Theory of Algorithms V

Introduction to Computational Complexity

Guoqiang Li

School of Software, Shanghai Jiao Tong University

A Concrete Turing Machine

Alan Turing

Alan Turing (23 Jun. 1912-7 Jun. 1954), an English student of Church, introduced a machine model for effective calculation in

"On Computable Numbers, with an Application to the Entsheidungs problem",

Proc. of the London Mathematical Society, **42**:230-265, 1936.

Turing Machine, Halting Problem, Turing Test

British Prime Minister Gordon Brown:

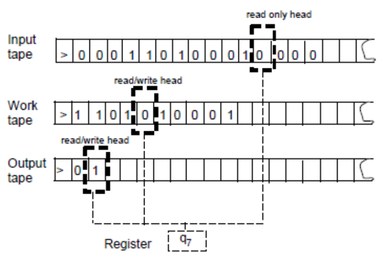
“…I am pleased to have the chance to say how deeply sorry I and we all are for what happened to him ... So on behalf of the British government, and all those who live freely thanks to Alan’s work, I am very proud to say: we’re sorry, you deserved so much better.”

Turing Machine

A k-tape Turing Machine $\mathbb{M}$ has k-tapes such that

- The first tape is the read-only input tape.
- The other $k-1$ tapes are the read/write work tapes.
- The k-th tape is also used as the output tape.

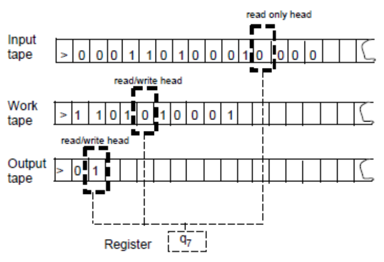
Every tape comes with a read/write head.



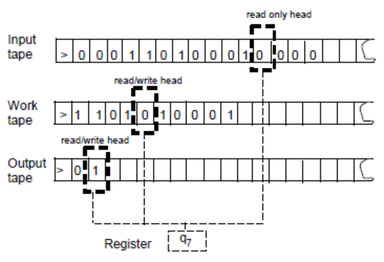
Turing Machine

The machine is described by a tuple (Γ, Q, δ) containing

- A finite set Γ, called alphabet, of symbols. It contains a blank symbol $\square$, a start symbol $\triangleright$, and the digits 0 and 1.
- A finite set Q of states. It contains a start state q_{s} and a halting state q_{h}.
- A transition function $\delta : Q \times \Gamma^k \to Q \times \Gamma^{k-1} \times \{\leftarrow, -, \rightarrow\}^k$, describing the rules of each computation step.



Computation and Configuration



Configuration, initial configuration, final configuration, computation step

An Example

$Q = \{q_s, q_h, q_1, q_2, q_3\}$, $\Gamma = \{0, 1, \square, \triangleright\}$,
and δ is as follows:

$p \in Q$	$\sigma \in \Gamma$	$\delta(p, \sigma)$
q_s	0	$(q_s, 0, \rightarrow)$
q_s	1	$(q_s, 1, \rightarrow)$
q_s	$\square$	$(q_1, \square, \leftarrow)$
q_s	$\triangleright$	$(q_s, \triangleright, \rightarrow)$
q_1	0	$(q_2, \square, \rightarrow)$
q_1	1	$(q_3, \square, \rightarrow)$
q_1	$\square$	$(q_1, \square, -)$
q_1	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_2	0	$(q_s, 0, \leftarrow)$
q_2	1	$(q_s, 0, \leftarrow)$
q_2	$\square$	$(q_s, 0, \leftarrow)$
q_2	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_3	0	$(q_s, 1, \leftarrow)$
q_3	1	$(q_s, 1, \leftarrow)$
q_3	$\square$	$(q_s, 1, \leftarrow)$
q_3	$\triangleright$	$(q_h, \triangleright, \rightarrow)$

An Example

$Q = \{q_s, q_h, q_1, q_2, q_3\}$, $\Gamma = \{0, 1, \square, \triangleright\}$,
and δ is as follows:

$p \in Q$	$\sigma \in \Gamma$	$\delta(p, \sigma)$
q_s	0	$(q_s, 0, \rightarrow)$
q_s	1	$(q_s, 1, \rightarrow)$
q_s	$\square$	$(q_1, \square, \leftarrow)$
q_s	$\triangleright$	$(q_s, \triangleright, \rightarrow)$
q_1	0	$(q_2, \square, \rightarrow)$
q_1	1	$(q_3, \square, \rightarrow)$
q_1	$\square$	$(q_1, \square, -)$
q_1	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_2	0	$(q_s, 0, \leftarrow)$
q_2	1	$(q_s, 0, \leftarrow)$
q_2	$\square$	$(q_s, 0, \leftarrow)$
q_2	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_3	0	$(q_s, 1, \leftarrow)$
q_3	1	$(q_s, 1, \leftarrow)$
q_3	$\square$	$(q_s, 1, \leftarrow)$
q_3	$\triangleright$	$(q_h, \triangleright, \rightarrow)$

