Involutory Turing Machines

Keisuke Nakano

RIEC, Tohoku University

RC 2020 @ Online

Involution f

$$\forall x \in \mathrm{dom}(f).\ \ f(f(x)) = x$$

- **Involution (a.k.a. involutory function)**
  - A function that is own inverse
  - I.e., $f(x) = y$ if and only if $f(y) = x$
  - A particular kind of reversible (≈ injective) function
- **Application of involution**
  - Mathematical proofs / cryptographic systems / Bidirectional transformation (mentioned later)

Overview

functions

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

injective functions

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

involutory functions

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

Computable

Characterized by Turing Machine (TM) etc.

w/ function semantics [AxelsenGlück16]

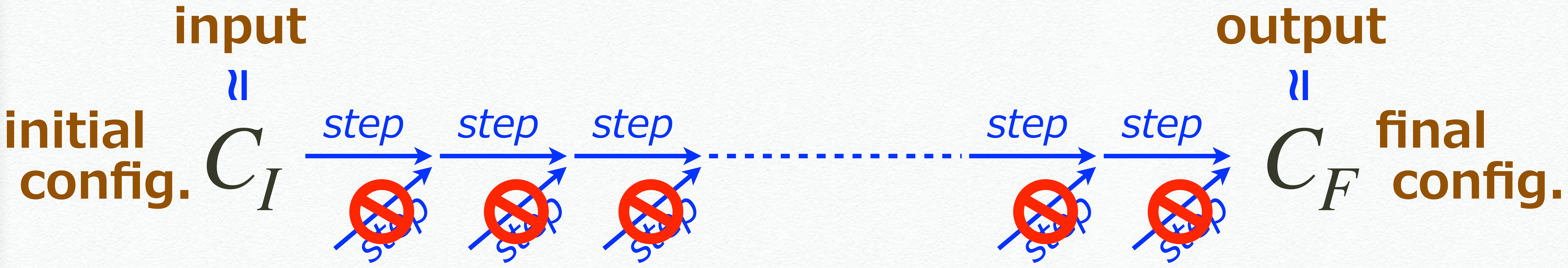
Characterized by Reversible TM (RTM)

↓ This work!

Characterized by Involutory TM (ITM)

Prior work on RTM [AxelsenGlück16]

- **RTM : backward-deterministic TM**



- RTM always computes an *injective* function
- Any computable *injective* function can be computed by an RTM.
- A *universal* RTM exists which simulates any RTM.

This work on ITM

- **ITM : *somehow restricted* TM**
  - ITM always computes an *involutory* function
  - Any computable *involutory* function can be computed by an ITM.
  - A *universal* ITM exists which simulates any ITM.

Rest of This Talk

- **Involutory Turing Machine (ITM)**
  - Definition and Semantics of TM
  - Results on Reversible TM
  - Definition of ITM
- **Properties of ITM**
  - Expressiveness (Tape Reduction / Universality)
- **Application of ITM**
  - Relationship with Bidirectional Transformation
- **Conclusion**

Turing Machine (TM)

