Algorithms and Computability

Lecture 8: The Church-Turing Thesis

Christian Schilling (christianms@cs.aau.dk)

slides courtesy of Martin Zimmermann

Last Lecture in Algorithms and Computability

We have seen

- Problems = Formal Languages
- Deterministic Turing machines (DTM) as an abstract model of computation
- The difference between computably-enumerable and computable languages:
  - L is computably-enumerable $\Leftrightarrow$ there exists a DTM M such that $L(M) = L$, i.e.,
    - $w \in L \Rightarrow M$ accepts w,
    - but $w \notin L \Rightarrow M$ rejects w or loops
  - L is computable $\Leftrightarrow$ there exists a halting DTM M such that $L(M) = L$, i.e.,
    - $w \in L \Rightarrow M$ accepts w and
    - $w \notin L \Rightarrow M$ rejects w

Conceptual View

A Turing Machine:

- An infinite tape of paper, divided into squares (often called cells)

Conceptual View

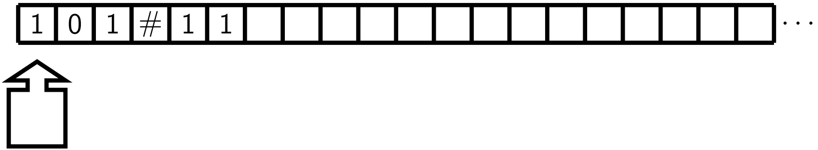
A Turing Machine:

1	0	1	#	1	1															···

- An infinite tape of paper, divided into squares (often called cells)
- Symbols in some squares

Conceptual View

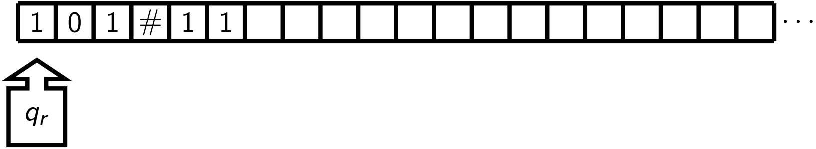
A Turing Machine:



- An infinite tape of paper, divided into squares (often called cells)
- Symbols in some squares
- A single square currently observed (with a reading/writing head)

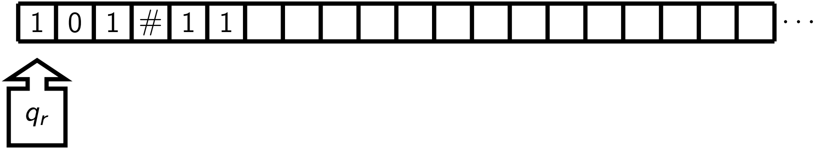
Conceptual View

A Turing Machine:



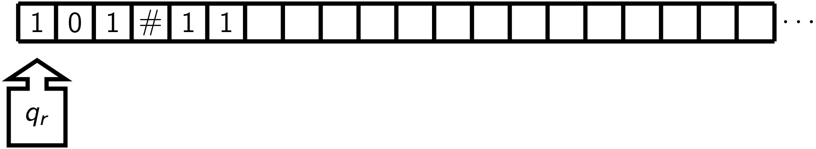
- An infinite tape of paper, divided into squares (often called cells)
- Symbols in some squares
- A single square currently observed (with a reading/writing head)
- A "state of mind"

A Turing Machine:



- An infinite tape of paper, divided into squares (often called cells)
- Symbols in some squares
- A single square currently observed (with a reading/writing head)
- A "state of mind"
- Rules updating the state and currently observed square

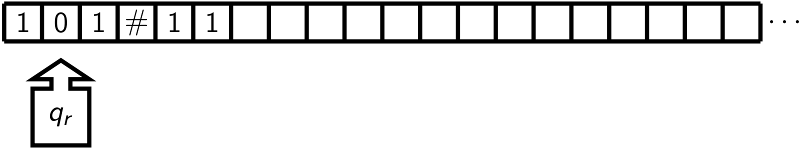
A Turing Machine:



- If state is q_r and symbol is 0 then change to state q_r, change symbol to 0, and move in direction 'right'
- If state is q_r and symbol is 1 then change to state q_r, change symbol to 1, and move in direction 'right'
- If state is q_r and symbol is # then change to state q_r, change symbol to #, and move in direction 'right'
- If state is q_r and symbol is 'empty' then change to state q_s, change symbol to 'empty', and move in direction 'left'

Conceptual View

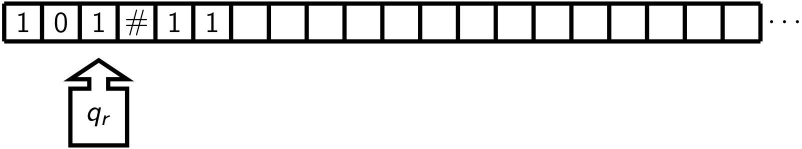
A Turing Machine:



- If state is q_r and symbol is 0 then change to state q_r, change symbol to 0, and move in direction 'right'
- If state is q_r and symbol is 1 then change to state q_r, change symbol to 1, and move in direction 'right'
- If state is q_r and symbol is # then change to state q_r, change symbol to #, and move in direction 'right'
- If state is q_r and symbol is 'empty' then change to state q_s, change symbol to 'empty', and move in direction 'left'

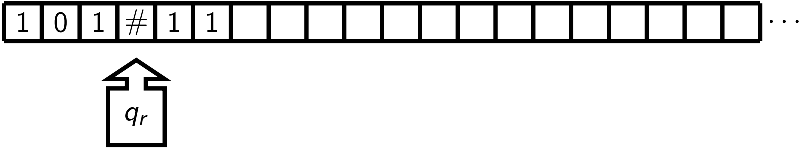
Conceptual View

A Turing Machine:



- If state is q_r and symbol is 0 then change to state q_r, change symbol to 0, and move in direction 'right'
- If state is q_r and symbol is 1 then change to state q_r, change symbol to 1, and move in direction 'right'
- If state is q_r and symbol is # then change to state q_r, change symbol to #, and move in direction 'right'
- If state is q_r and symbol is 'empty' then change to state q_s, change symbol to 'empty', and move in direction 'left'

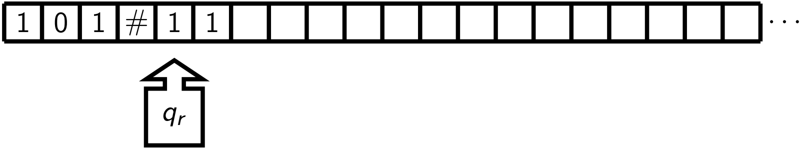
A Turing Machine:



- `If state is` q_r `and symbol is` 0 `then change to state` q_r, `change symbol to` 0, `and move in direction 'right'`
- `If state is` q_r `and symbol is` 1 `then change to state` q_r, `change symbol to` 1, `and move in direction 'right'`
- `If state is` q_r `and symbol is` # `then change to state` q_r, `change symbol to` #, `and move in direction 'right'`
- `If state is` q_r `and symbol is 'empty' then change to state` q_s, `change symbol to 'empty'`, `and move in direction 'left'`

Conceptual View

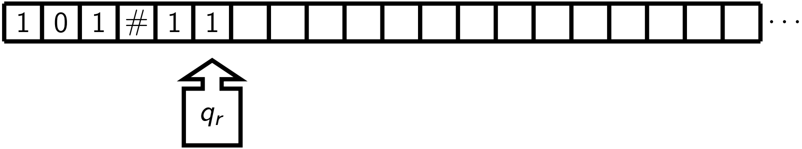
A Turing Machine:



- If state is q_r and symbol is 0 then change to state q_r, change symbol to 0, and move in direction 'right'
- If state is q_r and symbol is 1 then change to state q_r, change symbol to 1, and move in direction 'right'
- If state is q_r and symbol is # then change to state q_r, change symbol to #, and move in direction 'right'
- If state is q_r and symbol is 'empty' then change to state q_s, change symbol to 'empty', and move in direction 'left'

Conceptual View

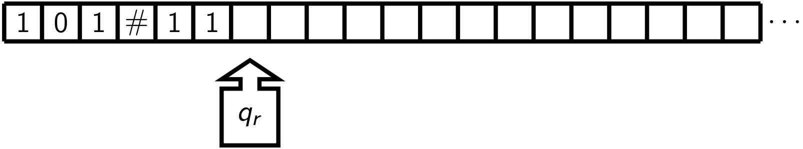
A Turing Machine:



- `If state is` q_r `and symbol is` 0 `then change to state` q_r, `change symbol to` 0, `and move in direction 'right'`
- `If state is` q_r `and symbol is` 1 `then change to state` q_r, `change symbol to` 1, `and move in direction 'right'`
- `If state is` q_r `and symbol is` # `then change to state` q_r, `change symbol to` #, `and move in direction 'right'`
- `If state is` q_r `and symbol is 'empty' then change to state` q_s, `change symbol to 'empty'`, `and move in direction 'left'`

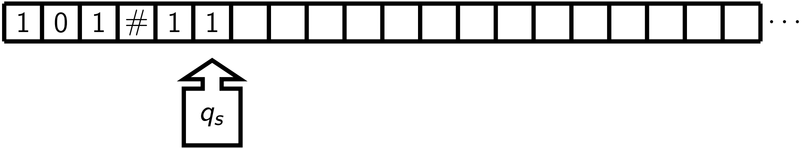
Conceptual View

A Turing Machine:



- If state is q_r and symbol is 0 then change to state q_r, change symbol to 0, and move in direction 'right'
- If state is q_r and symbol is 1 then change to state q_r, change symbol to 1, and move in direction 'right'
- If state is q_r and symbol is # then change to state q_r, change symbol to #, and move in direction 'right'
- If state is q_r and symbol is 'empty' then change to state q_s, change symbol to 'empty', and move in direction 'left'

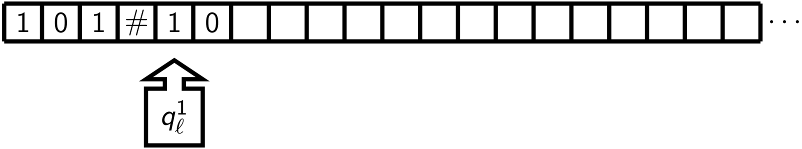
A Turing Machine:



- `If state is` q_s `and symbol is` 1 `then change to state` q_ℓ^1, `change symbol to` 0, `and move in direction 'left'`
- `If state is` q_ℓ^1 `and symbol is` 1 `then change to state` q_ℓ^1, `change symbol to` 1, `and move in direction 'left'`
- `If state is` q_ℓ^1 `and symbol is` $\#$ `then change to state` q_a^1, `change symbol to` $\#$, `and move in direction 'left'`
- `If state is` q_a^1 `and symbol is` 1 `then change to state` q_a^1, `change symbol to` 0, `and move in direction 'left'`

Conceptual View

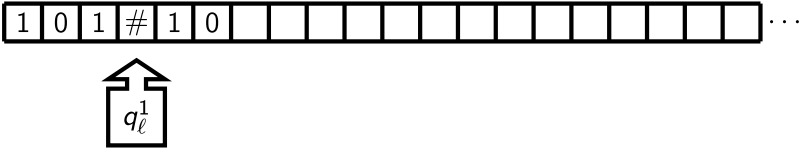
A Turing Machine:



- If state is q_s and symbol is 1 then change to state q_ℓ^1, change symbol to 0, and move in direction 'left'
- If state is q_ℓ^1 and symbol is 1 then change to state q_ℓ^1, change symbol to 1, and move in direction 'left'
- If state is q_ℓ^1 and symbol is # then change to state q_a^1, change symbol to #, and move in direction 'left'
- If state is q_a^1 and symbol is 1 then change to state q_a^1, change symbol to 0, and move in direction 'left'

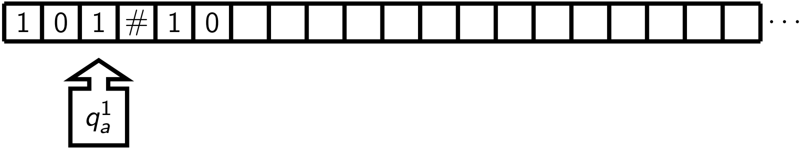
Conceptual View

A Turing Machine:



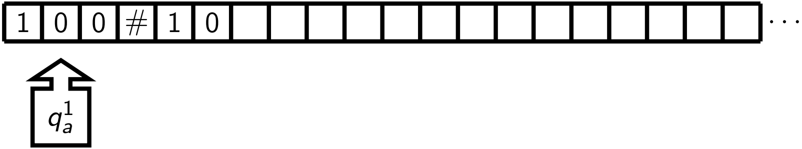
- If state is q_s and symbol is 1 then change to state q_ℓ^1, change symbol to 0, and move in direction 'left'
- If state is q_ℓ^1 and symbol is 1 then change to state q_ℓ^1, change symbol to 1, and move in direction 'left'
- If state is q_ℓ^1 and symbol is # then change to state q_a^1, change symbol to #, and move in direction 'left'
- If state is q_a^1 and symbol is 1 then change to state q_a^1, change symbol to 0, and move in direction 'left'

A Turing Machine:



- If state is q_s and symbol is 1 then change to state q_ℓ^1, change symbol to 0, and move in direction 'left'
- If state is q_ℓ^1 and symbol is 1 then change to state q_ℓ^1, change symbol to 1, and move in direction 'left'
- If state is q_ℓ^1 and symbol is # then change to state q_a^1, change symbol to #, and move in direction 'left'
- If state is q_a^1 and symbol is 1 then change to state q_a^1, change symbol to 0, and move in direction 'left'

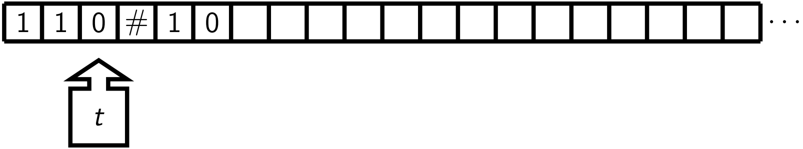
A Turing Machine:



- If state is q_a^1 and symbol is 0 then change to state t, change symbol to 1, and move in direction 'right'

Conceptual View

A Turing Machine:



- `If state is` q_a^1 `and symbol is` 0 `then change to state` t, `change symbol to` 1, `and move in direction 'right'`

Quiz 1

Suppose a DTM has a transition $\delta(q, a) = (q', b, d)$ and $q' \in \{t, r\}$, i.e., the transition leads to a halting configuration. Does it matter what b and d are?

Quiz 1

Suppose a DTM has a transition $\delta(q, a) = (q', b, d)$ and $q' \in \{t, r\}$, i.e., the transition leads to a halting configuration. Does it matter what b and d are?