Start the machine with input 010

An Example

$Q = \{q_s, q_h, q_1, q_2, q_3\}$, $\Gamma = \{0, 1, \Box, \triangleright\}$,
and δ is as follows:

$p \in Q$	$\sigma \in \Gamma$	$\delta(p, \sigma)$
q_s	0	$(q_s, 0, \rightarrow)$
q_s	1	$(q_s, 1, \rightarrow)$
q_s	$\Box$	$(q_1, \Box, \leftarrow)$
q_s	$\triangleright$	$(q_s, \triangleright, \rightarrow)$
q_1	0	$(q_2, \Box, \rightarrow)$
q_1	1	$(q_3, \Box, \rightarrow)$
q_1	$\Box$	$(q_1, \Box, -)$
q_1	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_2	0	$(q_s, 0, \leftarrow)$
q_2	1	$(q_s, 0, \leftarrow)$
q_2	$\Box$	$(q_s, 0, \leftarrow)$
q_2	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_3	0	$(q_s, 1, \leftarrow)$
q_3	1	$(q_s, 1, \leftarrow)$
q_3	$\Box$	$(q_s, 1, \leftarrow)$
q_3	$\triangleright$	$(q_h, \triangleright, \rightarrow)$

Start the machine with input 010

- q_s, $\underline{\triangleright}010$

An Example

$Q = \{q_s, q_h, q_1, q_2, q_3\}$, $\Gamma = \{0, 1, \Box, \triangleright\}$,
and δ is as follows:

$p \in Q$	$\sigma \in \Gamma$	$\delta(p, \sigma)$
q_s	0	$(q_s, 0, \rightarrow)$
q_s	1	$(q_s, 1, \rightarrow)$
q_s	$\Box$	$(q_1, \Box, \leftarrow)$
q_s	$\triangleright$	$(q_s, \triangleright, \rightarrow)$
q_1	0	$(q_2, \Box, \rightarrow)$
q_1	1	$(q_3, \Box, \rightarrow)$
q_1	$\Box$	$(q_1, \Box, -)$
q_1	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_2	0	$(q_s, 0, \leftarrow)$
q_2	1	$(q_s, 0, \leftarrow)$
q_2	$\Box$	$(q_s, 0, \leftarrow)$
q_2	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_3	0	$(q_s, 1, \leftarrow)$
q_3	1	$(q_s, 1, \leftarrow)$
q_3	$\Box$	$(q_s, 1, \leftarrow)$
q_3	$\triangleright$	$(q_h, \triangleright, \rightarrow)$

Start the machine with input 010

- $q_s, \underline{\triangleright}010$
- $q_s, \triangleright\underline{0}10$

An Example

$Q = \{q_s, q_h, q_1, q_2, q_3\}$, $\Gamma = \{0, 1, \square, \triangleright\}$,
and δ is as follows:

$p \in Q$	$\sigma \in \Gamma$	$\delta(p, \sigma)$
q_s	0	$(q_s, 0, \rightarrow)$
q_s	1	$(q_s, 1, \rightarrow)$
q_s	$\square$	$(q_1, \square, \leftarrow)$
q_s	$\triangleright$	$(q_s, \triangleright, \rightarrow)$
q_1	0	$(q_2, \square, \rightarrow)$
q_1	1	$(q_3, \square, \rightarrow)$
q_1	$\square$	$(q_1, \square, -)$
q_1	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_2	0	$(q_s, 0, \leftarrow)$
q_2	1	$(q_s, 0, \leftarrow)$
q_2	$\square$	$(q_s, 0, \leftarrow)$
q_2	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_3	0	$(q_s, 1, \leftarrow)$
q_3	1	$(q_s, 1, \leftarrow)$
q_3	$\square$	$(q_s, 1, \leftarrow)$
q_3	$\triangleright$	$(q_h, \triangleright, \rightarrow)$

Start the machine with input 010

- $q_s, \underline{\triangleright}010$
- $q_s, \triangleright\underline{0}10$
- $q_s, \triangleright 0\underline{1}0$

An Example

$Q = \{q_s, q_h, q_1, q_2, q_3\}$, $\Gamma = \{0, 1, \square, \triangleright\}$,
and δ is as follows:

$p \in Q$	$\sigma \in \Gamma$	$\delta(p, \sigma)$
q_s	0	$(q_s, 0, \rightarrow)$
q_s	1	$(q_s, 1, \rightarrow)$
q_s	$\square$	$(q_1, \square, \leftarrow)$
q_s	$\triangleright$	$(q_s, \triangleright, \rightarrow)$
q_1	0	$(q_2, \square, \rightarrow)$
q_1	1	$(q_3, \square, \rightarrow)$
q_1	$\square$	$(q_1, \square, -)$
q_1	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_2	0	$(q_s, 0, \leftarrow)$
q_2	1	$(q_s, 0, \leftarrow)$
q_2	$\square$	$(q_s, 0, \leftarrow)$
q_2	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_3	0	$(q_s, 1, \leftarrow)$
q_3	1	$(q_s, 1, \leftarrow)$
q_3	$\square$	$(q_s, 1, \leftarrow)$
q_3	$\triangleright$	$(q_h, \triangleright, \rightarrow)$

