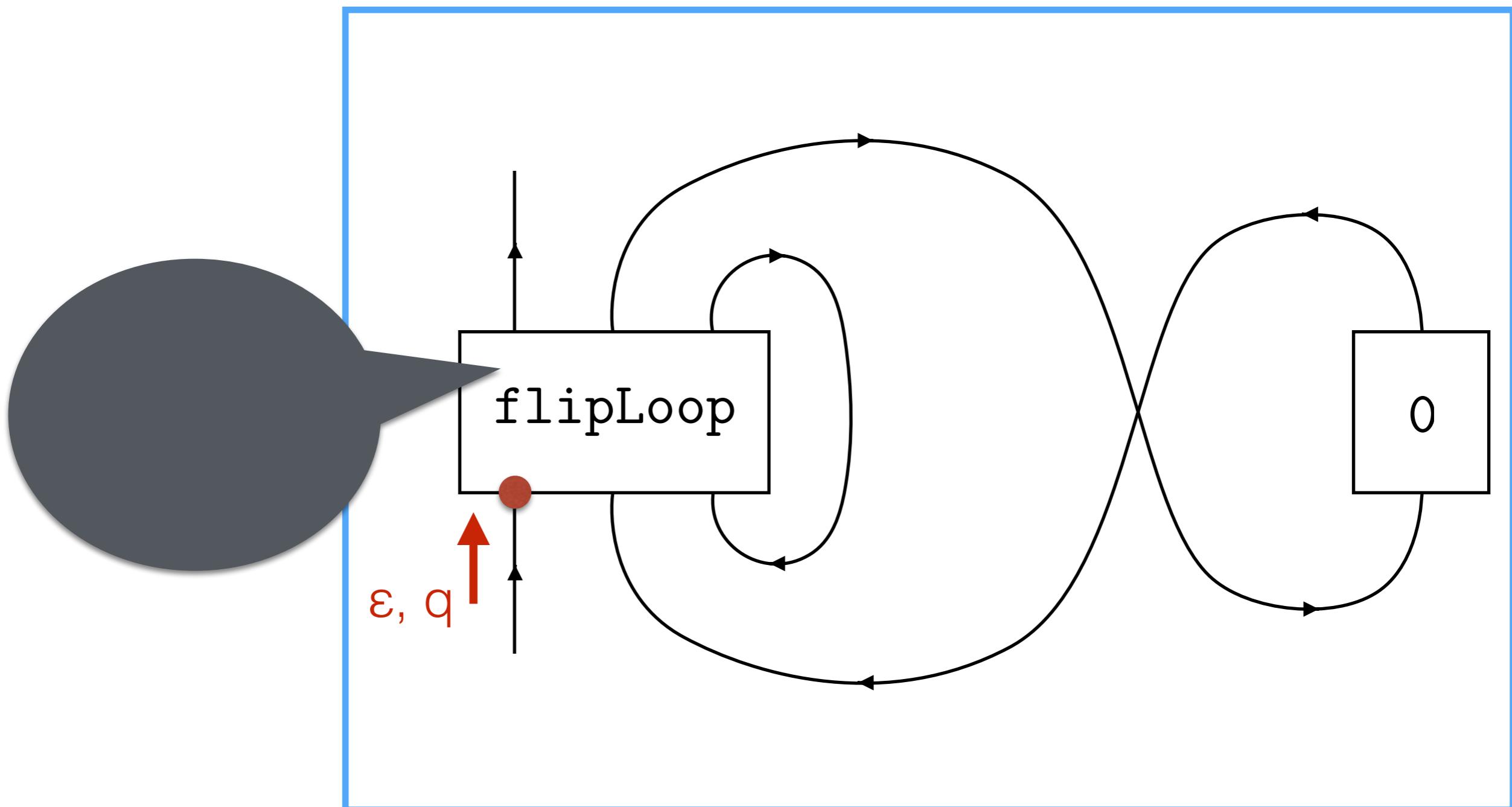
# 例：再帰的な確率プログラム

```
(rec(flipLoop, x).choose0.4(x, flipLoop(x + 1))) 0
```