$$T = (Q, \Sigma, q_I, q_F, \Delta)$$

set of states

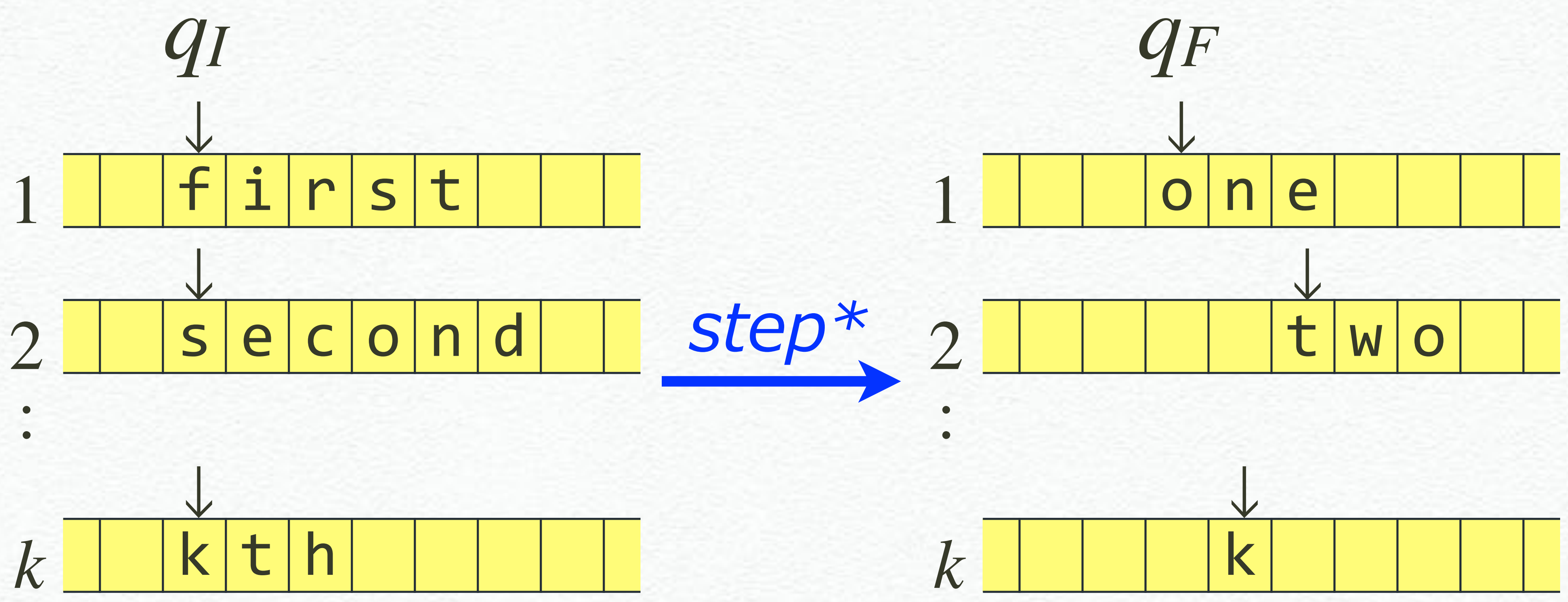
set of tape symbols (except blank)

initial state in Q

final state in Q

set of transitions

- **Working on multiple doubly-infinite tapes**
  - Each tape has a head.
  - All left cells of the head are blank *initially* and *finally*.