Start the machine with input 010

- q_s, $\underline{\triangleright}010$
- q_s, $\triangleright\underline{0}10$
- q_s, $\triangleright 0\underline{1}0$
- q_s, $\triangleright 01\underline{0}$

An Example

$Q = \{q_s, q_h, q_1, q_2, q_3\}$, $\Gamma = \{0, 1, \Box, \triangleright\}$,
and δ is as follows:

$p \in Q$	$\sigma \in \Gamma$	$\delta(p, \sigma)$
q_s	0	$(q_s, 0, \rightarrow)$
q_s	1	$(q_s, 1, \rightarrow)$
q_s	$\Box$	$(q_1, \Box, \leftarrow)$
q_s	$\triangleright$	$(q_s, \triangleright, \rightarrow)$
q_1	0	$(q_2, \Box, \rightarrow)$
q_1	1	$(q_3, \Box, \rightarrow)$
q_1	$\Box$	$(q_1, \Box, -)$
q_1	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_2	0	$(q_s, 0, \leftarrow)$
q_2	1	$(q_s, 0, \leftarrow)$
q_2	$\Box$	$(q_s, 0, \leftarrow)$
q_2	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_3	0	$(q_s, 1, \leftarrow)$
q_3	1	$(q_s, 1, \leftarrow)$
q_3	$\Box$	$(q_s, 1, \leftarrow)$
q_3	$\triangleright$	$(q_h, \triangleright, \rightarrow)$

Start the machine with input 010

- $q_s, \underline{\triangleright}010$
- $q_s, \triangleright\underline{0}10$
- $q_s, \triangleright 0\underline{1}0$
- $q_s, \triangleright 01\underline{0}$
- $q_s, \triangleright 010\underline{\Box}$

An Example

$Q = \{q_s, q_h, q_1, q_2, q_3\}$, $\Gamma = \{0, 1, \square, \triangleright\}$,
and δ is as follows:

$p \in Q$	$\sigma \in \Gamma$	$\delta(p, \sigma)$
q_s	0	$(q_s, 0, \rightarrow)$
q_s	1	$(q_s, 1, \rightarrow)$
q_s	$\square$	$(q_1, \square, \leftarrow)$
q_s	$\triangleright$	$(q_s, \triangleright, \rightarrow)$
q_1	0	$(q_2, \square, \rightarrow)$
q_1	1	$(q_3, \square, \rightarrow)$
q_1	$\square$	$(q_1, \square, -)$
q_1	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_2	0	$(q_s, 0, \leftarrow)$
q_2	1	$(q_s, 0, \leftarrow)$
q_2	$\square$	$(q_s, 0, \leftarrow)$
q_2	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_3	0	$(q_s, 1, \leftarrow)$
q_3	1	$(q_s, 1, \leftarrow)$
q_3	$\square$	$(q_s, 1, \leftarrow)$
q_3	$\triangleright$	$(q_h, \triangleright, \rightarrow)$

Start the machine with input 010

- $q_s, \underline{\triangleright}010$
- $q_s, \triangleright\underline{0}10$
- $q_s, \triangleright0\underline{1}0$
- $q_s, \triangleright01\underline{0}$
- $q_s, \triangleright010\underline{\square}$
- $q_1, \triangleright01\underline{0}\square$

An Example

$Q = \{q_s, q_h, q_1, q_2, q_3\}$, $\Gamma = \{0, 1, \square, \triangleright\}$,
and δ is as follows:

$p \in Q$	$\sigma \in \Gamma$	$\delta(p, \sigma)$
q_s	0	$(q_s, 0, \rightarrow)$
q_s	1	$(q_s, 1, \rightarrow)$
q_s	$\square$	$(q_1, \square, \leftarrow)$
q_s	$\triangleright$	$(q_s, \triangleright, \rightarrow)$
q_1	0	$(q_2, \square, \rightarrow)$
q_1	1	$(q_3, \square, \rightarrow)$
q_1	$\square$	$(q_1, \square, -)$
q_1	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_2	0	$(q_s, 0, \leftarrow)$
q_2	1	$(q_s, 0, \leftarrow)$
q_2	$\square$	$(q_s, 0, \leftarrow)$
q_2	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_3	0	$(q_s, 1, \leftarrow)$
q_3	1	$(q_s, 1, \leftarrow)$
q_3	$\square$	$(q_s, 1, \leftarrow)$
q_3	$\triangleright$	$(q_h, \triangleright, \rightarrow)$

Start the machine with input 010

- $q_s, \underline{\triangleright}010$
- $q_s, \triangleright\underline{0}10$
- $q_s, \triangleright 0\underline{1}0$
- $q_s, \triangleright 01\underline{0}$
- $q_s, \triangleright 010\underline{\square}$
- $q_1, \triangleright 01\underline{0}\square$
- $q_2, \triangleright 01\square\underline{\square}$