No, the final content of the tape and position of the head are irrelevant

Exercise 5, Tutorial 1

Consider the language $L = \{w\#w \mid w \in \{0,1\}^*\}$

1. Give a halting DTM for L. Explain your solution in natural language
2. Give the accepting run on $\# \in L$
3. Give the accepting run on $011\#011 \in L$
4. Give the rejecting run on $01\#00 \notin L$

Intuition

$L = \{w\#w \mid w \in \{0,1\}^*\}$

1. If current symbol is 0 or 1: remember it as b, replace it by X
2. Go right to leftmost non-X symbol right of $\#$
3. If it is not b, reject
4. If it is b, replace it by X
5. Go left to the leftmost non-X symbol left of $\#$ (if there is none go to step 7)
6. Go to step 1
7. Check that there is no 0 or 1 left

Intuition

$L = \{w\#w \mid w \in \{0,1\}^*\}$

1. If current symbol is 0 or 1: remember it as b, replace it by X
   Use states s, q_0, q_1
2. Go right to leftmost non-X symbol right of $\#$
3. If it is not b, reject
4. If it is b, replace it by X
5. Go left to the leftmost non-X symbol left of $\#$ (if there is none go to step 7)
6. Go to step 1
7. Check that there is no 0 or 1 left

Intuition

$L = \{w\#w \mid w \in \{0,1\}^*\}$

1. If current symbol is 0 or 1: remember it as b, replace it by X
   Use states s, q_0, q_1
2. Go right to leftmost non-X symbol right of $\#$
   Use states q_0, q_1 and $q_0^\#$ $q_1^\#$ (after $\#$)
3. If it is not b, reject
4. If it is b, replace it by X
5. Go left to the leftmost non-X symbol left of $\#$ (if there is none go to step 7)
6. Go to step 1
7. Check that there is no 0 or 1 left

Intuition

$L = \{w\#w \mid w \in \{0,1\}^*\}$

1. If current symbol is 0 or 1: remember it as b, replace it by X
   Use states s, q_0, q_1
2. Go right to leftmost non-X symbol right of $\#$
   Use states q_0, q_1 and $q_0^\#$ $q_1^\#$ (after $\#$)
3. If it is not b, reject
4. If it is b, replace it by X
5. Go left to the leftmost non-X symbol left of $\#$ (if there is none go to step 7)
   Use states q_ℓ, $q_\ell^\#$
6. Go to step 1
7. Check that there is no 0 or 1 left

Intuition

$L = \{w\#w \mid w \in \{0,1\}^*\}$

1. If current symbol is 0 or 1: remember it as b, replace it by X
   Use states s, q_0, q_1
2. Go right to leftmost non-X symbol right of $\#$
   Use states q_0, q_1 and $q_0^\#$ $q_1^\#$ (after $\#$)
3. If it is not b, reject
4. If it is b, replace it by X
5. Go left to the leftmost non-X symbol left of $\#$ (if there is none go to step 7)
   Use states q_ℓ, $q_\ell^\#$
6. Go to step 1
7. Check that there is no 0 or 1 left
   Use state q_s

Full Solution

$(Q, \Sigma, \Gamma, s, t, r, \delta)$ with

- $Q = \{s, q_0, q_1, q_0^{\#}, q_1^{\#}, q_\ell, q_\ell^{\#}, q_s, t, r\}$,
- $\Sigma = \{0, 1, \#\}$,
- $\Gamma = \{0, 1, \#, X, \sqcup\}$,
- and the following δ:

Full Solution

$(Q, \Sigma, \Gamma, s, t, r, \delta)$ with

- $Q = \{s, q_0, q_1, q_0^{\#}, q_1^{\#}, q_\ell, q_\ell^{\#}, q_s, t, r\}$,
- $\Sigma = \{0, 1, \#\}$,
- $\Gamma = \{0, 1, \#, X, \llcorner\!\lrcorner\}$,
- and the following δ:

$$
\begin{aligned}
\delta(s, 0) &= (q_0, X, +1) && \texttt{//remember 0} \\
\delta(s, 1) &= (q_1, X, +1) && \texttt{//remember 1} \\
\delta(s, \#) &= (q_s, \#, +1) && \texttt{//check that no more 0/1 on tape} \\
\delta(s, \llcorner\!\lrcorner) &= (r, \llcorner\!\lrcorner, +1) && \texttt{//empty word} \\
\delta(s, X) &= (r, X, +1) && \texttt{//error}
\end{aligned}
$$

Full Solution

$(Q, \Sigma, \Gamma, s, t, r, \delta)$ with

- $Q = \{s, q_0, q_1, q_0^{\#}, q_1^{\#}, q_\ell, q_\ell^{\#}, q_s, t, r\}$,
- $\Sigma = \{0, 1, \#\}$,
- $\Gamma = \{0, 1, \#, X, \llcorner\!\lrcorner\}$,
- and the following δ:

for $b \in \{0, 1\}$:

$\delta(q_b, 0) = (q_b, 0, +1)$ //go right until #

$\delta(q_b, 1) = (q_b, 1, +1)$ //go right until #

$\delta(q_b, \#) = (q_b^{\#}, \#, +1)$ //reached #

$\delta(q_b, \llcorner\!\lrcorner) = (r, \llcorner\!\lrcorner, +1)$ //no # found

$\delta(q_b, X) = (r, X, +1)$ //error

Full Solution

$(Q, \Sigma, \Gamma, s, t, r, \delta)$ with

- $Q = \{s, q_0, q_1, q_0^{\#}, q_1^{\#}, q_\ell, q_\ell^{\#}, q_s, t, r\}$,
- $\Sigma = \{0, 1, \#\}$,
- $\Gamma = \{0, 1, \#, X, \text{␣}\}$,
- and the following δ:

for $b \in \{0, 1\}$:

$\delta(q_b^{\#}, X) = (q_b^{\#}, X, +1)$ `//go right until first 0/1`

$\delta(q_b^{\#}, b) = (q_\ell, X, -1)$ `//found b, go back left`

$\delta(q_b^{\#}, 1 - b) = (r, 1 - b, -1)$ `//wrong symbol found`

$\delta(q_b^{\#}, \text{␣}) = (r, \text{␣}, -1)$ `//no 0/1 found`

$\delta(q_b^{\#}, \#) = (r, X, +1)$ `//error`

Full Solution

$(Q, \Sigma, \Gamma, s, t, r, \delta)$ with

- $Q = \{s, q_0, q_1, q_0^{\#}, q_1^{\#}, q_\ell, q_\ell^{\#}, q_s, t, r\}$,
- $\Sigma = \{0, 1, \#\}$,
- $\Gamma = \{0, 1, \#, X, \text{␣}\}$,
- and the following δ:

$$
\begin{aligned}
\delta(q_\ell, X) &= (q_\ell, X, -1) && \texttt{//go left until } \# \\
\delta(q_\ell, \#) &= (q_\ell^{\#}, \#, -1) && \texttt{//reached } \# \\
\delta(q_\ell, 0) &= (r, 0, -1) && \texttt{//error} \\
\delta(q_\ell, 1) &= (r, 1, -1) && \texttt{//error} \\
\delta(q_\ell, \text{␣}) &= (r, \text{␣}, -1) && \texttt{//error}
\end{aligned}
$$

Full Solution

$(Q, \Sigma, \Gamma, s, t, r, \delta)$ with

- $Q = \{s, q_0, q_1, q_0^{\#}, q_1^{\#}, q_\ell, q_\ell^{\#}, q_s, t, r\}$,
- $\Sigma = \{0, 1, \#\}$,
- $\Gamma = \{0, 1, \#, X, \text{␣}\}$,
- and the following δ:

$\delta(q_\ell^{\#}, 0) = (q_\ell^{\#}, 0, -1)$ `//go left until first X`

$\delta(q_\ell^{\#}, 1) = (q_\ell^{\#}, 1, -1)$ `//go left until first X`

$\delta(q_\ell^{\#}, X) = (s, X, +1)$ `//found X, check next symbol`

$\delta(q_\ell^{\#}, \text{␣}) = (r, \text{␣}, -1)$ `//error`

$\delta(q_\ell^{\#}, \#) = (r, \#, -1)$ `//error`

Full Solution

$(Q, \Sigma, \Gamma, s, t, r, \delta)$ with

- $Q = \{s, q_0, q_1, q_0^{\#}, q_1^{\#}, q_\ell, q_\ell^{\#}, q_s, t, r\}$,
- $\Sigma = \{0, 1, \#\}$,
- $\Gamma = \{0, 1, \#, X, \llcorner\!\lrcorner\}$,
- and the following δ:

$$\begin{aligned}
\delta(q_s, 0) &= (r, 0, -1) && \texttt{//reject if still a 0 on tape} \\
\delta(q_s, 1) &= (r, 1, -1) && \texttt{//reject if still a 1 on tape} \\
\delta(q_s, X) &= (q_s, X, +1) && \texttt{//check next cell} \\
\delta(q_s, \#) &= (r, \#, +1) && \texttt{//error} \\
\delta(q_s, \llcorner\!\lrcorner) &= (t, \llcorner\!\lrcorner, -1) && \texttt{//no 0/1 found}
\end{aligned}$$

Example Runs

Note: To save some space, we underline the head position instead of specifying it explicitly in a configuration and drop the $\sqcup \ldots$ in the end of the tape content!

- M accepts $\# \in L$:

Example Runs

Note: To save some space, we underline the head position instead of specifying it explicitly in a configuration and drop the ␣ . . . in the end of the tape content!

- M accepts $\# \in L$:

$$[s, \underline{\#}]$$

Example Runs

Note: To save some space, we underline the head position instead of specifying it explicitly in a configuration and drop the ␣ . . . in the end of the tape content!

- M accepts $\# \in L$:

$$[s, \underline{\#}] \ \vdash_M \ [q_s, \#\underline{␣}]$$

Example Runs

Note: To save some space, we underline the head position instead of specifying it explicitly in a configuration and drop the $\sqcup \ldots$ in the end of the tape content!

- M accepts $\# \in L$:

$$[s, \underline{\#}] \vdash_M [q_s, \#\underline{\sqcup}] \vdash_M [t, \underline{\#}]$$

Example Runs

Note: To save some space, we underline the head position instead of specifying it explicitly in a configuration and drop the $\sqcup \ldots$ in the end of the tape content!

- M accepts $\# \in L$:

$$[s, \underline{\#}] \vdash_M [q_s, \#\underline{\sqcup}] \vdash_M [t, \underline{\#}]$$

- M rejects $01\#00$:

Example Runs

Note: To save some space, we underline the head position instead of specifying it explicitly in a configuration and drop the $\llcorner \ldots$ in the end of the tape content!

- M accepts $\# \in L$:

$$[s, \underline{\#}] \ \vdash_M \ [q_s, \#\underline{\llcorner}] \ \vdash_M \ [t, \underline{\#}]$$

- M rejects $01\#00$:

  $[s, \underline{0}1\#00]$

Example Runs

Note: To save some space, we underline the head position instead of specifying it explicitly in a configuration and drop the $\sqcup \ldots$ in the end of the tape content!

- M accepts $\# \in L$:

$$[s, \underline{\#}] \vdash_M [q_s, \#\underline{\sqcup}] \vdash_M [t, \underline{\#}]$$

- M rejects $01\#00$:

  $[s, \underline{0}1\#00] \vdash_M [q_0, X\underline{1}\#00]$

Example Runs

Note: To save some space, we underline the head position instead of specifying it explicitly in a configuration and drop the $\sqcup \ldots$ in the end of the tape content!

- M accepts $\# \in L$:

$$[s, \underline{\#}] \vdash_M [q_s, \#\underline{\sqcup}] \vdash_M [t, \underline{\#}]$$

- M rejects $01\#00$:

$$[s, \underline{0}1\#00] \vdash_M [q_0, X\underline{1}\#00] \vdash_M [q_0, X1\underline{\#}00]$$

Example Runs

Note: To save some space, we underline the head position instead of specifying it explicitly in a configuration and drop the $\sqcup \ldots$ in the end of the tape content!

- M accepts $\# \in L$:

$$[s, \underline{\#}] \vdash_M [q_s, \#\underline{\sqcup}] \vdash_M [t, \underline{\#}]$$

- M rejects $01\#00$:

$[s, \underline{0}1\#00] \vdash_M [q_0, X\underline{1}\#00] \vdash_M [q_0, X1\underline{\#}00] \vdash_M$

$[q_0^{\#}, X1\#\underline{0}0]$

Example Runs

Note: To save some space, we underline the head position instead of specifying it explicitly in a configuration and drop the $\llcorner \ldots$ in the end of the tape content!

- M accepts $\# \in L$:

$$[s, \underline{\#}] \ \vdash_M \ [q_s, \#\underline{\llcorner}] \ \vdash_M \ [t, \underline{\#}]$$

- M rejects $01\#00$:

$[s, \underline{0}1\#00] \ \vdash_M \ [q_0, X\underline{1}\#00] \ \vdash_M \ [q_0, X1\underline{\#}00] \ \vdash_M$

$[q_0^{\#}, X1\#\underline{0}0] \ \vdash_M \ [q_\ell, X1\underline{\#}X0]$

Example Runs

Note: To save some space, we underline the head position instead of specifying it explicitly in a configuration and drop the $\sqcup \ldots$ in the end of the tape content!

- M accepts $\# \in L$:

$$[s, \underline{\#}] \ \vdash_M \ [q_s, \#\underline{\sqcup}] \ \vdash_M \ [t, \underline{\#}]$$

- M rejects $01\#00$:

$$[s, \underline{0}1\#00] \ \vdash_M \ [q_0, X\underline{1}\#00] \ \vdash_M \ [q_0, X1\underline{\#}00] \ \vdash_M$$

$$[q_0^{\#}, X1\#\underline{0}0] \ \vdash_M \ [q_\ell, X1\underline{\#}X0] \ \vdash_M \ [q_\ell^{\#}, X\underline{1}\#X0]$$

Example Runs

Note: To save some space, we underline the head position instead of specifying it explicitly in a configuration and drop the $\sqcup \ldots$ in the end of the tape content!

- M accepts $\# \in L$:

$$[s, \underline{\#}] \vdash_M [q_s, \#\underline{\sqcup}] \vdash_M [t, \underline{\#}]$$

- M rejects $01\#00$:

$[s, \underline{0}1\#00] \vdash_M [q_0, X\underline{1}\#00] \vdash_M [q_0, X1\underline{\#}00] \vdash_M$

$[q_0^{\#}, X1\#\underline{0}0] \vdash_M [q_\ell, X1\underline{\#}X0] \vdash_M [q_\ell^{\#}, X\underline{1}\#X0] \vdash_M$

$[q_\ell^{\#}, \underline{X}1\#X0]$

Example Runs

Note: To save some space, we underline the head position instead of specifying it explicitly in a configuration and drop the $\llcorner\!\lrcorner \ldots$ in the end of the tape content!

- M accepts $\# \in L$:

$$[s, \underline{\#}] \vdash_M [q_s, \#\underline{\llcorner\!\lrcorner}] \vdash_M [t, \underline{\#}]$$

- M rejects $01\#00$:

$[s, \underline{0}1\#00] \vdash_M [q_0, X\underline{1}\#00] \vdash_M [q_0, X1\underline{\#}00] \vdash_M$

$[q_0^{\#}, X1\#\underline{0}0] \vdash_M [q_\ell, X1\underline{\#}X0] \vdash_M [q_\ell^{\#}, X\underline{1}\#X0] \vdash_M$

$[q_\ell^{\#}, \underline{X}1\#X0] \vdash_M [s, X\underline{1}\#X0]$

Example Runs

Note: To save some space, we underline the head position instead of specifying it explicitly in a configuration and drop the $\llcorner \ldots$ in the end of the tape content!

- M accepts $\# \in L$:

$$[s, \underline{\#}] \vdash_M [q_s, \#\underline{\llcorner}] \vdash_M [t, \underline{\#}]$$

- M rejects $01\#00$:

$$[s, \underline{0}1\#00] \vdash_M [q_0, X\underline{1}\#00] \vdash_M [q_0, X1\underline{\#}00] \vdash_M$$

$$[q_0^{\#}, X1\#\underline{0}0] \vdash_M [q_\ell, X1\underline{\#}X0] \vdash_M [q_\ell^{\#}, X\underline{1}\#X0] \vdash_M$$

$$[q_\ell^{\#}, \underline{X}1\#X0] \vdash_M [s, X\underline{1}\#X0] \vdash_M [q_1, XX\underline{\#}X0]$$

Example Runs

Note: To save some space, we underline the head position instead of specifying it explicitly in a configuration and drop the $\sqcup \ldots$ in the end of the tape content!

