A theoretical computer science approach to entropy

ENS de Lyon meets **SISSA**

Nathalie Aubrun

LIP, ENS de Lyon, CNRS

5th December 2017

LIP – Laboratoire de l'Informatique du Parallélisme

Computer Science laboratory of the ENS de Lyon.

- **AriC**: Floating-point arithmetic, Certified computing and computer algebra, Cryptography and lattices.
- **Avalon**: Algorithms and Software Architectures for Distributed and HPC Platforms.
- **DANTE**: Graph-based signal processing, Dynamic graph theory, Distributed Algorithms for dynamic networks.
- **MC2**: Discrete and algebraic algorithms, Complexity theory, Combinatorics.
- **PLUME**: Logical Foundations of Programming Languages, Theory of Computing Systems.
- **ROMA**: Resource Optimization: Models, Algorithms, and Scheduling.
- **CASH**: Analysis and compilation for High Performance software and hardware.

LIP – Laboratoire de l'Informatique du Parallélisme

Computer Science laboratory of the ENS de Lyon.

- **AriC**: Floating-point arithmetic, Certified computing and computer algebra, Cryptography and lattices.
- **Avalon**: Algorithms and Software Architectures for Distributed and HPC Platforms.
- **DANTE**: Graph-based signal processing, Dynamic graph theory, Distributed Algorithms for dynamic networks.
- **MC2**: Discrete and algebraic algorithms, Complexity theory, Combinatorics.
- **PLUME**: Logical Foundations of Programming Languages, Theory of Computing Systems.
- **ROMA**: Resource Optimization: Models, Algorithms, and Scheduling.
- **CASH**: Analysis and compilation for High Performance software and hardware.

Introduction

A theoretical computer science approach to entropy

- Subshifts of finite type on $\mathbb{Z}^d$.
- Topological entropy.
- How does theoretical computer sciences help in understanding this entropy?

Subshifts of finite type (I)

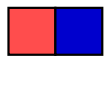
Subshifts of finite type (SFTs), topological Markov shifts, Bernoulli sub-flows, sets of tilings with Wang tiles...

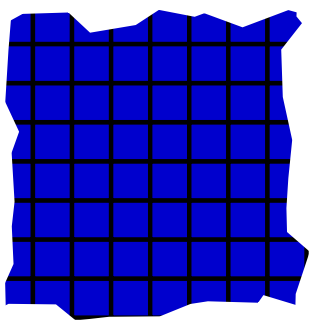
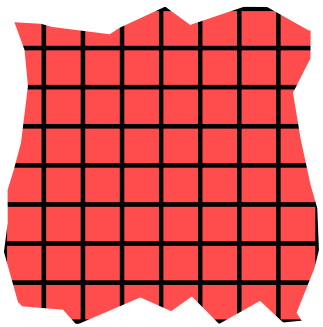
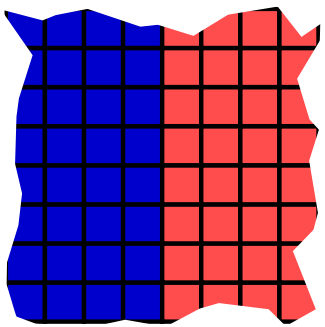
- $A^{\mathbb{Z}^{\boldsymbol{d}}}$: colorings of $\mathbb{Z}^d$ by some finite alphabet A
- SFTs = subsets of $A^{\mathbb{Z}^{\boldsymbol{d}}}$ defined by excluding finitely many patterns

Subshifts of finite type (I)

Subshifts of finite type (SFTs), topological Markov shifts, Bernoulli sub-flows, sets of tilings with Wang tiles...

- $A^{\mathbb{Z}^d}$: colorings of $\mathbb{Z}^d$ by some finite alphabet A
- SFTs = subsets of $A^{\mathbb{Z}^d}$ defined by excluding finitely many patterns

Example 1: The SFT $X_{\{$  $\}}$ contains the following configurations (in 2D):

Subshifts of finite type (III)

Example 2: The golden mean shift: $A = \{0, 1\}$, avoids any pair of adjacent 1's.

Subshifts of finite type (III)

Example 2: The golden mean shift: $A = \{0, 1\}$, avoids any pair of adjacent 1's.

In 1D, admissible colorings include

$$\ldots \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad \ldots$$

and

$$\ldots \quad 0 \quad 0 \quad 1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad \ldots$$

and

$$\ldots \quad 0 \quad 0 \quad 1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 \quad 1 \quad \ldots$$

etc. . .

Subshifts of finite type (III)

Example 2: The golden mean shift: $A = \{0, 1\}$, avoids any pair of adjacent 1's.

In 2D, admissible colorings include

Subshifts of finite type (III)

Example 2: The golden mean shift: $A = \{0, 1\}$, avoids any pair of adjacent 1's.

In 2D, admissible colorings include

Subshifts of finite type (III)

Example 2: The golden mean shift: $A = \{0, 1\}$, avoids any pair of adjacent 1's.

In 2D, admissible colorings include

$$
\begin{array}{cccccccccccccc}
1 & . & . & 1 & . & . & . & . & 1 & . & . & . & . & . \\
. & . & 1 & . & . & 1 & . & . & . & 1 & . & . & . & . \\
1 & . & . & . & . & . & . & . & . & . & . & . & . & . \\
. & . & . & . & 1 & . & . & 1 & . & . & . & . & . & . \\
. & . & . & . & . & . & . & . & . & . & . & 1 & . & . \\
. & . & . & . & 1 & . & . & 1 & . & . & . & . & . & . \\
. & 1 & . & . & . & . & . & . & . & 1 & . & . & . & .
\end{array}
$$

Subshifts of finite type (III)

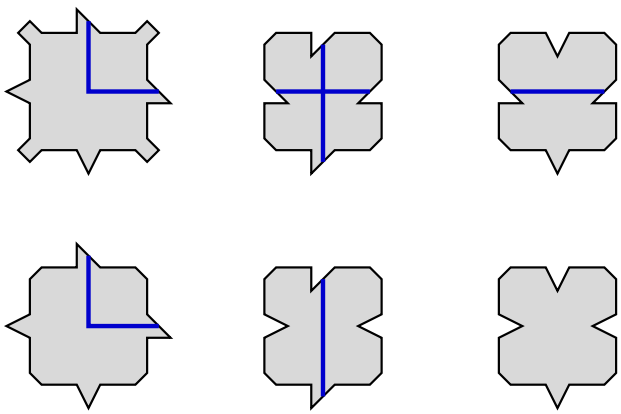
Example 2: The golden mean shift: $A = \{0, 1\}$, avoids any pair of adjacent 1's.

In 2D, admissible colorings include

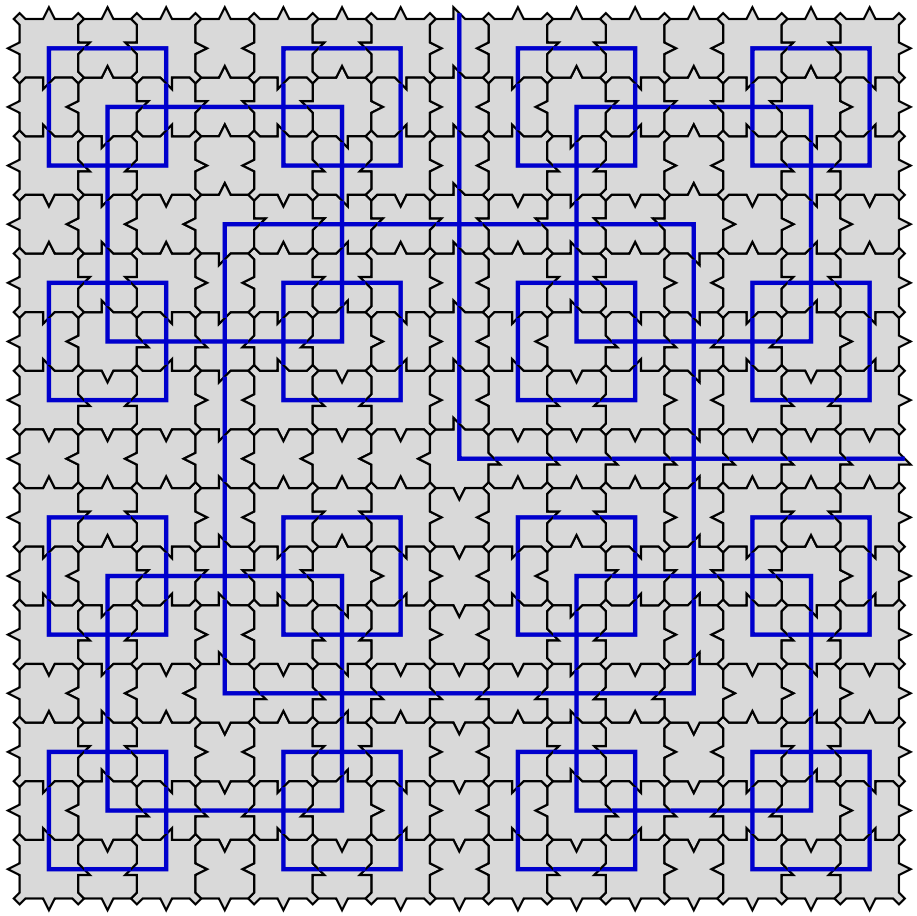
```
1 . 1 . 1 . 1 . 1 . 1 . 1 . 1 .
. 1 . 1 . 1 . 1 . 1 . 1 . 1 . 1
1 . 1 . 1 . 1 . 1 . 1 . 1 . 1 .
. 1 . 1 . 1 . 1 . 1 . 1 . 1 . 1
1 . 1 . 1 . 1 . 1 . 1 . 1 . 1 .
. 1 . 1 . 1 . 1 . 1 . 1 . 1 . 1
1 . 1 . 1 . 1 . 1 . 1 . 1 . 1 .
```

Subshifts of finite type (III)

Example 3: Robinson's SFT

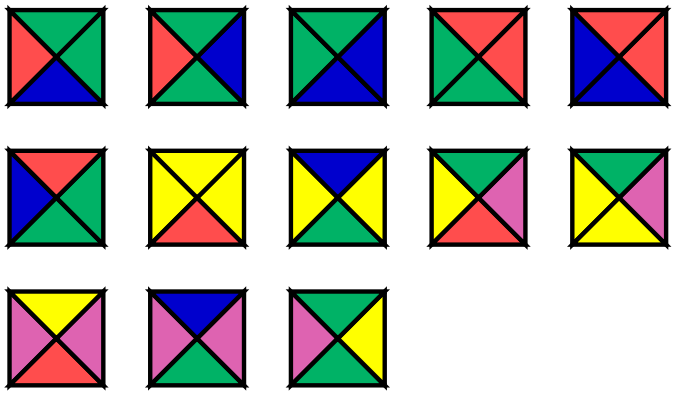


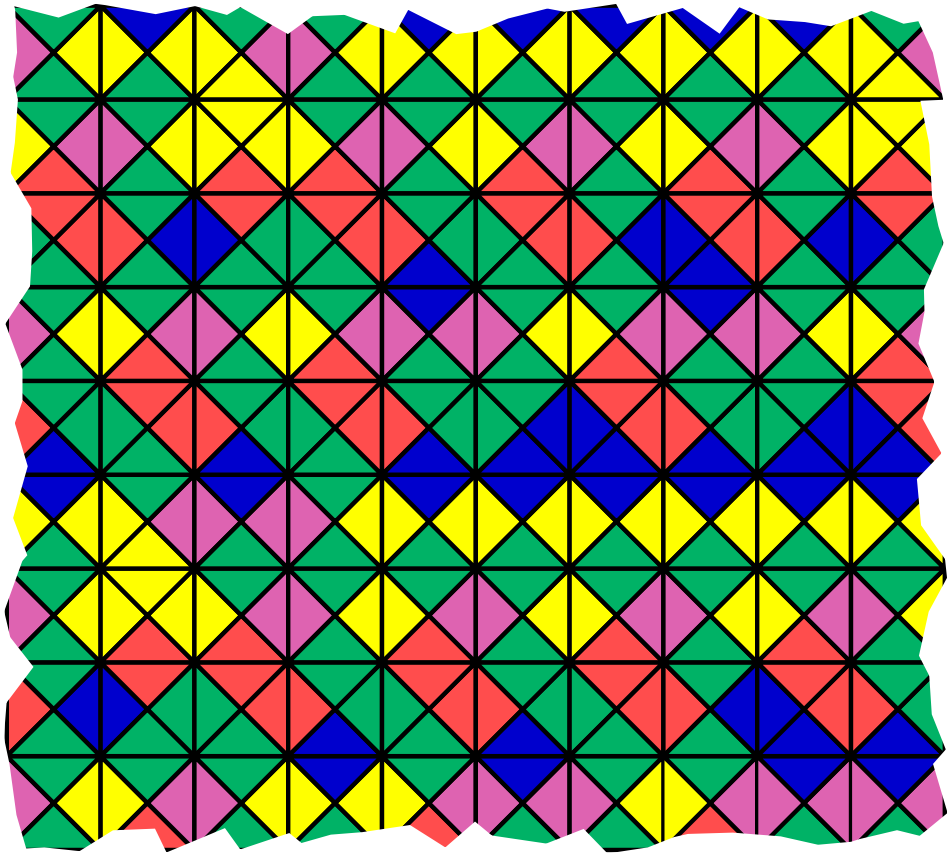
+ rotations & reflections



Subshifts of finite type (IV)

Example 4: (Kari & Culik)'s SFT





Entropy of SFTs

If $X \subseteq A^{\mathbb{Z}^{\boldsymbol{d}}}$, denote $\mathcal{L}_n(X)$ the set of patterns with shape $[1; n]^d$ that appear in X.