Transition Rule

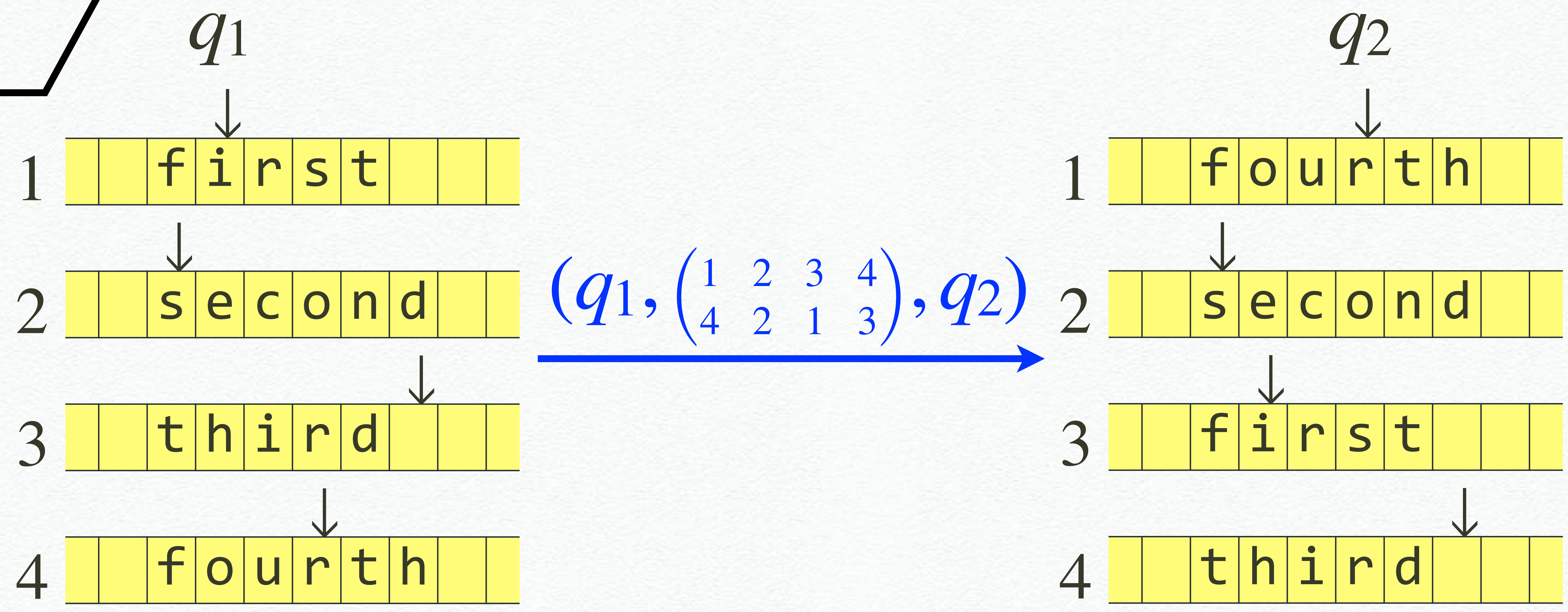
$$(q_1, a, q_2) \in \Delta$$

source state in $Q-\{q_F\}$

target state in $Q-\{q_I\}$

action

- symbol
  $a \equiv s_1 \Rightarrow s_2$
- move
  $a \equiv \leftarrow$ or $\bullet$ or $\rightarrow$
- permutation
  $$a \equiv \begin{pmatrix} 1 & 2 & \dots & k \\ i_1 & i_2 & \dots & i_k \end{pmatrix}$$



Permute the order of the tapes
with preserving contents

Deterministic/Reversible TM

- **Deterministic TM (DTM, or TM simply)**
  - (Locally) forward-deterministic TM

    $$(q, a_1, -) \neq (q, a_2, -) \in \Delta$$

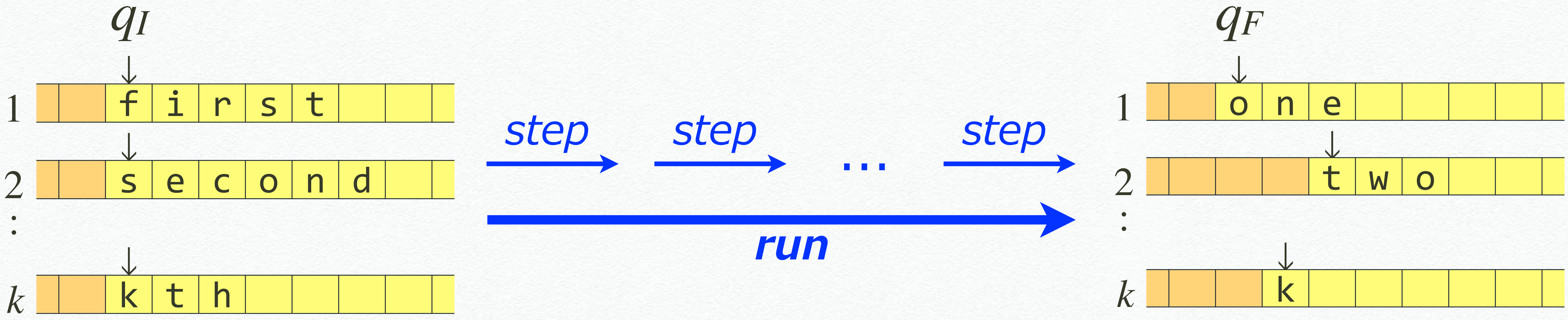
    $\Longrightarrow$ a_1 and a_2 are symbol actions w/ different *inputs*

- **Reversible TM (RTM)**
  - (Locally) forward- & backward-deterministic TM

    $$(-, a_1, q) \neq (-, a_2, q) \in \Delta$$

    $\Longrightarrow$ a_1 and a_2 are symbol actions w/ different *outputs*

Semantics of TM



$$[\![T]\!](\texttt{first}, \texttt{second}, \ldots, \texttt{kth}) = (\texttt{one}, \texttt{two}, \ldots, \texttt{k})$$

- **Function semantics** [AxelsenGlück16]
  - Input/output ≈ initial/final configuration

Convention.
When $[\![T]\!](x_1, \ldots, x_k) = (y_1, \ldots, y_k)$ implies $x_{i+1} = \ldots = x_k = y_{j+1} = \ldots = y_k = \varepsilon$, we may identify the function with $[\![T]\!](x_1, \ldots, x_i) = (y_1, \ldots, y_j)$.