- M accepts $\# \in L$:

$$[s, \underline{\#}] \vdash_M [q_s, \#\underline{\sqcup}] \vdash_M [t, \underline{\#}]$$

- M rejects $01\#00$:

$$[s, \underline{0}1\#00] \vdash_M [q_0, X\underline{1}\#00] \vdash_M [q_0, X1\underline{\#}00] \vdash_M$$
$$[q_0^{\#}, X1\#\underline{0}0] \vdash_M [q_\ell, X1\underline{\#}X0] \vdash_M [q_\ell^{\#}, X\underline{1}\#X0] \vdash_M$$
$$[q_\ell^{\#}, \underline{X}1\#X0] \vdash_M [s, X\underline{1}\#X0] \vdash_M [q_1, XX\underline{\#}X0] \vdash_M$$
$$[q_1^{\#}, XX\#\underline{X}0]$$

Example Runs

Note: To save some space, we underline the head position instead of specifying it explicitly in a configuration and drop the $\sqcup \ldots$ in the end of the tape content!

- M accepts $\# \in L$:

$$[s, \underline{\#}] \vdash_M [q_s, \#\underline{\sqcup}] \vdash_M [t, \underline{\#}]$$

- M rejects $01\#00$:

$[s, \underline{0}1\#00] \vdash_M [q_0, X\underline{1}\#00] \vdash_M [q_0, X1\underline{\#}00] \vdash_M$

$[q_0^{\#}, X1\#\underline{0}0] \vdash_M [q_\ell, X1\underline{\#}X0] \vdash_M [q_\ell^{\#}, X\underline{1}\#X0] \vdash_M$

$[q_\ell^{\#}, \underline{X}1\#X0] \vdash_M [s, X\underline{1}\#X0] \vdash_M [q_1, XX\underline{\#}X0] \vdash_M$

$[q_1^{\#}, XX\#\underline{X}0] \vdash_M [q_1^{\#}, XX\#X\underline{0}]$

Example Runs

Note: To save some space, we underline the head position instead of specifying it explicitly in a configuration and drop the $\textvisiblespace \ldots$ in the end of the tape content!

- M accepts $\# \in L$:

$$[s, \underline{\#}] \vdash_M [q_s, \#\underline{\textvisiblespace}] \vdash_M [t, \underline{\#}]$$

- M rejects $01\#00$:

$$[s, \underline{0}1\#00] \vdash_M [q_0, X\underline{1}\#00] \vdash_M [q_0, X1\underline{\#}00] \vdash_M$$

$$[q_0^{\#}, X1\#\underline{0}0] \vdash_M [q_\ell, X1\underline{\#}X0] \vdash_M [q_\ell^{\#}, X\underline{1}\#X0] \vdash_M$$

$$[q_\ell^{\#}, \underline{X}1\#X0] \vdash_M [s, X\underline{1}\#X0] \vdash_M [q_1, XX\underline{\#}X0] \vdash_M$$

$$[q_1^{\#}, XX\#\underline{X}0] \vdash_M [q_1^{\#}, XX\#X\underline{0}] \vdash_M [r, XX\#\underline{X}0]$$

Another Run

- M accepts $011\#011 \in L$:

Another Run

- M accepts $011\#011 \in L$:

$[s, \underline{0}11\#011]$

Another Run

- M accepts $011\#011 \in L$:

$$[s, \underline{0}11\#011] \ \vdash_M \ [q_0, X\underline{1}1\#011]$$

Another Run

- M accepts $011\#011 \in L$:

$$[s, \underline{0}11\#011] \;\vdash_M\; [q_0, X\underline{1}1\#011] \;\vdash_M\; [q_0, X1\underline{1}\#011]$$

Another Run

- M accepts $011\#011 \in L$:

$$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011]$$

Another Run

- M accepts $011\#011 \in L$:

$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$

$[q_0^{\#}, X11\#\underline{0}11]$

Another Run

- M accepts $011\#011 \in L$:

$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$

$[q_0^{\#}, X11\#\underline{0}11] \vdash_M [q_\ell, X11\underline{\#}X11]$

Another Run

- M accepts $011\#011 \in L$:

$$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$$

$$[q_0^{\#}, X11\#\underline{0}11] \vdash_M [q_\ell, X11\underline{\#}X11] \vdash_M [q_\ell^{\#}, X1\underline{1}\#X11]$$

Another Run

- M accepts $011\#011 \in L$:

$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$

$[q_0^{\#}, X11\#\underline{0}11] \vdash_M [q_\ell, X11\underline{\#}X11] \vdash_M [q_\ell^{\#}, X1\underline{1}\#X11] \vdash_M [q_\ell^{\#}, X\underline{1}1\#X11]$

Another Run

- M accepts $011\#011 \in L$:

$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$

$[q_0^{\#}, X11\#\underline{0}11] \vdash_M [q_\ell, X11\underline{\#}X11] \vdash_M [q_\ell^{\#}, X1\underline{1}\#X11] \vdash_M [q_\ell^{\#}, X\underline{1}1\#X11] \vdash_M$

$[q_\ell^{\#}, \underline{X}11\#X11]$

Another Run

- M accepts $011\#011 \in L$:

$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$

$[q_0^{\#}, X11\#\underline{0}11] \vdash_M [q_\ell, X11\underline{\#}X11] \vdash_M [q_\ell^{\#}, X1\underline{1}\#X11] \vdash_M [q_\ell^{\#}, X\underline{1}1\#X11] \vdash_M$

$[q_\ell^{\#}, \underline{X}11\#X11] \vdash_M [s, X\underline{1}1\#X11]$

Another Run

- M accepts $011\#011 \in L$:

$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$
$[q_0^{\#}, X11\#\underline{0}11] \vdash_M [q_\ell, X11\underline{\#}X11] \vdash_M [q_\ell^{\#}, X1\underline{1}\#X11] \vdash_M [q_\ell^{\#}, X\underline{1}1\#X11] \vdash_M$
$[q_\ell^{\#}, \underline{X}11\#X11] \vdash_M [s, X\underline{1}1\#X11] \vdash_M [q_1, XX\underline{1}\#X11]$

Another Run

- M accepts $011\#011 \in L$:

$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$

$[q_0^{\#}, X11\#\underline{0}11] \vdash_M [q_\ell, X11\underline{\#}X11] \vdash_M [q_\ell^{\#}, X1\underline{1}\#X11] \vdash_M [q_\ell^{\#}, X\underline{1}1\#X11] \vdash_M$

$[q_\ell^{\#}, \underline{X}11\#X11] \vdash_M [s, X\underline{1}1\#X11] \vdash_M [q_1, XX\underline{1}\#X11] \vdash_M [q_1, XX1\underline{\#}X11]$

Another Run

- M accepts $011\#011 \in L$:

$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$
$[q_0^{\#}, X11\#\underline{0}11] \vdash_M [q_\ell, X11\underline{\#}X11] \vdash_M [q_\ell^{\#}, X1\underline{1}\#X11] \vdash_M [q_\ell^{\#}, X\underline{1}1\#X11] \vdash_M$
$[q_\ell^{\#}, \underline{X}11\#X11] \vdash_M [s, X\underline{1}1\#X11] \vdash_M [q_1, XX\underline{1}\#X11] \vdash_M [q_1, XX1\underline{\#}X11] \vdash_M$
$[q_1^{\#}, XX1\#\underline{X}11]$

Another Run

- M accepts $011\#011 \in L$:

$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$
$[q_0^{\#}, X11\#\underline{0}11] \vdash_M [q_\ell, X11\underline{\#}X11] \vdash_M [q_\ell^{\#}, X1\underline{1}\#X11] \vdash_M [q_\ell^{\#}, X\underline{1}1\#X11] \vdash_M$
$[q_\ell^{\#}, \underline{X}11\#X11] \vdash_M [s, X\underline{1}1\#X11] \vdash_M [q_1, XX\underline{1}\#X11] \vdash_M [q_1, XX1\underline{\#}X11] \vdash_M$
$[q_1^{\#}, XX1\#\underline{X}11] \vdash_M [q_1^{\#}, XX1\#X\underline{1}1]$

Another Run

- M accepts $011\#011 \in L$:

$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$

$[q_0^\#, X11\#\underline{0}11] \vdash_M [q_\ell, X11\underline{\#}X11] \vdash_M [q_\ell^\#, X1\underline{1}\#X11] \vdash_M [q_\ell^\#, X\underline{1}1\#X11] \vdash_M$

$[q_\ell^\#, \underline{X}11\#X11] \vdash_M [s, X\underline{1}1\#X11] \vdash_M [q_1, XX\underline{1}\#X11] \vdash_M [q_1, XX1\underline{\#}X11] \vdash_M$

$[q_1^\#, XX1\#\underline{X}11] \vdash_M [q_1^\#, XX1\#X\underline{1}1] \vdash_M [q_\ell, XX1\#\underline{X}X1]$

Another Run

- M accepts $011\#011 \in L$:

$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$
$[q_0^\#, X11\#\underline{0}11] \vdash_M [q_\ell, X11\underline{\#}X11] \vdash_M [q_\ell^\#, X1\underline{1}\#X11] \vdash_M [q_\ell^\#, X\underline{1}1\#X11] \vdash_M$
$[q_\ell^\#, \underline{X}11\#X11] \vdash_M [s, X\underline{1}1\#X11] \vdash_M [q_1, XX\underline{1}\#X11] \vdash_M [q_1, XX1\underline{\#}X11] \vdash_M$
$[q_1^\#, XX1\#\underline{X}11] \vdash_M [q_1^\#, XX1\#X\underline{1}1] \vdash_M [q_\ell, XX1\#\underline{X}X1] \vdash_M [q_\ell, XX1\underline{\#}XX1]$

Another Run

- M accepts $011\#011 \in L$:

$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$
$[q_0^\#, X11\#\underline{0}11] \vdash_M [q_\ell, X11\underline{\#}X11] \vdash_M [q_\ell^\#, X1\underline{1}\#X11] \vdash_M [q_\ell^\#, X\underline{1}1\#X11] \vdash_M$
$[q_\ell^\#, \underline{X}11\#X11] \vdash_M [s, X\underline{1}1\#X11] \vdash_M [q_1, XX\underline{1}\#X11] \vdash_M [q_1, XX1\underline{\#}X11] \vdash_M$
$[q_1^\#, XX1\#\underline{X}11] \vdash_M [q_1^\#, XX1\#X\underline{1}1] \vdash_M [q_\ell, XX1\#\underline{X}X1] \vdash_M [q_\ell, XX1\underline{\#}XX1] \vdash_M$
$[q_\ell^\#, XX\underline{1}\#XX1]$

Another Run

- M accepts $011\#011 \in L$:

$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$
$[q_0^\#, X11\#\underline{0}11] \vdash_M [q_\ell, X11\underline{\#}X11] \vdash_M [q_\ell^\#, X1\underline{1}\#X11] \vdash_M [q_\ell^\#, X\underline{1}1\#X11] \vdash_M$
$[q_\ell^\#, \underline{X}11\#X11] \vdash_M [s, X\underline{1}1\#X11] \vdash_M [q_1, XX\underline{1}\#X11] \vdash_M [q_1, XX1\underline{\#}X11] \vdash_M$
$[q_1^\#, XX1\#\underline{X}11] \vdash_M [q_1^\#, XX1\#X\underline{1}1] \vdash_M [q_\ell, XX1\#\underline{X}X1] \vdash_M [q_\ell, XX1\underline{\#}XX1] \vdash_M$
$[q_\ell^\#, XX\underline{1}\#XX1] \vdash_M [q_\ell^\#, X\underline{X}1\#XX1]$

Another Run

- M accepts $011\#011 \in L$:

$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$
$[q_0^\#, X11\#\underline{0}11] \vdash_M [q_\ell, X11\underline{\#}X11] \vdash_M [q_\ell^\#, X1\underline{1}\#X11] \vdash_M [q_\ell^\#, X\underline{1}1\#X11] \vdash_M$
$[q_\ell^\#, \underline{X}11\#X11] \vdash_M [s, X\underline{1}1\#X11] \vdash_M [q_1, XX\underline{1}\#X11] \vdash_M [q_1, XX1\underline{\#}X11] \vdash_M$
$[q_1^\#, XX1\#\underline{X}11] \vdash_M [q_1^\#, XX1\#X\underline{1}1] \vdash_M [q_\ell, XX1\#\underline{X}X1] \vdash_M [q_\ell, XX1\underline{\#}XX1] \vdash_M$
$[q_\ell^\#, XX\underline{1}\#XX1] \vdash_M [q_\ell^\#, X\underline{X}1\#XX1] \vdash_M [s, XX\underline{1}\#XX1]$

Another Run

- M accepts $011\#011 \in L$:

$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$
$[q_0^\#, X11\#\underline{0}11] \vdash_M [q_\ell, X11\underline{\#}X11] \vdash_M [q_\ell^\#, X1\underline{1}\#X11] \vdash_M [q_\ell^\#, X\underline{1}1\#X11] \vdash_M$
$[q_\ell^\#, \underline{X}11\#X11] \vdash_M [s, X\underline{1}1\#X11] \vdash_M [q_1, XX\underline{1}\#X11] \vdash_M [q_1, XX1\underline{\#}X11] \vdash_M$
$[q_1^\#, XX1\#\underline{X}11] \vdash_M [q_1^\#, XX1\#X\underline{1}1] \vdash_M [q_\ell, XX1\#\underline{X}X1] \vdash_M [q_\ell, XX1\underline{\#}XX1] \vdash_M$
$[q_\ell^\#, XX\underline{1}\#XX1] \vdash_M [q_\ell^\#, X\underline{X}1\#XX1] \vdash_M [s, XX\underline{1}\#XX1] \vdash_M [q_1, XXX\underline{\#}XX1]$

Another Run

- M accepts $011\#011 \in L$:

$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$
$[q_0^{\#}, X11\#\underline{0}11] \vdash_M [q_\ell, X11\underline{\#}X11] \vdash_M [q_\ell^{\#}, X1\underline{1}\#X11] \vdash_M [q_\ell^{\#}, X\underline{1}1\#X11] \vdash_M$
$[q_\ell^{\#}, \underline{X}11\#X11] \vdash_M [s, X\underline{1}1\#X11] \vdash_M [q_1, XX\underline{1}\#X11] \vdash_M [q_1, XX1\underline{\#}X11] \vdash_M$
$[q_1^{\#}, XX1\#\underline{X}11] \vdash_M [q_1^{\#}, XX1\#X\underline{1}1] \vdash_M [q_\ell, XX1\#\underline{X}X1] \vdash_M [q_\ell, XX1\underline{\#}XX1] \vdash_M$
$[q_\ell^{\#}, XX\underline{1}\#XX1] \vdash_M [q_\ell^{\#}, X\underline{X}1\#XX1] \vdash_M [s, XX\underline{1}\#XX1] \vdash_M [q_1, XXX\underline{\#}XX1] \vdash_M$
$[q_1^{\#}, XXX\#\underline{X}X1]$

Another Run

- M accepts $011\#011 \in L$:

$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$
$[q_0^\#, X11\#\underline{0}11] \vdash_M [q_\ell, X11\underline{\#}X11] \vdash_M [q_\ell^\#, X1\underline{1}\#X11] \vdash_M [q_\ell^\#, X\underline{1}1\#X11] \vdash_M$
$[q_\ell^\#, \underline{X}11\#X11] \vdash_M [s, X\underline{1}1\#X11] \vdash_M [q_1, XX\underline{1}\#X11] \vdash_M [q_1, XX1\underline{\#}X11] \vdash_M$
$[q_1^\#, XX1\#\underline{X}11] \vdash_M [q_1^\#, XX1\#X\underline{1}1] \vdash_M [q_\ell, XX1\#\underline{X}X1] \vdash_M [q_\ell, XX1\underline{\#}XX1] \vdash_M$
$[q_\ell^\#, XX\underline{1}\#XX1] \vdash_M [q_\ell^\#, X\underline{X}1\#XX1] \vdash_M [s, XX\underline{1}\#XX1] \vdash_M [q_1, XXX\underline{\#}XX1] \vdash_M$
$[q_1^\#, XXX\#\underline{X}X1] \vdash_M [q_1^\#, XXX\#X\underline{X}1]$

Another Run

- M accepts $011\#011 \in L$:

$$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$$
$$[q_0^\#, X11\#\underline{0}11] \vdash_M [q_\ell, X11\underline{\#}X11] \vdash_M [q_\ell^\#, X1\underline{1}\#X11] \vdash_M [q_\ell^\#, X\underline{1}1\#X11] \vdash_M$$
$$[q_\ell^\#, \underline{X}11\#X11] \vdash_M [s, X\underline{1}1\#X11] \vdash_M [q_1, XX\underline{1}\#X11] \vdash_M [q_1, XX1\underline{\#}X11] \vdash_M$$
$$[q_1^\#, XX1\#\underline{X}11] \vdash_M [q_1^\#, XX1\#X\underline{1}1] \vdash_M [q_\ell, XX1\#\underline{X}X1] \vdash_M [q_\ell, XX1\underline{\#}XX1] \vdash_M$$
$$[q_\ell^\#, XX\underline{1}\#XX1] \vdash_M [q_\ell^\#, X\underline{X}1\#XX1] \vdash_M [s, XX\underline{1}\#XX1] \vdash_M [q_1, XXX\underline{\#}XX1] \vdash_M$$
$$[q_1^\#, XXX\#\underline{X}X1] \vdash_M [q_1^\#, XXX\#X\underline{X}1] \vdash_M [q_1^\#, XXX\#XX\underline{1}]$$

Another Run

- M accepts $011\#011 \in L$:

$$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$$

$$[q_0^\#, X11\#\underline{0}11] \vdash_M [q_\ell, X11\underline{\#}X11] \vdash_M [q_\ell^\#, X1\underline{1}\#X11] \vdash_M [q_\ell^\#, X\underline{1}1\#X11] \vdash_M$$

$$[q_\ell^\#, \underline{X}11\#X11] \vdash_M [s, X\underline{1}1\#X11] \vdash_M [q_1, XX\underline{1}\#X11] \vdash_M [q_1, XX1\underline{\#}X11] \vdash_M$$

$$[q_1^\#, XX1\#\underline{X}11] \vdash_M [q_1^\#, XX1\#X\underline{1}1] \vdash_M [q_\ell, XX1\#\underline{X}X1] \vdash_M [q_\ell, XX1\underline{\#}XX1] \vdash_M$$

$$[q_\ell^\#, XX\underline{1}\#XX1] \vdash_M [q_\ell^\#, X\underline{X}1\#XX1] \vdash_M [s, XX\underline{1}\#XX1] \vdash_M [q_1, XXX\underline{\#}XX1] \vdash_M$$

$$[q_1^\#, XXX\#\underline{X}X1] \vdash_M [q_1^\#, XXX\#X\underline{X}1] \vdash_M [q_1^\#, XXX\#XX\underline{1}] \vdash_M [q_\ell, XXX\#X\underline{X}X]$$

Another Run

- M accepts $011\#011 \in L$:

$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$
$[q_0^\#, X11\#\underline{0}11] \vdash_M [q_\ell, X11\underline{\#}X11] \vdash_M [q_\ell^\#, X1\underline{1}\#X11] \vdash_M [q_\ell^\#, X\underline{1}1\#X11] \vdash_M$
$[q_\ell^\#, \underline{X}11\#X11] \vdash_M [s, X\underline{1}1\#X11] \vdash_M [q_1, XX\underline{1}\#X11] \vdash_M [q_1, XX1\underline{\#}X11] \vdash_M$
$[q_1^\#, XX1\#\underline{X}11] \vdash_M [q_1^\#, XX1\#X\underline{1}1] \vdash_M [q_\ell, XX1\#\underline{X}X1] \vdash_M [q_\ell, XX1\underline{\#}XX1] \vdash_M$
$[q_\ell^\#, XX\underline{1}\#XX1] \vdash_M [q_\ell^\#, X\underline{X}1\#XX1] \vdash_M [s, XX\underline{1}\#XX1] \vdash_M [q_1, XXX\underline{\#}XX1] \vdash_M$
$[q_1^\#, XXX\#\underline{X}X1] \vdash_M [q_1^\#, XXX\#X\underline{X}1] \vdash_M [q_1^\#, XXX\#XX\underline{1}] \vdash_M [q_\ell, XXX\#X\underline{X}X] \vdash_M$
$[q_\ell, XXX\#\underline{X}XX]$

Another Run

- M accepts $011\#011 \in L$:

$$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$$
$$[q_0^\#, X11\#\underline{0}11] \vdash_M [q_\ell, X11\underline{\#}X11] \vdash_M [q_\ell^\#, X1\underline{1}\#X11] \vdash_M [q_\ell^\#, X\underline{1}1\#X11] \vdash_M$$
$$[q_\ell^\#, \underline{X}11\#X11] \vdash_M [s, X\underline{1}1\#X11] \vdash_M [q_1, XX\underline{1}\#X11] \vdash_M [q_1, XX1\underline{\#}X11] \vdash_M$$
$$[q_1^\#, XX1\#\underline{X}11] \vdash_M [q_1^\#, XX1\#X\underline{1}1] \vdash_M [q_\ell, XX1\#\underline{X}X1] \vdash_M [q_\ell, XX1\underline{\#}XX1] \vdash_M$$
$$[q_\ell^\#, XX\underline{1}\#XX1] \vdash_M [q_\ell^\#, X\underline{X}1\#XX1] \vdash_M [s, XX\underline{1}\#XX1] \vdash_M [q_1, XXX\underline{\#}XX1] \vdash_M$$
$$[q_1^\#, XXX\#\underline{X}X1] \vdash_M [q_1^\#, XXX\#X\underline{X}1] \vdash_M [q_1^\#, XXX\#XX\underline{1}] \vdash_M [q_\ell, XXX\#X\underline{X}X] \vdash_M$$
$$[q_\ell, XXX\#\underline{X}XX] \vdash_M [q_\ell, XXX\underline{\#}XXX]$$

Another Run

- M accepts $011\#011 \in L$:

$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$
$[q_0^\#, X11\#\underline{0}11] \vdash_M [q_\ell, X11\underline{\#}X11] \vdash_M [q_\ell^\#, X1\underline{1}\#X11] \vdash_M [q_\ell^\#, X\underline{1}1\#X11] \vdash_M$
$[q_\ell^\#, \underline{X}11\#X11] \vdash_M [s, X\underline{1}1\#X11] \vdash_M [q_1, XX\underline{1}\#X11] \vdash_M [q_1, XX1\underline{\#}X11] \vdash_M$
$[q_1^\#, XX1\#\underline{X}11] \vdash_M [q_1^\#, XX1\#X\underline{1}1] \vdash_M [q_\ell, XX1\#\underline{X}X1] \vdash_M [q_\ell, XX1\underline{\#}XX1] \vdash_M$
$[q_\ell^\#, XX\underline{1}\#XX1] \vdash_M [q_\ell^\#, X\underline{X}1\#XX1] \vdash_M [s, XX\underline{1}\#XX1] \vdash_M [q_1, XXX\underline{\#}XX1] \vdash_M$
$[q_1^\#, XXX\#\underline{X}X1] \vdash_M [q_1^\#, XXX\#X\underline{X}1] \vdash_M [q_1^\#, XXX\#XX\underline{1}] \vdash_M [q_\ell, XXX\#X\underline{X}X] \vdash_M$
$[q_\ell, XXX\#\underline{X}XX] \vdash_M [q_\ell, XXX\underline{\#}XXX] \vdash_M [q_\ell^\#, XX\underline{X}\#XXX]$

Another Run

- M accepts $011\#011 \in L$:

$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$
$[q_0^\#, X11\#\underline{0}11] \vdash_M [q_\ell, X11\underline{\#}X11] \vdash_M [q_\ell^\#, X1\underline{1}\#X11] \vdash_M [q_\ell^\#, X\underline{1}1\#X11] \vdash_M$
$[q_\ell^\#, \underline{X}11\#X11] \vdash_M [s, X\underline{1}1\#X11] \vdash_M [q_1, XX\underline{1}\#X11] \vdash_M [q_1, XX1\underline{\#}X11] \vdash_M$
$[q_1^\#, XX1\#\underline{X}11] \vdash_M [q_1^\#, XX1\#X\underline{1}1] \vdash_M [q_\ell, XX1\#\underline{X}X1] \vdash_M [q_\ell, XX1\underline{\#}XX1] \vdash_M$
$[q_\ell^\#, XX\underline{1}\#XX1] \vdash_M [q_\ell^\#, X\underline{X}1\#XX1] \vdash_M [s, XX\underline{1}\#XX1] \vdash_M [q_1, XXX\underline{\#}XX1] \vdash_M$
$[q_1^\#, XXX\#\underline{X}X1] \vdash_M [q_1^\#, XXX\#X\underline{X}1] \vdash_M [q_1^\#, XXX\#XX\underline{1}] \vdash_M [q_\ell, XXX\#X\underline{X}X] \vdash_M$
$[q_\ell, XXX\#\underline{X}XX] \vdash_M [q_\ell, XXX\underline{\#}XXX] \vdash_M [q_\ell^\#, XX\underline{X}\#XXX] \vdash_M [s, XXX\underline{\#}XXX]$

Another Run

- M accepts $011\#011 \in L$:

$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$
$[q_0^\#, X11\#\underline{0}11] \vdash_M [q_\ell, X11\underline{\#}X11] \vdash_M [q_\ell^\#, X1\underline{1}\#X11] \vdash_M [q_\ell^\#, X\underline{1}1\#X11] \vdash_M$
$[q_\ell^\#, \underline{X}11\#X11] \vdash_M [s, X\underline{1}1\#X11] \vdash_M [q_1, XX\underline{1}\#X11] \vdash_M [q_1, XX1\underline{\#}X11] \vdash_M$
$[q_1^\#, XX1\#\underline{X}11] \vdash_M [q_1^\#, XX1\#X\underline{1}1] \vdash_M [q_\ell, XX1\#\underline{X}X1] \vdash_M [q_\ell, XX1\underline{\#}XX1] \vdash_M$
$[q_\ell^\#, XX\underline{1}\#XX1] \vdash_M [q_\ell^\#, X\underline{X}1\#XX1] \vdash_M [s, XX\underline{1}\#XX1] \vdash_M [q_1, XXX\underline{\#}XX1] \vdash_M$
$[q_1^\#, XXX\#\underline{X}X1] \vdash_M [q_1^\#, XXX\#X\underline{X}1] \vdash_M [q_1^\#, XXX\#XX\underline{1}] \vdash_M [q_\ell, XXX\#X\underline{X}X] \vdash_M$
$[q_\ell, XXX\#\underline{X}XX] \vdash_M [q_\ell, XXX\underline{\#}XXX] \vdash_M [q_\ell^\#, XX\underline{X}\#XXX] \vdash_M [s, XXX\underline{\#}XXX] \vdash_M$
$[q_s, XXX\#\underline{X}XX]$

Another Run

- M accepts $011\#011 \in L$:

$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$
$[q_0^\#, X11\#\underline{0}11] \vdash_M [q_\ell, X11\underline{\#}X11] \vdash_M [q_\ell^\#, X1\underline{1}\#X11] \vdash_M [q_\ell^\#, X\underline{1}1\#X11] \vdash_M$
$[q_\ell^\#, \underline{X}11\#X11] \vdash_M [s, X\underline{1}1\#X11] \vdash_M [q_1, XX\underline{1}\#X11] \vdash_M [q_1, XX1\underline{\#}X11] \vdash_M$
$[q_1^\#, XX1\#\underline{X}11] \vdash_M [q_1^\#, XX1\#X\underline{1}1] \vdash_M [q_\ell, XX1\#\underline{X}X1] \vdash_M [q_\ell, XX1\underline{\#}XX1] \vdash_M$
$[q_\ell^\#, XX\underline{1}\#XX1] \vdash_M [q_\ell^\#, X\underline{X}1\#XX1] \vdash_M [s, XX\underline{1}\#XX1] \vdash_M [q_1, XXX\underline{\#}XX1] \vdash_M$
$[q_1^\#, XXX\#\underline{X}X1] \vdash_M [q_1^\#, XXX\#X\underline{X}1] \vdash_M [q_1^\#, XXX\#XX\underline{1}] \vdash_M [q_\ell, XXX\#X\underline{X}X] \vdash_M$
$[q_\ell, XXX\#\underline{X}XX] \vdash_M [q_\ell, XXX\underline{\#}XXX] \vdash_M [q_\ell^\#, XX\underline{X}\#XXX] \vdash_M [s, XXX\underline{\#}XXX] \vdash_M$
$[q_s, XXX\#\underline{X}XX] \vdash_M [q_s, XXX\#X\underline{X}X]$

Another Run

- M accepts $011\#011 \in L$:

$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$
$[q_0^\#, X11\#\underline{0}11] \vdash_M [q_\ell, X11\underline{\#}X11] \vdash_M [q_\ell^\#, X1\underline{1}\#X11] \vdash_M [q_\ell^\#, X\underline{1}1\#X11] \vdash_M$
$[q_\ell^\#, \underline{X}11\#X11] \vdash_M [s, X\underline{1}1\#X11] \vdash_M [q_1, XX\underline{1}\#X11] \vdash_M [q_1, XX1\underline{\#}X11] \vdash_M$
$[q_1^\#, XX1\#\underline{X}11] \vdash_M [q_1^\#, XX1\#X\underline{1}1] \vdash_M [q_\ell, XX1\#\underline{X}X1] \vdash_M [q_\ell, XX1\underline{\#}XX1] \vdash_M$
$[q_\ell^\#, XX\underline{1}\#XX1] \vdash_M [q_\ell^\#, X\underline{X}1\#XX1] \vdash_M [s, XX\underline{1}\#XX1] \vdash_M [q_1, XXX\underline{\#}XX1] \vdash_M$
$[q_1^\#, XXX\#\underline{X}X1] \vdash_M [q_1^\#, XXX\#X\underline{X}1] \vdash_M [q_1^\#, XXX\#XX\underline{1}] \vdash_M [q_\ell, XXX\#X\underline{X}X] \vdash_M$
$[q_\ell, XXX\#\underline{X}XX] \vdash_M [q_\ell, XXX\underline{\#}XXX] \vdash_M [q_\ell^\#, XX\underline{X}\#XXX] \vdash_M [s, XXX\underline{\#}XXX] \vdash_M$
$[q_s, XXX\#\underline{X}XX] \vdash_M [q_s, XXX\#X\underline{X}X] \vdash_M [q_s, XXX\#XX\underline{X}]$