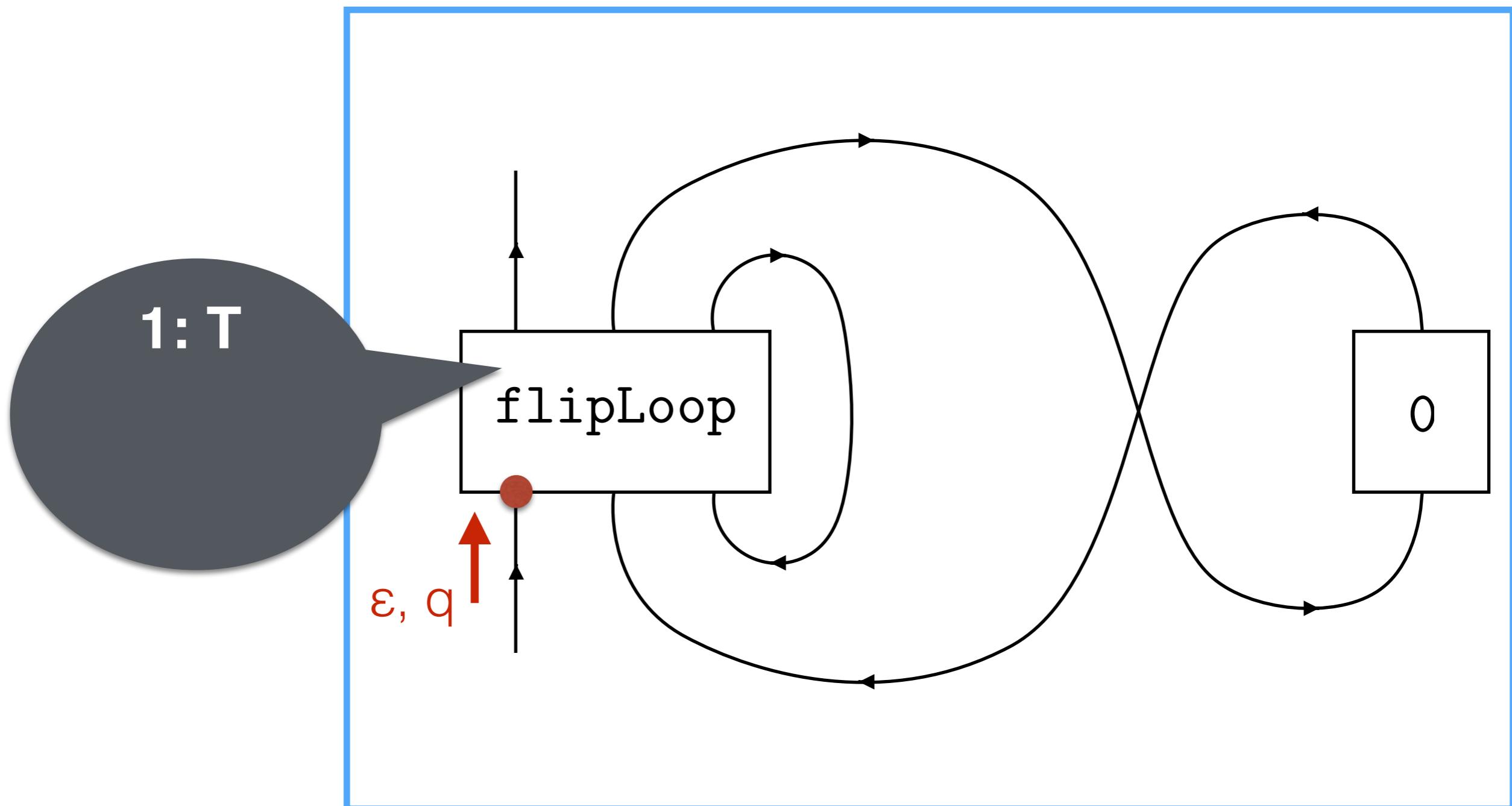
# 例：再帰的な確率プログラム

```
(rec(flipLoop, x).choose0.4(x, flipLoop(x + 1))) 0
```



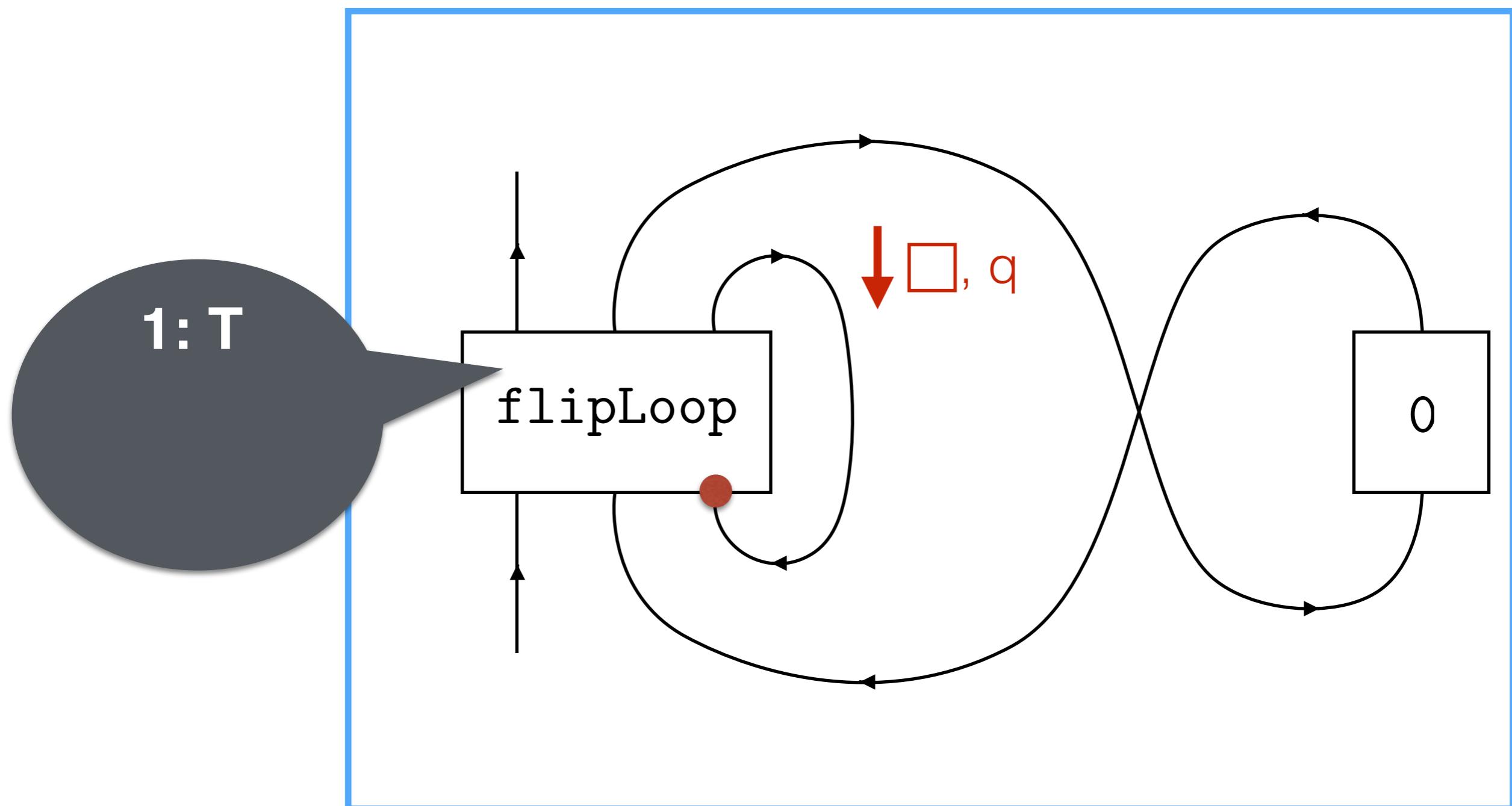
# 例：再帰的な確率プログラム

```
(rec(flipLoop, x).choose0.4(x, flipLoop(x + 1))) 0
```



