Lecture 6

Universal Turing Machine

Oblivious Turing Machine

Oblivious Turing Machine

Definition: A TM M is called an **oblivious TM**, if on every input x and $i \in \mathbb{N}$, the location

Oblivious Turing Machine

Definition: A TM M is called an **oblivious TM**, if on every input x and $i \in \mathbb{N}$, the location of each of M's heads at the ith step of execution on x is a function of $|x|$ and i.

Oblivious Turing Machine

Definition: A TM M is called an **oblivious TM**, if on every input x and $i \in \mathbb{N}$, the location of each of M's heads at the ith step of execution on x is a function of $|x|$ and i.

Claim: Every TM M that runs in time-constructible time $T(n)$ can be simulated by an

Oblivious Turing Machine

Definition: A TM M is called an **oblivious TM**, if on every input x and $i \in \mathbb{N}$, the location of each of M's heads at the ith step of execution on x is a function of $|x|$ and i.

Claim: Every TM M that runs in time-constructible time $T(n)$ can be simulated by an oblivious TM M' that runs in $O(T(n)^2)$ time.

Oblivious Turing Machine

Definition: A TM M is called an **oblivious TM**, if on every input x and $i \in \mathbb{N}$, the location of each of M's heads at the ith step of execution on x is a function of $|x|$ and i.

Claim: Every TM M that runs in time-constructible time $T(n)$ can be simulated by an oblivious TM M' that runs in $O(T(n)^2)$ time.

Proof:

Oblivious Turing Machine

Definition: A TM M is called an **oblivious TM**, if on every input x and $i \in \mathbb{N}$, the location of each of M's heads at the ith step of execution on x is a function of $|x|$ and i.

Claim: Every TM M that runs in time-constructible time $T(n)$ can be simulated by an oblivious TM M' that runs in $O(T(n)^2)$ time.

Proof: Left as an exercise. Use the idea of the last claim of the last lecture.

Oblivious Turing Machine

Definition: A TM M is called an **oblivious TM**, if on every input x and $i \in \mathbb{N}$, the location of each of M's heads at the ith step of execution on x is a function of $|x|$ and i.

Claim: Every TM M that runs in time-constructible time $T(n)$ can be simulated by an oblivious TM M' that runs in $O(T(n)^2)$ time.

Proof: Left as an exercise. Use the idea of the last claim of the last lecture. ∎

Encoding a Turing Machine

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map:

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots)$

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \quad \rightarrow$

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$

$(0,1, \triangleright, \sqcup)$

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$

$(0,1, \triangleright, \sqcup) \rightarrow$

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$

$$(0,1, \triangleright, \sqcup) \rightarrow (1,2,3,4)$$

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$

$$(0,1, \triangleright, \sqcup) \rightarrow (1,2,3,4)$$

$$(L, R, S)$$

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map:

$$(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$$

$$(0,1, \triangleright, \sqcup) \rightarrow (1,2,3,4)$$

$$(L, R, S) \rightarrow$$

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$

$$(0,1, \triangleright, \sqcup) \rightarrow (1,2,3,4)$$

$$(L, R, S) \rightarrow (1,2,3)$$

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map:

$$(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$$

$$(0,1, \triangleright, \sqcup) \rightarrow (1,2,3,4)$$

$$(L, R, S) \rightarrow (1,2,3)$$

Encoding an entry of δ:

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$

$$(0,1, \triangleright, \sqcup) \rightarrow (1,2,3,4)$$

$$(L, R, S) \rightarrow (1,2,3)$$

Encoding an entry of δ: Encode an entry of δ as encoding a tuple of integers.

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$

$$(0,1, \triangleright, \sqcup) \rightarrow (1,2,3,4)$$

$$(L, R, S) \rightarrow (1,2,3)$$

Encoding an entry of δ: Encode an entry of δ as encoding a tuple of integers.

$$\delta(q_1,0,1) = (q_2,0,L,R)$$

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map:

$$(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$$

$$(0,1, \triangleright, \sqcup) \rightarrow (1,2,3,4)$$

$$(L, R, S) \rightarrow (1,2,3)$$

Encoding an entry of δ: Encode an entry of δ as encoding a tuple of integers.

$$\delta(q_1,0,1) = (q_2,0,L,R) \rightarrow$$

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$

$$(0,1, \triangleright, \sqcup) \rightarrow (1,2,3,4)$$

$$(L, R, S) \rightarrow (1,2,3)$$

Encoding an entry of δ: Encode an entry of δ as encoding a tuple of integers.

$$\delta(q_1,0,1) = (q_2,0,L,R) \rightarrow 1111$$

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$

$$(0,1,\triangleright,\sqcup) \rightarrow (1,2,3,4)$$

$$(L,R,S) \rightarrow (1,2,3)$$

Encoding an entry of δ: Encode an entry of δ as encoding a tuple of integers.

$$\delta(q_1,0,1) = (q_2,0,L,R) \rightarrow 111101$$

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$

$$(0,1,\triangleright,\sqcup) \rightarrow (1,2,3,4)$$

$$(L,R,S) \rightarrow (1,2,3)$$

Encoding an entry of δ: Encode an entry of δ as encoding a tuple of integers.

$$\delta(q_1,0,1) = (q_2,0,L,R) \rightarrow 11110111$$

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$

$$(0,1, \triangleright, \sqcup) \rightarrow (1,2,3,4)$$

$$(L, R, S) \rightarrow (1,2,3)$$

Encoding an entry of δ: Encode an entry of δ as encoding a tuple of integers.

$$\delta(q_1,0,1) = (q_2,0,L,R) \rightarrow 1111011101$$

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$

$$(0,1,\triangleright,\sqcup) \rightarrow (1,2,3,4)$$

$$(L,R,S) \rightarrow (1,2,3)$$

Encoding an entry of δ: Encode an entry of δ as encoding a tuple of integers.

$$\delta(q_1,0,1) = (q_2,0,L,R) \rightarrow 1111011101111100$$

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$

$$(0,1, \triangleright, \sqcup) \rightarrow (1,2,3,4)$$

$$(L, R, S) \rightarrow (1,2,3)$$

Encoding an entry of δ: Encode an entry of δ as encoding a tuple of integers.

$$\delta(q_1,0,1) = (q_2,0,L,R) \rightarrow 1111\,01\,111\,01\,1100\,01$$

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$

$$(0,1,\triangleright,\sqcup) \rightarrow (1,2,3,4)$$

$$(L,R,S) \rightarrow (1,2,3)$$

Encoding an entry of δ: Encode an entry of δ as encoding a tuple of integers.

$$\delta(q_1,0,1) = (q_2,0,L,R) \rightarrow 1111\,01\,11\,01\,1100\,01\,110000$$

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$

$$(0,1, \triangleright, \sqcup) \rightarrow (1,2,3,4)$$

$$(L, R, S) \rightarrow (1,2,3)$$

Encoding an entry of δ: Encode an entry of δ as encoding a tuple of integers.

$$\delta(q_1,0,1) = (q_2,0,L,R) \rightarrow 1111\,01\,11\,01\,1100\,01\,110000\,01$$

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$

$$(0,1, \triangleright, \sqcup) \rightarrow (1,2,3,4)$$

$$(L, R, S) \rightarrow (1,2,3)$$

Encoding an entry of δ: Encode an entry of δ as encoding a tuple of integers.

$$\delta(q_1,0,1) = (q_2,0,L,R) \rightarrow 1111011101110001110000 0111$$

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$

$$(0,1,\triangleright,\sqcup) \rightarrow (1,2,3,4)$$

$$(L, R, S) \rightarrow (1,2,3)$$

Encoding an entry of δ: Encode an entry of δ as encoding a tuple of integers.

$$\delta(q_1,0,1) = (q_2,0,L,R) \rightarrow 1111\,01\,11\,01\,1100\,01\,110000\,01\,11\,01$$

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$

$$(0,1, \triangleright, \sqcup) \rightarrow (1,2,3,4)$$

$$(L, R, S) \rightarrow (1,2,3)$$

Encoding an entry of δ: Encode an entry of δ as encoding a tuple of integers.