An Example

$Q = \{q_s, q_h, q_1, q_2, q_3\}$, $\Gamma = \{0, 1, \square, \triangleright\}$,
and δ is as follows:

$p \in Q$	$\sigma \in \Gamma$	$\delta(p, \sigma)$
q_s	0	$(q_s, 0, \rightarrow)$
q_s	1	$(q_s, 1, \rightarrow)$
q_s	$\square$	$(q_1, \square, \leftarrow)$
q_s	$\triangleright$	$(q_s, \triangleright, \rightarrow)$
q_1	0	$(q_2, \square, \rightarrow)$
q_1	1	$(q_3, \square, \rightarrow)$
q_1	$\square$	$(q_1, \square, -)$
q_1	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_2	0	$(q_s, 0, \leftarrow)$
q_2	1	$(q_s, 0, \leftarrow)$
q_2	$\square$	$(q_s, 0, \leftarrow)$
q_2	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_3	0	$(q_s, 1, \leftarrow)$
q_3	1	$(q_s, 1, \leftarrow)$
q_3	$\square$	$(q_s, 1, \leftarrow)$
q_3	$\triangleright$	$(q_h, \triangleright, \rightarrow)$

Start the machine with input 010

- $q_s, \underline{\triangleright}010$
- $q_s, \triangleright\underline{0}10$
- $q_s, \triangleright 0\underline{1}0$
- $q_s, \triangleright 01\underline{0}$
- $q_s, \triangleright 010\underline{\square}$
- $q_1, \triangleright 01\underline{0}\square$
- $q_2, \triangleright 01\square\underline{\square}$
- $q_s, \triangleright 01\underline{\square}0$

An Example

$Q = \{q_s, q_h, q_1, q_2, q_3\}$, $\Gamma = \{0, 1, \square, \triangleright\}$,
and δ is as follows:

$p \in Q$	$\sigma \in \Gamma$	$\delta(p, \sigma)$
q_s	0	$(q_s, 0, \rightarrow)$
q_s	1	$(q_s, 1, \rightarrow)$
q_s	$\square$	$(q_1, \square, \leftarrow)$
q_s	$\triangleright$	$(q_s, \triangleright, \rightarrow)$
q_1	0	$(q_2, \square, \rightarrow)$
q_1	1	$(q_3, \square, \rightarrow)$
q_1	$\square$	$(q_1, \square, -)$
q_1	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_2	0	$(q_s, 0, \leftarrow)$
q_2	1	$(q_s, 0, \leftarrow)$
q_2	$\square$	$(q_s, 0, \leftarrow)$
q_2	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_3	0	$(q_s, 1, \leftarrow)$
q_3	1	$(q_s, 1, \leftarrow)$
q_3	$\square$	$(q_s, 1, \leftarrow)$
q_3	$\triangleright$	$(q_h, \triangleright, \rightarrow)$

Start the machine with input 010

- $q_s, \underline{\triangleright}010$
- $q_s, \triangleright\underline{0}10$
- $q_s, \triangleright0\underline{1}0$
- $q_s, \triangleright01\underline{0}$
- $q_s, \triangleright010\underline{\square}$
- $q_1, \triangleright01\underline{0}\square$
- $q_2, \triangleright01\square\underline{\square}$
- $q_s, \triangleright01\underline{\square}0$
- $q_1, \triangleright0\underline{1}\square0$

An Example

$Q = \{q_s, q_h, q_1, q_2, q_3\}$, $\Gamma = \{0, 1, \square, \triangleright\}$,
and δ is as follows:

$p \in Q$	$\sigma \in \Gamma$	$\delta(p, \sigma)$
q_s	0	$(q_s, 0, \rightarrow)$
q_s	1	$(q_s, 1, \rightarrow)$
q_s	$\square$	$(q_1, \square, \leftarrow)$
q_s	$\triangleright$	$(q_s, \triangleright, \rightarrow)$
q_1	0	$(q_2, \square, \rightarrow)$
q_1	1	$(q_3, \square, \rightarrow)$
q_1	$\square$	$(q_1, \square, -)$
q_1	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_2	0	$(q_s, 0, \leftarrow)$
q_2	1	$(q_s, 0, \leftarrow)$
q_2	$\square$	$(q_s, 0, \leftarrow)$
q_2	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_3	0	$(q_s, 1, \leftarrow)$
q_3	1	$(q_s, 1, \leftarrow)$
q_3	$\square$	$(q_s, 1, \leftarrow)$
q_3	$\triangleright$	$(q_h, \triangleright, \rightarrow)$

Start the machine with input 010

- q_s, $\underline{\triangleright}010$
- q_s, $\triangleright\underline{0}10$
- q_s, $\triangleright0\underline{1}0$
- q_s, $\triangleright01\underline{0}$
- q_s, $\triangleright010\underline{\square}$
- q_1, $\triangleright01\underline{0}\square$
- q_2, $\triangleright01\square\underline{\square}$
- q_s, $\triangleright01\underline{\square}0$
- q_1, $\triangleright0\underline{1}\square0$
- q_3, $\triangleright0\square\underline{\square}0$