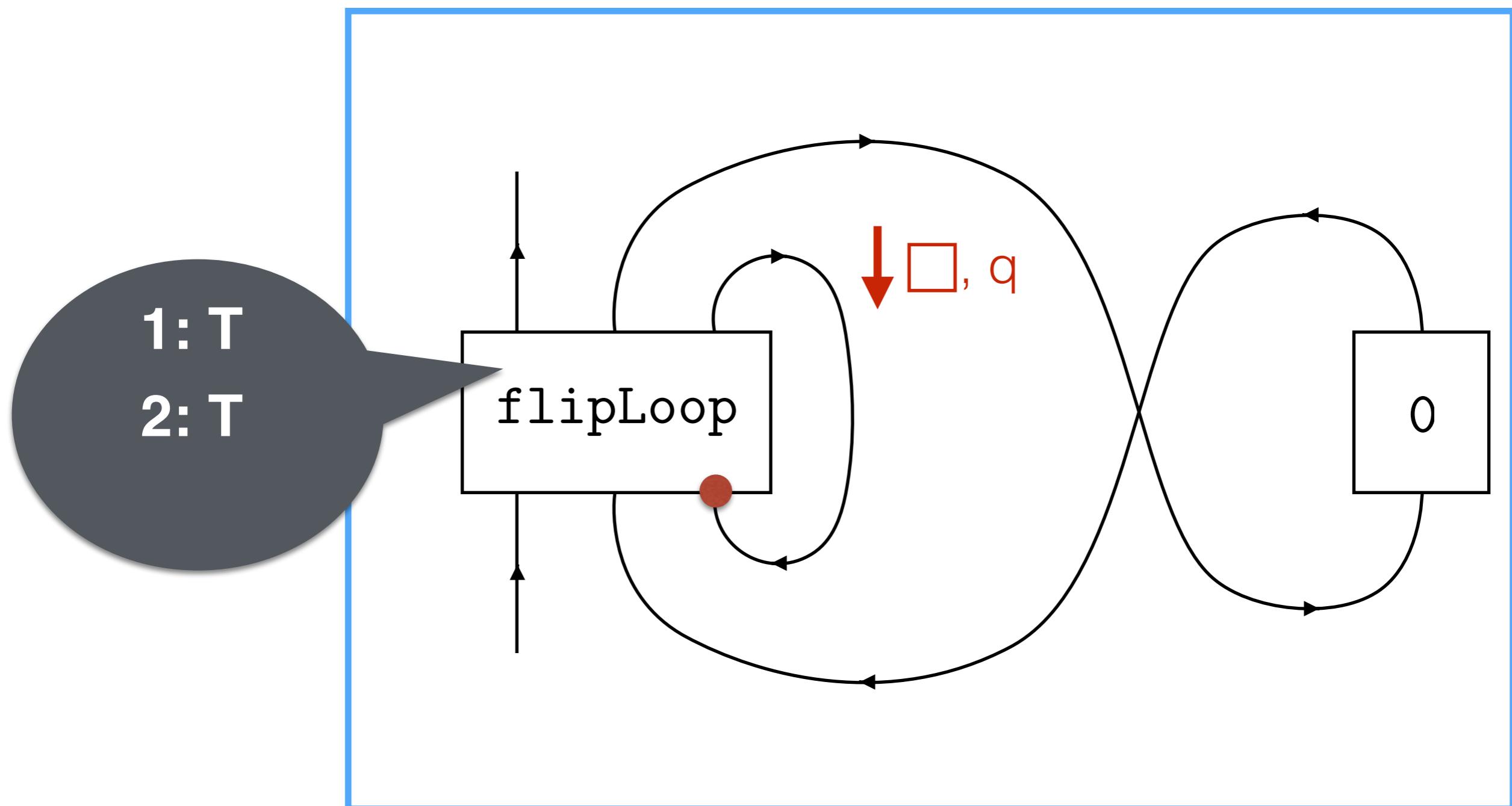
# 例：再帰的な確率プログラム

```
(rec(flipLoop, x).choose0.4(x, flipLoop(x + 1))) 0
```