$$\delta(q_1,0,1) = (q_2,0,L,R) \rightarrow 1111\,01\,11\,01\,1100\,01\,110000\,01\,11\,01\,11$$

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$

$$(0,1,\triangleright,\sqcup) \rightarrow (1,2,3,4)$$

$$(L,R,S) \rightarrow (1,2,3)$$

Encoding an entry of δ: Encode an entry of δ as encoding a tuple of integers.

$$\delta(q_1,0,1) = (q_2,0,L,R) \rightarrow 1111011101110001110000011101101$$

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$

$$(0,1,\triangleright,\sqcup) \rightarrow (1,2,3,4)$$

$$(L,R,S) \rightarrow (1,2,3)$$

Encoding an entry of δ: Encode an entry of δ as encoding a tuple of integers.

$$\delta(q_1,0,1) = (q_2,0,L,R) \rightarrow 1111\,01\,11\,01\,1100\,01\,110000\,01\,11\,01\,11\,01\,1100$$

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$

$$(0,1,\triangleright,\sqcup) \rightarrow (1,2,3,4)$$

$$(L,R,S) \rightarrow (1,2,3)$$

Encoding an entry of δ: Encode an entry of δ as encoding a tuple of integers.

$$\delta(q_1,0,1) = (q_2,0,L,R) \rightarrow 1111011101110001110000011101110111100$$

Encoding the δ:

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$

$$(0,1,\triangleright,\sqcup) \rightarrow (1,2,3,4)$$

$$(L,R,S) \rightarrow (1,2,3)$$

Encoding an entry of δ: Encode an entry of δ as encoding a tuple of integers.

$$\delta(q_1,0,1) = (q_2,0,L,R) \rightarrow 1111011101110001110000011101110111100$$

Encoding the δ: Concatenate the encodings of all entries with 0101.

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map:

$$(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$$

$$(0,1, \triangleright, \sqcup) \rightarrow (1,2,3,4)$$

$$(L, R, S) \rightarrow (1,2,3)$$

Encoding an entry of δ: Encode an entry of δ as encoding a tuple of integers.

$$\delta(q_1,0,1) = (q_2,0,L,R) \rightarrow 1111\,01\,11\,01\,1100\,01\,110000\,01\,11\,01\,11\,01\,1100$$

Encoding the δ: Concatenate the encodings of all entries with 0101.

$enc(1st$ entry)

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$

$$(0,1, \triangleright, \sqcup) \rightarrow (1,2,3,4)$$

$$(L, R, S) \rightarrow (1,2,3)$$

Encoding an entry of δ: Encode an entry of δ as encoding a tuple of integers.

$$\delta(q_1,0,1) = (q_2,0,L,R) \rightarrow 1111\,01\,11\,01\,1100\,01\,110000\,01\,11\,01\,11\,01\,1100$$

Encoding the δ: Concatenate the encodings of all entries with 0101.

$enc(1st$ entry$)\,0101$

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$

$$(0,1,\triangleright,\sqcup) \rightarrow (1,2,3,4)$$

$$(L,R,S) \rightarrow (1,2,3)$$

Encoding an entry of δ: Encode an entry of δ as encoding a tuple of integers.

$$\delta(q_1,0,1) = (q_2,0,L,R) \rightarrow 1111011101110001110000011101110111100$$

Encoding the δ: Concatenate the encodings of all entries with 0101.

$enc(1st$ entry$)$ 0101 $enc(2st$ entry$)$

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$

$$(0,1,\triangleright,\sqcup) \rightarrow (1,2,3,4)$$

$$(L,R,S) \rightarrow (1,2,3)$$

Encoding an entry of δ: Encode an entry of δ as encoding a tuple of integers.

$$\delta(q_1,0,1) = (q_2,0,L,R) \rightarrow 1111\,01\,11\,01\,1100\,01\,110000\,01\,11\,01\,11\,01\,1100$$

Encoding the δ: Concatenate the encodings of all entries with 0101.

$$enc(1st \text{ entry})\,0101\,enc(2st \text{ entry}) \quad \ldots$$

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$

$(0,1, \triangleright, \sqcup) \rightarrow (1,2,3,4)$

$(L, R, S) \rightarrow (1,2,3)$

Encoding an entry of δ: Encode an entry of δ as encoding a tuple of integers.

$$\delta(q_1,0,1) = (q_2,0,L,R) \rightarrow 1111011101110001110000011101110111100$$

Encoding the δ: Concatenate the encodings of all entries with 0101.

$enc(1st \text{ entry})\,0101\,enc(2st \text{ entry}) \quad \ldots \quad 0101$

Encoding a Turing Machine

Observation: We can represent a TM M as a binary string by encoding the δ of M.

Map: $(q_{start}, q_{halt}, q_1, q_2, \ldots) \rightarrow (1,2,3,4,\ldots)$

$(0,1, \triangleright, \sqcup) \rightarrow (1,2,3,4)$

$(L, R, S) \rightarrow (1,2,3)$

Encoding an entry of δ: Encode an entry of δ as encoding a tuple of integers.

$$\delta(q_1,0,1) = (q_2,0,L,R) \rightarrow 1111011101110001110000011101110111100$$

Encoding the δ: Concatenate the encodings of all entries with 0101.

$$enc(1st \text{ entry})\, 0101\, enc(2st \text{ entry}) \quad \ldots \quad 0101\, enc(kth \text{ entry})$$

Encoding a Turing Machine

Encoding a Turing Machine

Notation:

Encoding a Turing Machine

Notation:

- $\langle M \rangle$ denotes the encoding of M.

Encoding a Turing Machine

Notation:

- $\langle M \rangle$ denotes the encoding of M.
- M_α denotes the TM that binary string α represents.

Encoding a Turing Machine

Notation:

- $\langle M \rangle$ denotes the encoding of M.
- M_α denotes the TM that binary string α represents.

Essential properties for encoding a TM:

Encoding a Turing Machine

Notation:

- $\langle M \rangle$ denotes the encoding of M.
- M_α denotes the TM that binary string α represents.

Essential properties for encoding a TM:

- Every binary string represents a TM.

Encoding a Turing Machine

Notation:

- $\langle M \rangle$ denotes the encoding of M.
- M_α denotes the TM that binary string α represents.

Essential properties for encoding a TM:

- Every binary string represents a TM. (*Map invalid strings to a trivial TM.*)

Encoding a Turing Machine

Notation:

- $\langle M \rangle$ denotes the encoding of M.
- M_α denotes the TM that binary string α represents.

Essential properties for encoding a TM:

- Every binary string represents a TM. (*Map invalid strings to a trivial TM.*)
- Every TM is represented by infinitely many binary strings.

Encoding a Turing Machine

Notation:

- $\langle M \rangle$ denotes the encoding of M.
- M_α denotes the TM that binary string α represents.

Essential properties for encoding a TM:

- Every binary string represents a TM. (*Map invalid strings to a trivial TM.*)
- Every TM is represented by infinitely many binary strings.

$$enc(1st\text{ entry})\,0101\,enc(2st\text{ entry}) \quad \ldots \quad 0101\,enc(kth\text{ entry})$$

Encoding a Turing Machine

Notation:

- $\langle M \rangle$ denotes the encoding of M.
- M_α denotes the TM that binary string α represents.

Essential properties for encoding a TM:

- Every binary string represents a TM. (*Map invalid strings to a trivial TM.*)
- Every TM is represented by infinitely many binary strings.

$$enc(1st\text{ entry})\, 0101\, enc(2st\text{ entry}) \quad \ldots \quad 0101\, enc(kth\text{ entry})\, 0101$$

Encoding a Turing Machine

Notation:

- $\langle M \rangle$ denotes the encoding of M.
- M_α denotes the TM that binary string α represents.