An Example

$Q = \{q_s, q_h, q_1, q_2, q_3\}$, $\Gamma = \{0, 1, \square, \triangleright\}$,
and δ is as follows:

$p \in Q$	$\sigma \in \Gamma$	$\delta(p, \sigma)$
q_s	0	$(q_s, 0, \rightarrow)$
q_s	1	$(q_s, 1, \rightarrow)$
q_s	$\square$	$(q_1, \square, \leftarrow)$
q_s	$\triangleright$	$(q_s, \triangleright, \rightarrow)$
q_1	0	$(q_2, \square, \rightarrow)$
q_1	1	$(q_3, \square, \rightarrow)$
q_1	$\square$	$(q_1, \square, -)$
q_1	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_2	0	$(q_s, 0, \leftarrow)$
q_2	1	$(q_s, 0, \leftarrow)$
q_2	$\square$	$(q_s, 0, \leftarrow)$
q_2	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_3	0	$(q_s, 1, \leftarrow)$
q_3	1	$(q_s, 1, \leftarrow)$
q_3	$\square$	$(q_s, 1, \leftarrow)$
q_3	$\triangleright$	$(q_h, \triangleright, \rightarrow)$

Start the machine with input 010

- $q_s, \underline{\triangleright}010$
- $q_s, \triangleright\underline{0}10$
- $q_s, \triangleright0\underline{1}0$
- $q_s, \triangleright01\underline{0}$
- $q_s, \triangleright010\underline{\square}$
- $q_1, \triangleright01\underline{0}\square$
- $q_2, \triangleright01\square\underline{\square}$
- $q_s, \triangleright01\underline{\square}0$
- $q_1, \triangleright0\underline{1}\square0$
- $q_3, \triangleright0\square\underline{\square}0$
- $q_s, \triangleright0\underline{\square}10$

An Example

$Q = \{q_s, q_h, q_1, q_2, q_3\}$, $\Gamma = \{0, 1, \Box, \triangleright\}$,
and δ is as follows:

$p \in Q$	$\sigma \in \Gamma$	$\delta(p, \sigma)$
q_s	0	$(q_s, 0, \rightarrow)$
q_s	1	$(q_s, 1, \rightarrow)$
q_s	$\Box$	$(q_1, \Box, \leftarrow)$
q_s	$\triangleright$	$(q_s, \triangleright, \rightarrow)$
q_1	0	$(q_2, \Box, \rightarrow)$
q_1	1	$(q_3, \Box, \rightarrow)$
q_1	$\Box$	$(q_1, \Box, -)$
q_1	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_2	0	$(q_s, 0, \leftarrow)$
q_2	1	$(q_s, 0, \leftarrow)$
q_2	$\Box$	$(q_s, 0, \leftarrow)$
q_2	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_3	0	$(q_s, 1, \leftarrow)$
q_3	1	$(q_s, 1, \leftarrow)$
q_3	$\Box$	$(q_s, 1, \leftarrow)$
q_3	$\triangleright$	$(q_h, \triangleright, \rightarrow)$

Start the machine with input 010

- $q_s, \underline{\triangleright}010$
- $q_s, \triangleright\underline{0}10$
- $q_s, \triangleright0\underline{1}0$
- $q_s, \triangleright01\underline{0}$
- $q_s, \triangleright010\underline{\Box}$
- $q_1, \triangleright01\underline{0}\Box$
- $q_2, \triangleright01\Box\underline{\Box}$
- $q_s, \triangleright01\underline{\Box}0$
- $q_1, \triangleright0\underline{1}\Box0$
- $q_3, \triangleright0\Box\underline{\Box}0$
- $q_s, \triangleright0\underline{\Box}10$
- $q_1, \triangleright\underline{0}\Box10$

An Example

$Q = \{q_s, q_h, q_1, q_2, q_3\}$, $\Gamma = \{0, 1, \Box, \triangleright\}$,
and δ is as follows:

$p \in Q$	$\sigma \in \Gamma$	$\delta(p, \sigma)$
q_s	0	$(q_s, 0, \rightarrow)$
q_s	1	$(q_s, 1, \rightarrow)$
q_s	$\Box$	$(q_1, \Box, \leftarrow)$
q_s	$\triangleright$	$(q_s, \triangleright, \rightarrow)$
q_1	0	$(q_2, \Box, \rightarrow)$
q_1	1	$(q_3, \Box, \rightarrow)$
q_1	$\Box$	$(q_1, \Box, -)$
q_1	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_2	0	$(q_s, 0, \leftarrow)$
q_2	1	$(q_s, 0, \leftarrow)$
q_2	$\Box$	$(q_s, 0, \leftarrow)$
q_2	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_3	0	$(q_s, 1, \leftarrow)$
q_3	1	$(q_s, 1, \leftarrow)$
q_3	$\Box$	$(q_s, 1, \leftarrow)$
q_3	$\triangleright$	$(q_h, \triangleright, \rightarrow)$