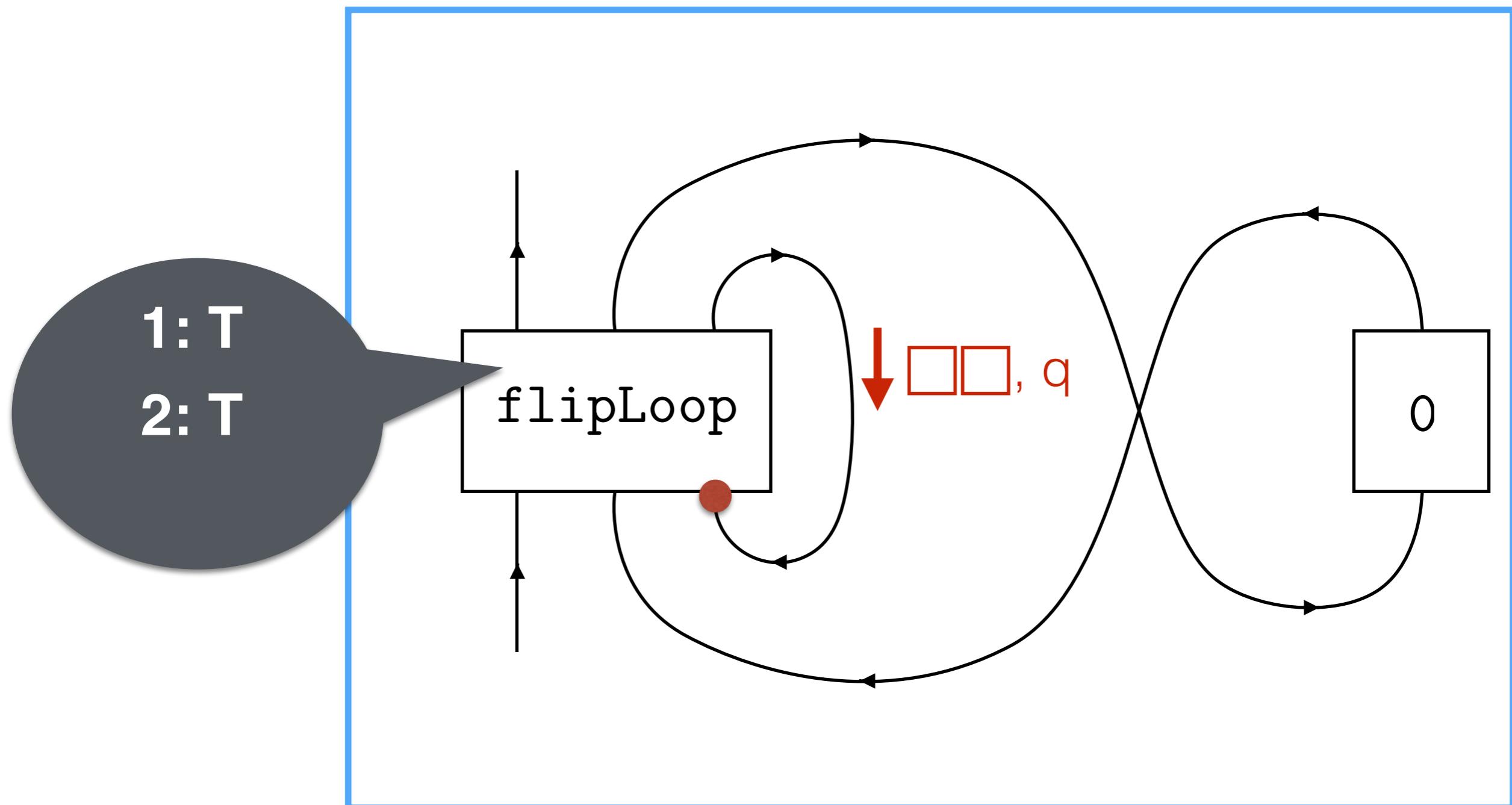
# 例：再帰的な確率プログラム

```
(rec(flipLoop, x).choose0.4(x, flipLoop(x + 1))) 0
```



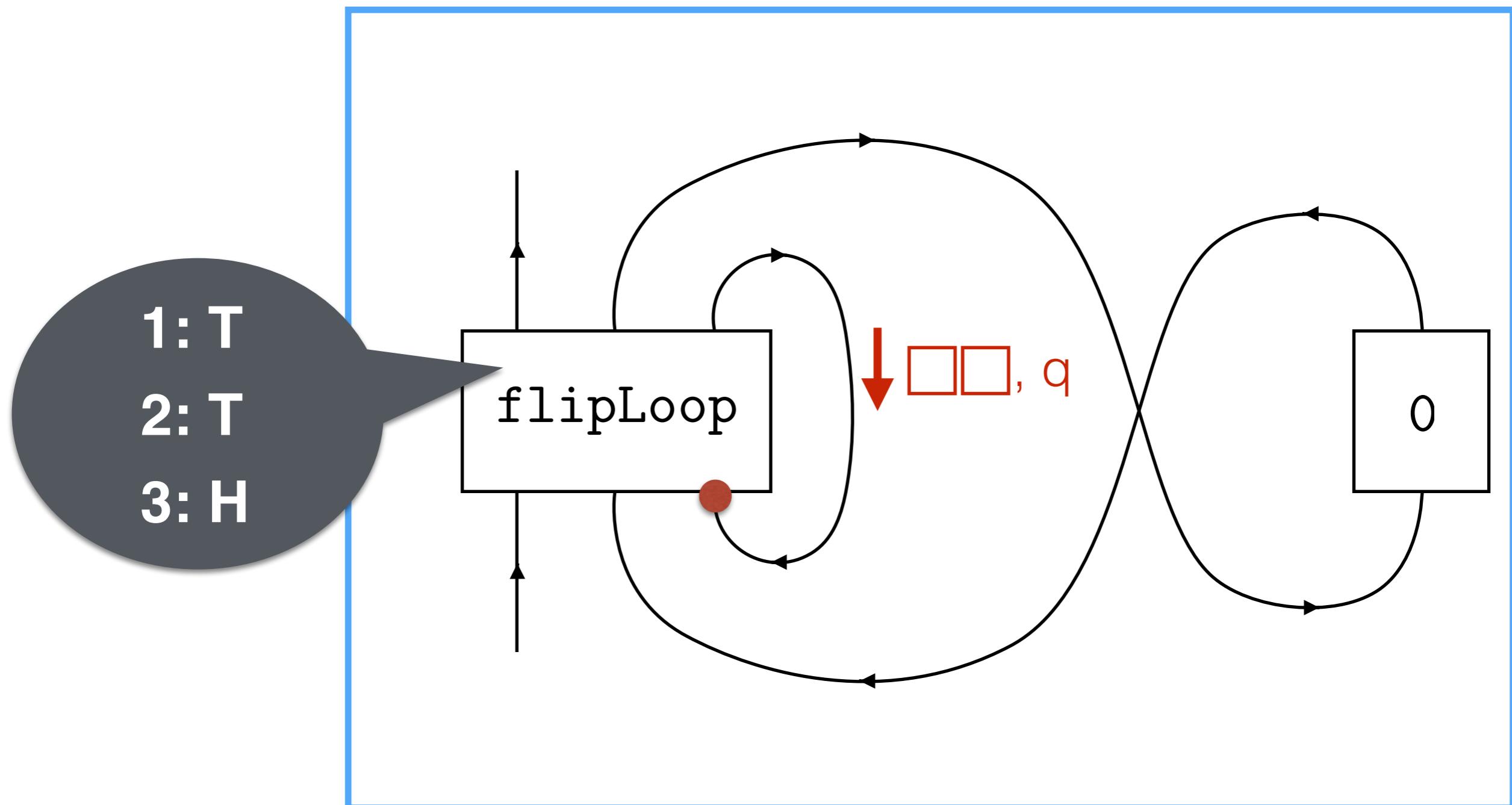
# 例：再帰的な確率プログラム

```
(rec(flipLoop, x).choose0.4(x, flipLoop(x + 1))) 0
```