Essential properties for encoding a TM:

- Every binary string represents a TM. (*Map invalid strings to a trivial TM.*)
- Every TM is represented by infinitely many binary strings.

$$enc(1st \text{ entry})\, 0101\, enc(2st \text{ entry}) \quad \dots \quad 0101\, enc(kth \text{ entry}) 01011$$

Encoding a Turing Machine

Notation:

- $\langle M \rangle$ denotes the encoding of M.
- M_α denotes the TM that binary string α represents.

Essential properties for encoding a TM:

- Every binary string represents a TM. (*Map invalid strings to a trivial TM.*)
- Every TM is represented by infinitely many binary strings.

$$enc(1st \text{ entry})\, 0101\, enc(2st \text{ entry}) \quad \ldots \quad 0101\, enc(kth \text{ entry})\, 010111$$

Encoding a Turing Machine

Notation:

- $\langle M \rangle$ denotes the encoding of M.
- M_α denotes the TM that binary string α represents.

Essential properties for encoding a TM:

- Every binary string represents a TM. (*Map invalid strings to a trivial TM.*)
- Every TM is represented by infinitely many binary strings.

$$enc(1st \text{ entry})\, 0101\, enc(2st \text{ entry}) \quad \ldots \quad 0101\, enc(kth \text{ entry})\, 0101111$$

Encoding a Turing Machine

Notation:

- $\langle M \rangle$ denotes the encoding of M.
- M_α denotes the TM that binary string α represents.

Essential properties for encoding a TM:

- Every binary string represents a TM. (*Map invalid strings to a trivial TM.*)
- Every TM is represented by infinitely many binary strings.

$$enc(1st \text{ entry})\, 0101\, enc(2st \text{ entry}) \quad \ldots \quad 0101\, enc(kth \text{ entry})\, 0101\, 1111$$

Encoding a Turing Machine

Notation:

- $\langle M \rangle$ denotes the encoding of M.
- M_α denotes the TM that binary string α represents.

Essential properties for encoding a TM:

- Every binary string represents a TM. (*Map invalid strings to a trivial TM.*)
- Every TM is represented by infinitely many binary strings.

$$enc(1st \text{ entry})\ 0101\ enc(2st \text{ entry}) \quad \ldots \quad 0101\ enc(kth \text{ entry})\ 0101\ 1111\ldots$$

Encoding a Turing Machine

Notation:

- $\langle M \rangle$ denotes the encoding of M.
- M_α denotes the TM that binary string α represents.

Essential properties for encoding a TM:

- Every binary string represents a TM. (*Map invalid strings to a trivial TM.*)
- Every TM is represented by infinitely many binary strings.

$$enc(1st \text{ entry})\,0101\,enc(2st \text{ entry}) \quad \ldots \quad 0101\,enc(kth \text{ entry})\,0101\,1111\ldots$$

Encoding a Turing Machine

Notation:

- $\langle M \rangle$ denotes the encoding of M.
- M_α denotes the TM that binary string α represents.

Essential properties for encoding a TM:

- Every binary string represents a TM. (*Map invalid strings to a trivial TM.*)
- Every TM is represented by infinitely many binary strings.

$$enc(1st \text{ entry})\, 0101\, enc(2st \text{ entry}) \quad \ldots \quad 0101\, enc(kth \text{ entry})\, 0101\, 1111\ldots$$

add arbitrary number of 1s in the end that are ignored

Universal Turing Machine

Universal Turing Machine

- Given (α, x) a **universal Turing machine** can simulate M_α on x.

Universal Turing Machine

- Given (α, x) a **universal Turing machine** can simulate M_α on x.
- Universal TM motivated the invention of general purpose electronic computers.

Universal Turing Machine

- Given (α, x) a **universal Turing machine** can simulate M_α on x.
- Universal TM motivated the invention of general purpose electronic computers.

Theorem: There exists a TM U such that $\forall x, \alpha \in \{0,1\}^*$, $U(x, \alpha) = M_\alpha(x)$.

Universal Turing Machine

- Given (α, x) a **universal Turing machine** can simulate M_α on x.

- Universal TM motivated the invention of general purpose electronic computers.

output of M_α on x

Theorem: There exists a TM U such that $\forall x, \alpha \in \{0,1\}^*$, $U(x, \alpha) = M_\alpha(x)$.

Universal Turing Machine

- Given (α, x) a **universal Turing machine** can simulate M_α on x.
- Universal TM motivated the invention of general purpose electronic computers.

output of M_α on x

Theorem: There exists a TM U such that $\forall x, \alpha \in \{0,1\}^*$, $U(x, \alpha) = M_\alpha(x)$. Moreover,

Universal Turing Machine

- Given (α, x) a **universal Turing machine** can simulate M_α on x.
- Universal TM motivated the invention of general purpose electronic computers.

output of M_α on x

Theorem: There exists a TM U such that $\forall x, \alpha \in \{0,1\}^*$, $U(x, \alpha) = M_\alpha(x)$. Moreover, if M_α halts on x within T steps,

Universal Turing Machine

- Given (α, x) a **universal Turing machine** can simulate M_α on x.
- Universal TM motivated the invention of general purpose electronic computers.

output of M_α on x

Theorem: There exists a TM U such that $\forall x, \alpha \in \{0,1\}^*$, $U(x, \alpha) = M_\alpha(x)$. Moreover, if M_α halts on x within T steps, then U on (α, x) halts in $CT \log T$ steps,

Universal Turing Machine

- Given (α, x) a **universal Turing machine** can simulate M_α on x.
- Universal TM motivated the invention of general purpose electronic computers.

output of M_α on x

Theorem: There exists a TM U such that $\forall x, \alpha \in \{0,1\}^*$, $U(x, \alpha) = M_\alpha(x)$. Moreover, if M_α halts on x within T steps, then U on (α, x) halts in $CT \log T$ steps, where C depends

Universal Turing Machine

- Given (α, x) a **universal Turing machine** can simulate M_α on x.
- Universal TM motivated the invention of general purpose electronic computers.

output of M_α on x

Theorem: There exists a TM U such that $\forall x, \alpha \in \{0,1\}^*$, $U(x, \alpha) = M_\alpha(x)$. Moreover, if M_α halts on x within T steps, then U on (α, x) halts in $CT \log T$ steps, where C depends only on $M'_\alpha s$ alphabet size, number of tapes, and number of states.

Universal Turing Machine

- Given (α, x) a **universal Turing machine** can simulate M_α on x.
- Universal TM motivated the invention of general purpose electronic computers.

output of M_α on x

Theorem: There exists a TM U such that $\forall x, \alpha \in \{0,1\}^*$, $U(x, \alpha) = M_\alpha(x)$. Moreover, if M_α halts on x within T steps, then U on (α, x) halts in $CT \log T$ steps, where C depends only on $M'_\alpha s$ alphabet size, number of tapes, and number of states.

Proof of easier version (CT^2 instead of $CT \log T$):

Universal Turing Machine

- Given (α, x) a **universal Turing machine** can simulate M_α on x.
- Universal TM motivated the invention of general purpose electronic computers.

output of M_α on x

Theorem: There exists a TM U such that $\forall x, \alpha \in \{0,1\}^*$, $U(x, \alpha) = M_\alpha(x)$. Moreover, if M_α halts on x within T steps, then U on (α, x) halts in $CT \log T$ steps, where C depends only on M_α's alphabet size, number of tapes, and number of states.