Then the **topological entropy** of X is

$$h(X) = \lim_{n \to \infty} \frac{1}{n^d} \log \left(|\mathcal{L}_n| \right).$$

Entropy of SFTs

If $X \subseteq A^{\mathbb{Z}^{\boldsymbol{d}}}$, denote $\mathcal{L}_n(X)$ the set of patterns with shape $[1; n]^d$ that appear in X.

Then the **topological entropy** of X is

$$h(X) = \lim_{n \to \infty} \frac{1}{n^{\boldsymbol{d}}} \log \left(|\mathcal{L}_n| \right).$$

Example: if $X = A^{\mathbb{Z}^{\boldsymbol{d}}}$, then $h(X) = \lim_{n \to \infty} \frac{1}{n^{\boldsymbol{d}}} \log \left(|A|^{n^{\boldsymbol{d}}} \right) = \log(|A|)$.

Why entropy?

Conjugacy invariant

Entropy is conjugacy invariant for SFTs: if X and Y are conjugate ($\approx$ the same up to conjugacy), then $h(X) = h(Y)$.

Why entropy?

Conjugacy invariant

Entropy is conjugacy invariant for SFTs: if X and Y are conjugate ($\approx$ the same up to conjugacy), then $h(X) = h(Y)$.

If $p \in A^{[1;n]^d}$, $[p] = \left\{ x \in A^{\mathbb{Z}^d} \mid x_{|[1;n]^d} = p \right\}$.

Variational principle

If μ is a shift-invariant probability measure on X, then

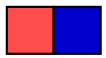
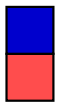
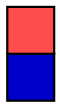
$$h(\mu) = \lim_{n \to \infty} \frac{-1}{n^d} \sum_{p \in A^{[1;n]^d}} \mu([p]) \log(\mu([p])).$$

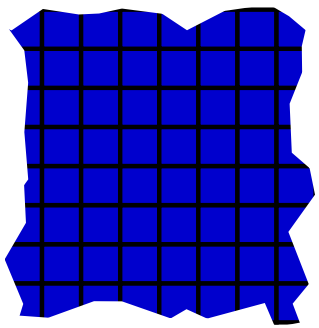
Variational principle:

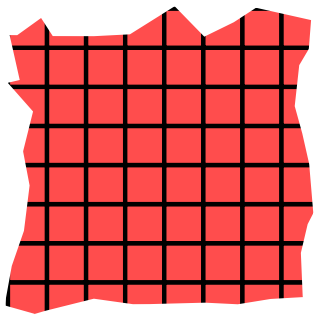
$$\sup_{\mu} h(\mu) = h(X)$$

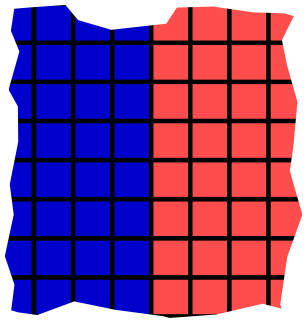
where μ ranges over all shift-invariant measures on X.

Entropy: Example 1

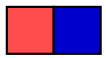
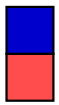
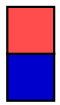
The SFT $X_{\{$  ,  ,  $\}$ contains the following configurations (in 2D):

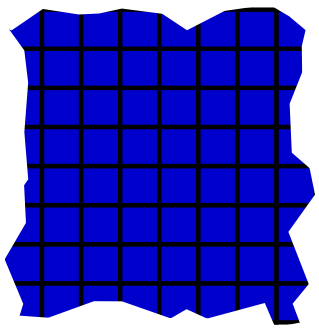
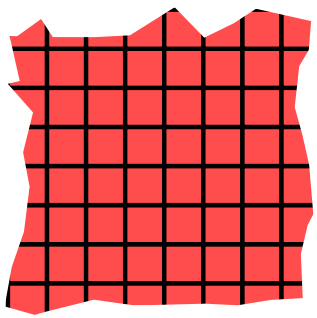
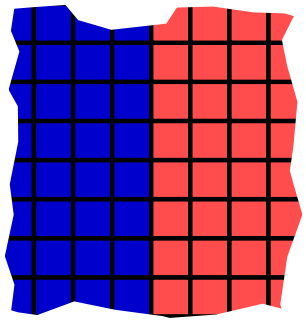






Entropy: Example 1

The SFT $X_{\{$  ,  ,  $\}$ contains the following configurations (in 2D):

- $\mathcal{L}_n =$? Choose where the limit between  and  lies: $n+1$ choices.
- $|\mathcal{L}_n| = n+1$
- $\Rightarrow \ h(X) = 0$

Entropy: Example 2

Example 2: The golden mean shift: $A = \{0, 1\}$, avoids any pair of adjacent 1's.

In 1D, admissible colorings include

```
1 . . 1 . . . . 1 . . . . .
. . 1 . . 1 . . . 1 . . . .
1 . . . . . . . . . . . . .
. . . . 1 . . 1 . . . . . .
. . . . . . . . . . 1 . . .
. . . . 1 . . 1 . . . . . .
. 1 . . . . . . . 1 . . . .
```

Entropy: Example 2

Example 2: The golden mean shift: $A = \{0, 1\}$, avoids any pair of adjacent 1's.

In 1D, admissible colorings include

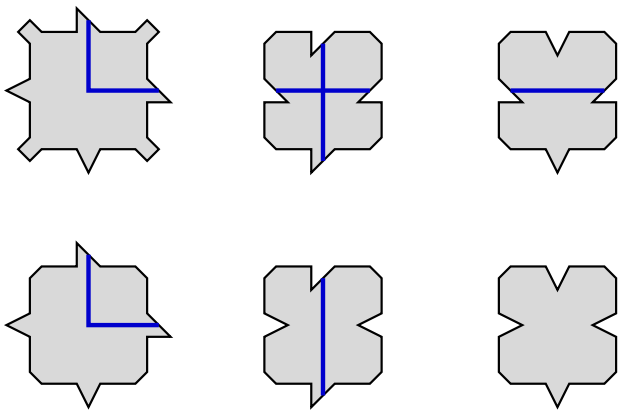
$$\begin{array}{cccccccccccccc}
1 & . & . & 1 & . & . & . & . & 1 & . & . & . & . & . \\
. & . & 1 & . & . & 1 & . & . & . & 1 & . & . & . & . \\
1 & . & . & . & . & . & . & . & . & . & . & . & . & . \\
. & . & . & . & 1 & . & . & 1 & . & . & . & . & . & . \\
. & . & . & . & . & . & . & . & . & . & . & 1 & . & . \\
. & . & . & . & 1 & . & . & 1 & . & . & . & . & . & . \\
. & 1 & . & . & . & . & . & . & . & 1 & . & . & . & .
\end{array}$$

The entropy is still unknown...

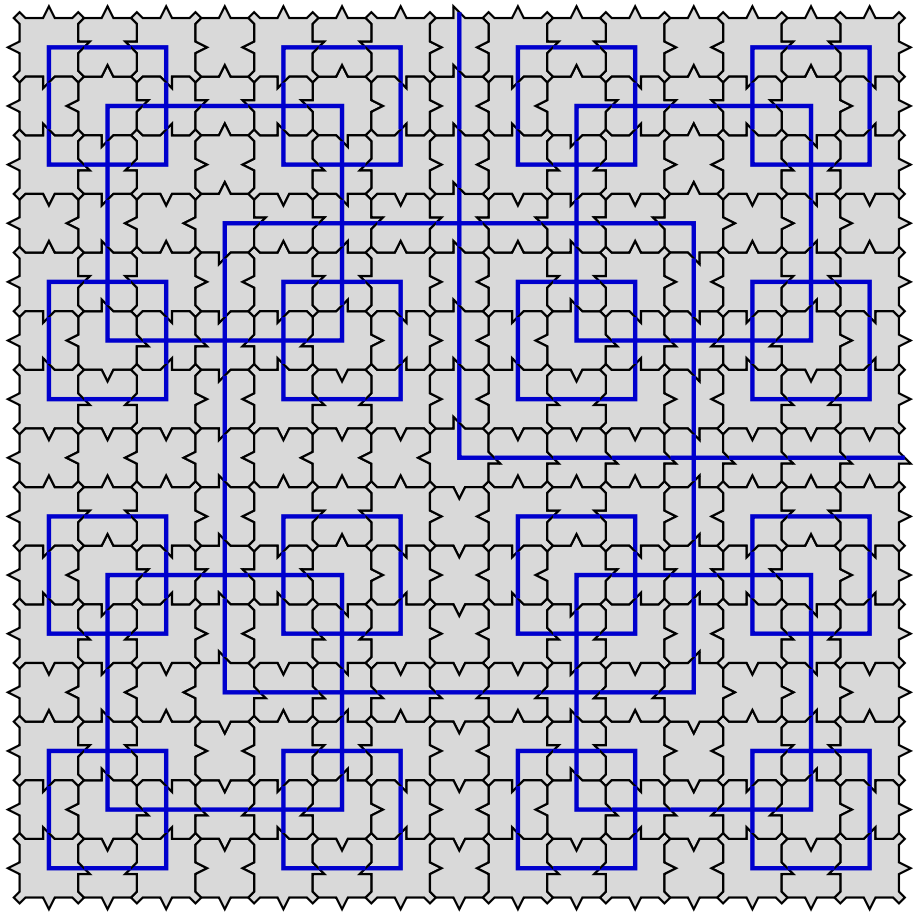
$$0.587891161775\mathbf{34} \leq h(X) \leq 0.587891161775\mathbf{35}$$

Entropy: Example 3

Example 3: Robinson's SFT



+ rotations & reflections

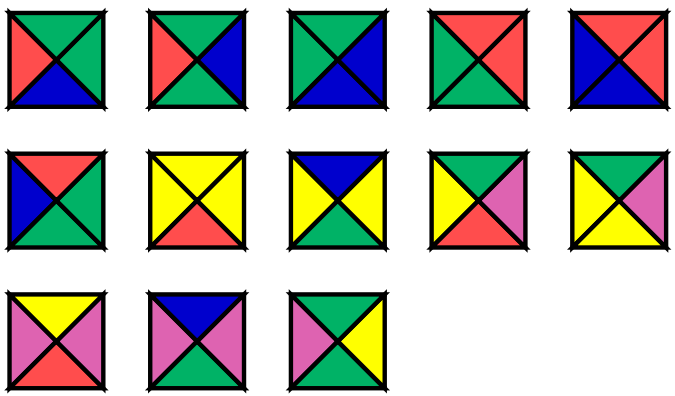


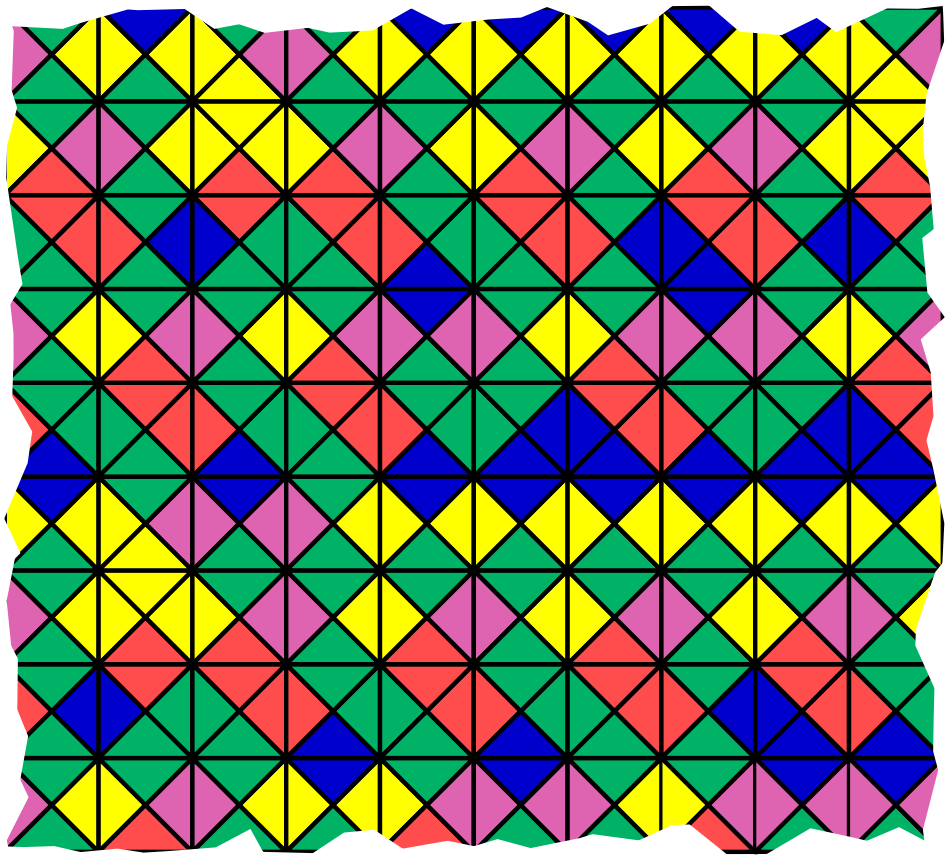
The entropy is known!

$$h(X) = 0$$

Subshifts of finite type (IV)

Example 4: (Kari & Culik)'s SFT





The entropy is unknown, but

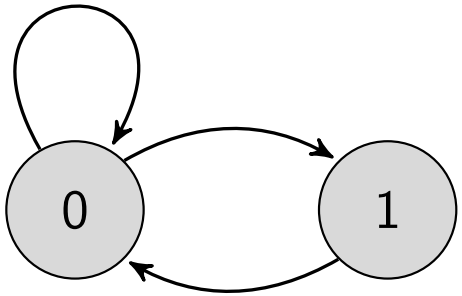
$$h(X) > 0$$

Entropy: Example 2, again

The 1D golden mean shift : 11 is forbidden.