Another Run

- M accepts $011\#011 \in L$:

$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$
$[q_0^\#, X11\#\underline{0}11] \vdash_M [q_\ell, X11\underline{\#}X11] \vdash_M [q_\ell^\#, X1\underline{1}\#X11] \vdash_M [q_\ell^\#, X\underline{1}1\#X11] \vdash_M$
$[q_\ell^\#, \underline{X}11\#X11] \vdash_M [s, X\underline{1}1\#X11] \vdash_M [q_1, XX\underline{1}\#X11] \vdash_M [q_1, XX1\underline{\#}X11] \vdash_M$
$[q_1^\#, XX1\#\underline{X}11] \vdash_M [q_1^\#, XX1\#X\underline{1}1] \vdash_M [q_\ell, XX1\#\underline{X}X1] \vdash_M [q_\ell, XX1\underline{\#}XX1] \vdash_M$
$[q_\ell^\#, XX\underline{1}\#XX1] \vdash_M [q_\ell^\#, X\underline{X}1\#XX1] \vdash_M [s, XX\underline{1}\#XX1] \vdash_M [q_1, XXX\underline{\#}XX1] \vdash_M$
$[q_1^\#, XXX\#\underline{X}X1] \vdash_M [q_1^\#, XXX\#X\underline{X}1] \vdash_M [q_1^\#, XXX\#XX\underline{1}] \vdash_M [q_\ell, XXX\#X\underline{X}X] \vdash_M$
$[q_\ell, XXX\#\underline{X}XX] \vdash_M [q_\ell, XXX\underline{\#}XXX] \vdash_M [q_\ell^\#, XX\underline{X}\#XXX] \vdash_M [s, XXX\underline{\#}XXX] \vdash_M$
$[q_s, XXX\#\underline{X}XX] \vdash_M [q_s, XXX\#X\underline{X}X] \vdash_M [q_s, XXX\#XX\underline{X}] \vdash_M [q_s, XXX\#XXX\underline{\sqcup}]$

Another Run

- M accepts $011\#011 \in L$:

$[s, \underline{0}11\#011] \vdash_M [q_0, X\underline{1}1\#011] \vdash_M [q_0, X1\underline{1}\#011] \vdash_M [q_0, X11\underline{\#}011] \vdash_M$
$[q_0^\#, X11\#\underline{0}11] \vdash_M [q_\ell, X11\underline{\#}X11] \vdash_M [q_\ell^\#, X1\underline{1}\#X11] \vdash_M [q_\ell^\#, X\underline{1}1\#X11] \vdash_M$
$[q_\ell^\#, \underline{X}11\#X11] \vdash_M [s, X\underline{1}1\#X11] \vdash_M [q_1, XX\underline{1}\#X11] \vdash_M [q_1, XX1\underline{\#}X11] \vdash_M$
$[q_1^\#, XX1\#\underline{X}11] \vdash_M [q_1^\#, XX1\#X\underline{1}1] \vdash_M [q_\ell, XX1\#\underline{X}X1] \vdash_M [q_\ell, XX1\underline{\#}XX1] \vdash_M$
$[q_\ell^\#, XX\underline{1}\#XX1] \vdash_M [q_\ell^\#, X\underline{X}1\#XX1] \vdash_M [s, XX\underline{1}\#XX1] \vdash_M [q_1, XXX\underline{\#}XX1] \vdash_M$
$[q_1^\#, XXX\#\underline{X}X1] \vdash_M [q_1^\#, XXX\#X\underline{X}1] \vdash_M [q_1^\#, XXX\#XX\underline{1}] \vdash_M [q_\ell, XXX\#X\underline{X}X] \vdash_M$
$[q_\ell, XXX\#\underline{X}XX] \vdash_M [q_\ell, XXX\underline{\#}XXX] \vdash_M [q_\ell^\#, XX\underline{X}\#XXX] \vdash_M [s, XXX\underline{\#}XXX] \vdash_M$
$[q_s, XXX\#\underline{X}XX] \vdash_M [q_s, XXX\#X\underline{X}X] \vdash_M [q_s, XXX\#XX\underline{X}] \vdash_M [q_s, XXX\#XXX\underline{\sqcup}] \vdash_M$
$[t, XXX\#XX\underline{X}]$

Today

Turing machines define the limits of computation:

Claim: Everything that can be computed can be computed by a Turing machine

Today

Turing machines define the limits of computation:

> *Claim: Everything that can be computed can be computed by a Turing machine*

How can we be so sure? What about

- parallel computing?
- quantum computing?
- neural networks?
- some technology we have not invented yet?

Agenda

1. **The Church-Turing Thesis**
2. Multi-tape Turing Machines
3. Nondeterministic Turing Machines

A Bit of History

The "Entscheidungsproblem" (Hilbert and Ackermann, 1928)

> *Is there an algorithm that, given a statement in some logical language (typically predicate logic), answers "Yes" or "No" according to whether the statement is universally valid?*

A Bit of History

The "Entscheidungsproblem" (Hilbert and Ackermann, 1928)

> *Is there an algorithm that, given a statement in some logical language (typically predicate logic), answers "Yes" or "No" according to whether the statement is universally valid?*

In our view: Is $\{w \mid w \text{ encodes a valid statement}\}$ computable?

A Bit of History

The "Entscheidungsproblem" (Hilbert and Ackermann, 1928)

> *Is there an algorithm that, given a statement in some logical language (typically predicate logic), answers "Yes" or "No" according to whether the statement is universally valid?*

In our view: Is $\{w \mid w \text{ encodes a valid statement}\}$ computable?

Requires the formalization of "algorithms"

Independent proposals:

- **Gödel, 1933**: μ-recursive functions
- **Church, 1936**: λ-calculus
- **Turing, 1936**: Turing machines

A Bit of History

The "Entscheidungsproblem" (Hilbert and Ackermann, 1928)

> *Is there an algorithm that, given a statement in some logical language (typically predicate logic), answers "Yes" or "No" according to whether the statement is universally valid?*

In our view: Is $\{w \mid w \text{ encodes a valid statement}\}$ computable?

Requires the formalization of "algorithms"

Independent proposals:

- **Gödel, 1933**: μ-recursive functions
- **Church, 1936**: λ-calculus
- **Turing, 1936**: Turing machines

Theorem (Church, Kleene, Turing)

All three formalizations compute the same functions (and therefore can deal with the same languages/problems)

Church-Turing Thesis

This equivalence led mathematicians to believe that the intuitive notion of algorithm is precisely captured by any of these three formalizations

Church-Turing Thesis

This equivalence led mathematicians to believe that the intuitive notion of algorithm is precisely captured by any of these three formalizations

The Church-Turing Thesis

Everything that can be computed
can be computed by a Turing machine

Church-Turing Thesis

This equivalence led mathematicians to believe that the intuitive notion of algorithm is precisely captured by any of these three formalizations

The Church-Turing Thesis

Everything that can be computed
can be computed by a Turing machine

- Many other formalizations have been proposed, all equivalent to Turing machines
- But there are "nonphysical" models that are stronger: Zeno machines (infinite computations in finite time), time-travelling Turing machines, etc.
- It is a thesis, **not** a definition and **not** a theorem, and may be refuted in the future

Turing-completeness

Definition

A formalization of computation is Turing-complete if it can simulate every Turing machine

In that way, a Turing-complete formalism can compute everything Turing machines can compute

Turing-completeness

Definition

A formalization of computation is Turing-complete if it can simulate every Turing machine

In that way, a Turing-complete formalism can compute everything Turing machines can compute

Examples

- λ-calculus, μ-recursive functions,
- Java, Python, and other programming languages (assuming the computer has infinite memory),
- Excel, PowerPoint, LaTeX, etc.,
- Game of Life, Minecraft, Magic: The Gathering,
- and many other formalisms

Robustness

We (informally) say that a formalization of computation is robust if no reasonable extension increases its power

Robustness

We (informally) say that a formalization of computation is robust if no reasonable extension increases its power

Turing machines are very robust. We will study three extensions (but there are many more!) and show that they are not more powerful than Turing machines

Robustness

We (informally) say that a formalization of computation is robust if no reasonable extension increases its power

Turing machines are very robust. We will study three extensions (but there are many more!) and show that they are not more powerful than Turing machines

To prove this, we simulate extended TMs by DTMs

Exercise 1, Tutorial 2

We want to add another "direction" for the reading head of a Turing machine, namely "0" for "stay." Thus, a transition of the form $\delta(q, a) = (q', b, 0)$ updates the state to q' and changes the symbol at the current cell to b but does not move the head

Show that every DTM with the "stay" direction can be simulated by a standard DTM (i.e., without the "stay" direction)

We want to add another "direction" for the reading head of a Turing machine, namely "0" for "stay." Thus, a transition of the form $\delta(q, a) = (q', b, 0)$ updates the state to q' and changes the symbol at the current cell to b but does not move the head

Show that every DTM with the "stay" direction can be simulated by a standard DTM (i.e., without the "stay" direction)

Note

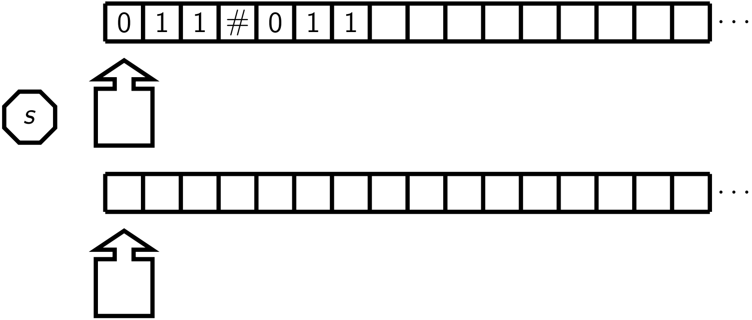
We will use the direction 0 from now on in our Turing machines (whenever convenient)

Agenda

1. The Church-Turing Thesis
2. **Multi-tape Turing Machines**
3. Nondeterministic Turing Machines

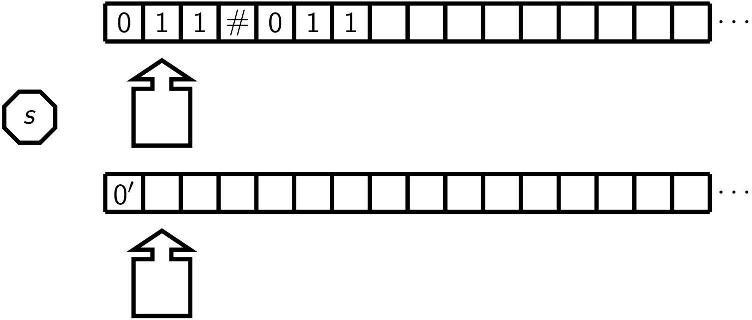
Conceptual View

A 2-tape DTM for $\{w\#w \mid w \in \{0,1\}^*\}$:



- Copy from first to second tape until first $\#$ (additionally marking the first cell by a prime)

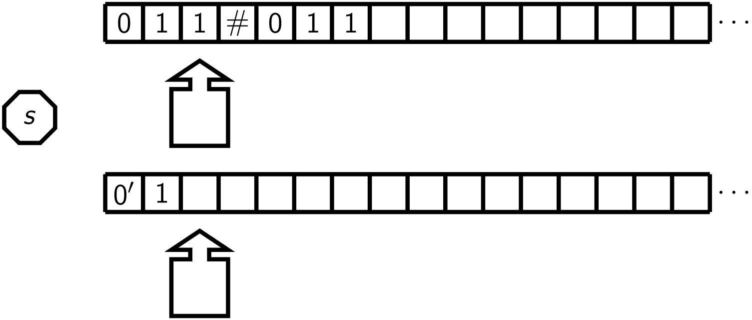
A 2-tape DTM for $\{w\#w \mid w \in \{0,1\}^*\}$:



- Copy from first to second tape until first $\#$ (additionally marking the first cell by a prime)

Conceptual View

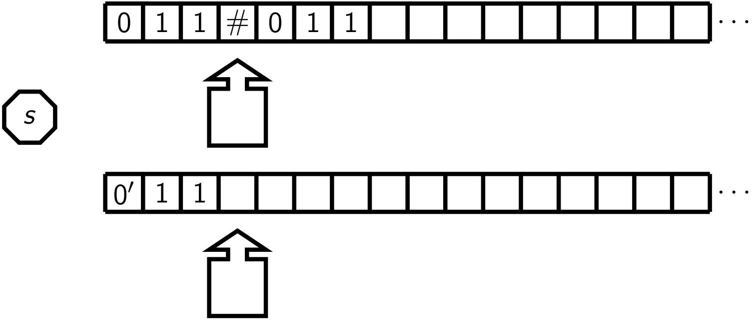
A 2-tape DTM for $\{w\#w \mid w \in \{0,1\}^*\}$:



- Copy from first to second tape until first $\#$ (additionally marking the first cell by a prime)

Conceptual View

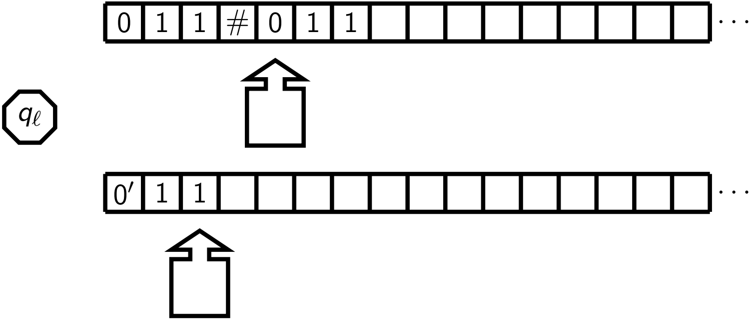
A 2-tape DTM for $\{w\#w \mid w \in \{0,1\}^*\}$:



- Copy from first to second tape until first $\#$ (additionally marking the first cell by a prime)
- Move second head back to the primed letter and the first one one cell to the right

Conceptual View

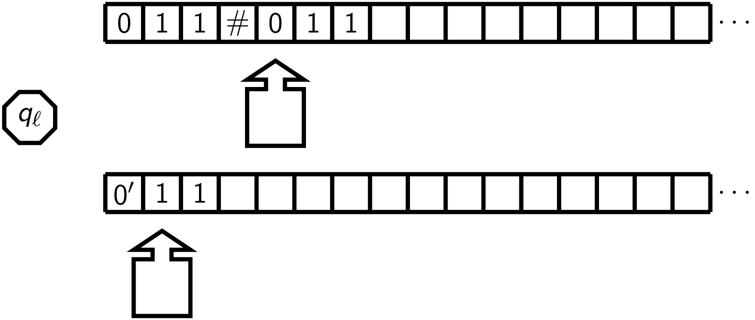
A 2-tape DTM for $\{w\#w \mid w \in \{0,1\}^*\}$:



- Copy from first to second tape until first # (additionally marking the first cell by a prime)
- Move second head back to the primed letter and the first one one cell to the right