Start the machine with input 010

- $q_s, \underline{\triangleright}010$
- $q_s, \triangleright\underline{0}10$
- $q_s, \triangleright0\underline{1}0$
- $q_s, \triangleright01\underline{0}$
- $q_s, \triangleright010\underline{\Box}$
- $q_1, \triangleright01\underline{0}\Box$
- $q_2, \triangleright01\Box\underline{\Box}$
- $q_s, \triangleright01\underline{\Box}0$
- $q_1, \triangleright0\underline{1}\Box0$
- $q_3, \triangleright0\Box\underline{\Box}0$
- $q_s, \triangleright0\underline{\Box}10$
- $q_1, \triangleright\underline{0}\Box10$
- $q_2, \triangleright\Box\underline{\Box}10$

An Example

$Q = \{q_s, q_h, q_1, q_2, q_3\}$, $\Gamma = \{0, 1, \square, \triangleright\}$,
and δ is as follows:

$p \in Q$	$\sigma \in \Gamma$	$\delta(p, \sigma)$
q_s	0	$(q_s, 0, \rightarrow)$
q_s	1	$(q_s, 1, \rightarrow)$
q_s	$\square$	$(q_1, \square, \leftarrow)$
q_s	$\triangleright$	$(q_s, \triangleright, \rightarrow)$
q_1	0	$(q_2, \square, \rightarrow)$
q_1	1	$(q_3, \square, \rightarrow)$
q_1	$\square$	$(q_1, \square, -)$
q_1	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_2	0	$(q_s, 0, \leftarrow)$
q_2	1	$(q_s, 0, \leftarrow)$
q_2	$\square$	$(q_s, 0, \leftarrow)$
q_2	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_3	0	$(q_s, 1, \leftarrow)$
q_3	1	$(q_s, 1, \leftarrow)$
q_3	$\square$	$(q_s, 1, \leftarrow)$
q_3	$\triangleright$	$(q_h, \triangleright, \rightarrow)$

Start the machine with input 010

- $q_s, \underline{\triangleright}010$
- $q_s, \triangleright\underline{0}10$
- $q_s, \triangleright0\underline{1}0$
- $q_s, \triangleright01\underline{0}$
- $q_s, \triangleright010\underline{\square}$
- $q_1, \triangleright01\underline{0}\square$
- $q_2, \triangleright01\square\underline{\square}$
- $q_s, \triangleright01\underline{\square}0$
- $q_1, \triangleright0\underline{1}\square0$
- $q_3, \triangleright0\square\underline{\square}0$
- $q_s, \triangleright0\underline{\square}10$
- $q_1, \triangleright\underline{0}\square10$
- $q_2, \triangleright\square\underline{\square}10$
- $q_0, \triangleright\underline{\square}010$

An Example

$Q = \{q_s, q_h, q_1, q_2, q_3\}$, $\Gamma = \{0, 1, \square, \triangleright\}$,
and δ is as follows:

$p \in Q$	$\sigma \in \Gamma$	$\delta(p, \sigma)$
q_s	0	$(q_s, 0, \rightarrow)$
q_s	1	$(q_s, 1, \rightarrow)$
q_s	$\square$	$(q_1, \square, \leftarrow)$
q_s	$\triangleright$	$(q_s, \triangleright, \rightarrow)$
q_1	0	$(q_2, \square, \rightarrow)$
q_1	1	$(q_3, \square, \rightarrow)$
q_1	$\square$	$(q_1, \square, -)$
q_1	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_2	0	$(q_s, 0, \leftarrow)$
q_2	1	$(q_s, 0, \leftarrow)$
q_2	$\square$	$(q_s, 0, \leftarrow)$
q_2	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_3	0	$(q_s, 1, \leftarrow)$
q_3	1	$(q_s, 1, \leftarrow)$
q_3	$\square$	$(q_s, 1, \leftarrow)$
q_3	$\triangleright$	$(q_h, \triangleright, \rightarrow)$

Start the machine with input 010

- $q_s, \underline{\triangleright}010$
- $q_s, \triangleright\underline{0}10$
- $q_s, \triangleright 0\underline{1}0$
- $q_s, \triangleright 01\underline{0}$
- $q_s, \triangleright 010\underline{\square}$
- $q_1, \triangleright 01\underline{0}\square$
- $q_2, \triangleright 01\square\underline{\square}$
- $q_s, \triangleright 01\underline{\square}0$
- $q_1, \triangleright 0\underline{1}\square 0$
- $q_3, \triangleright 0\square\underline{\square}0$
- $q_s, \triangleright 0\underline{\square}10$
- $q_1, \triangleright\underline{0}\square 10$
- $q_2, \triangleright\square\underline{\square}10$
- $q_0, \triangleright\underline{\square}010$
- $q_1, \underline{\triangleright}\square 010$

An Example

$Q = \{q_s, q_h, q_1, q_2, q_3\}$, $\Gamma = \{0, 1, \square, \triangleright\}$,
and δ is as follows:

$p \in Q$	$\sigma \in \Gamma$	$\delta(p, \sigma)$
q_s	0	$(q_s, 0, \rightarrow)$
q_s	1	$(q_s, 1, \rightarrow)$
q_s	$\square$	$(q_1, \square, \leftarrow)$
q_s	$\triangleright$	$(q_s, \triangleright, \rightarrow)$
q_1	0	$(q_2, \square, \rightarrow)$
q_1	1	$(q_3, \square, \rightarrow)$
q_1	$\square$	$(q_1, \square, -)$
q_1	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_2	0	$(q_s, 0, \leftarrow)$
q_2	1	$(q_s, 0, \leftarrow)$
q_2	$\square$	$(q_s, 0, \leftarrow)$
q_2	$\triangleright$	$(q_h, \triangleright, \rightarrow)$
q_3	0	$(q_s, 1, \leftarrow)$
q_3	1	$(q_s, 1, \leftarrow)$
q_3	$\square$	$(q_s, 1, \leftarrow)$
q_3	$\triangleright$	$(q_h, \triangleright, \rightarrow)$

Start the machine with input 010

- $q_s, \underline{\triangleright}010$
- $q_s, \triangleright\underline{0}10$
- $q_s, \triangleright0\underline{1}0$
- $q_s, \triangleright01\underline{0}$
- $q_s, \triangleright010\underline{\square}$
- $q_1, \triangleright01\underline{0}\square$
- $q_2, \triangleright01\square\underline{\square}$
- $q_s, \triangleright01\underline{\square}0$
- $q_1, \triangleright0\underline{1}\square0$
- $q_3, \triangleright0\square\underline{\square}0$
- $q_s, \triangleright0\underline{\square}10$
- $q_1, \triangleright\underline{0}\square10$
- $q_2, \triangleright\square\underline{\square}10$
- $q_0, \triangleright\underline{\square}010$
- $q_1, \underline{\triangleright}\square010$
- $q_h, \triangleright\underline{\square}010$

The Second Example

$Q = \{q_s, q_h, q_1\}$, $\Gamma = \{0, 1, \square, \triangleright\}$, and δ is as follows:

$p \in Q$	$\sigma \in \Gamma$	$\delta(p, \sigma)$
q_s	0	$(q_s, 0, \rightarrow)$
q_s	1	$(q_s, 1, \rightarrow)$
q_s	$\square$	$(q_1, \square, \leftarrow)$
q_s	$\triangleright$	$(q_s, \triangleright, \rightarrow)$
q_1	0	$(q_h, 1, -)$
q_1	1	$(q_1, 0, \leftarrow)$
q_1	$\triangleright$	$(q_h, \triangleright, \rightarrow)$

The Third Example

$Q = \{q_s, q_h, q_c, q_l, q_t\}$; $\Gamma = \{\Box, \triangleright, 0, 1\}$; two work tapes.

The Third Example

$Q = \{q_s, q_h, q_c, q_l, q_t\}$; $\Gamma = \{\Box, \rhd, 0, 1\}$; two work tapes.

$\langle q_s, \rhd, \rhd, \rhd\rangle \rightarrow \langle q_c, \rhd, \rhd, \rightarrow, \rightarrow, \rightarrow\rangle$

$\langle q_c, 0, \Box, \Box\rangle \rightarrow \langle q_c, 0, \Box, \rightarrow, \rightarrow, -\rangle$
$\langle q_c, 1, \Box, \Box\rangle \rightarrow \langle q_c, 1, \Box, \rightarrow, \rightarrow, -\rangle$
$\langle q_c, \Box, \Box, \Box\rangle \rightarrow \langle q_l, \Box, \Box, \leftarrow, -, -\rangle$

$\langle q_l, 0, \Box, \Box\rangle \rightarrow \langle q_l, \Box, \Box, \leftarrow, -, -\rangle$
$\langle q_l, 1, \Box, \Box\rangle \rightarrow \langle q_l, \Box, \Box, \leftarrow, -, -\rangle$
$\langle q_l, \rhd, \Box, \Box\rangle \rightarrow \langle q_t, \Box, \Box, \rightarrow, \leftarrow, -\rangle$

$\langle q_t, \Box, \rhd, \Box\rangle \rightarrow \langle q_h, \rhd, 1, -, -, -\rangle$
$\langle q_t, 0, 1, \Box\rangle \rightarrow \langle q_h, 1, 0, -, -, -\rangle$
$\langle q_t, 1, 0, \Box\rangle \rightarrow \langle q_h, 0, 0, -, -, -\rangle$
$\langle q_t, 0, 0, \Box\rangle \rightarrow \langle q_t, 0, \Box, \rightarrow, \leftarrow, -\rangle$
$\langle q_t, 1, 1, \Box\rangle \rightarrow \langle q_t, 1, \Box, \rightarrow, \leftarrow, -\rangle$

$\{0, 1, \Box, \triangleright\}$ vs. Larger Alphabets

Suppose $\mathbb{M}$ has k tapes with the alphabet Γ.