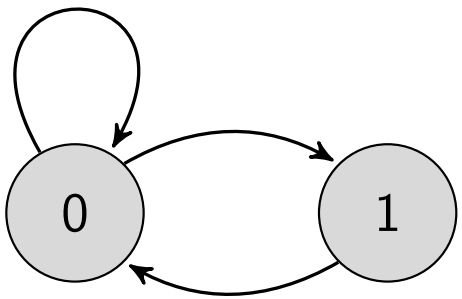
Entropy: Example 2, again

The 1D golden mean shift : 11 is forbidden.



Entropy: Example 2, again

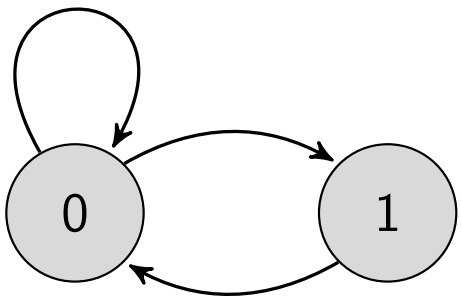
The 1D golden mean shift : 11 is forbidden.



$$M = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$$

Entropy: Example 2, again

The 1D golden mean shift : 11 is forbidden.

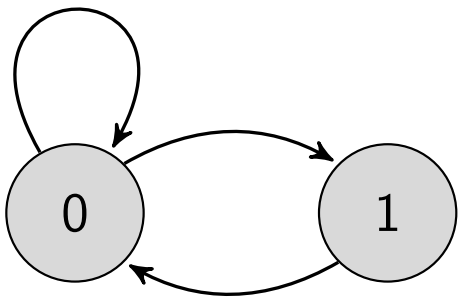


$$M = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$$

$$Sp(M) = \left\{ \frac{1+\sqrt{5}}{2}, \frac{1-\sqrt{5}}{2} \right\}$$

Entropy: Example 2, again

The 1D golden mean shift : 11 is forbidden.



$$M = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$$

$$h(X) = \log\left(\frac{1+\sqrt{5}}{2}\right) \qquad Sp(M) = \left\{\frac{1+\sqrt{5}}{2}, \frac{1-\sqrt{5}}{2}\right\}$$

Computing the entropy of 1D SFTs

- Compute the graph/matrix representation of the SFT $\rightsquigarrow$ components (C_i)
- Compute λ_i the eigenvalue of C_i s.t. $\lambda_i > 1$ (Perron-Frobenius)
- $h(X) = \log(\max_i \lambda_i)$

Computing the entropy of 1D SFTs

- Compute the graph/matrix representation of the SFT $\rightsquigarrow$ components (C_i)
- Compute λ_i the eigenvalue of C_i s.t. $\lambda_i > 1$ (Perron-Frobenius)
- $h(X) = \log(\max_i \lambda_i)$

$\Rightarrow$ valid algorithm **for all** 1D SFTs.

Computing the entropy of 1D SFTs

- Compute the graph/matrix representation of the SFT $\rightsquigarrow$ components (C_i)
- Compute λ_i the eigenvalue of C_i s.t. $\lambda_i > 1$ (Perron-Frobenius)
- $h(X) = \log(\max_i \lambda_i)$

$\Rightarrow$ valid algorithm **for all** 1D SFTs.

Theorem

Entropy is computable for 1D SFTs, and possible value are logarithm of Perron numbers (special kind of algebraic number).

What about dimension 2?

- No good representation of SFTs with graph/matrix.
- Variational principle does not help: not necessarily a unique MME, hard to give explicit description of MME.
- Even for very simple examples of SFTs, no close formula in general.
- ...

What about dimension 2?

- No good representation of SFTs with graph/matrix.
- Variational principle does not help: not necessarily a unique MME, hard to give explicit description of MME.
- Even for very simple examples of SFTs, no close formula in general.
- ...

Indeed, computing the entropy is **very** hard:

Theorem

Entropy is non computable for 2D SFTs.

Where does uncomputability come from?

Theorem

Entropy is non computable for 2D SFTs.

$$h(X) = \lim_{n \to \infty} \frac{1}{n^d} \log \left(|\mathcal{L}_n| \right).$$

Where does uncomputability come from?

Theorem

Entropy is non computable for 2D SFTs.

$$h(X) = \lim_{n\to\infty} \frac{1}{n^d} \log\left(|\mathcal{L}_n|\right).$$

Where does uncomputability come from?

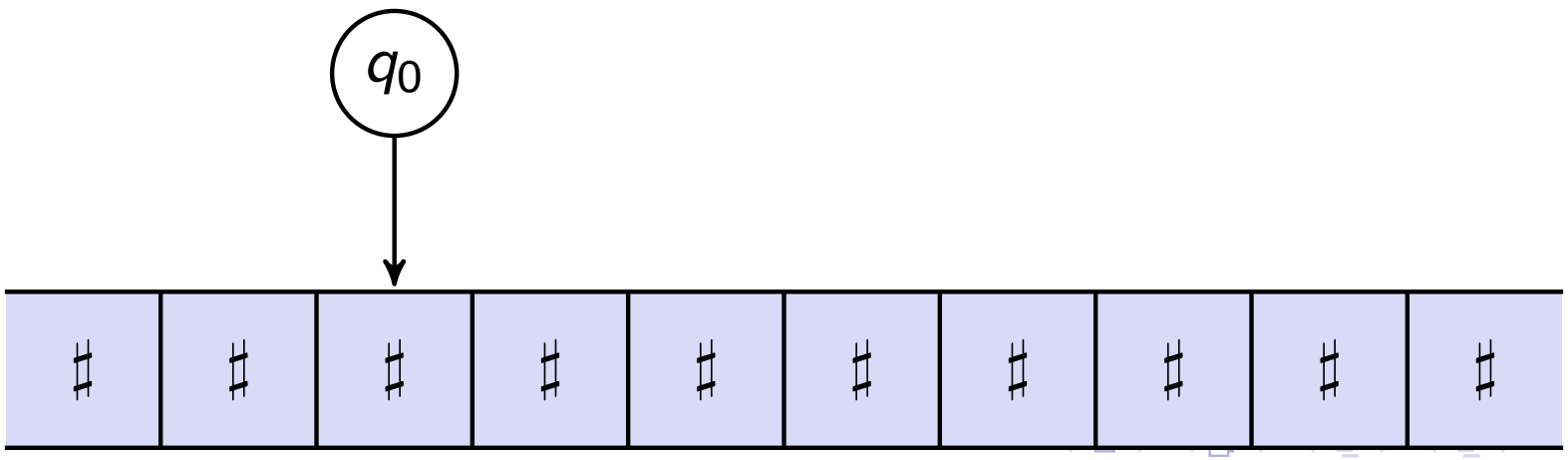
$\rightsquigarrow$ 2D SFTs can be used to encode Turing machines computations.

Turing machines (I)

Turing machines (I)

An example :

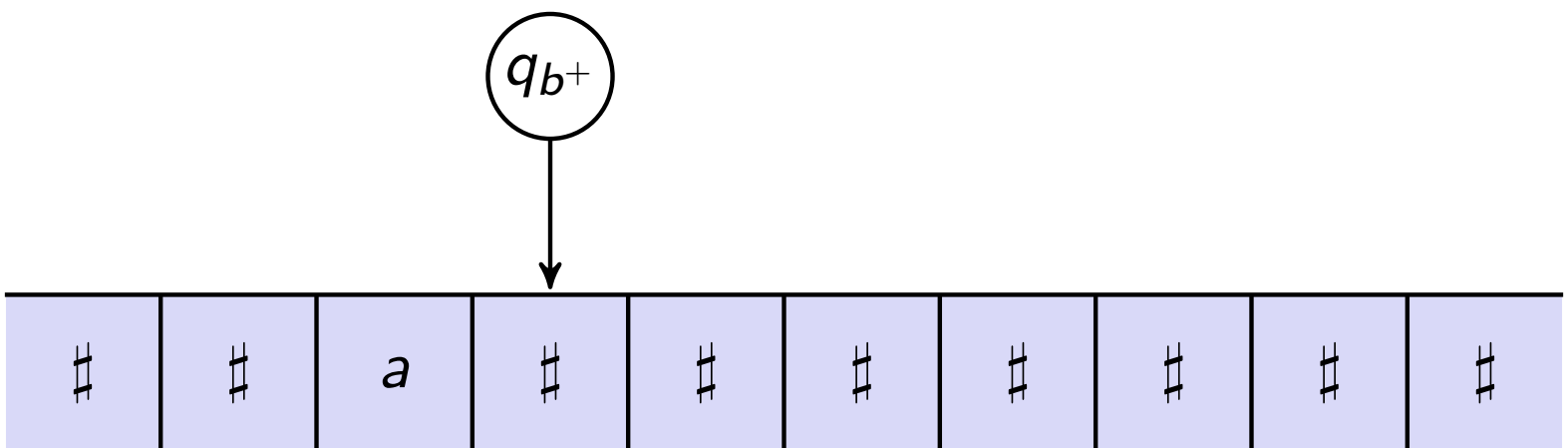
$\delta(q, x)$		Symbol x			
		a	b	$\|\|$	$\sharp$
State q	q_0	$\perp$	$\perp$	$\perp$	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	$\perp$	$(q_{b^{++}}, a, \rightarrow)$	$\perp$	$\perp$
	q_{b^+}	$\perp$	$\perp$	$\perp$	$(q_{\|\|}, b, \rightarrow)$
	$q_{b^{++}}$	$\perp$	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	$\perp$
	$q_{\|\|}$	$(q_{a^+}, a, \rightarrow)$	$(q_{\|\|}, b, \leftarrow)$	$(q_{\|\|}, \|\|, \leftarrow)$	$(q_{\|\|}, \|\|, \cdot)$



Turing machines (I)

An example :

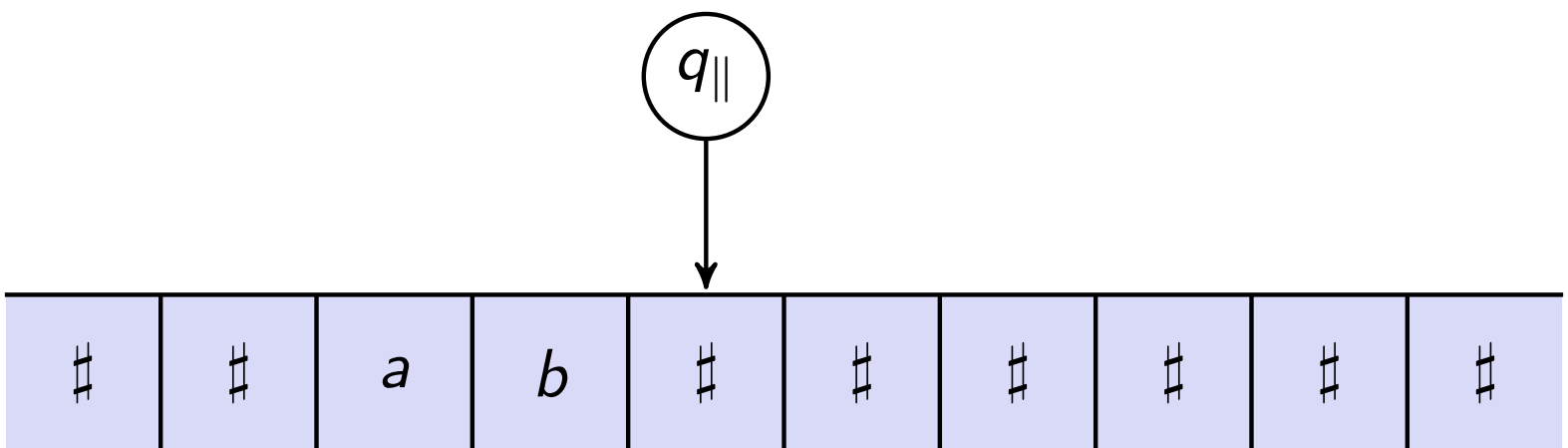
$\delta(q, x)$		Symbol x			
		a	b	$\|\|$	$\sharp$
State q	q_0	$\perp$	$\perp$	$\perp$	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	$\perp$	$(q_{b^{++}}, a, \rightarrow)$	$\perp$	$\perp$
	q_{b^+}	$\perp$	$\perp$	$\perp$	$(q_{\|\|}, b, \rightarrow)$
	$q_{b^{++}}$	$\perp$	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	$\perp$
	$q_{\|\|}$	$(q_{a^+}, a, \rightarrow)$	$(q_{\|\|}, b, \leftarrow)$	$(q_{\|\|}, \|\|, \leftarrow)$	$(q_{\|\|}, \|\|, \cdot)$