Proof of easier version (CT^2 instead of $CT \log T$): Next slide…

Universal Turing Machine

Universal Turing Machine

U

Universal Turing Machine

U

Universal Turing Machine

Input tape

U

Universal Turing Machine

U {

Input tape

Work tape 1

Universal Turing Machine

U {

Input tape

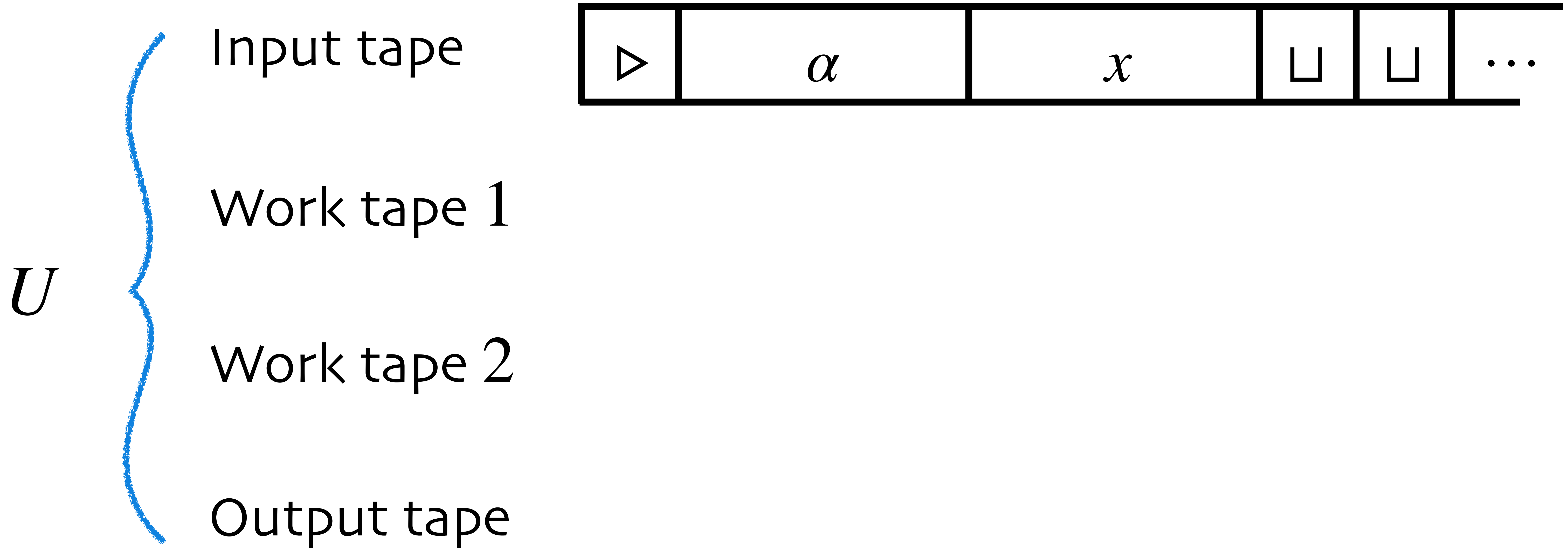
Work tape 1

Work tape 2

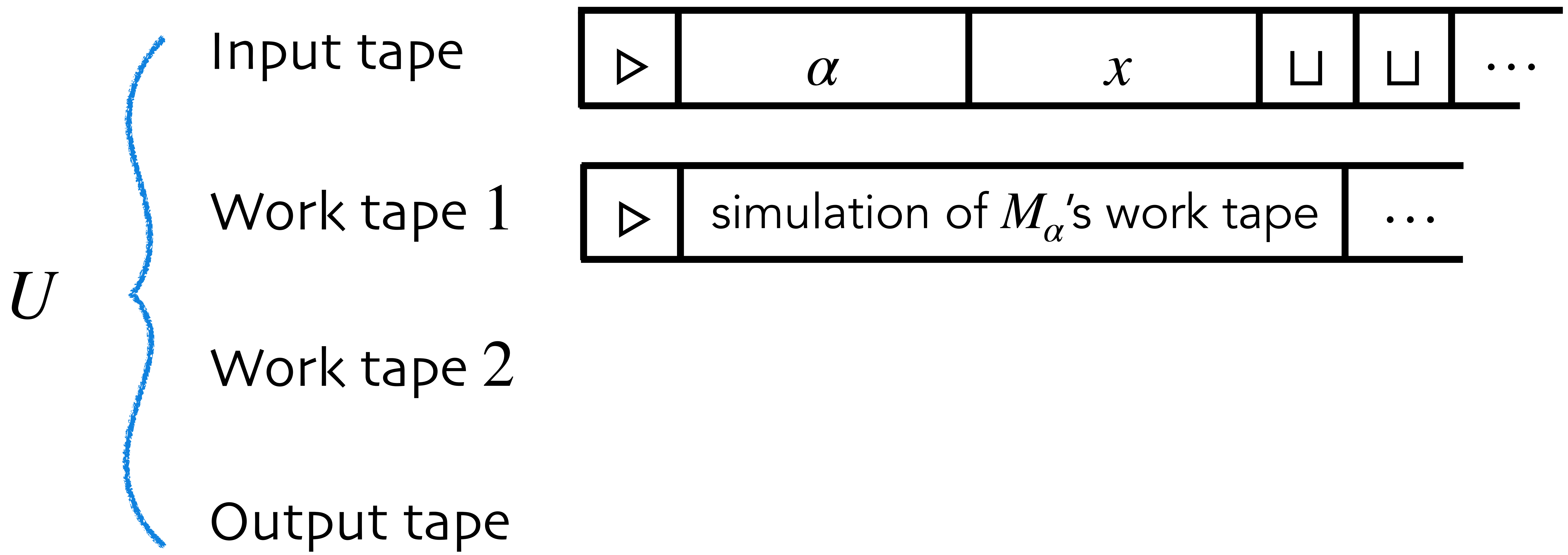