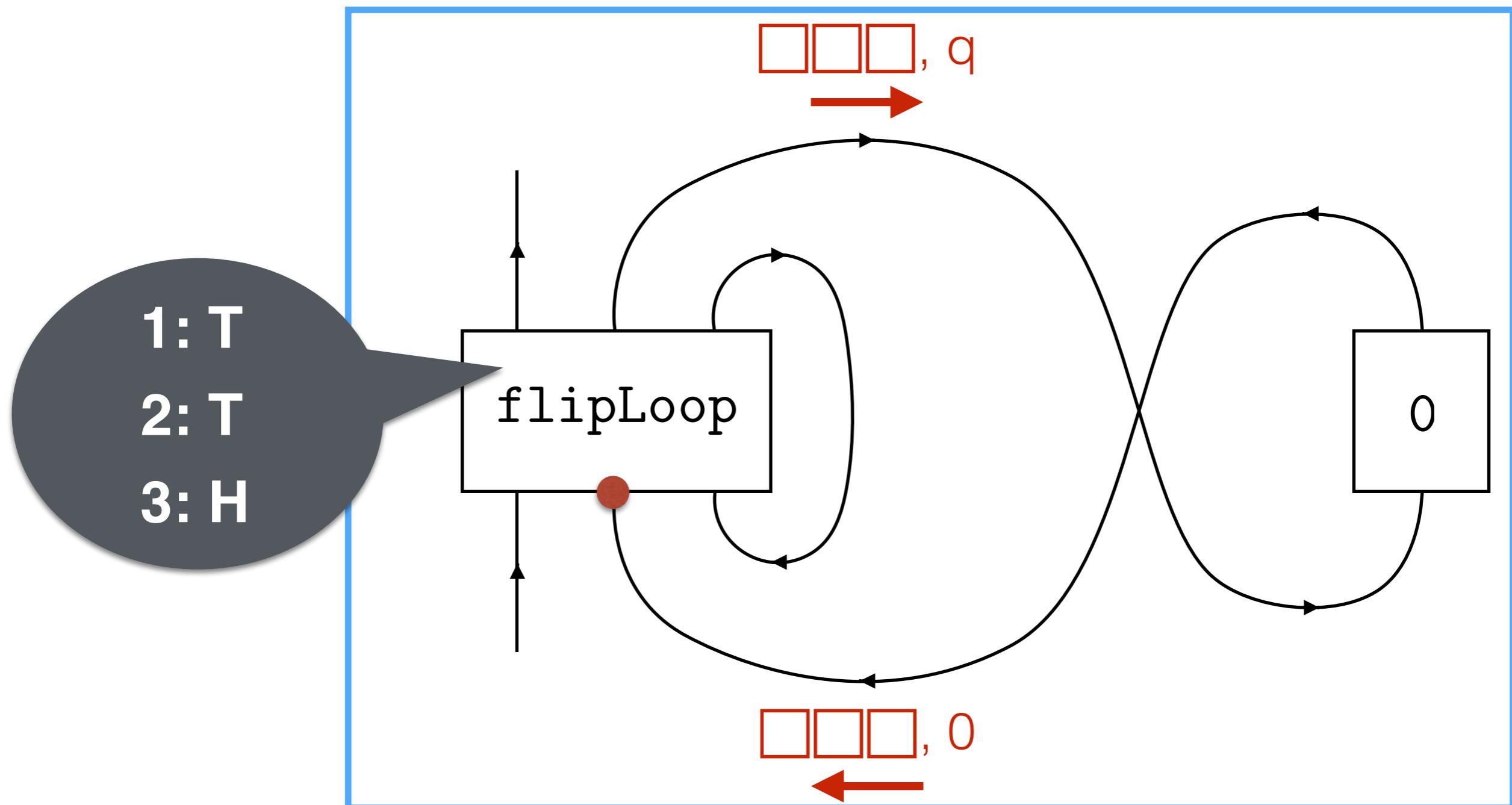
# 例：再帰的な確率プログラム

```
(rec(flipLoop, x).choose0.4(x, flipLoop(x + 1))) 0
```