Turing machines (I)

An example :

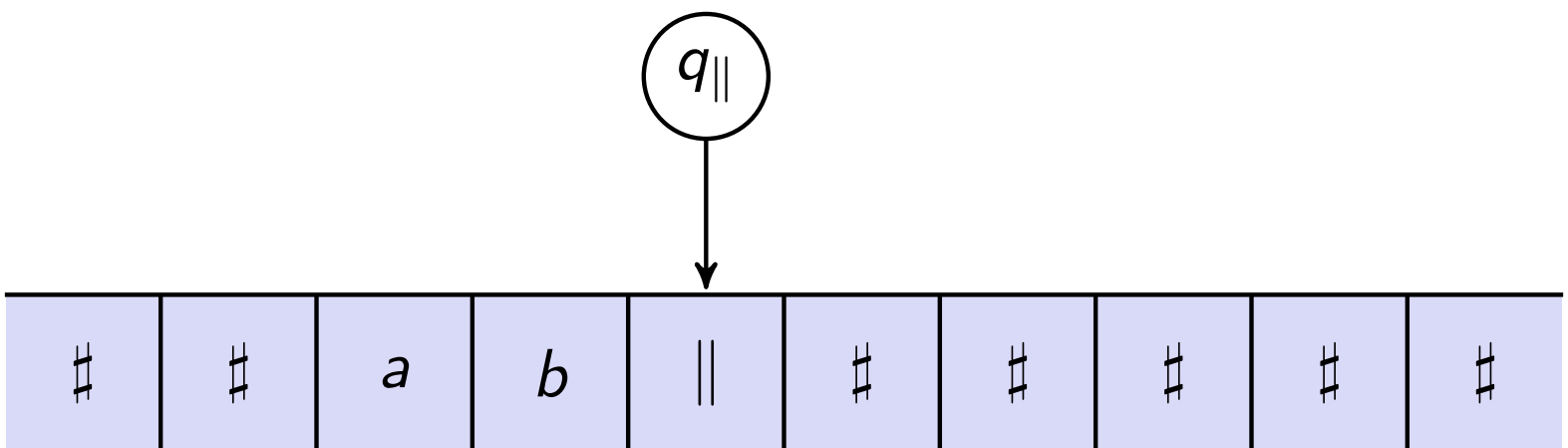
$\delta(q,x)$		Symbol x			
		a	b	$\|\|$	$\sharp$
State q	q_0	$\perp$	$\perp$	$\perp$	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	$\perp$	$(q_{b^{++}}, a, \rightarrow)$	$\perp$	$\perp$
	q_{b^+}	$\perp$	$\perp$	$\perp$	$(q_{\|\|}, b, \rightarrow)$
	$q_{b^{++}}$	$\perp$	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	$\perp$
	$q_{\|\|}$	$(q_{a^+}, a, \rightarrow)$	$(q_{\|\|}, b, \leftarrow)$	$(q_{\|\|}, \|\|, \leftarrow)$	$(q_{\|\|}, \|\|, \cdot)$



Turing machines (I)

An example :

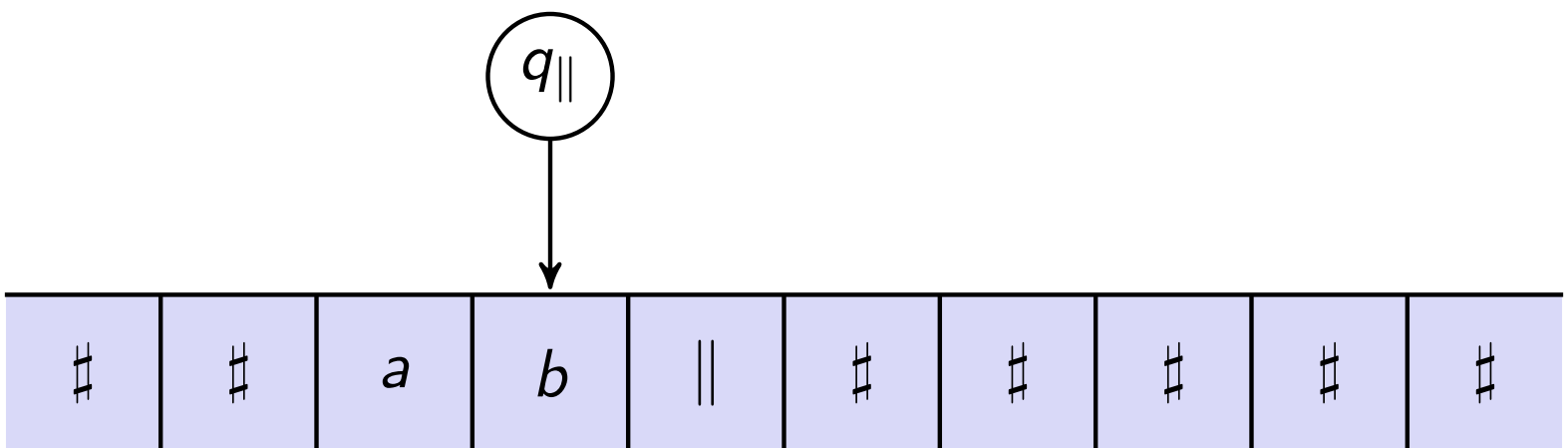
$\delta(q,x)$		Symbol x			
		a	b	$\|\|$	$\sharp$
State q	q_0	$\perp$	$\perp$	$\perp$	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	$\perp$	$(q_{b^{++}}, a, \rightarrow)$	$\perp$	$\perp$
	q_{b^+}	$\perp$	$\perp$	$\perp$	$(q_{\|\|}, b, \rightarrow)$
	$q_{b^{++}}$	$\perp$	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	$\perp$
	$q_{\|\|}$	$(q_{a^+}, a, \rightarrow)$	$(q_{\|\|}, b, \leftarrow)$	$(q_{\|\|}, \|\|, \leftarrow)$	$(q_{\|\|}, \|\|, \cdot)$



Turing machines (I)

An example :

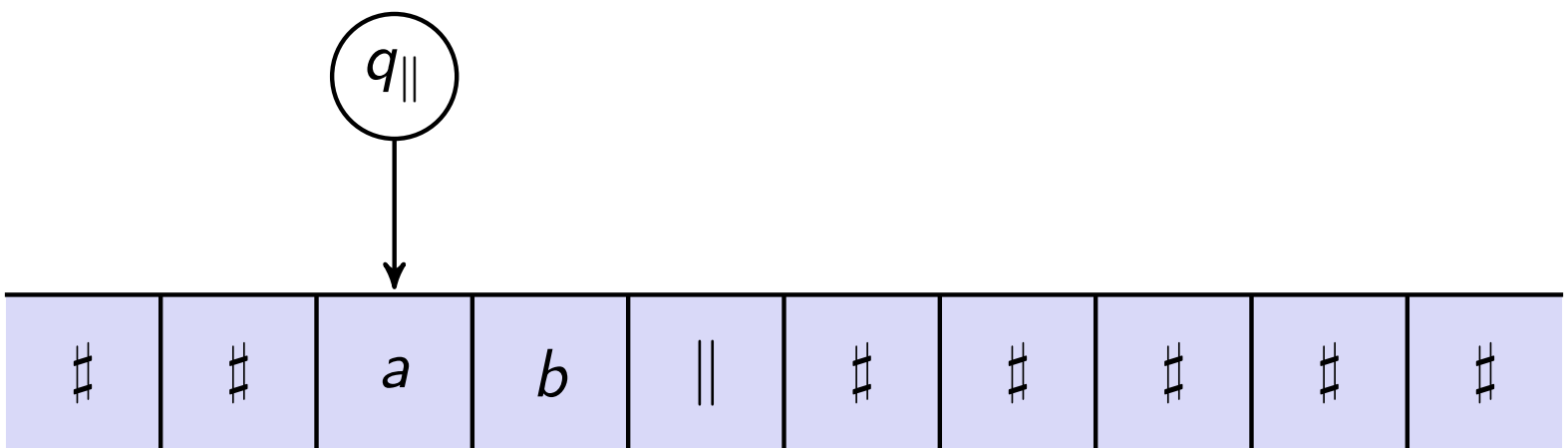
$\delta(q,x)$		Symbol x			
		a	b	$\|\|$	$\sharp$
State q	q_0	$\perp$	$\perp$	$\perp$	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	$\perp$	$(q_{b^{++}}, a, \rightarrow)$	$\perp$	$\perp$
	q_{b^+}	$\perp$	$\perp$	$\perp$	$(q_{\|\|}, b, \rightarrow)$
	$q_{b^{++}}$	$\perp$	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	$\perp$
	$q_{\|\|}$	$(q_{a^+}, a, \rightarrow)$	$(q_{\|\|}, b, \leftarrow)$	$(q_{\|\|}, \|\|, \leftarrow)$	$(q_{\|\|}, \|\|, \cdot)$



Turing machines (I)

An example :

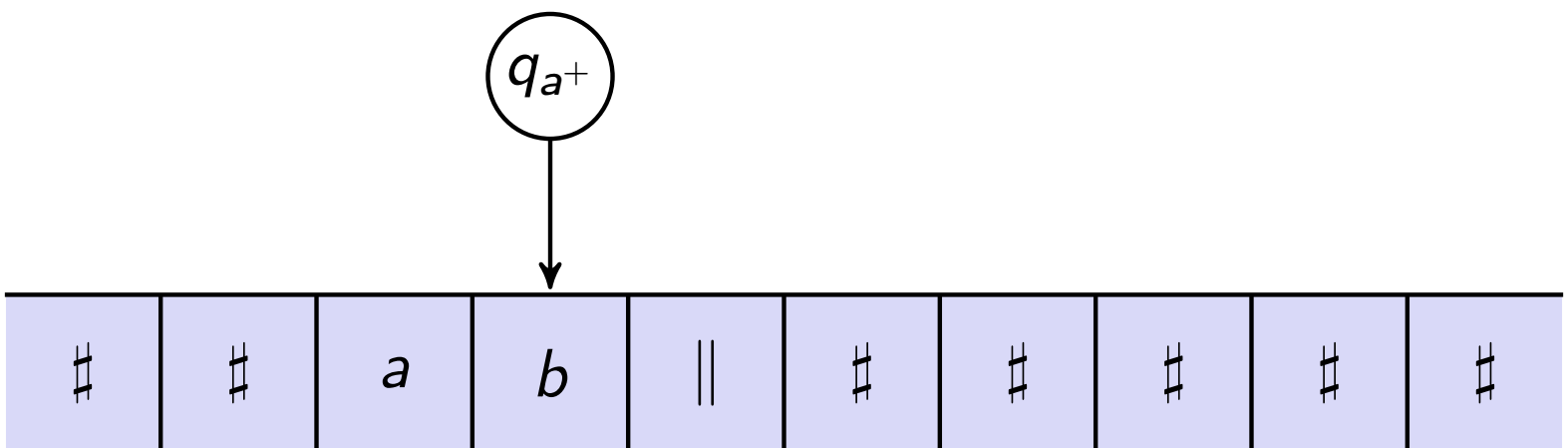
$\delta(q,x)$		Symbol x			
		a	b	$\|\|$	$\sharp$
State q	q_0	$\bot$	$\bot$	$\bot$	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	$\bot$	$(q_{b^{++}}, a, \rightarrow)$	$\bot$	$\bot$
	q_{b^+}	$\bot$	$\bot$	$\bot$	$(q_{\|\|}, b, \rightarrow)$
	$q_{b^{++}}$	$\bot$	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	$\bot$
	$q_{\|\|}$	$(q_{a^+}, a, \rightarrow)$	$(q_{\|\|}, b, \leftarrow)$	$(q_{\|\|}, \|\|, \leftarrow)$	$(q_{\|\|}, \|\|, \cdot)$



Turing machines (I)

An example :