Syntactic Inverse of TM

For $T = (Q, \Sigma, q_I, q_F, \Delta)$,

$$T^{-1} \stackrel{\text{def}}{=} (Q, \Sigma, q_F, q_I, \Delta^{-1})$$

where $\Delta^{-1} = \{ (q_2, a^{-1}, q_1) \mid (q_1, a, q_2) \in \Delta \}$

$(s_1 \Rightarrow s_2)^{-1} = s_2 \Rightarrow s_1$

$(\leftarrow)^{-1} = \rightarrow$, $(\bullet)^{-1} = \bullet$, $(\rightarrow)^{-1} = \leftarrow$

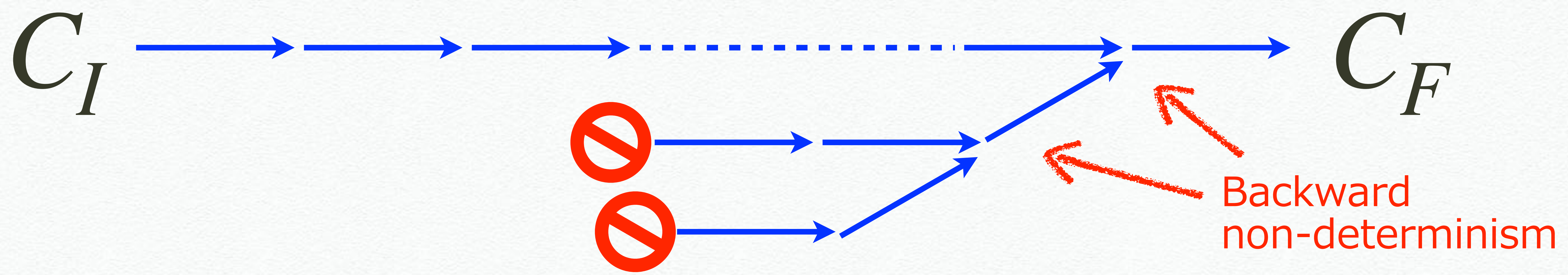
$$\begin{pmatrix} 1 & \dots & k \\ i_1 & \dots & i_k \end{pmatrix}^{-1} = \begin{pmatrix} i_1 & \dots & i_k \\ 1 & \dots & k \end{pmatrix}$$

Proposition. Let T be an RTM.
T^{-1} is an RTM s.t. $[\![T^{-1}]\!] = [\![T]\!]^{-1}$.

Expressiveness of RTM [Axelsen+16]

Proposition. If the semantics of TM T is injective, then there exists an RTM T' s.t. $[\![T']\!] = [\![T]\!]$.

- **Semantics of non-RTM can be injective.**



- Proposition implies "*an equivalent RTM always exists.*"

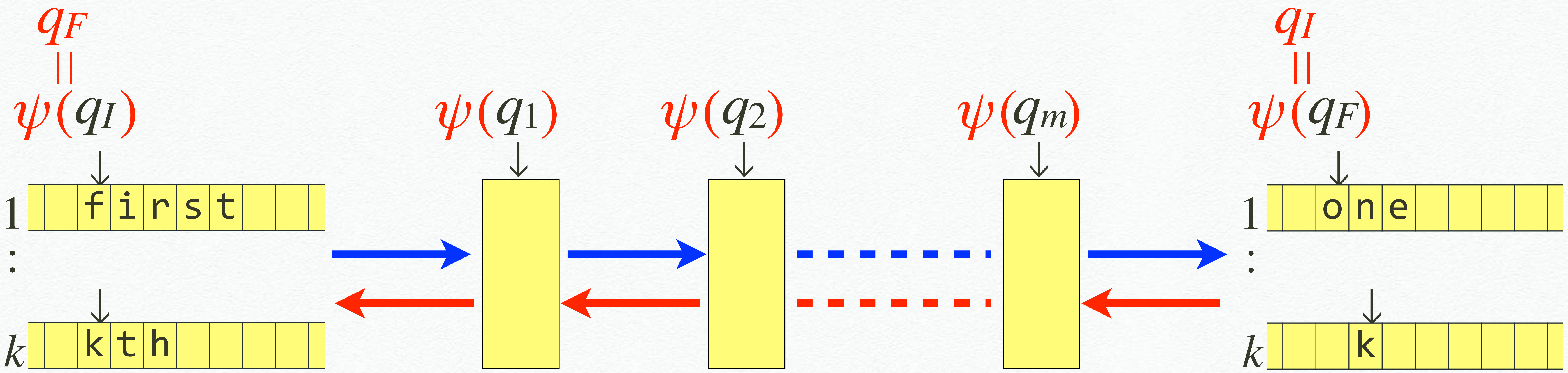
Cororally.
Any computable injective function can be computed by an RTM.