Universal Turing Machine

U {
- Input tape
- Work tape 1
- Work tape 2
- Output tape

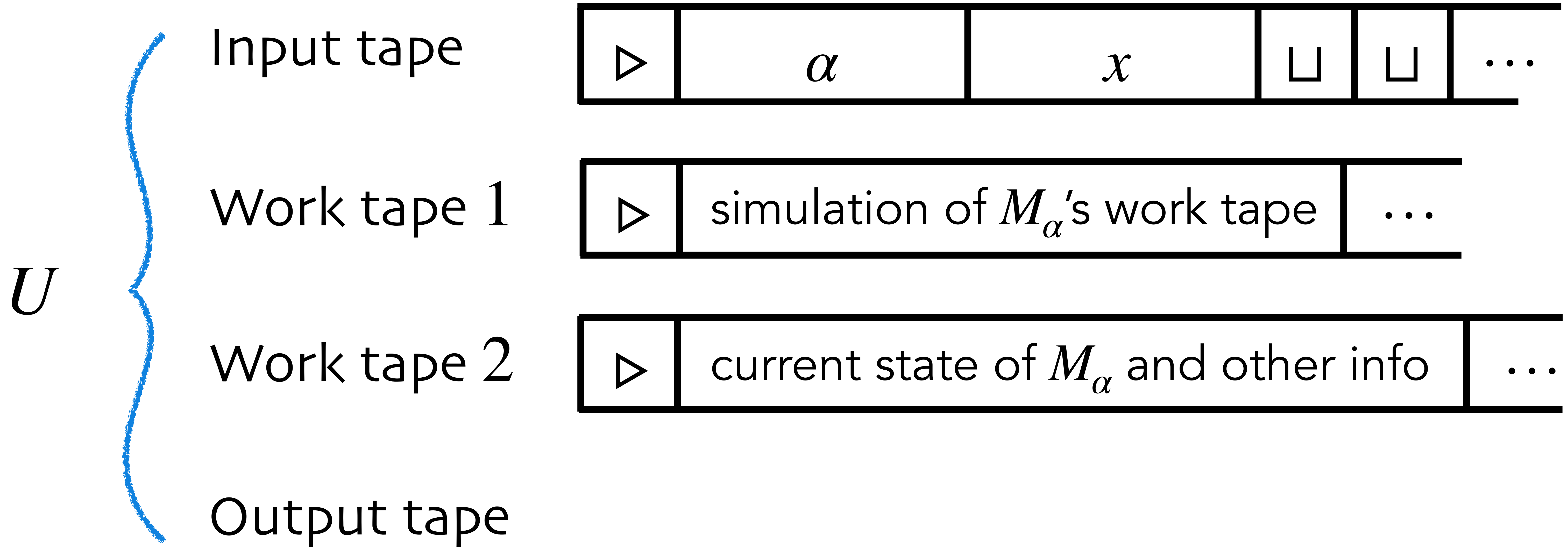
Universal Turing Machine



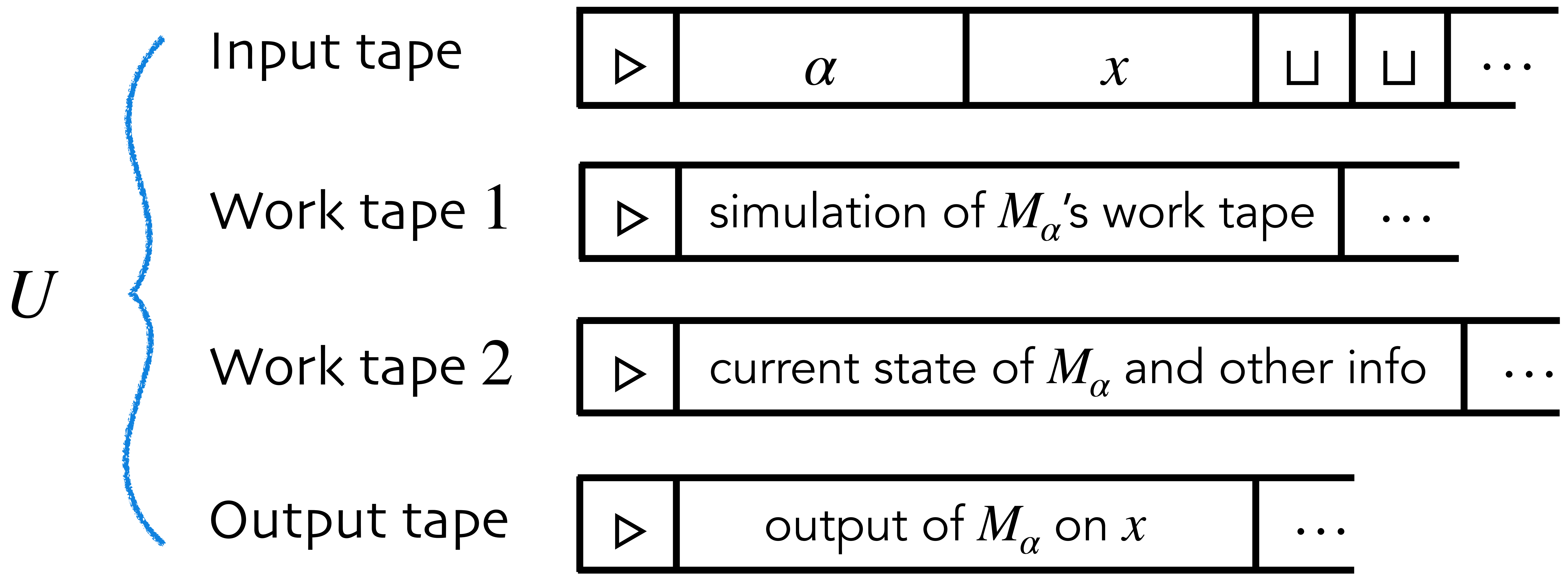
Universal Turing Machine



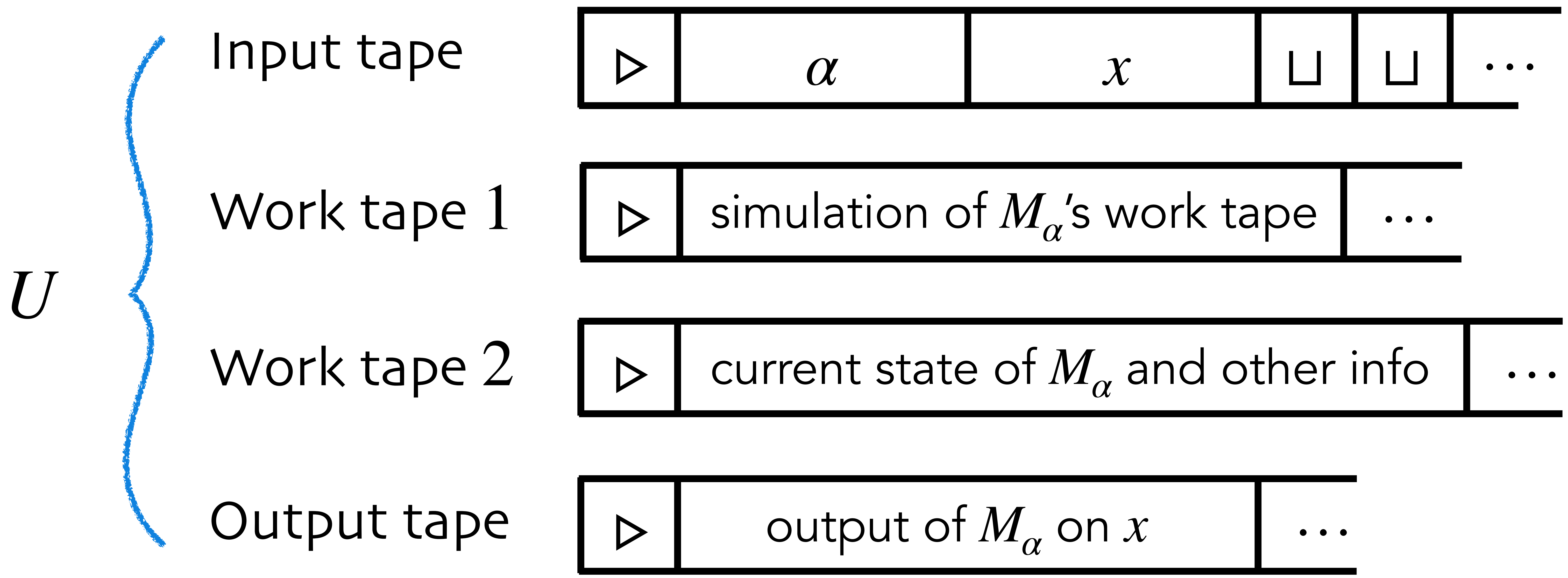
Universal Turing Machine



Universal Turing Machine

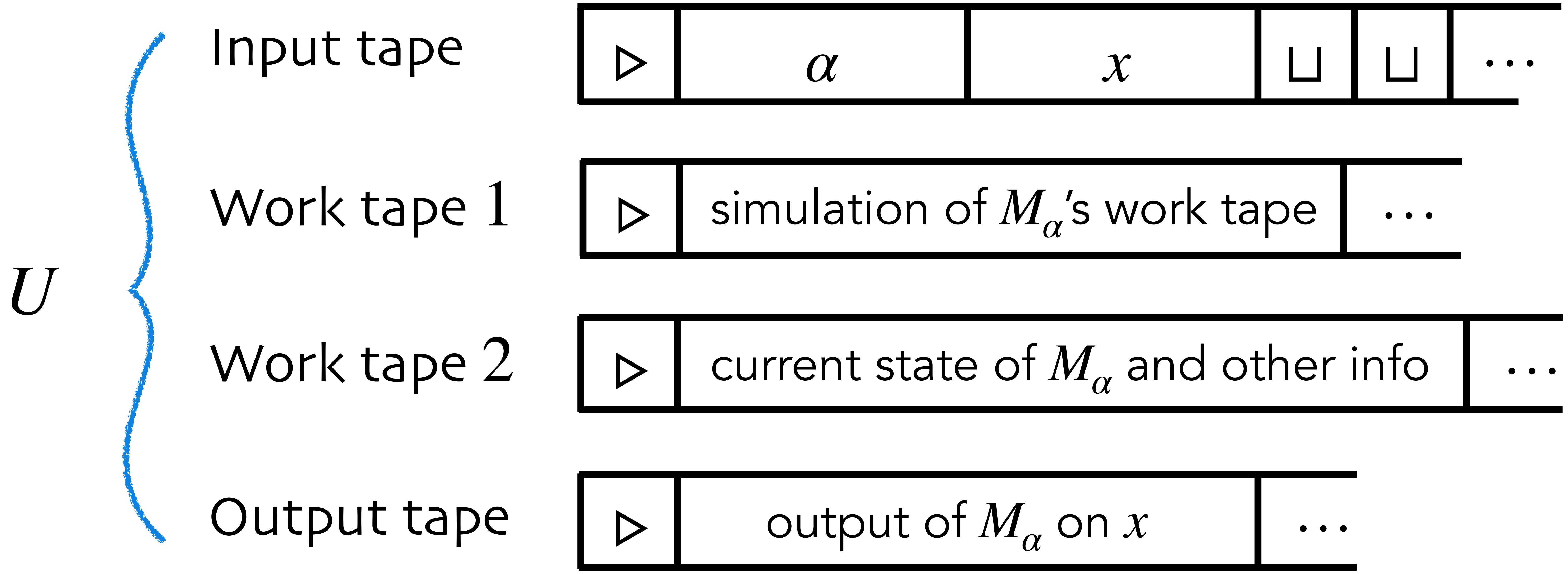


Universal Turing Machine



U's simulation of one step of M_α:

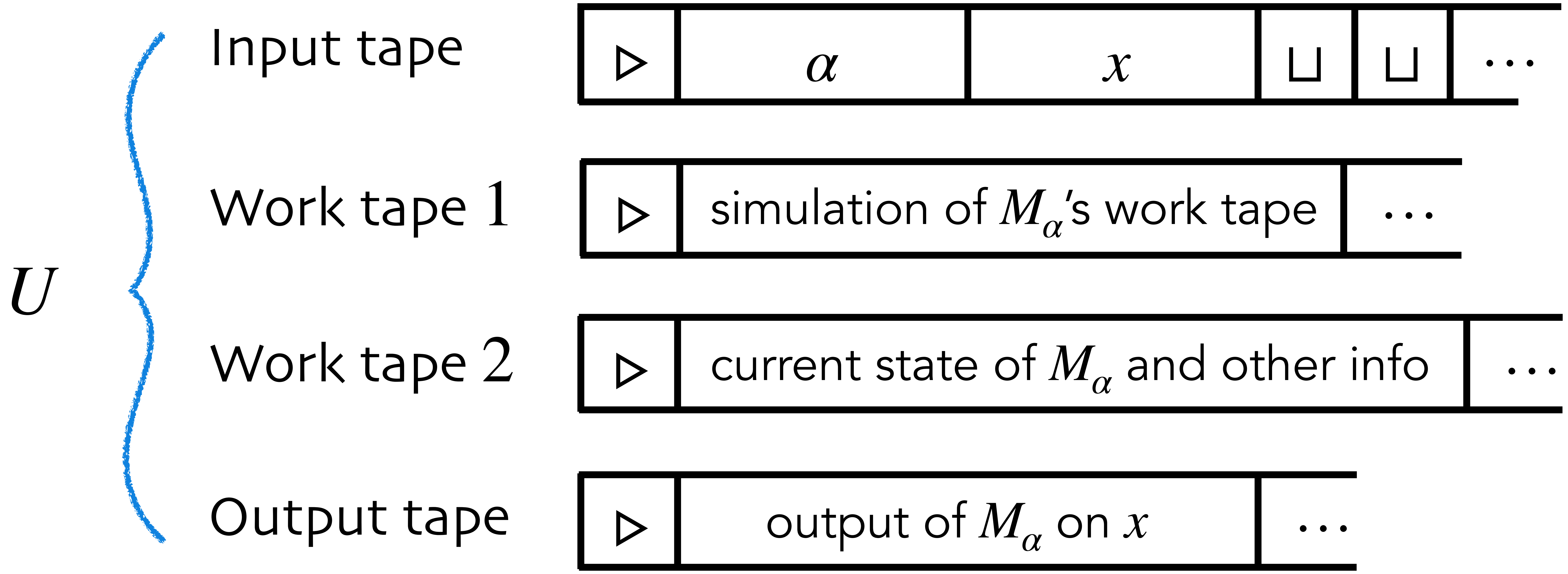
Universal Turing Machine



U's simulation of one step of M_α:

- U reads the current symbols from *WT* 1 and writes them on *WT* 2.

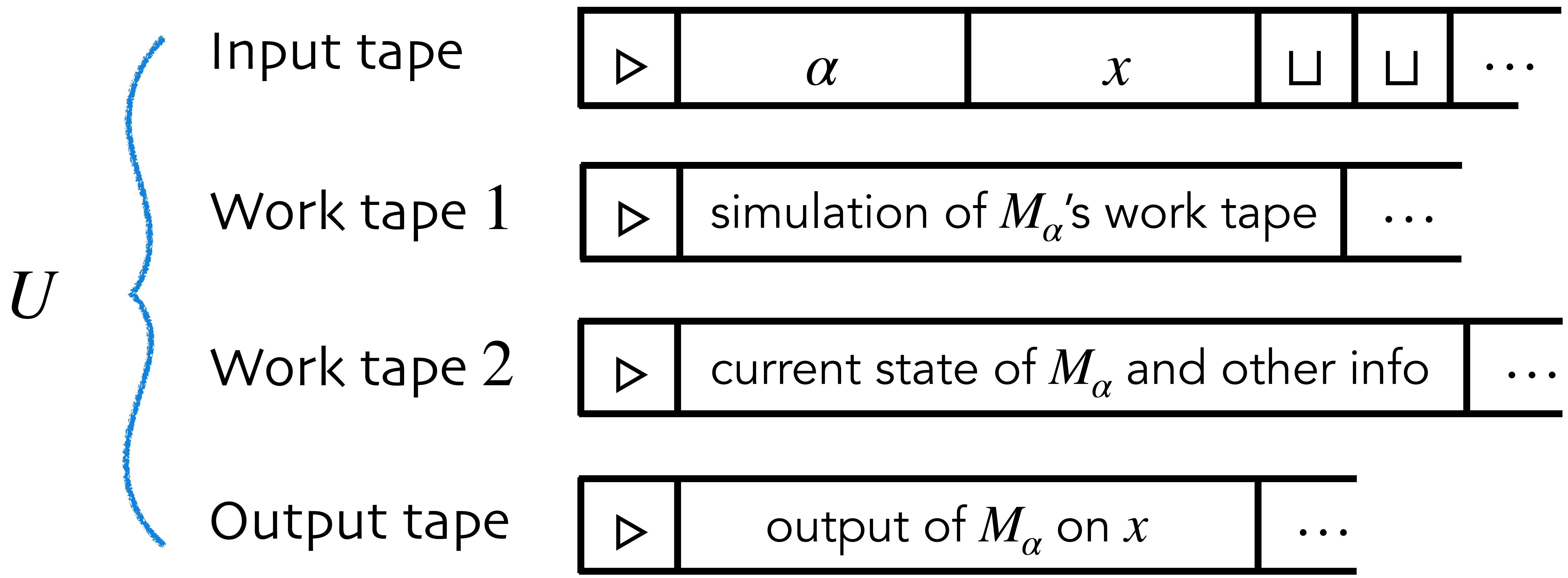
Universal Turing Machine



U's simulation of one step of M_α:

- U reads the current symbols from *WT* 1 and writes them on *WT* 2.
- U scans the *IT* to find the entry of δ that matches with the current state and symbols.