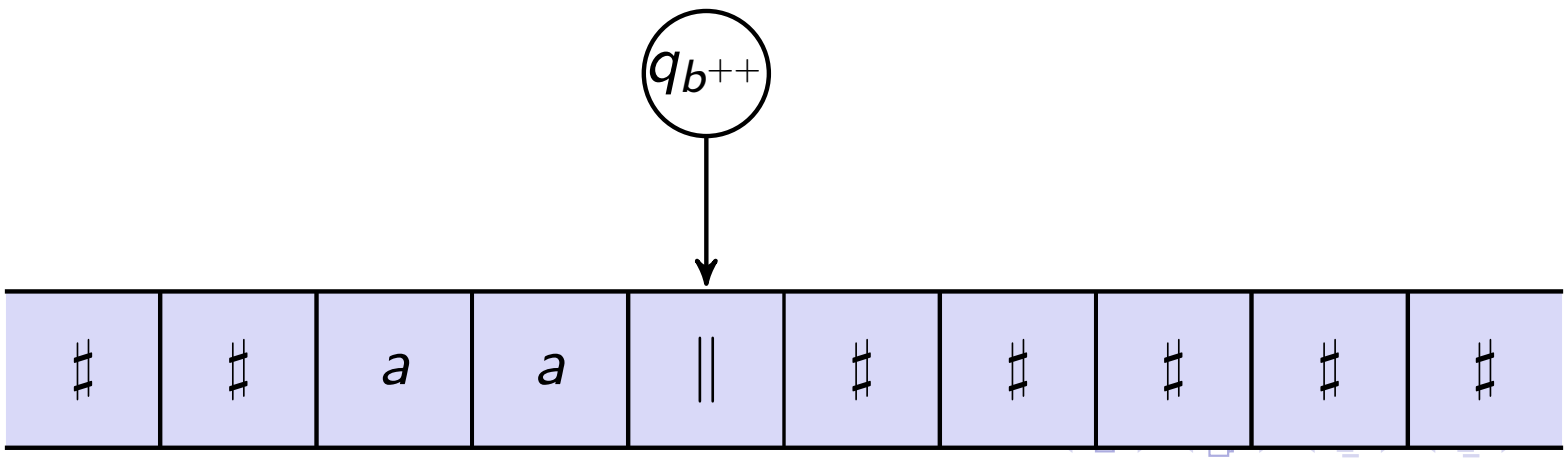
$\delta(q,x)$		Symbol x			
		a	b	$\|\|$	$\sharp$
State q	q_0	$\perp$	$\perp$	$\perp$	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	$\perp$	$(q_{b^{++}}, a, \rightarrow)$	$\perp$	$\perp$
	q_{b^+}	$\perp$	$\perp$	$\perp$	$(q_{\|\|}, b, \rightarrow)$
	$q_{b^{++}}$	$\perp$	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	$\perp$
	$q_{\|\|}$	$(q_{a^+}, a, \rightarrow)$	$(q_{\|\|}, b, \leftarrow)$	$(q_{\|\|}, \|\|, \leftarrow)$	$(q_{\|\|}, \|\|, \cdot)$



Turing machines (I)

An example :

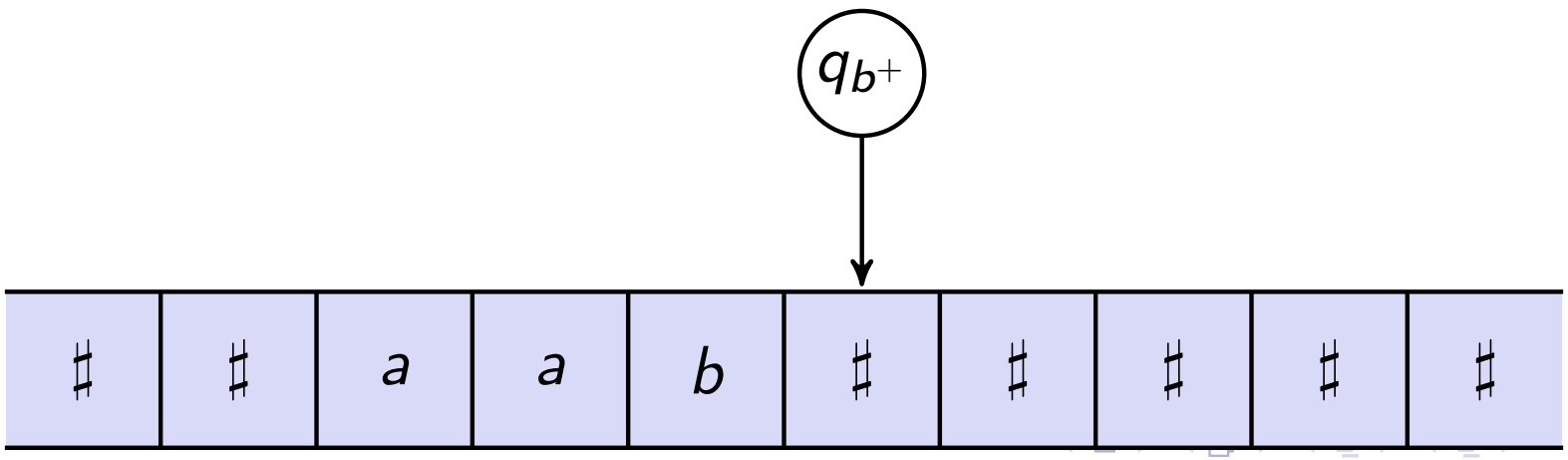
$\delta(q,x)$		Symbol x			
		a	b	$\|\|$	$\sharp$
State q	q_0	$\perp$	$\perp$	$\perp$	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	$\perp$	$(q_{b^{++}}, a, \rightarrow)$	$\perp$	$\perp$
	q_{b^+}	$\perp$	$\perp$	$\perp$	$(q_{\|\|}, b, \rightarrow)$
	$q_{b^{++}}$	$\perp$	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	$\perp$
	$q_{\|\|}$	$(q_{a^+}, a, \rightarrow)$	$(q_{\|\|}, b, \leftarrow)$	$(q_{\|\|}, \|\|, \leftarrow)$	$(q_{\|\|}, \|\|, \cdot)$



Turing machines (I)

An example :

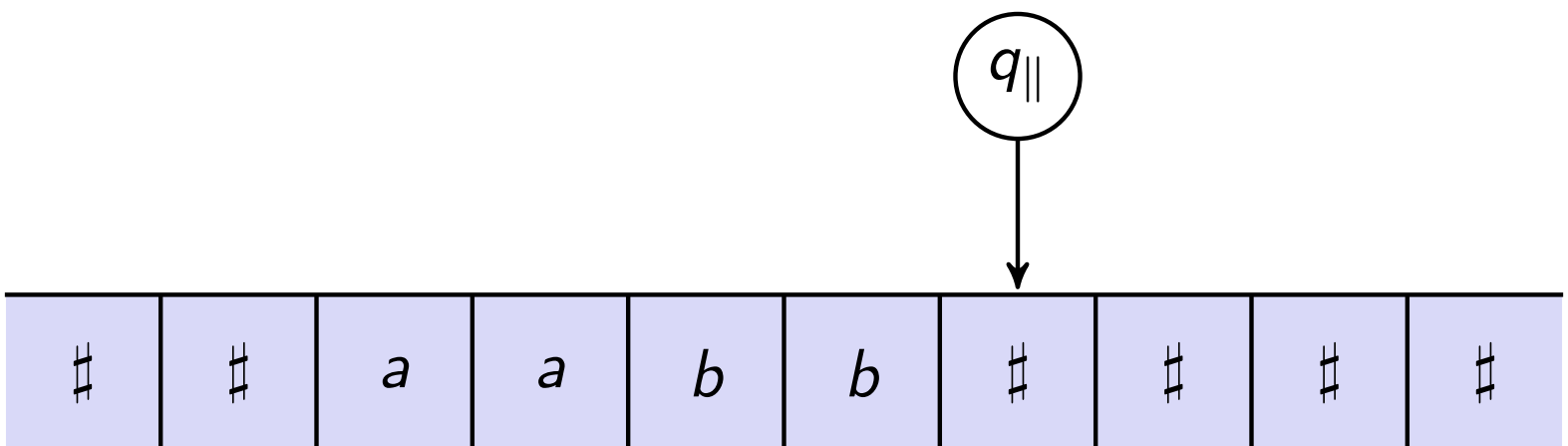
$\delta(q,x)$		Symbol x			
		a	b	$\|\|$	$\sharp$
State q	q_0	$\perp$	$\perp$	$\perp$	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	$\perp$	$(q_{b^{++}}, a, \rightarrow)$	$\perp$	$\perp$
	q_{b^+}	$\perp$	$\perp$	$\perp$	$(q_{\|\|}, b, \rightarrow)$
	$q_{b^{++}}$	$\perp$	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	$\perp$
	$q_{\|\|}$	$(q_{a^+}, a, \rightarrow)$	$(q_{\|\|}, b, \leftarrow)$	$(q_{\|\|}, \|\|, \leftarrow)$	$(q_{\|\|}, \|\|, \cdot)$



Turing machines (I)

An example :

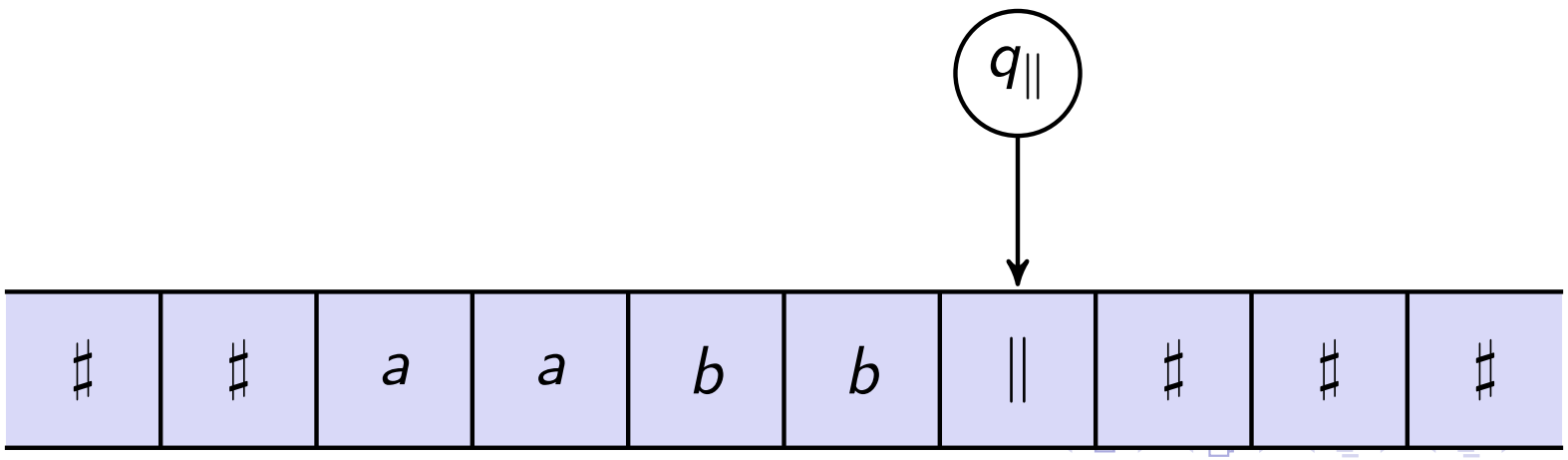
$\delta(q,x)$		Symbol x			
		a	b	$\|\|$	$\sharp$
State q	q_0	$\perp$	$\perp$	$\perp$	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	$\perp$	$(q_{b^{++}}, a, \rightarrow)$	$\perp$	$\perp$
	q_{b^+}	$\perp$	$\perp$	$\perp$	$(q_{\|\|}, b, \rightarrow)$
	$q_{b^{++}}$	$\perp$	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	$\perp$
	$q_{\|\|}$	$(q_{a^+}, a, \rightarrow)$	$(q_{\|\|}, b, \leftarrow)$	$(q_{\|\|}, \|\|, \leftarrow)$	$(q_{\|\|}, \|\|, \cdot)$



Turing machines (I)

An example :