Conceptual View

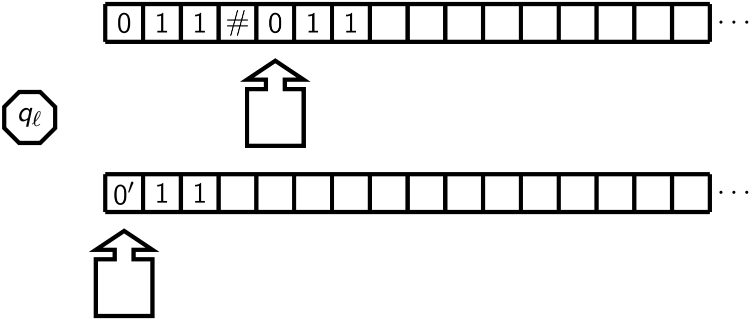
A 2-tape DTM for $\{w\#w \mid w \in \{0,1\}^*\}$:



- Copy from first to second tape until first # (additionally marking the first cell by a prime)
- Move second head back to the primed letter and the first one one cell to the right

Conceptual View

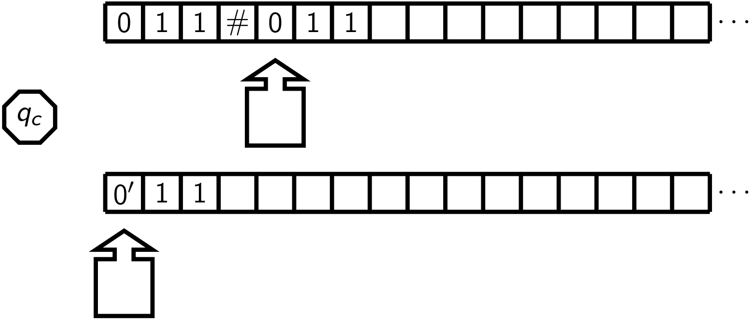
A 2-tape DTM for $\{w\#w \mid w \in \{0,1\}^*\}$:



- Copy from first to second tape until first # (additionally marking the first cell by a prime)
- Move second head back to the primed letter and the first one one cell to the right

Conceptual View

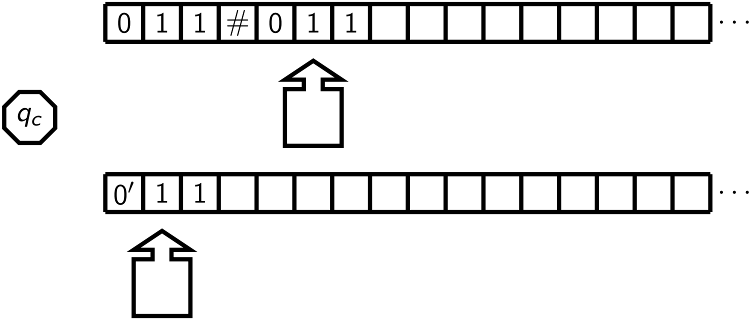
A 2-tape DTM for $\{w\#w \mid w \in \{0,1\}^*\}$:



- Copy from first to second tape until first $\#$ (additionally marking the first cell by a prime)
- Move second head back to the primed letter and the first one one cell to the right
- Compare content on first and second tape. Accept if equal (ignoring the prime), otherwise reject

Conceptual View

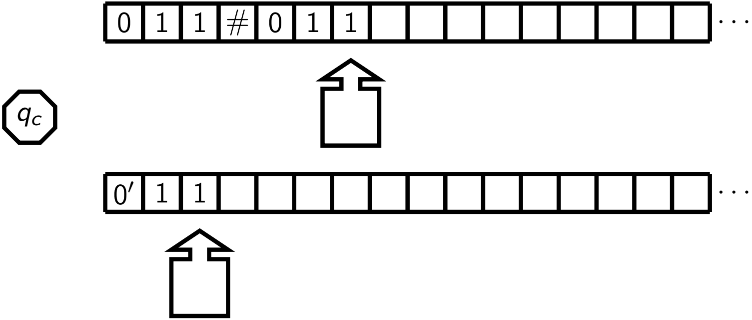
A 2-tape DTM for $\{w\#w \mid w \in \{0,1\}^*\}$:



- Copy from first to second tape until first # (additionally marking the first cell by a prime)
- Move second head back to the primed letter and the first one one cell to the right
- Compare content on first and second tape. Accept if equal (ignoring the prime), otherwise reject

Conceptual View

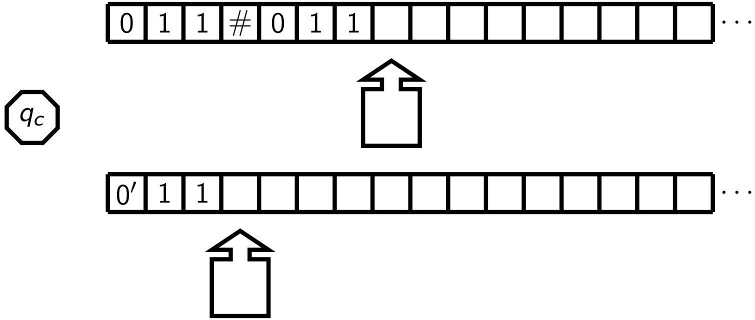
A 2-tape DTM for $\{w\#w \mid w \in \{0,1\}^*\}$:



- Copy from first to second tape until first $\#$ (additionally marking the first cell by a prime)
- Move second head back to the primed letter and the first one one cell to the right
- Compare content on first and second tape. Accept if equal (ignoring the prime), otherwise reject

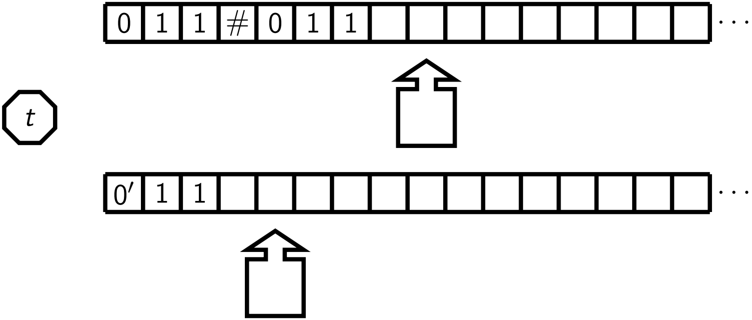
Conceptual View

A 2-tape DTM for $\{w\#w \mid w \in \{0,1\}^*\}$:



- Copy from first to second tape until first # (additionally marking the first cell by a prime)
- Move second head back to the primed letter and the first one one cell to the right
- Compare content on first and second tape. Accept if equal (ignoring the prime), otherwise reject

A 2-tape DTM for $\{w\#w \mid w \in \{0,1\}^*\}$:



- Copy from first to second tape until first # (additionally marking the first cell by a prime)
- Move second head back to the primed letter and the first one one cell to the right
- Compare content on first and second tape. Accept if equal (ignoring the prime), otherwise reject

Definition

Definition (full definition in book)

Let $k \geq 1$. A k-tape DTM has the form $(Q, \Sigma, \Gamma, s, t, r, \delta)$ where Q, Σ, Γ, s, t and r are as for DTMs and where

$$\delta \colon (Q \setminus \{t, r\}) \times \Gamma^k \to Q \times \Gamma^k \times \{-1, +1\}^k$$

- **Configuration**: **One** state and k tapes with k (independent) heads
- **Initial configuration**: Input word on first tape, all other tapes empty
- **Successor configuration**: Update state, update current cell on each tape, move head on each tape

Definition

Definition (full definition in book)

Let $k \geq 1$. A k-tape DTM has the form $(Q, \Sigma, \Gamma, s, t, r, \delta)$ where Q, Σ, Γ, s, t and r are as for DTMs and where

$$\delta\colon (Q \setminus \{t, r\}) \times \Gamma^k \to Q \times \Gamma^k \times \{-1, +1\}^k$$

- **Configuration**: **One** state and k tapes with k (independent) heads
- **Initial configuration**: Input word on first tape, all other tapes empty
- **Successor configuration**: Update state, update current cell on each tape, move head on each tape

Remark

1-tape DTM = DTM as defined in the previous lecture

Simulation

A machine M' outcome-simulates a machine M if we have the following for every input w:

- If M halts on w, then M' halts on w, and
- M accepts w if and only if M' accepts w

Simulation

A machine M' outcome-simulates a machine M if we have the following for every input w:

- If M halts on w, then M' halts on w, and
- M accepts w if and only if M' accepts w

Theorem

For every k-tape DTM M there is a (standard) DTM that outcome-simulates M

Simulation

A machine M' outcome-simulates a machine M if we have the following for every input w:

- If M halts on w, then M' halts on w, and
- M accepts w if and only if M' accepts w

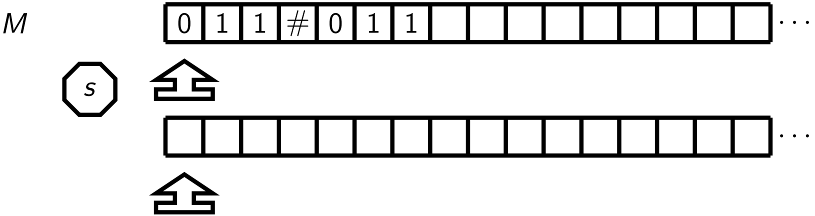
Theorem

For every k-tape DTM M there is a (standard) DTM that outcome-simulates M

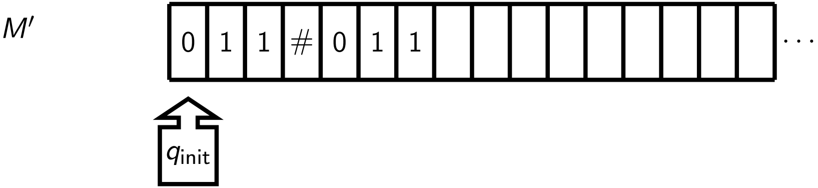
The language of a multi-tape DTM and multi-tape halting DTMs are defined as expected

Corollary

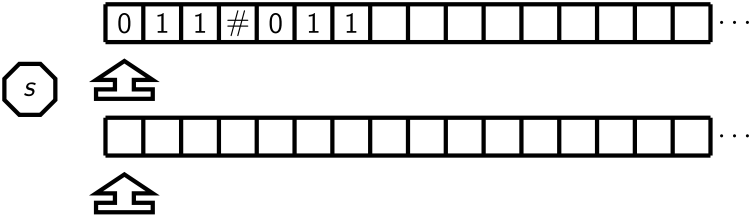
1. *A language is computably-enumerable if and only if it is the language of some multi-tape DTM*
2. *A language is computable if and only if it is the language of some halting multi-tape DTM*



M' simulates M by storing the n-th cells of M's tapes in the n-th cell of its tape; head positions are marked with a “ˆ” $\Rightarrow$ requires initialization!

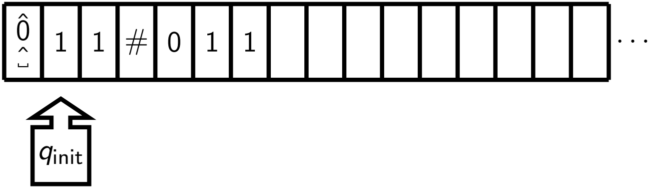


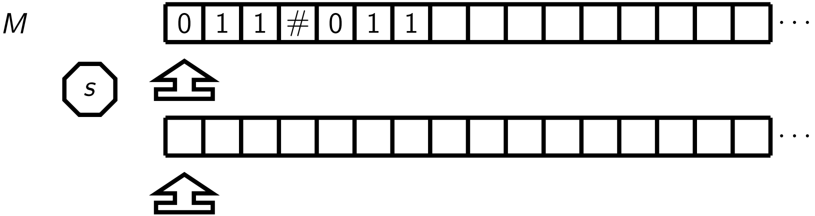
M



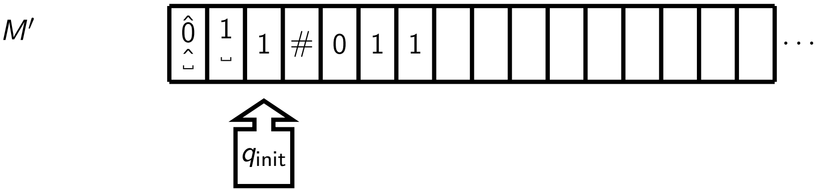
M' simulates M by storing the n-th cells of M's tapes in the n-th cell of its tape; head positions are marked with a “$\hat{\ }$” $\Rightarrow$ requires initialization!

M'

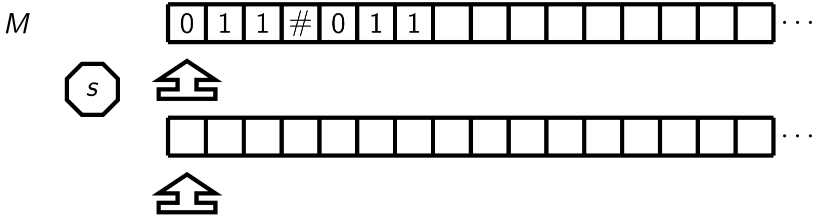




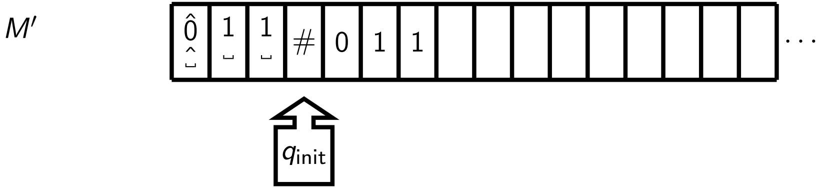
M' simulates M by storing the n-th cells of M's tapes in the n-th cell of its tape; head positions are marked with a “ˆ” $\Rightarrow$ requires initialization!

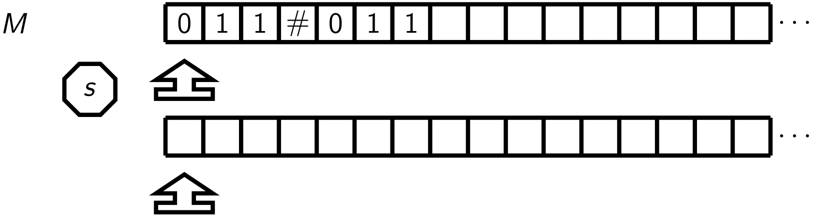


Proof Sketch

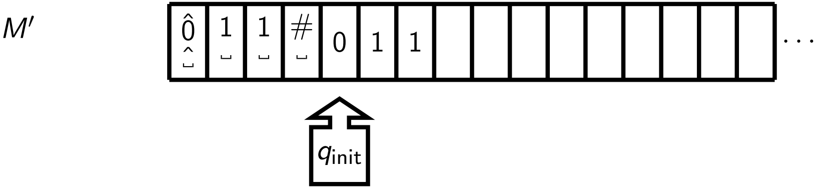


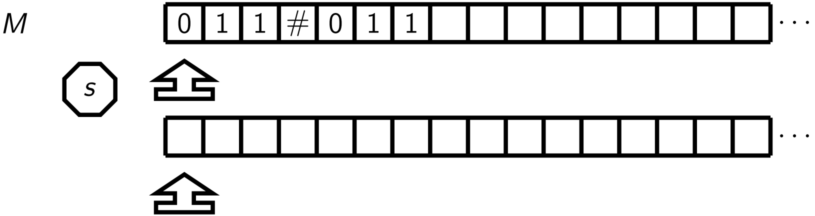
M' simulates M by storing the n-th cells of M's tapes in the n-th cell of its tape; head positions are marked with a "ˆ" ⇒ requires initialization!