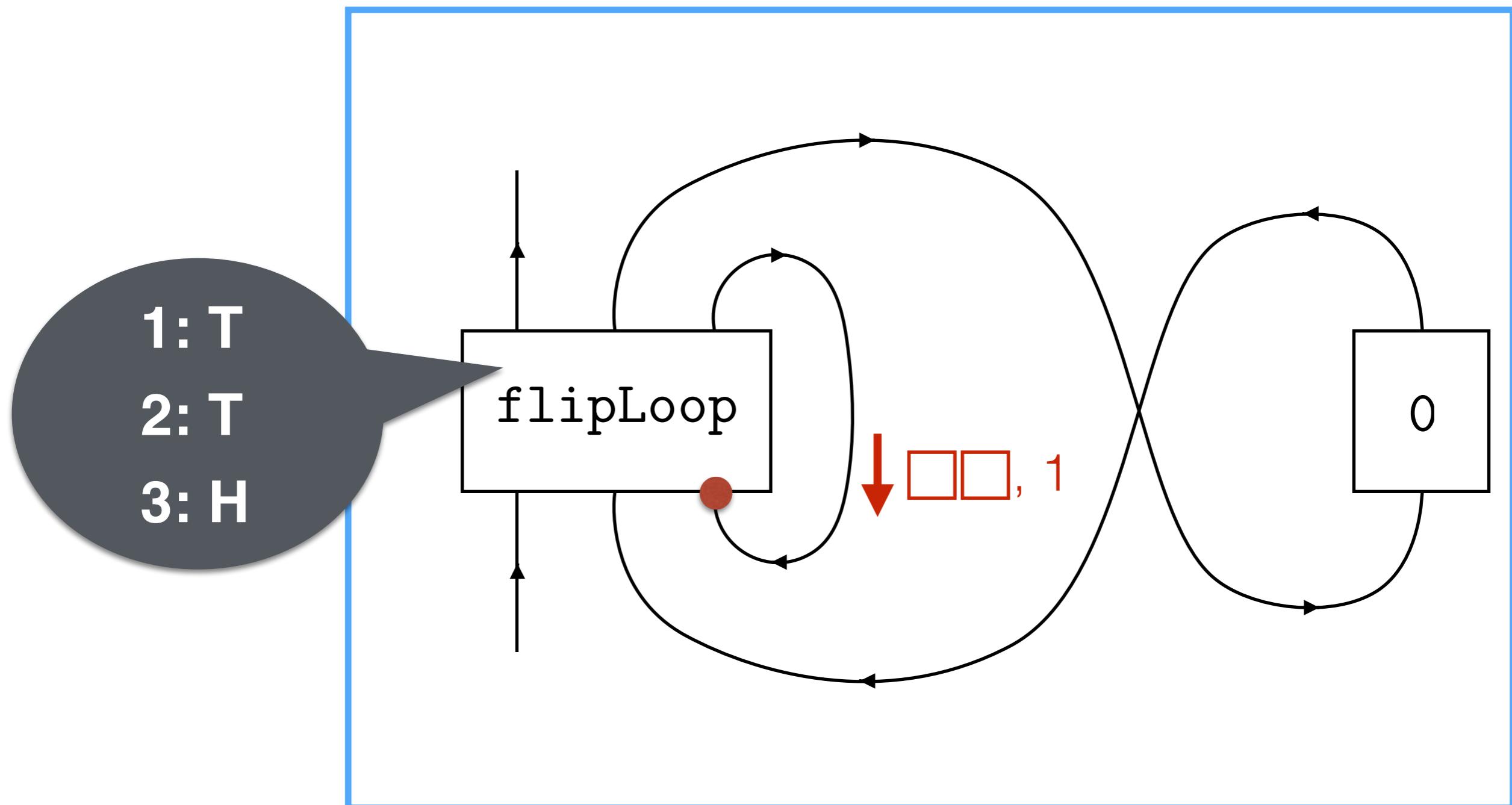
# 例：再帰的な確率プログラム

```
(rec(flipLoop, x).choose0.4(x, flipLoop(x + 1))) 0
```



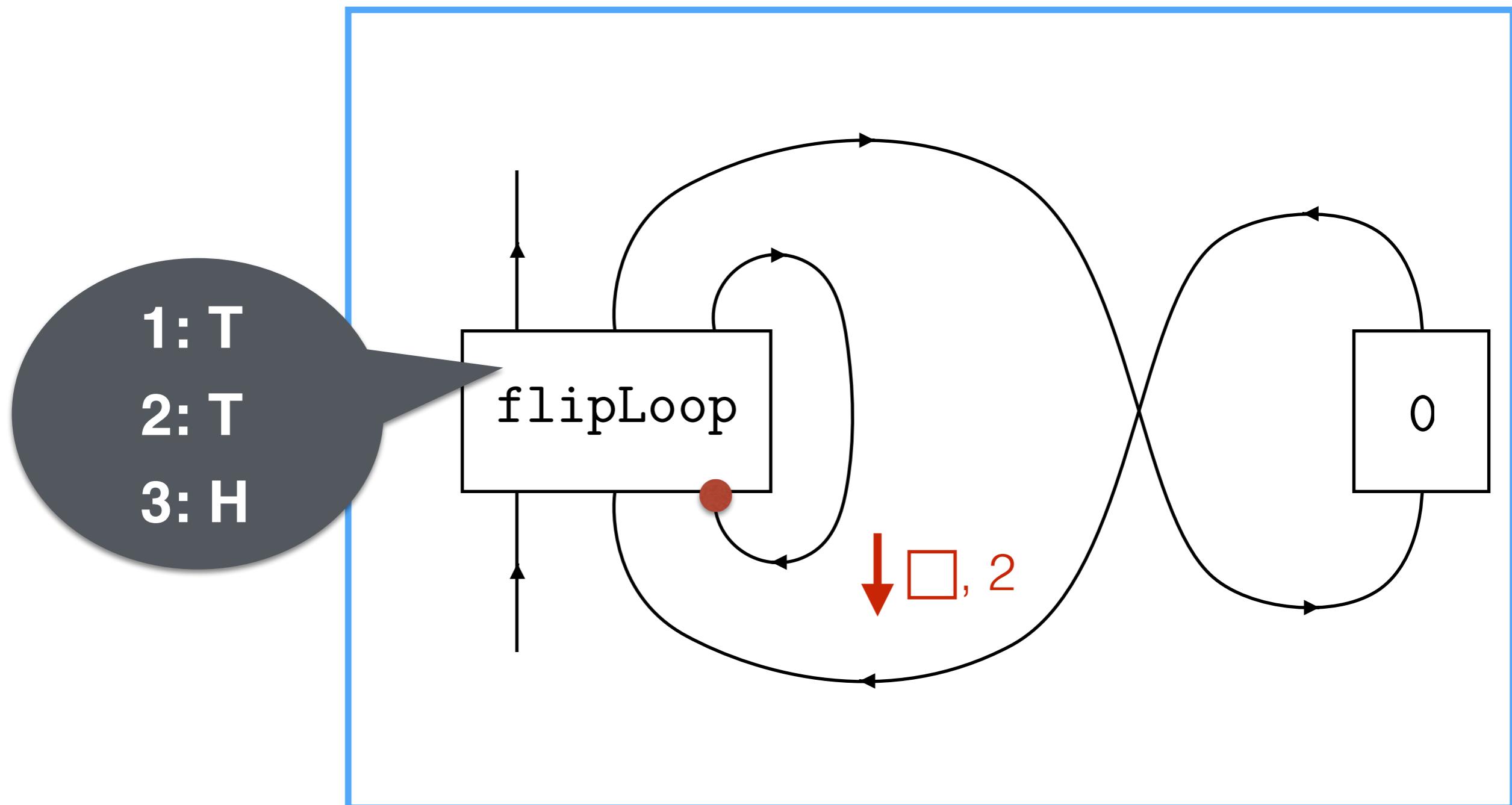
# 例：再帰的な確率プログラム

```
(rec(flipLoop, x).choose0.4(x, flipLoop(x + 1))) 0
```