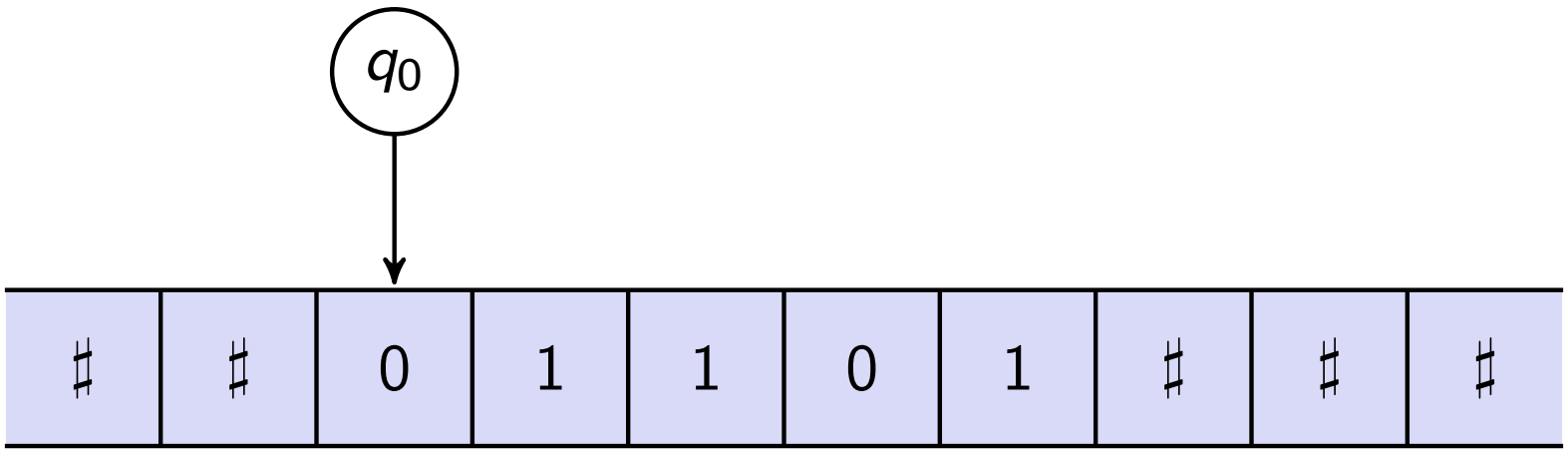
$\delta(q,x)$		Symbol x			
		a	b	$\|\|$	$\sharp$
State q	q_0	$\perp$	$\perp$	$\perp$	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	$\perp$	$(q_{b^{++}}, a, \rightarrow)$	$\perp$	$\perp$
	q_{b^+}	$\perp$	$\perp$	$\perp$	$(q_{\|\|}, b, \rightarrow)$
	$q_{b^{++}}$	$\perp$	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	$\perp$
	$q_{\|\|}$	$(q_{a^+}, a, \rightarrow)$	$(q_{\|\|}, b, \leftarrow)$	$(q_{\|\|}, \|\|, \leftarrow)$	$(q_{\|\|}, \|\|, \cdot)$



Turing machines (II)

Another example :

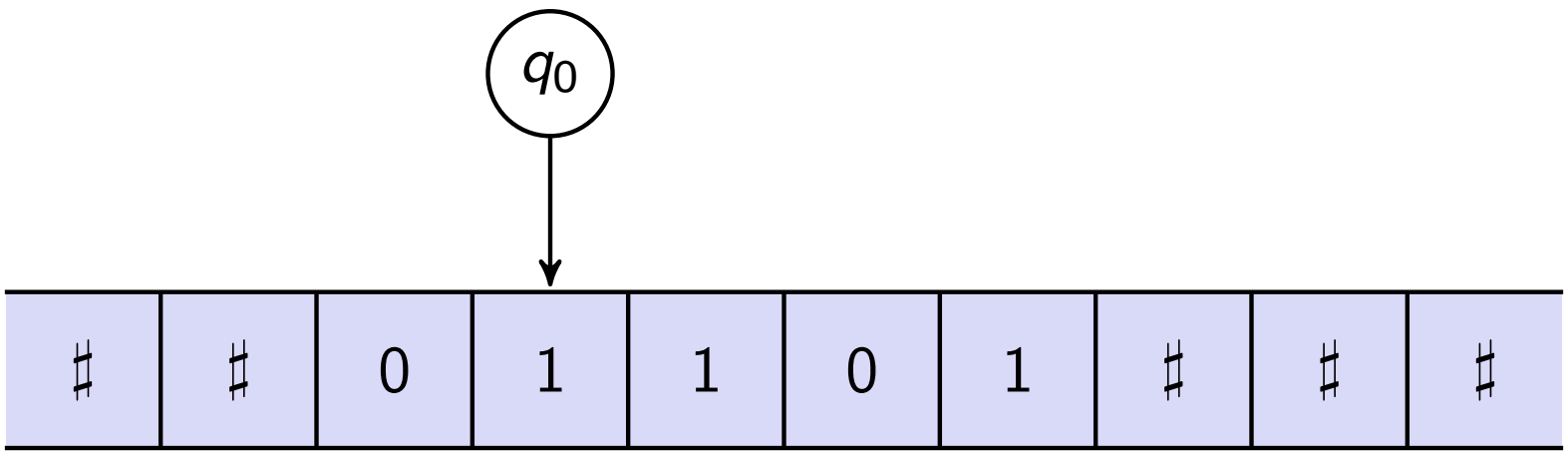
$\delta(q,x)$		Symbol x		
		0	1	♯
State q	q_0	$(q_0, 0, \rightarrow)$	$(q_0, 1, \rightarrow)$	$(q_+, \sharp, \leftarrow)$
	q_+	$(q_f, 1, .)$	$(q_+, 0, \leftarrow)$	$(q_f, 1, .)$
	q_f	$(q_f, 0, .)$	$(q_f, 1, .)$	$(q_f, \sharp, .)$



Turing machines (II)

Another example :

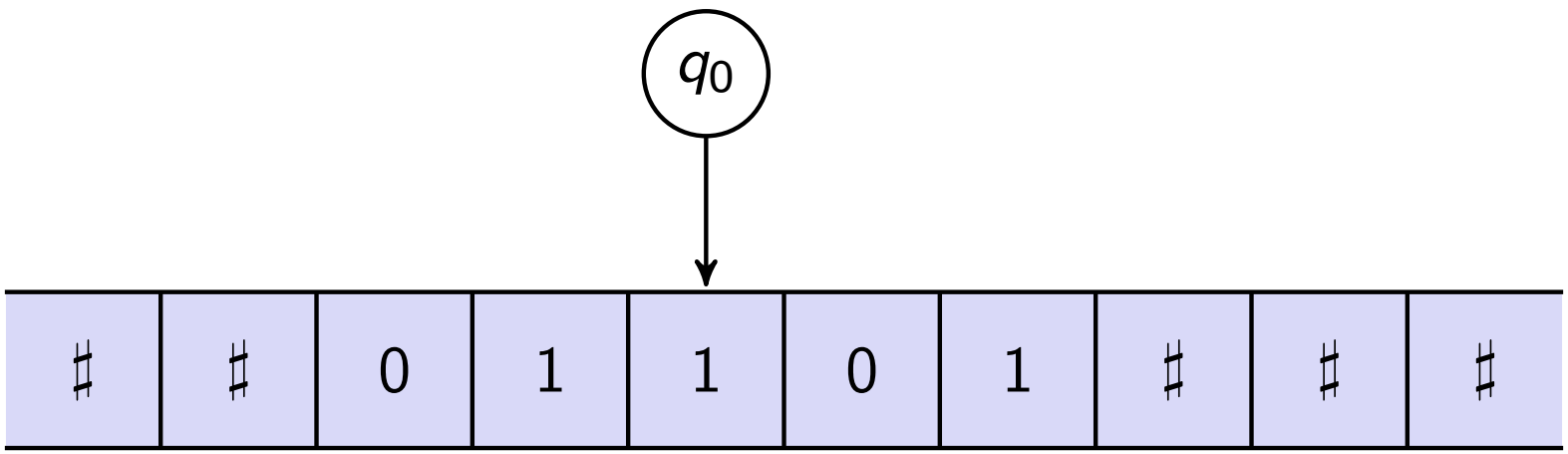
$\delta(q,x)$		Symbol x		
		0	1	♯
State q	q_0	$(q_0, 0, \rightarrow)$	$(q_0, 1, \rightarrow)$	$(q_+, \sharp, \leftarrow)$
	q_+	$(q_f, 1, .)$	$(q_+, 0, \leftarrow)$	$(q_f, 1, .)$
	q_f	$(q_f, 0, .)$	$(q_f, 1, .)$	$(q_f, \sharp, .)$



Turing machines (II)

Another example :

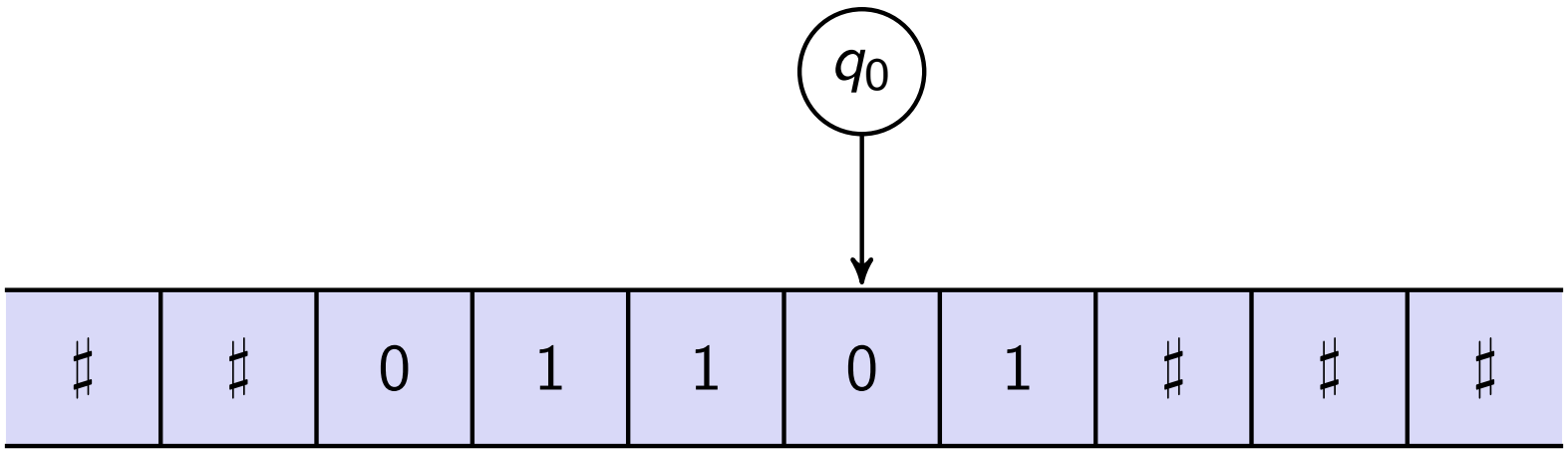
$\delta(q,x)$		Symbol x		
		0	1	♯
State q	q_0	$(q_0, 0, \rightarrow)$	$(q_0, 1, \rightarrow)$	$(q_+, \sharp, \leftarrow)$
	q_+	$(q_f, 1, .)$	$(q_+, 0, \leftarrow)$	$(q_f, 1, .)$
	q_f	$(q_f, 0, .)$	$(q_f, 1, .)$	$(q_f, \sharp, .)$



Turing machines (II)

Another example :

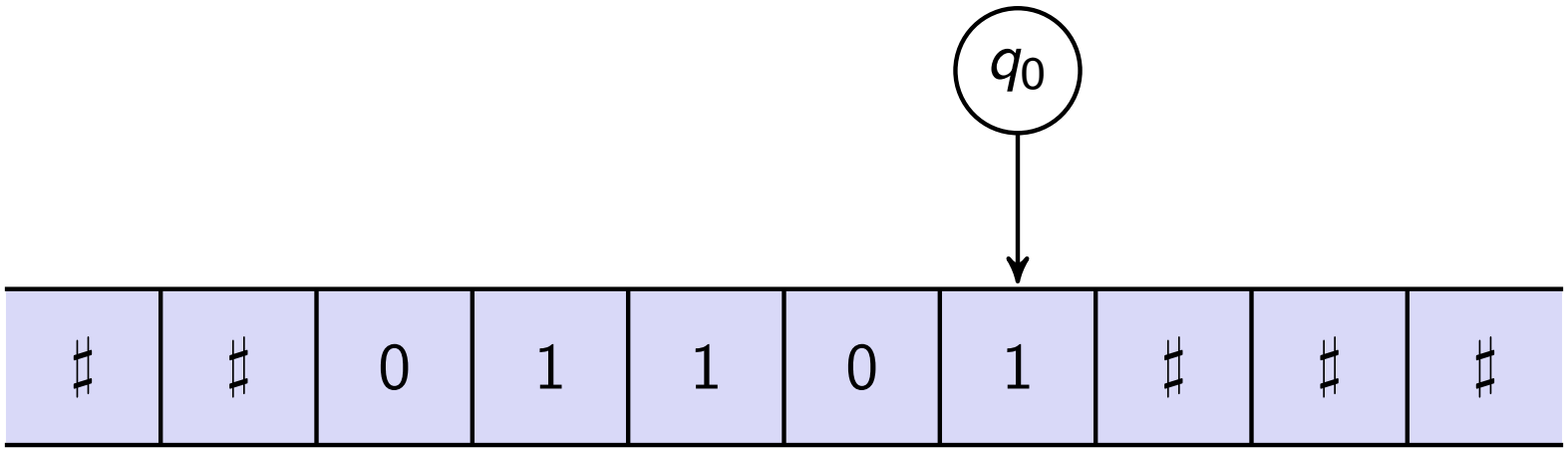
$\delta(q,x)$		Symbol x		
		0	1	♯
State q	q_0	$(q_0, 0, \rightarrow)$	$(q_0, 1, \rightarrow)$	$(q_+, \sharp, \leftarrow)$
	q_+	$(q_f, 1, .)$	$(q_+, 0, \leftarrow)$	$(q_f, 1, .)$
	q_f	$(q_f, 0, .)$	$(q_f, 1, .)$	$(q_f, \sharp, .)$



Turing machines (II)

Another example :