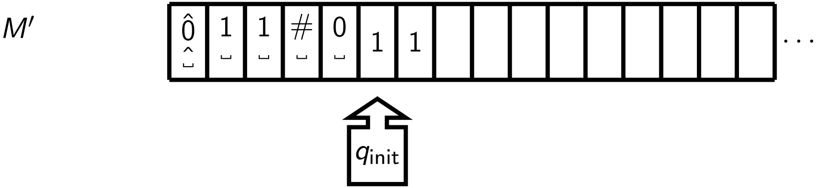


M' simulates M by storing the n-th cells of M's tapes in the n-th cell of its tape; head positions are marked with a “ˆ” $\Rightarrow$ requires initialization!

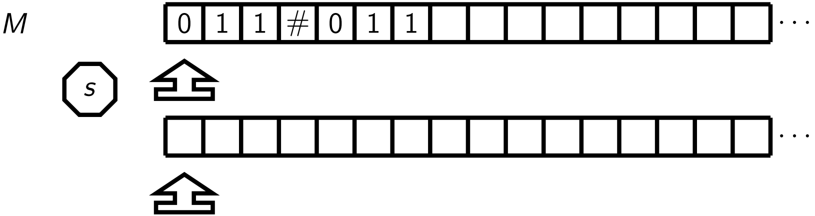




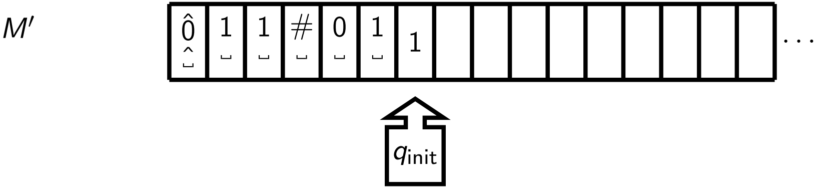
M' simulates M by storing the n-th cells of M's tapes in the n-th cell of its tape; head positions are marked with a "ˆ" $\Rightarrow$ requires initialization!



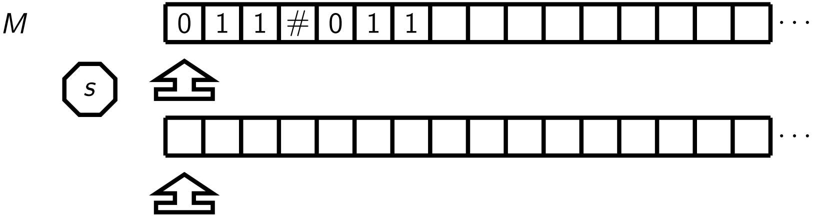
Proof Sketch



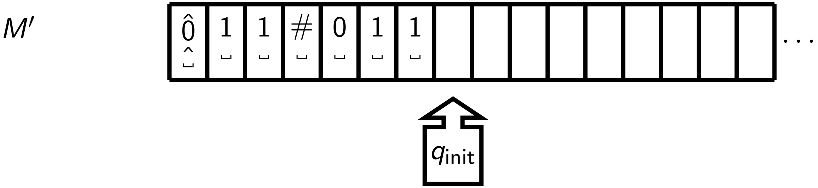
M' simulates M by storing the n-th cells of M's tapes in the n-th cell of its tape; head positions are marked with a “ˆ” $\Rightarrow$ requires initialization!



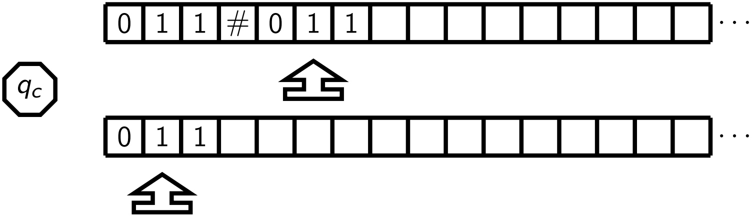
Proof Sketch



M' simulates M by storing the n-th cells of M's tapes in the n-th cell of its tape; head positions are marked with a “^” $\Rightarrow$ requires initialization!

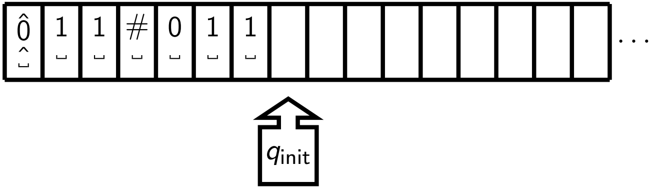


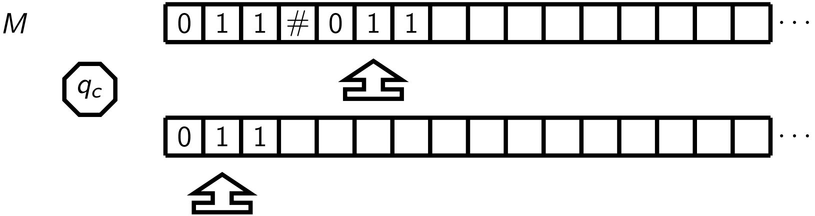
M



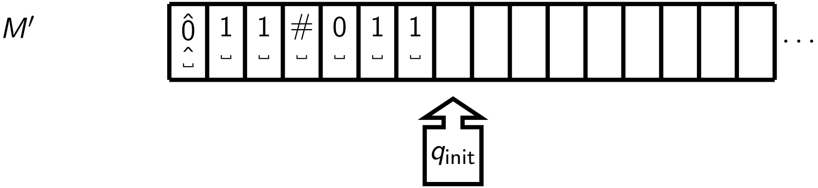
Let us simulate a transition of M.

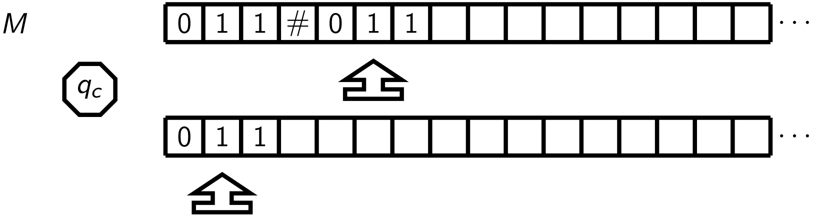
M'



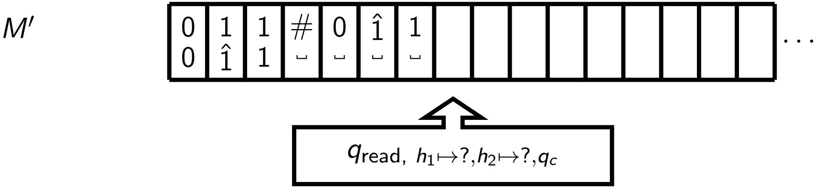


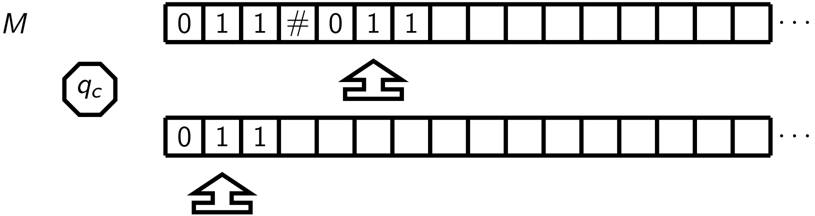
Let us simulate a transition of M. First, M' has to read its tape to determine the letters marked by ˆ on each tape



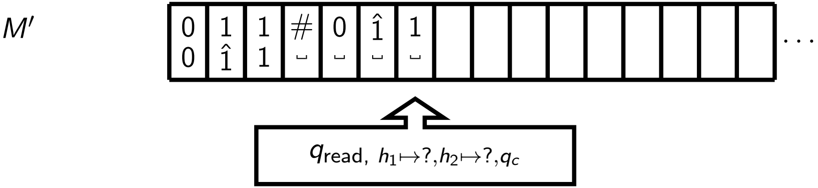


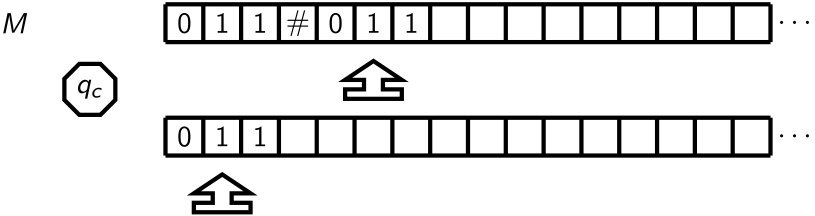
Let us simulate a transition of M. First, M' has to read its tape to determine the letters marked by ˆ on each tape



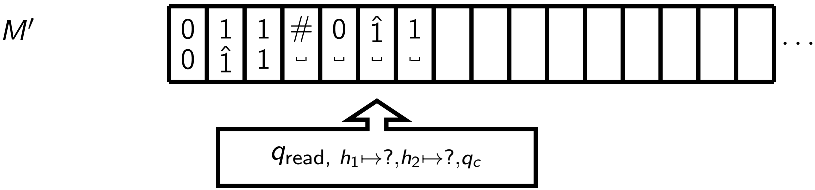


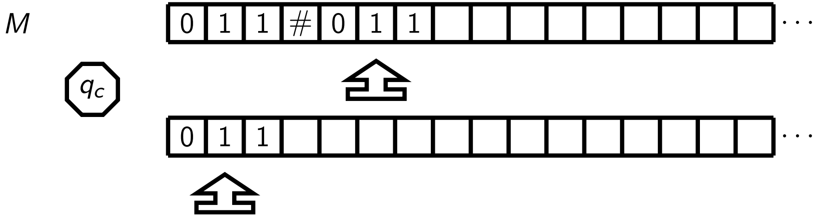
Let us simulate a transition of M. First, M' has to read its tape to determine the letters marked by ˆ on each tape



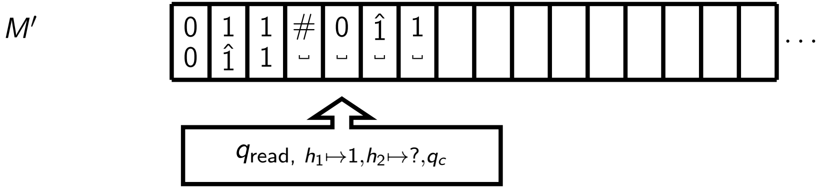


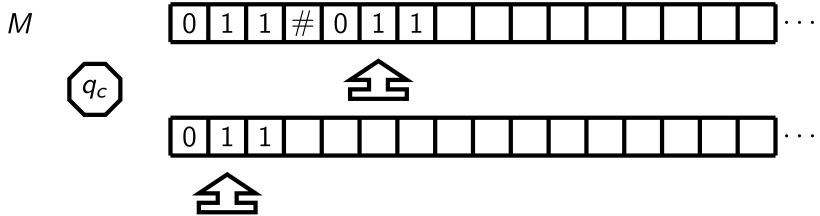
Let us simulate a transition of M. First, M' has to read its tape to determine the letters marked by ˆ on each tape



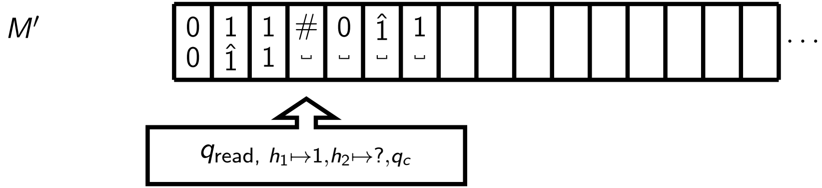


Let us simulate a transition of M. First, M' has to read its tape to determine the letters marked by ˆ on each tape

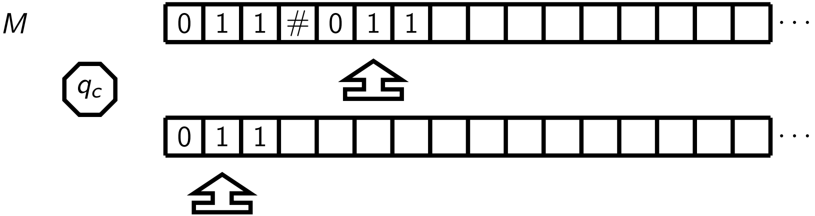




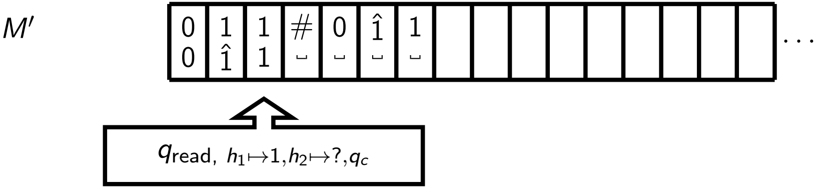
Let us simulate a transition of M. First, M' has to read its tape to determine the letters marked by ˆ on each tape



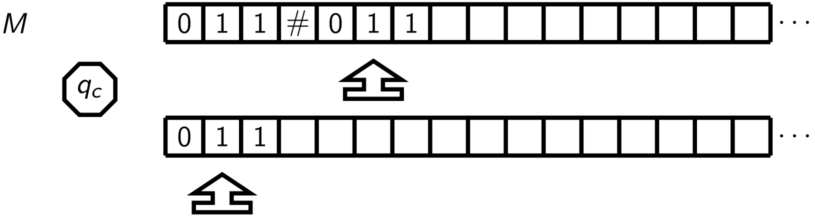
Proof Sketch



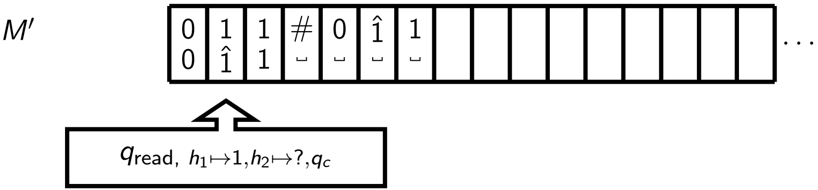
Let us simulate a transition of M. First, M' has to read its tape to determine the letters marked by ˆ on each tape

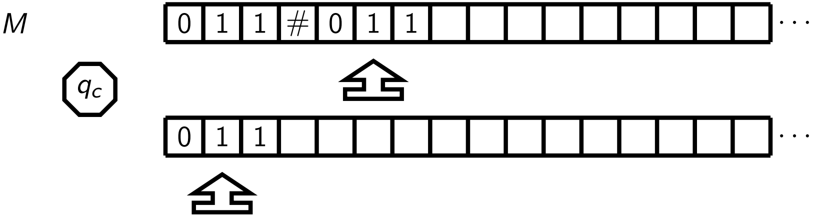


Proof Sketch

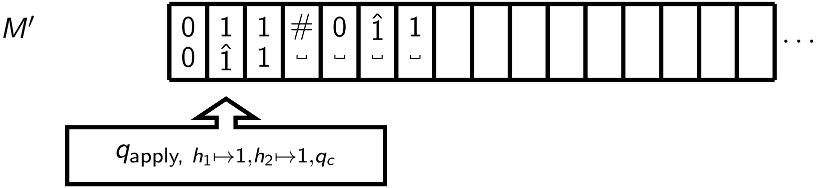


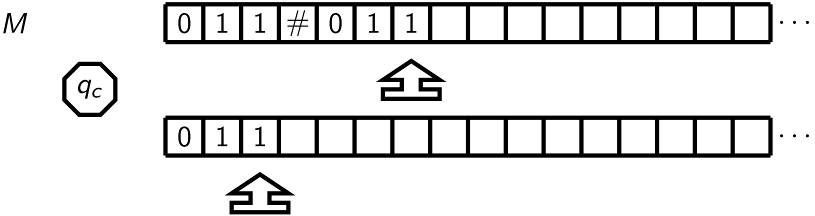
Let us simulate a transition of M. First, M' has to read its tape to determine the letters marked by ˆ on each tape