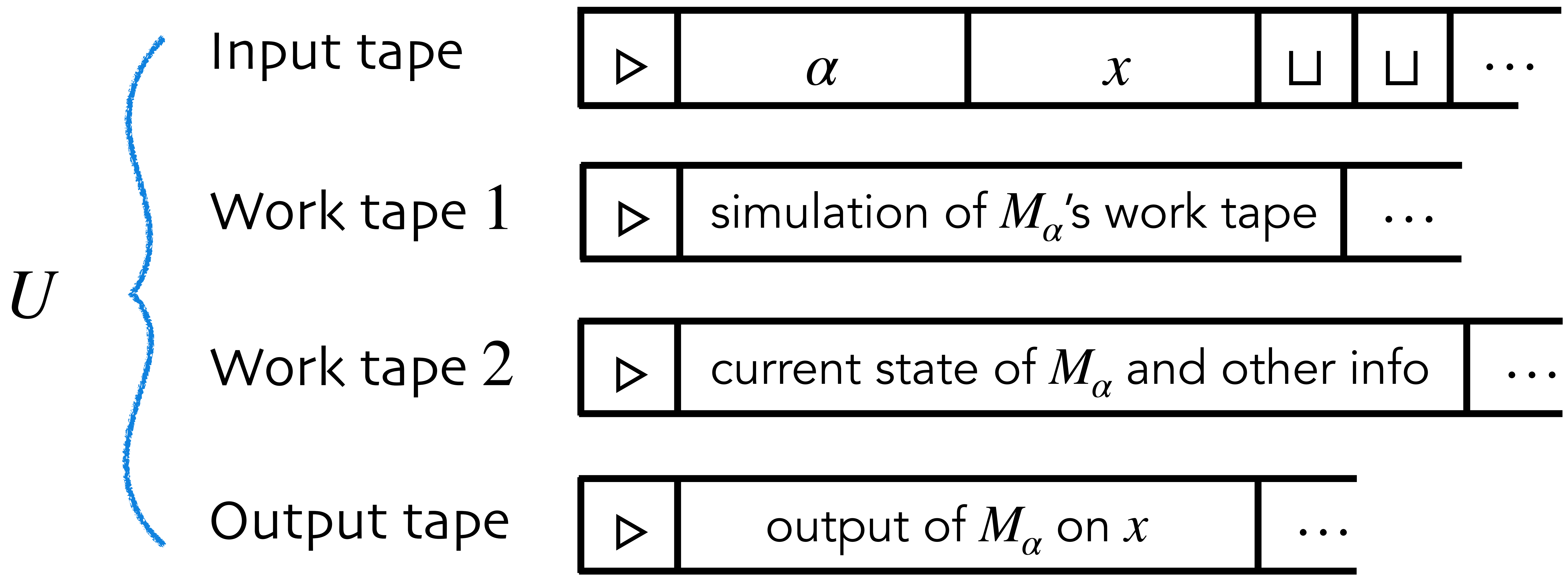
Universal Turing Machine



U's simulation of one step of M_α:

- U reads the current symbols from *WT* 1 and writes them on *WT* 2.
- U scans the *IT* to find the entry of δ that matches with the current state and symbols. and writes the next state, new symbols, head movements on *WT* 2.

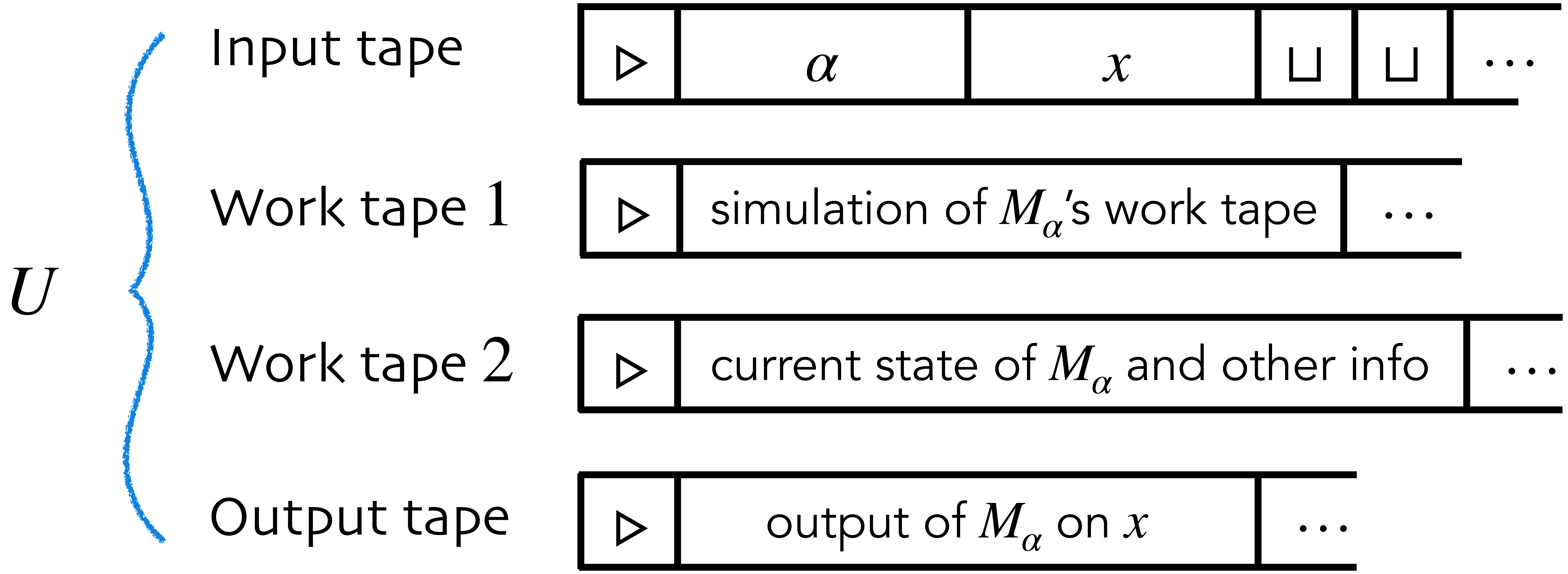
Universal Turing Machine



U's simulation of one step of M_α:

- U reads the current symbols from *WT* 1 and writes them on *WT* 2.
- U scans the *IT* to find the entry of δ that matches with the current state and symbols. and writes the next state, new symbols, head movements on *WT* 2.
- U changes the symbols on WT 1 and move tape heads (using ^).

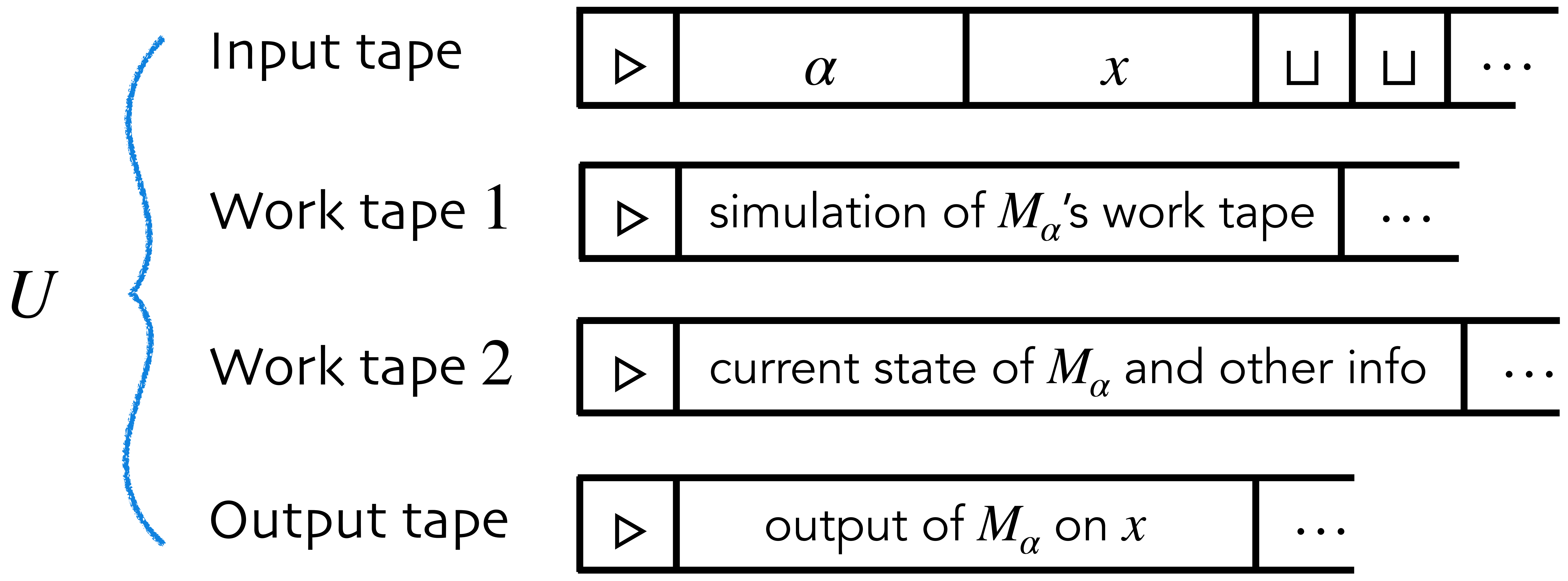
Universal Turing Machine



U's simulation of one step of M_α:

- U reads the current symbols from *WT* 1 and writes them on *WT* 2. $(O(\log|\Gamma|\,.\,k\,.\,T))$
- U scans the *IT* to find the entry of δ that matches with the current state and symbols. and writes the next state, new symbols, head movements on *WT* 2.
- U changes the symbols on $WT\ 1$ and move tape heads (using ^).