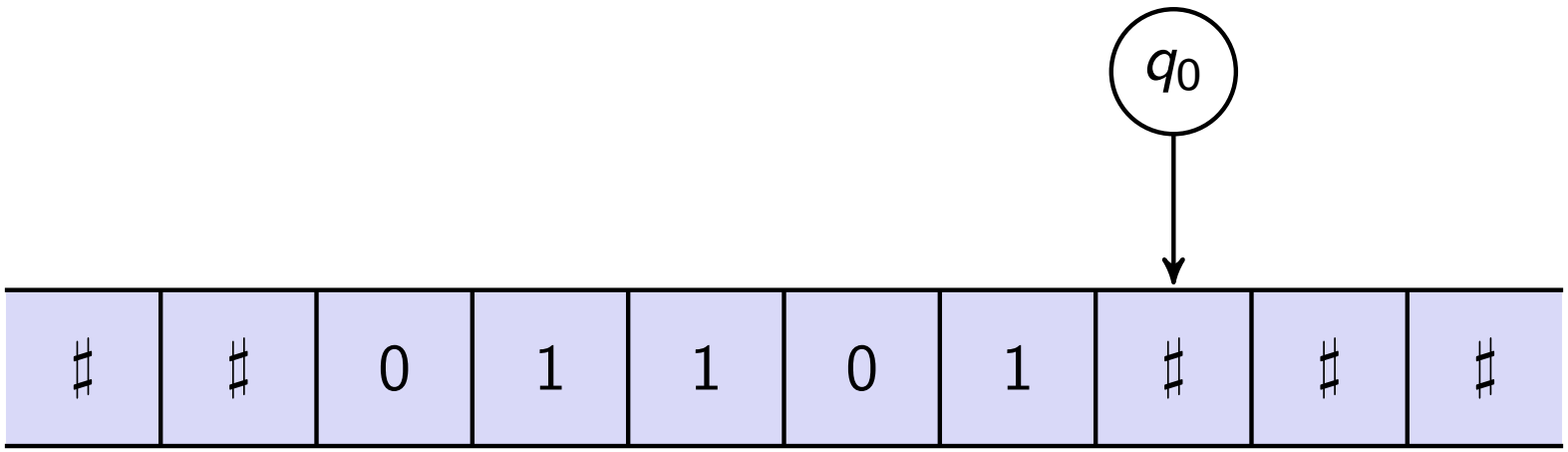
$\delta(q,x)$		Symbol x		
		0	1	♯
State q	q_0	$(q_0, 0, \rightarrow)$	$(q_0, 1, \rightarrow)$	$(q_+, \sharp, \leftarrow)$
	q_+	$(q_f, 1, .)$	$(q_+, 0, \leftarrow)$	$(q_f, 1, .)$
	q_f	$(q_f, 0, .)$	$(q_f, 1, .)$	$(q_f, \sharp, .)$



Turing machines (II)

Another example :

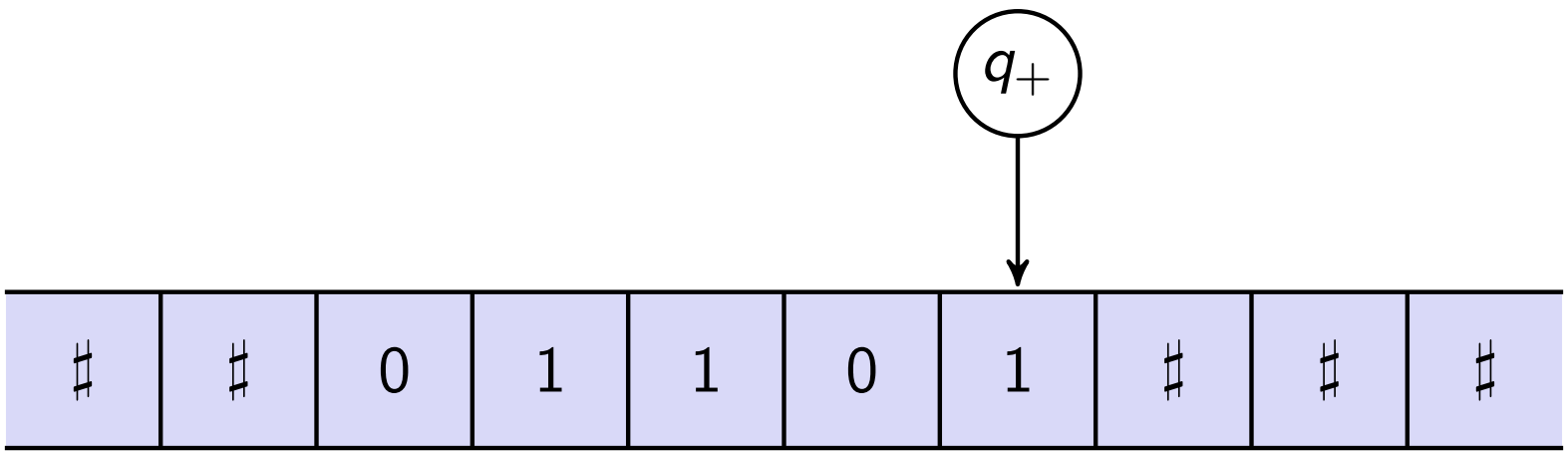
$\delta(q,x)$		Symbol x		
		0	1	♯
State q	q_0	$(q_0, 0, \rightarrow)$	$(q_0, 1, \rightarrow)$	$(q_+, \sharp, \leftarrow)$
	q_+	$(q_f, 1, .)$	$(q_+, 0, \leftarrow)$	$(q_f, 1, .)$
	q_f	$(q_f, 0, .)$	$(q_f, 1, .)$	$(q_f, \sharp, .)$



Turing machines (II)

Another example :

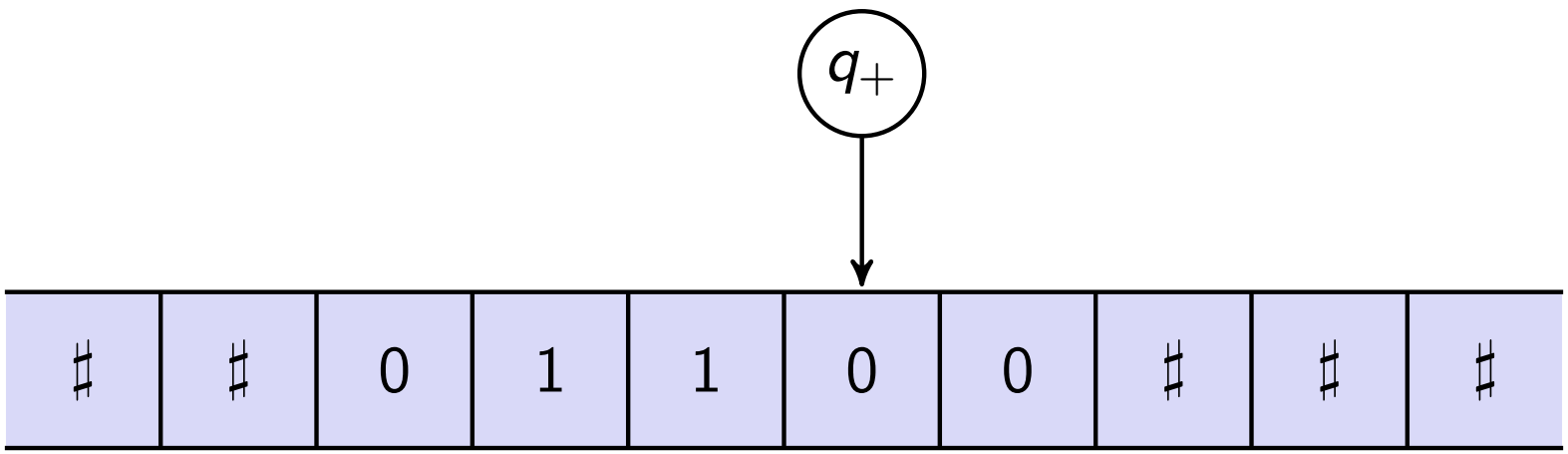
$\delta(q,x)$		Symbol x		
		0	1	♯
State q	q_0	$(q_0, 0, \rightarrow)$	$(q_0, 1, \rightarrow)$	$(q_+, \sharp, \leftarrow)$
	q_+	$(q_f, 1, .)$	$(q_+, 0, \leftarrow)$	$(q_f, 1, .)$
	q_f	$(q_f, 0, .)$	$(q_f, 1, .)$	$(q_f, \sharp, .)$



Turing machines (II)

Another example :

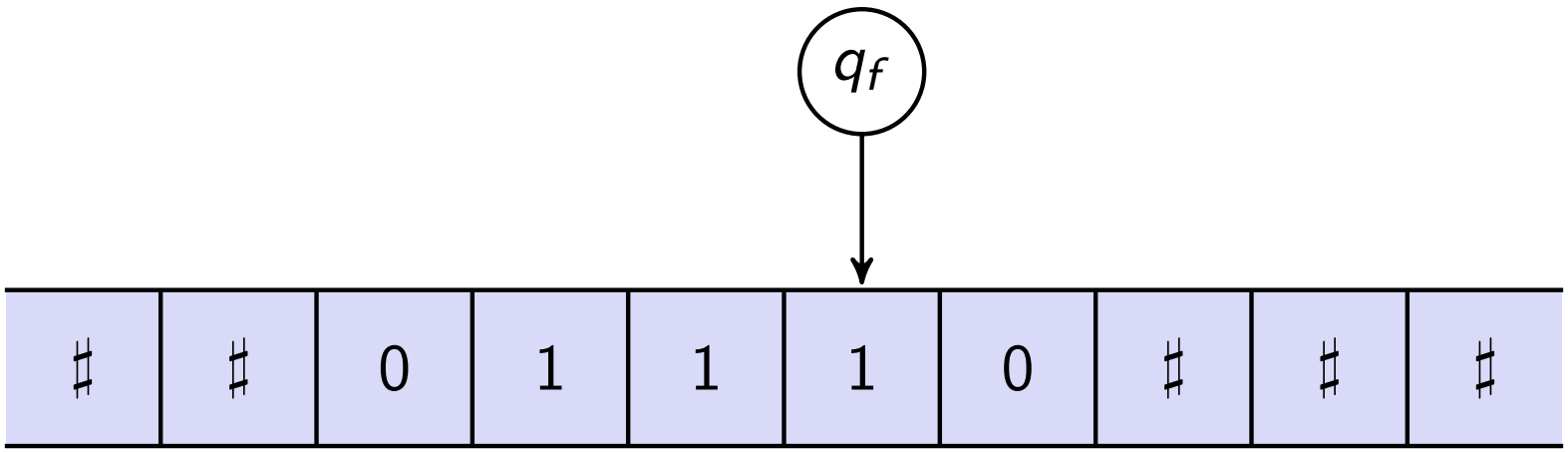
$\delta(q,x)$		Symbol x		
		0	1	♯
State q	q_0	$(q_0, 0, \rightarrow)$	$(q_0, 1, \rightarrow)$	$(q_+, \sharp, \leftarrow)$
	q_+	$(q_f, 1, .)$	$(q_+, 0, \leftarrow)$	$(q_f, 1, .)$
	q_f	$(q_f, 0, .)$	$(q_f, 1, .)$	$(q_f, \sharp, .)$



Turing machines (II)

Another example :

$\delta(q,x)$		Symbol x		
		0	1	♯
State q	q_0	$(q_0, 0, \rightarrow)$	$(q_0, 1, \rightarrow)$	$(q_+, \sharp, \leftarrow)$
	q_+	$(q_f, 1, .)$	$(q_+, 0, \leftarrow)$	$(q_f, 1, .)$
	q_f	$(q_f, 0, .)$	$(q_f, 1, .)$	$(q_f, \sharp, .)$



Turing machines (II)

Another example :

$\delta(q,x)$		Symbol x		
		0	1	♯
State q	q_0	$(q_0, 0, \rightarrow)$	$(q_0, 1, \rightarrow)$	$(q_+, \sharp, \leftarrow)$
	q_+	$(q_f, 1, .)$	$(q_+, 0, \leftarrow)$	$(q_f, 1, .)$
	q_f	$(q_f, 0, .)$	$(q_f, 1, .)$	$(q_f, \sharp, .)$

- State q_f is a special state, called **final state**.
- If there are only finitely many 0's and 1's on the tape around the computation head, the machine adds 1 to the binary number written on the tape, reaches state q_f and stops.
- Otherwise, the machine never halts and computes forever.

Computing with Turing machines

Church–Turing thesis

Functions that can be computed by an algorithm/computer (with no limitation of space nor time) are exactly functions that can be computed by Turing machines.

Computing with Turing machines

Church–Turing thesis

Functions that can be computed by an algorithm/computer (with no limitation of space nor time) are exactly functions that can be computed by Turing machines.