Involutory TM (ITM)

TM $T = (Q, \Sigma, q_I, q_F, \Delta)$ is ***involutory*** if

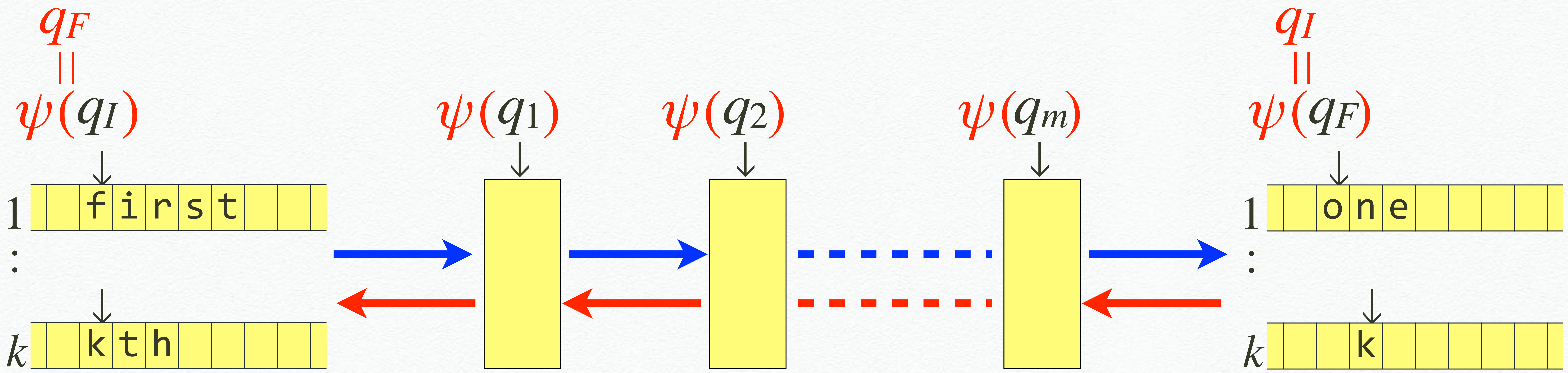
$\exists\psi$: involution over Q s.t.

- $\psi(q_I) = q_F$
- $(q_1, a, q_2) \in \Delta \Longrightarrow (\psi(q_2), a^{-1}, \psi(q_1)) \in \Delta$



Involutory TM (ITM)

Theorem. Let T be an ITM.
Then $[\![T]\!]$ is involutory, i.e., $[\![T]\!] = [\![T]\!]^{-1}$ holds.

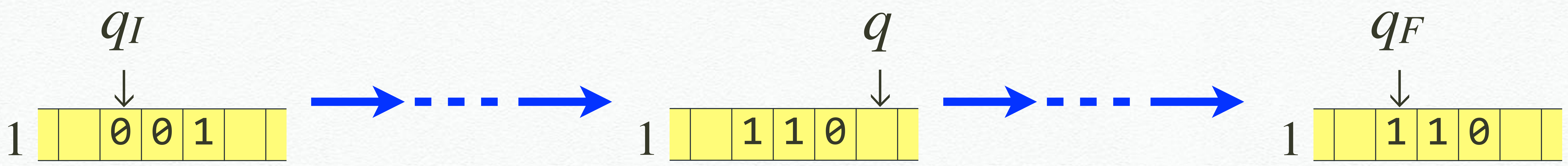


Expressiveness of ITM

Theorem. If the semantics of TM T is involutory, then there exists an ITM T' s.t. $[\![T']\!] = [\![T]\!]$.

Semantics of non-ITM can be involutory.

- Imagine a TM that computes bitwise negation by negating bits left-to-right and moving back.



- Obviously, this is not an ITM.

Cororally.
Any computable involution can be computed by an ITM.

Proof of ITM Expressiveness

Theorem. If the semantics of TM T is involutory, then there exists an ITM T' s.t. $[\![T']\!] = [\![T]\!]$.