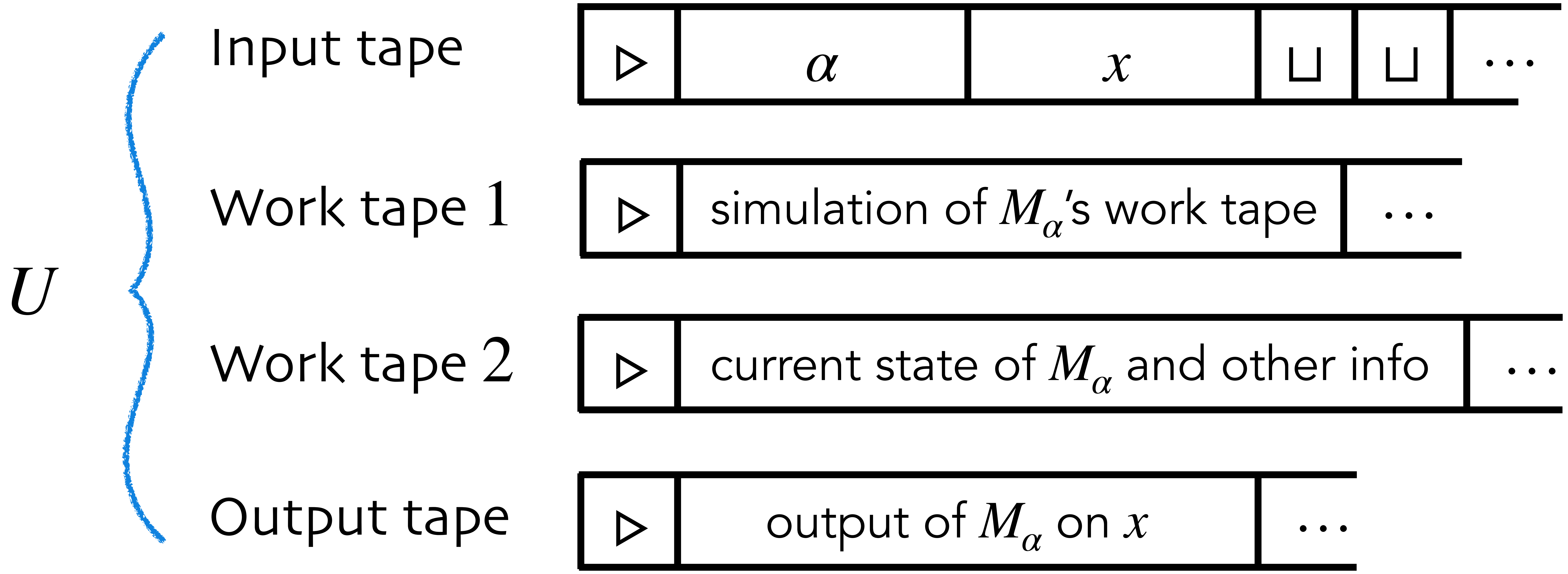
Universal Turing Machine



U's simulation of one step of M_α:

- U reads the current symbols from *WT* 1 and writes them on *WT* 2. $(O(\log|\Gamma|\,.\,k\,.\,T))$
- U scans the *IT* to find the entry of δ that matches with the current state and symbols. and writes the next state, new symbols, head movements on *WT* 2. $(O(C'))$
- U changes the symbols on WT 1 and move tape heads (using ^).

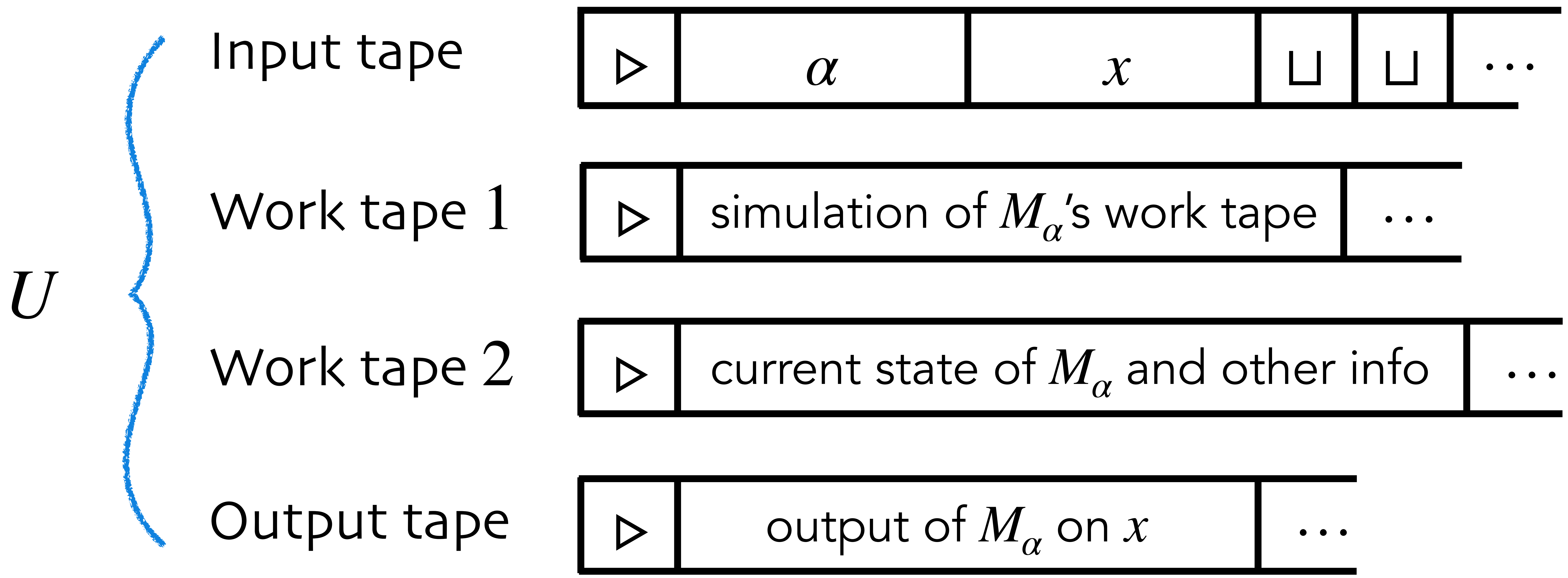
Universal Turing Machine



U's simulation of one step of M_α:

- U reads the current symbols from *WT* 1 and writes them on *WT* 2. $(O(\log|\Gamma| . k . T))$
- U scans the *IT* to find the entry of δ that matches with the current state and symbols. and writes the next state, new symbols, head movements on *WT* 2. $(O(C'))$
- U changes the symbols on WT 1 and move tape heads (using ^). $(O(\log|\Gamma| . k . T))$

Universal Turing Machine



U's simulation of one step of M_α:

- U reads the current symbols from *WT* 1 and writes them on *WT* 2. $(O(\log|\Gamma|\,.\,k\,.\,T))$
- U scans the *IT* to find the entry of δ that matches with the current state and symbols. and writes the next state, new symbols, head movements on *WT* 2. $(O(C'))$
- U changes the symbols on WT 1 and move tape heads (using ˆ). $(O(\log|\Gamma|\,.\,k\,.\,T))$

■