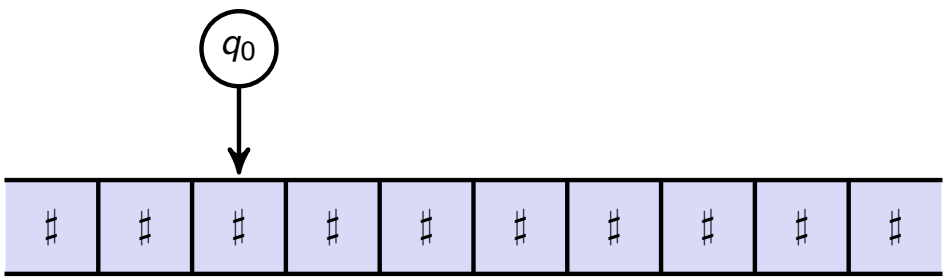
Theorem (Turing, 1936)

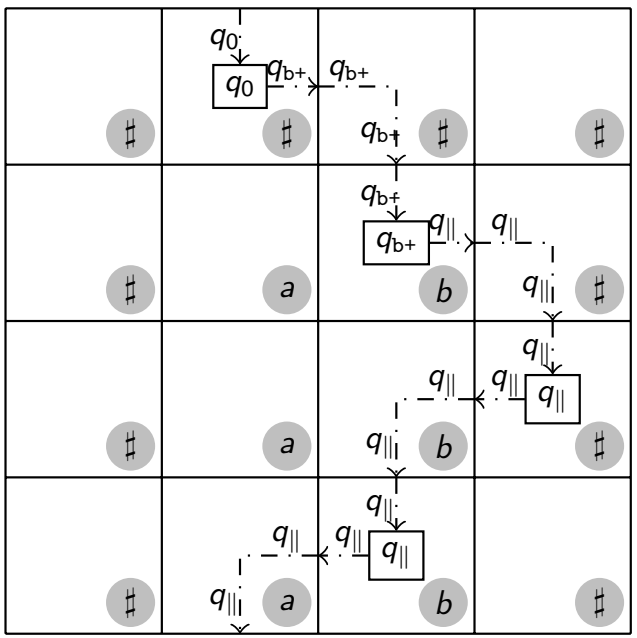
The halting problem is not computable.

There is no program/algorithm/Turing machine that can determine whether a program/algorithm/Turing machine will stop or keep computing forever.

Turing machines inside 2D SFTs (I)

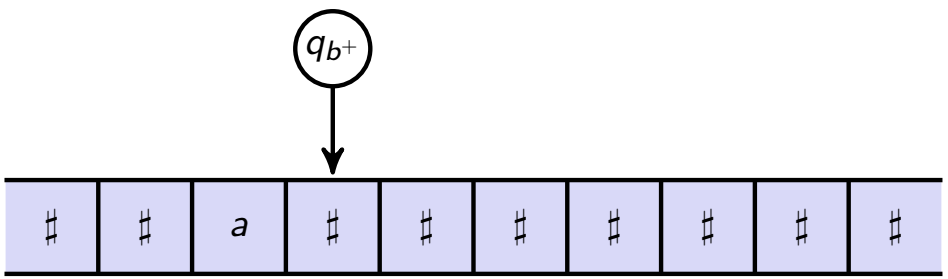
$\delta(q,x)$		Symbol x			
		a	b	$\|\|$	$\sharp$
State q	q_0	$\perp$	$\perp$	$\perp$	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	$\perp$	$(q_{b^{++}}, a, \rightarrow)$	$\perp$	$\perp$
	q_{b^+}	$\perp$	$\perp$	$\perp$	$(q_{\|\|}, b, \rightarrow)$
	$q_{b^{++}}$	$\perp$	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	$\perp$
	$q_{\|\|}$	$(q_{a^+}, a, \rightarrow)$	$(q_{\|\|}, b, \leftarrow)$	$(q_{\|\|}, \|\|, \leftarrow)$	$(q_{\|\|}, \|\|, \cdot)$

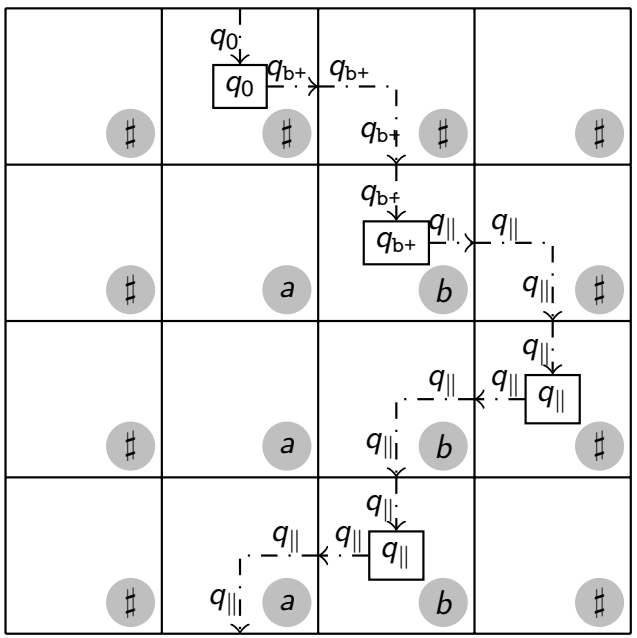




Turing machines inside 2D SFTs (I)

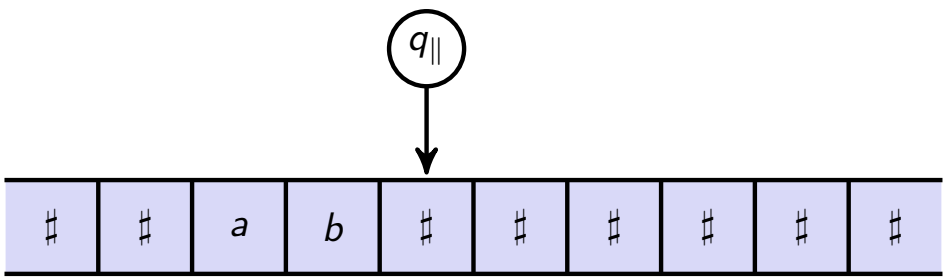
$\delta(q,x)$		Symbol x			
		a	b	$\|\|$	$\sharp$
State q	q_0	$\perp$	$\perp$	$\perp$	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	$\perp$	$(q_{b^{++}}, a, \rightarrow)$	$\perp$	$\perp$
	q_{b^+}	$\perp$	$\perp$	$\perp$	$(q_{\|\|}, b, \rightarrow)$
	$q_{b^{++}}$	$\perp$	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	$\perp$
	$q_{\|\|}$	$(q_{a^+}, a, \rightarrow)$	$(q_{\|\|}, b, \leftarrow)$	$(q_{\|\|}, \|\|, \leftarrow)$	$(q_{\|\|}, \|\|, \cdot)$

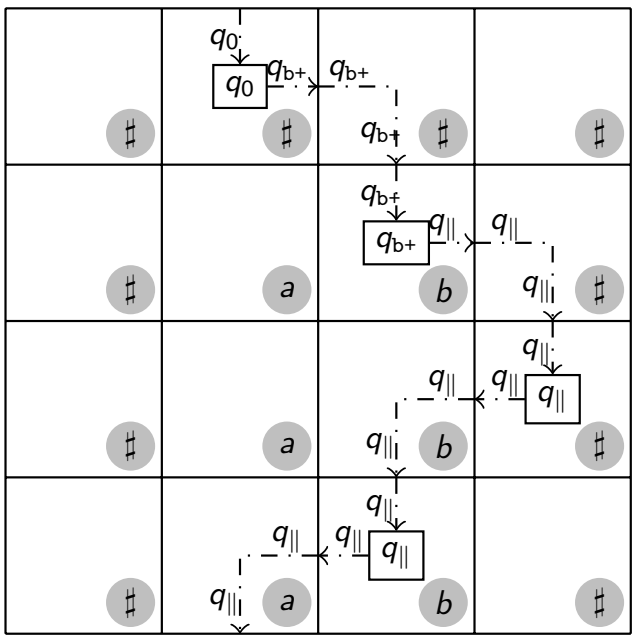




Turing machines inside 2D SFTs (I)

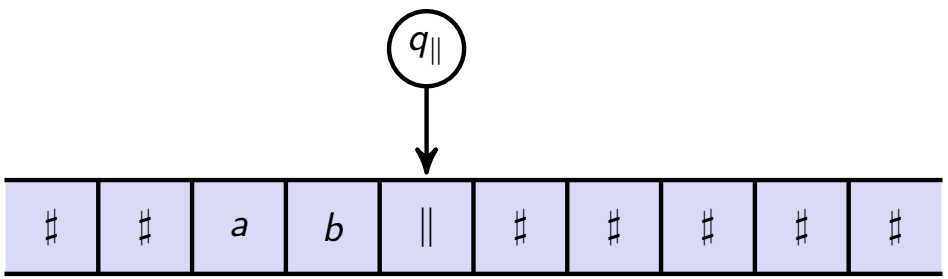
$\delta(q,x)$		Symbol x			
		a	b	$\|\|$	$\sharp$
State q	q_0	$\perp$	$\perp$	$\perp$	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	$\perp$	$(q_{b^{++}}, a, \rightarrow)$	$\perp$	$\perp$
	q_{b^+}	$\perp$	$\perp$	$\perp$	$(q_{\|\|}, b, \rightarrow)$
	$q_{b^{++}}$	$\perp$	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	$\perp$
	$q_{\|\|}$	$(q_{a^+}, a, \rightarrow)$	$(q_{\|\|}, b, \leftarrow)$	$(q_{\|\|}, \|\|, \leftarrow)$	$(q_{\|\|}, \|\|, \cdot)$

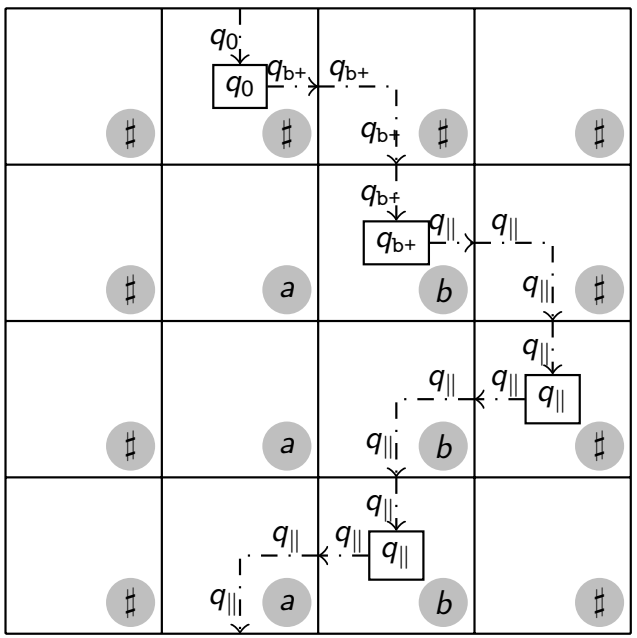




Turing machines inside 2D SFTs (I)

$\delta(q,x)$		Symbol x			
		a	b	$\|\|$	$\sharp$
State q	q_0	$\perp$	$\perp$	$\perp$	$(q_{b^+}, a, \rightarrow)$
	q_{a^+}	$\perp$	$(q_{b^{++}}, a, \rightarrow)$	$\perp$	$\perp$
	q_{b^+}	$\perp$	$\perp$	$\perp$	$(q_{\|\|}, b, \rightarrow)$
	$q_{b^{++}}$	$\perp$	$(q_{b^{++}}, b, \rightarrow)$	$(q_{b^+}, b, \rightarrow)$	$\perp$
	$q_{\|\|}$	$(q_{a^+}, a, \rightarrow)$	$(q_{\|\|}, b, \leftarrow)$	$(q_{\|\|}, \|\|, \leftarrow)$	$(q_{\|\|}, \|\|, \cdot)$





Turing machines inside 2D SFTs (II)

Given a program/algorithm/Turing machine, we can construct a 2D SFT that mimics the computations of the program/algorithm/Turing machine.

Turing machines inside 2D SFTs (II)

Given a program/algorithm/Turing machine, we can construct a 2D SFT that mimics the computations of the program/algorithm/Turing machine.

(technical stuff swept under the rug)

Turing machines inside 2D SFTs (II)

Given a program/algorithm/Turing machine, we can construct a 2D SFT that mimics the computations of the program/algorithm/Turing machine.

(technical stuff swept under the rug)

Consequences:

- No algorithm to compute the entropy of 2D SFTs (Hochman & Meyerovitch, 2010).
- Possible values for entropy of 2D SFTs are exactly right recursively enumerable numbers (Hochman & Meyerovitch, 2010).
- No algorithm to decide whether a 2D SFT is empty or not (Berger, 1964).
- ...

Conclusion

- No general algorithm that works for all 2D SFTs, but it exists for some subclasses (irreducible SFTs) and some techniques are known for particular examples (transfer matrix method, corner transfer matrix method,. . .)
- Entropy of 2D SFTs: example of question solved with tools from theoretical computer science.

Conclusion

- No general algorithm that works for all 2D SFTs, but it exists for some subclasses (irreducible SFTs) and some techniques are known for particular examples (transfer matrix method, corner transfer matrix method,...)
- Entropy of 2D SFTs: example of question solved with tools from theoretical computer science.

Thank you for your attention !!