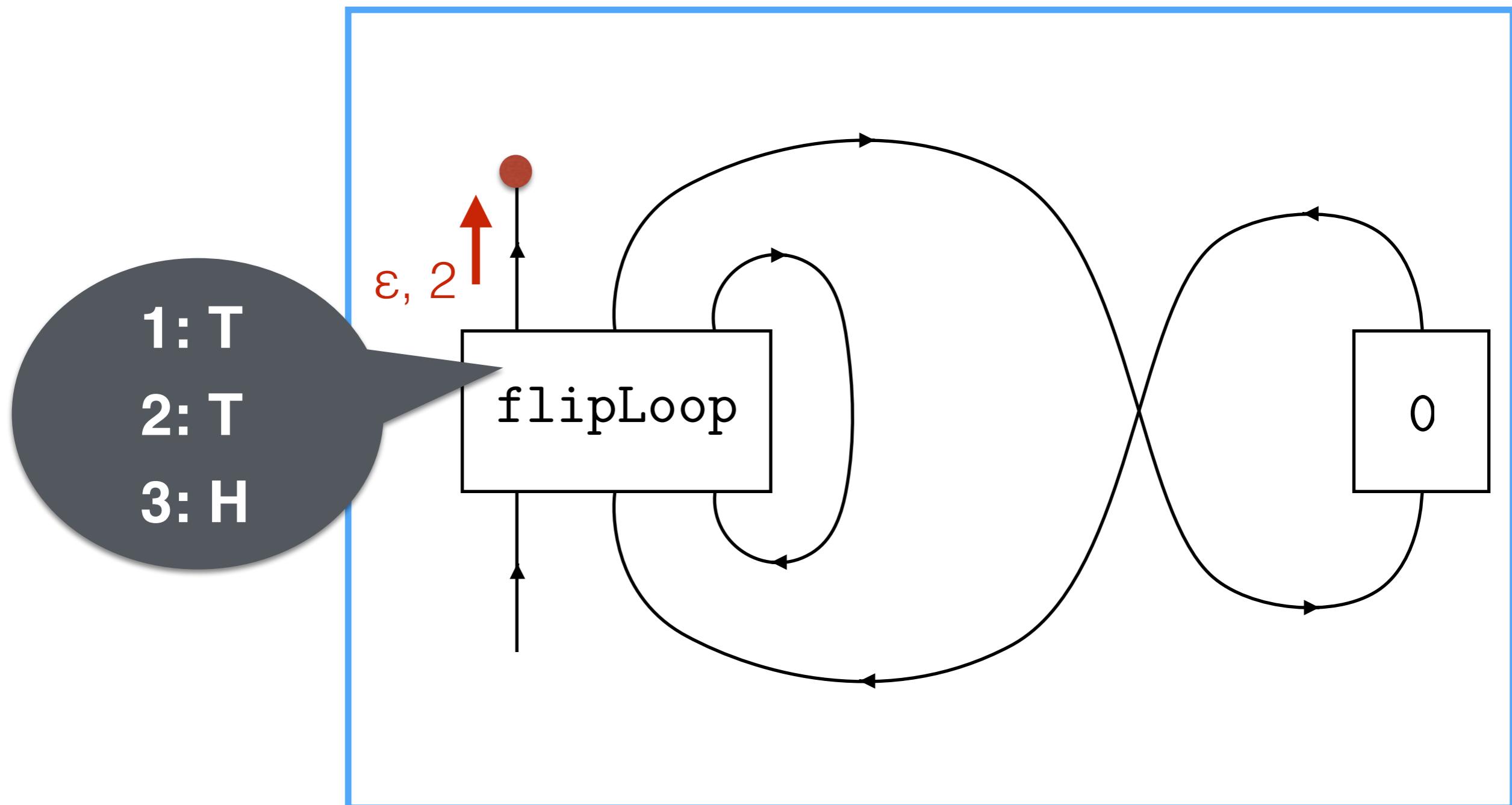
# 例：再帰的な確率プログラム

```
(rec(flipLoop, x).choose0.4(x, flipLoop(x + 1))) 0
```



# 例：再帰的な確率プログラム

```
(rec(flipLoop, x).choose0.4(x, flipLoop(x + 1))) 0
```



# 可視化ツール *TtT*

<http://koko-m.github.io/TtT/>

The screenshot shows a web-based visualization tool for terms in the *memoryful Go!* framework. At the top, a menu bar includes 'QuickTime Player', 'File', 'Edit', 'View', 'Window', and 'Help'. Below the menu is a browser-like header with tabs ('TtT'), a search bar ('Search'), and a URL field ('koko-m.github.io/TtT/'). The main content area has a dark red header bar with the text 'TtT (Terms to Transducers)'. Below this, a text input field contains the term '((rec(flipLoopSimple x) (choose(0.4) x (flipLoopSimple x))) 0)' and features a play/pause control bar with buttons for play, pause, and volume, along with a zoom slider set to 300%. A large gray rectangular area below the input field represents the visualization canvas. To the right of the main content area are two empty white boxes.

Enter a term, or type ";ex" to select one from 13 examples. [[read documents](#)]

((rec(flipLoopSimple x) (choose(0.4) x (flipLoopSimple x))) 0)

300

This is a simulation tool of the [memoryful Go!](#) framework.  
Implemented by [Koko Muroya](#), using [Processing.js](#) v1.4.8 and [PEG.js](#) v0.8.0.

# 可視化ツール *TtT*

<http://koko-m.github.io/TtT/>

The screenshot shows a web-based visualization tool for the *memoryful Go!* framework. At the top, a menu bar includes QuickTime Player, File, Edit, View, Window, and Help. The title bar reads "TtT". The address bar shows the URL "koko-m.github.io/TtT/". Below the title bar, a large red header bar contains the text "TtT (Terms to Transducers)". A message prompt "Enter a term, or type ";ex" to select one from 13 examples. [read documents]" is displayed. To the right of the message is a control panel with a play/pause button, a progress bar set to 300, and a zoom-in icon. A large gray rectangular area below the message is currently empty. On the right side of the screen, there are two empty white boxes.