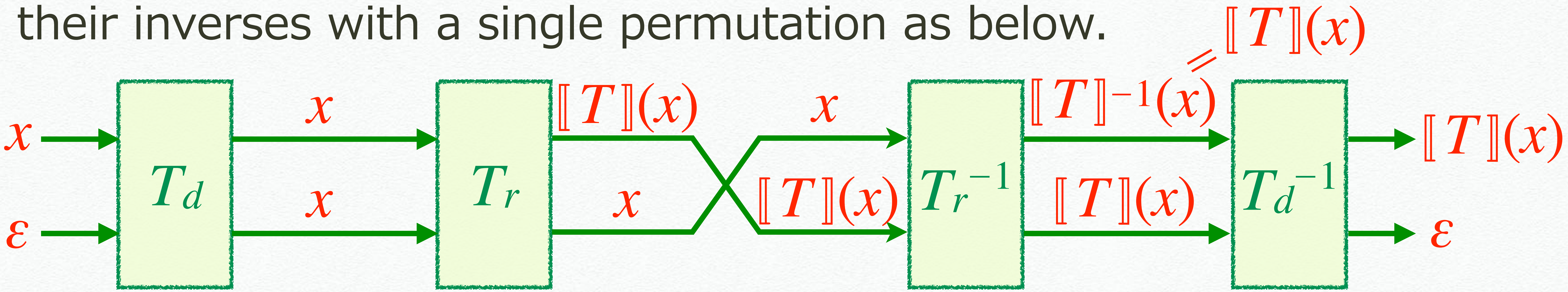
Proof sketch.

Let T be a TM s.t. $[\![T]\!]$ is involutory.

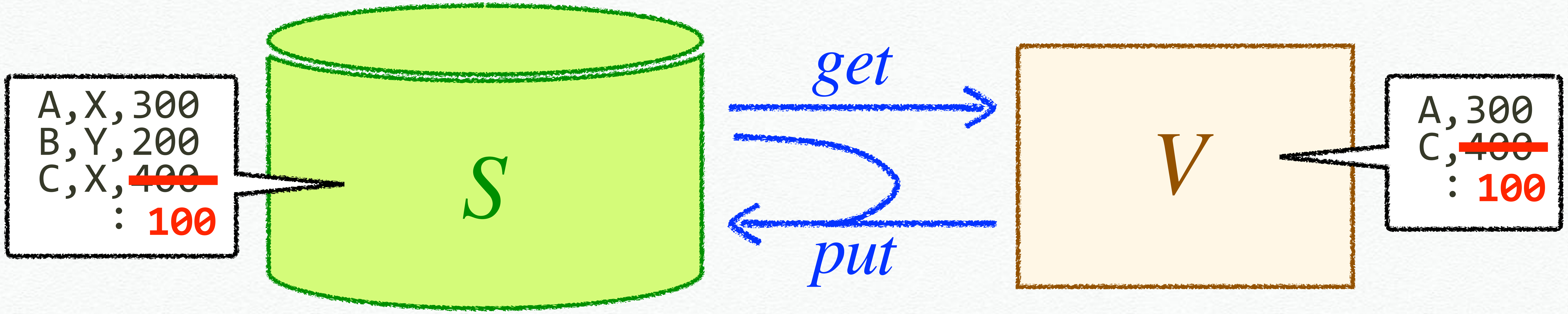
Let d and r be functions s.t. $d(x,\varepsilon) = (x,x)$ and $r(x,y) = ([\![T]\!](x), y)$.

Due to their injectivity, we have RTMs T_d and T_r s.t. $[\![T_d]\!]=d$, $[\![T_r]\!]=r$.

An ITM T' we want is obtained by concatenating T_d, T_r and their inverses with a single permutation as below.



Applications to BX



- **BX: bidirectional transformation**
  - Pair of $get : S \to V$ and $put : S \times V \to S$
  - Characterized by $pg : S \times V \to S \times V$ such that $pg(s, v) = (put(s, v), get(s))$
- **Consistency forces involutoriness of pg**
  - $pg(pg(s, v)) = (s, v)$ holds (※ for *very-well-behaved lens*)

Conclusion

- **Involutory Turing machine is presented.**
  - ITM always computes involution.
  - Any computable involution is computed by an ITM.
    - Permutation rule plays an important role for this.
  - Universal involutory Turing machine exists.
    - It can be efficiently constructed by Bennet's trick.
- **The work is motivated by my BX research.**
  - Exact computational model of BX is coming soon.