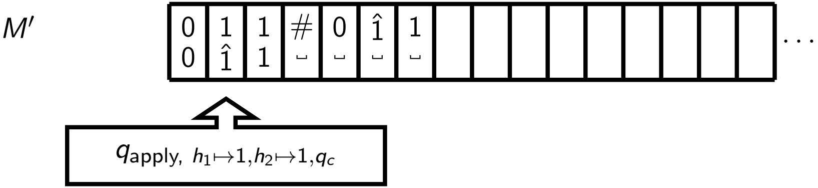


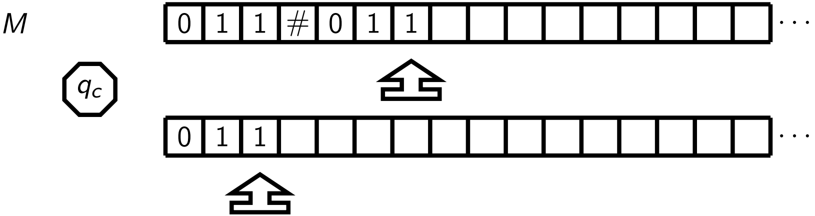
Let us simulate a transition of M. First, M' has to read its tape to determine the letters marked by ˆ on each tape



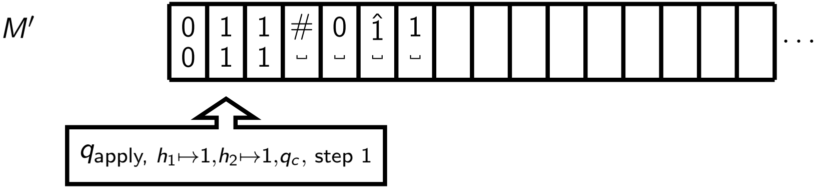


Let us simulate a transition of M. Then, M' can simulate the transition by updating cells and head positions

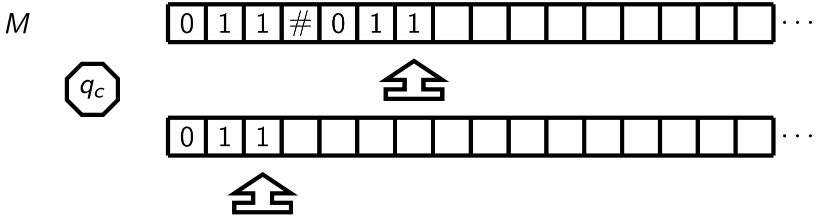




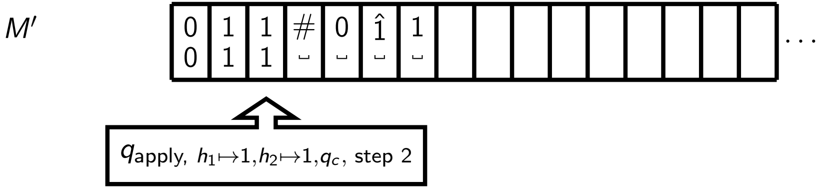
Let us simulate a transition of M. Then, M' can simulate the transition by updating cells and head positions

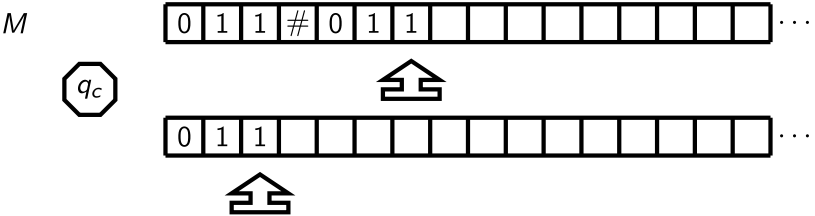


Proof Sketch

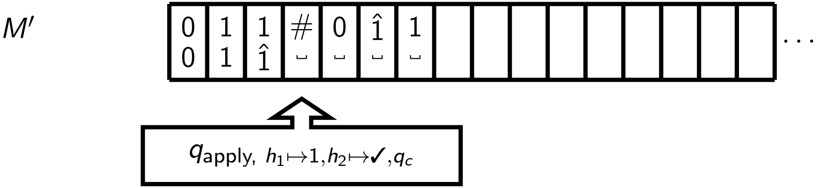


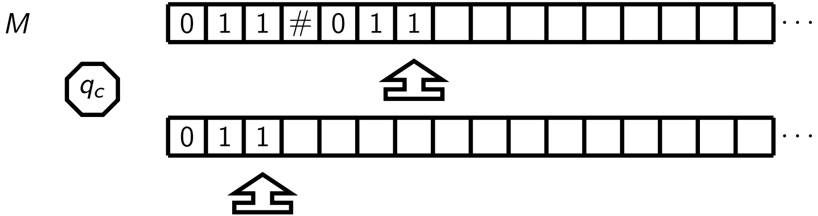
Let us simulate a transition of M. Then, M' can simulate the transition by updating cells and head positions



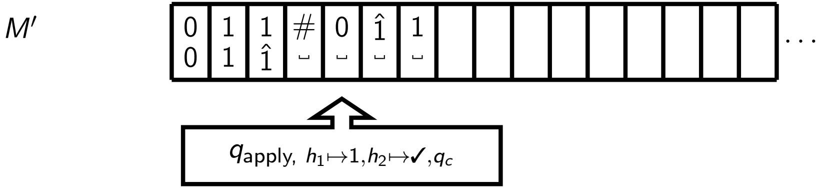


Let us simulate a transition of M. Then, M' can simulate the transition by updating cells and head positions

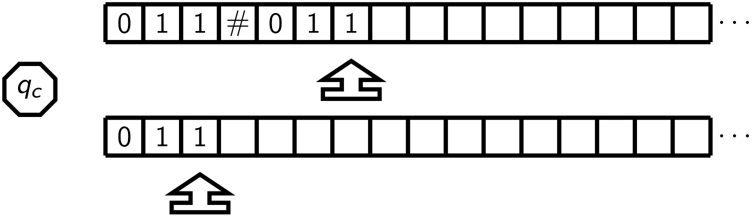




Let us simulate a transition of M. Then, M' can simulate the transition by updating cells and head positions

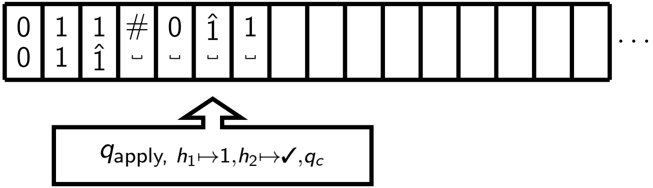


M

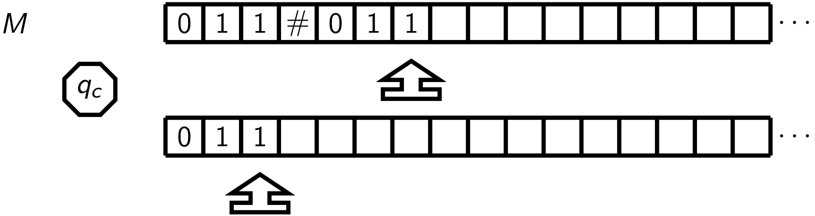


Let us simulate a transition of M. Then, M' can simulate the transition by updating cells and head positions

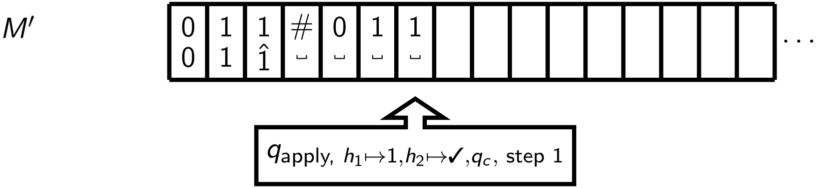
M'

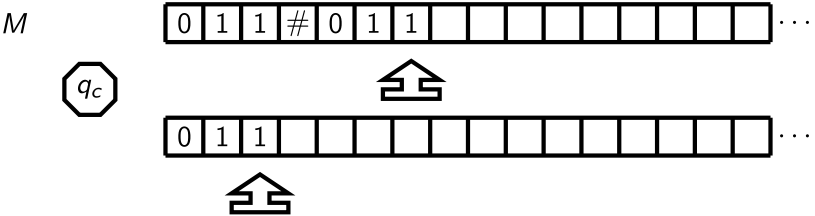


Proof Sketch

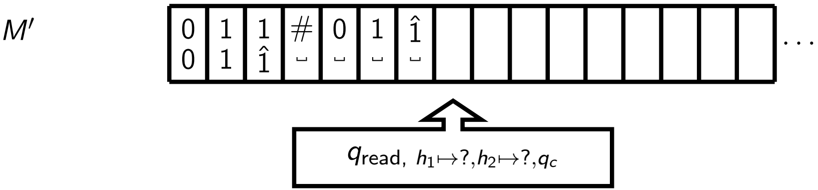


Let us simulate a transition of M. Then, M' can simulate the transition by updating cells and head positions



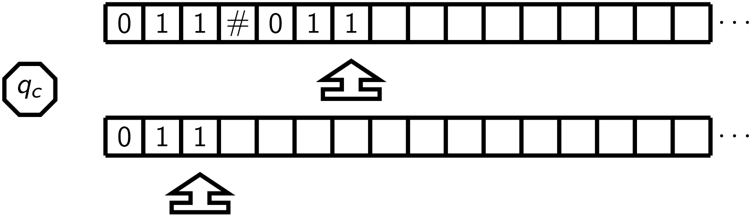


Let us simulate a transition of M. Then, M' can simulate the transition by updating cells and head positions



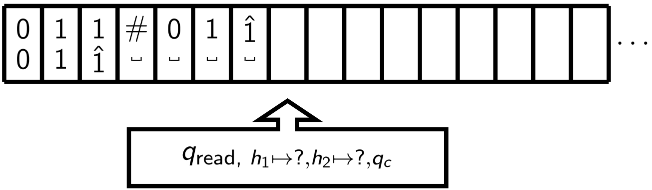
Proof Sketch

M



This process is repeated until M halts. M' accepts/rejects if and only if M does so

M'



Agenda

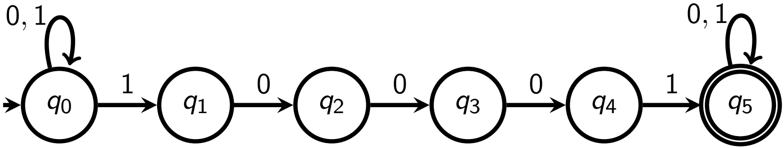
1. The Church-Turing Thesis
2. Multi-tape Turing Machines
3. **Nondeterministic Turing Machines**

A nondeterministic finite automaton (NFA) has the form $(Q, \Sigma, q_I, \delta, F)$ where

- Q is a finite set of states,
- Σ is an alphabet,
- $q_I \in Q$ is the initial state,
- $\delta \colon Q \times \Sigma \to 2^Q$ is the transition function, and
- $F \subseteq Q$ is a set of accepting states

Example

An NFA for the language $\{\{0,1\}^* 10001 \{0,1\}^* \mid n \geq 0\}$:



Definition (full definition in book)

A nondeterministic Turing machine (NTM) has the form $(Q, \Sigma, \Gamma, s, t, r, \delta)$ where Q, Σ, Γ, s, t and r are as for DTMs and where

$$\delta\colon (Q \setminus \{t, r\}) \times \Gamma \to 2^{Q \times \Gamma \times \{-1,+1\}}$$

Intuition:

- Configurations and initial configuration as for DTM
- But: A configuration can have multiple successor configurations
- Acceptance: Some accepting configuration is reachable from the initial configuration

Intuition

$\delta(q, b) = \{(q_1, c, 0), (q_2, d, +1), (q_3, e, -1)\}$:

$$[q, abcw' \llcorner \ldots, 2]$$

Intuition

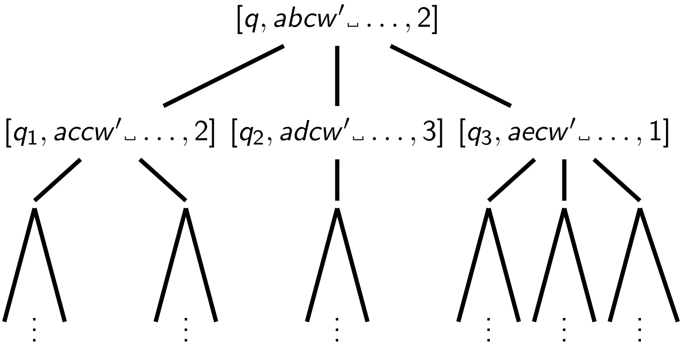
$\delta(q, b) = \{(q_1, c, 0), (q_2, d, +1), (q_3, e, -1)\}$:

$$[q, abcw' \textvisiblespace \ldots, 2]$$

$$[q_1, accw' \textvisiblespace \ldots, 2] \quad [q_2, adcw' \textvisiblespace \ldots, 3] \quad [q_3, aecw' \textvisiblespace \ldots, 1]$$

Intuition

$\delta(q, b) = \{(q_1, c, 0), (q_2, d, +1), (q_3, e, -1)\}$:



Computation tree of an NTM on an input w:

- Root: Initial configuration on w
- Children of a configuration: All its successor configurations
- May be infinite (if and only if it has an infinite branch)

More Definitions

Definition

An NTM M accepts an input w if the computation tree of M on w contains an accepting configuration

As before:

$$L(M) = \{w \in \Sigma^* \mid M \text{ accepts } w\}$$

More Definitions

Definition

An NTM M accepts an input w if the computation tree of M on w contains an accepting configuration

As before:

$$L(M) = \{w \in \Sigma^* \mid M \text{ accepts } w\}$$

Definition

An NTM M is a halting NTM if the computation tree of M is finite for every input w

So, every branch ends in an accepting or rejecting configuration

Quiz 2

When does a halting NTM reject an input?

Quiz 2

When does a halting NTM reject an input?

When all branches end in a rejecting configuration

Simulation

Theorem

For every NTM M there is a (standard) DTM that outcome-simulates M

Theorem

For every NTM M there is a (standard) DTM that outcome-simulates M

Proof sketch:

Let M' be a DTM that does the following when given an input w:

- $\Gamma_0 := \{\alpha_w\}$, where α_w is the initial configuration of M on w
- $i := 0$
- Iterate:
  - If Γ_i contains an accepting configuration, accept
  - If Γ_i is empty, reject
  - $\Gamma_{i+1} := \{\gamma' \mid \gamma \vdash_M \gamma' \text{ for some } \gamma \in \Gamma_i\}$ (the set of successor configurations of the configurations in Γ_i)
  - $i := i + 1$

Simulation

Theorem

For every NTM M there is a (standard) DTM that outcome-simulates M

Corollary

1. *A language is computably-enumerable if and only if it is the language of some NTM*
2. *A language is computable if and only if it is the language of some halting NTM*

Conclusion

We have seen

- the Church-Turing Thesis,
- multi-tape DTMs and
- NTMs, and
- their equivalence

- Not covered: multi-tape NTMs (can also be simulated by DTMs)

Reading:

- Sections 2.6 and 2.7.1 of "Computability and Complexity" (pages 98 to 115)

Optional watching:

- Tom Wildenhain: On The Turing Completeness of PowerPoint