Enter a term, or type ";ex" to select one from 13 examples. [[read documents](#)]

((rec(flipLoopSimple x) (choose(0.4) x (flipLoopSimple x))) 0)

300

This is a simulation tool of the [memoryful Go!](#) framework.  
Implemented by [Koko Muroya](#), using [Processing.js](#) v1.4.8 and [PEG.js](#) v0.8.0.

# 概要

コンパイラ / 高位合成  
[Mackie '95] [Ghica '07]

暗黙的計算量理論  
[Dal Lago '06]

Gol  
[Girard '89]

memoryful Gol  
[Hoshino, M. & Hasuo '14]

memoryful Gol  
with recursion



# 今後の課題

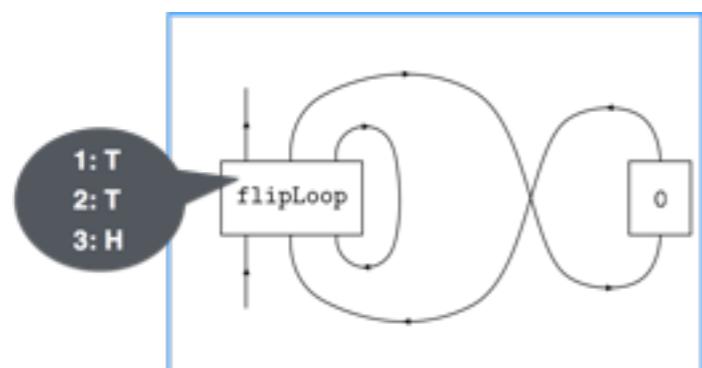
関数型プログラム + 計算副作用 + 再帰

```
(rec(flipLoop, x). choose0..4(x, flipLoop(x + 1))) 0
```

memoryful Go  
with recursion



メモリつきトーケンマシン



- 必要なリソース量の見積もり
  - 状態空間 / トークンのデータ
  - 実行コストの削減
  - プログラム評価コストとの対応

# 論文

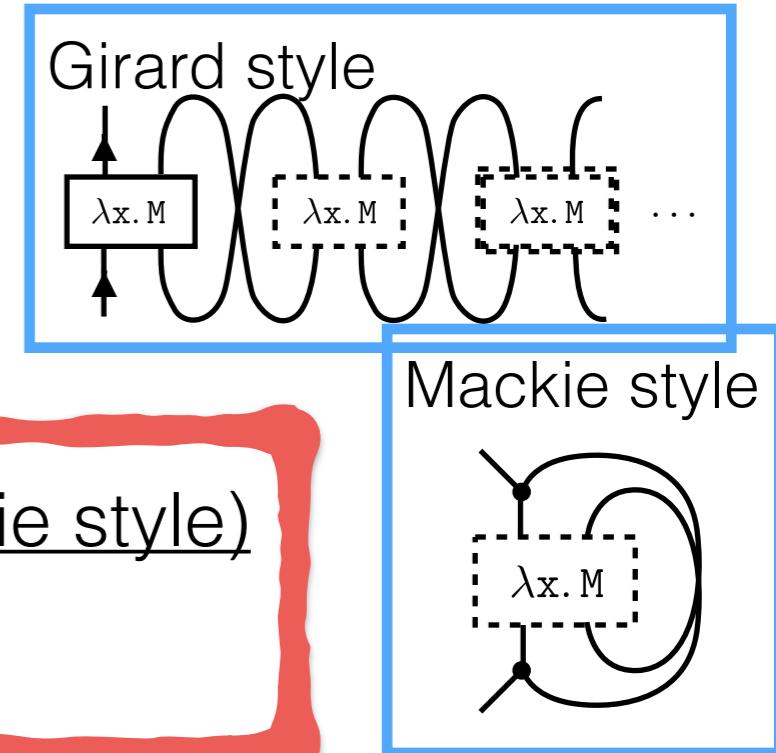
- Naohiko Hoshino, Koko Muroya & Ichiro Hasuo, “Memoryful Geometry of Interaction: from coalgebraic components to algebraic effects.”  
In Proc. of the *Joint Meeting of the 23rd EACSL Annual Conference on Computer Science Logic and the 29th Annual ACM/IEEE Symposium on Logic in Computer Science (CSL-LICS 2014)*, paper 52, 2014.
- Koko Muroya, Naohiko Hoshino & Ichiro Hasuo, “Memoryful Geometry of Interaction II: recursion and adequacy.”  
In Proc. of the *43rd ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL 2016)*, p. 748–760, 2016.

# 口頭発表

- “Compiling effectful terms to transducers: prototype implementation of memoryful Geometry of Interaction” (preliminary report).
  - *the 5th Workshop on Syntax and Semantics of Low-Level Languages (LOLA 2014)*, Vienna, Austria, 2014 年 7 月.
  - 第 25 回代数、論理、幾何と情報科学研究集会 (ALGI 25), 神奈川大学, 2014 年 8 月.
- “Memoryful Geometry of Interaction with recursion” (preliminary report).
  - 理論計算機科学と圈論ワークショップ (CSCAT 2015), 鹿児島大学, 2015 年 3 月.
  - *the 6th Workshop on Syntax and Semantics of Low-Level Languages (LOLA 1015)*, Kyoto, Japan, 2015 年 7 月.
- “Memoryful Geometry of Interaction II: recursion and adequacy.”
  - *the 43rd ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL 2016)*, St. Petersburg, FL, USA, 2016 年 1 月.

# 結論：memoryful GoI with recursion

関数型プログラム + 計算副作用 + 再帰



**Thm. (coincidence)** of Girard style & Mackie style)

$$(\mathbb{M})_{\text{Girard}} \simeq (\mathbb{M})_{\text{Mackie}}$$

**Theorem 6.4** (adequacy). Any closed term  $M$  of base type nat satisfies  $\llbracket |M| \rrbracket = (\mathbb{M})^\dagger$ .

メモリつきトークンマシン

可視化ツール TtT

<http://koko-m.github.io/TtT/>

# 研究計画