$\{0, 1, \square, \triangleright\}$ vs. Larger Alphabets

Suppose $\mathbb{M}$ has k tapes with the alphabet Γ.

A symbol of $\mathbb{M}$ is encoded by a string $\sigma \in \{0, 1\}^*$ of length $\log |\Gamma|$.

$\{0, 1, \Box, \triangleright\}$ vs. Larger Alphabets

Suppose $\mathbb{M}$ has k tapes with the alphabet Γ.

A symbol of $\mathbb{M}$ is encoded by a string $\sigma \in \{0,1\}^*$ of length $\log|\Gamma|$.

States: A state q is turned into states q, $\langle q, \sigma_1^1, \ldots, \sigma_1^k \rangle$ where $|\sigma_1^1| = \ldots = |\sigma_1^k| = 1$, ..., $\langle q, \sigma_{\log|\Gamma|}^1, \ldots, \sigma_{\log|\Gamma|}^k \rangle$ where $|\sigma_{\log|\Gamma|}^1| = \ldots = |\sigma_{\log|\Gamma|}^k| = \log|\Gamma|$.

$\{0, 1, \Box, \triangleright\}$ vs. Larger Alphabets

Suppose $\mathbb{M}$ has k tapes with the alphabet Γ.

A symbol of $\mathbb{M}$ is encoded by a string $\sigma \in \{0, 1\}^*$ of length $\log |\Gamma|$.

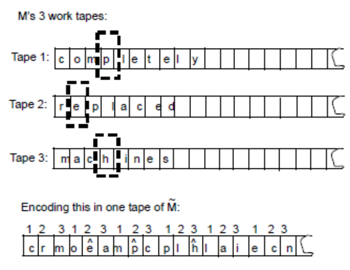
States: A state q is turned into states q, $\langle q, \sigma_1^1, \ldots, \sigma_1^k \rangle$ where $|\sigma_1^1| = \ldots = |\sigma_1^k| = 1$, …, $\langle q, \sigma_{\log |\Gamma|}^1, \ldots, \sigma_{\log |\Gamma|}^k \rangle$ where $|\sigma_{\log |\Gamma|}^1| = \ldots = |\sigma_{\log |\Gamma|}^k| = \log |\Gamma|$.

A computation step of $\mathbb{M}$ is simulated in $\widetilde{\mathbb{M}}$ by $\log |\Gamma|$ steps to read, $\log |\Gamma|$ steps to write, and $\log |\Gamma|$ steps to relocate the heads.

One Tape vs. Many Tapes

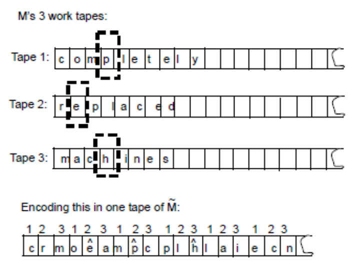
One Tape vs. Many Tapes

The basic idea is to interleave k tapes into one tape.
The first $n+1$ cells are reserved for the input.



One Tape vs. Many Tapes

The basic idea is to interleave k tapes into one tape.
The first $n+1$ cells are reserved for the input.



Every symbol a of $\mathbb{M}$ is turned into two symbols $a, \widehat{a}$ in $\widetilde{\mathbb{M}}$, with $\widehat{a}$ used to indicate head position.

One Tape vs. Many Tapes

One Tape vs. Many Tapes

The machine $\widetilde{\mathbb{M}}$ copies the input bits to the first imaginary tape. The head then moves left to the $(n+2)$-th cell.

One Tape vs. Many Tapes

The machine $\widetilde{\mathbb{M}}$ copies the input bits to the first imaginary tape. The head then moves left to the $(n+2)$-th cell.

Sweeping the tape cells from left to right. Record in the register the k symbols marked with the hat $\widehat{\_}$.

One Tape vs. Many Tapes

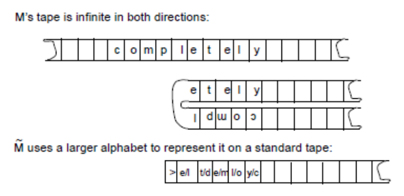
The machine $\widetilde{\mathbb{M}}$ copies the input bits to the first imaginary tape. The head then moves left to the $(n+2)$-th cell.

Sweeping the tape cells from left to right. Record in the register the k symbols marked with the hat $\widehat{\_}$.

Sweeping the tape cells from right to left to update using the transitions of $\mathbb{M}$.

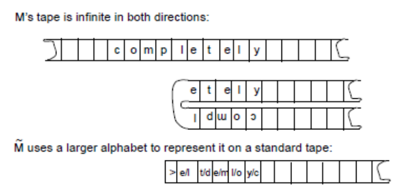
One Unidirectional vs. Bidirectional Tape

The idea is that $\widetilde{\mathbb{M}}$ makes use of the alphabet $\Gamma \times \Gamma$.



One Unidirectional vs. Bidirectional Tape

The idea is that $\widetilde{\mathbb{M}}$ makes use of the alphabet $\Gamma \times \Gamma$.



Every state q of $\mathbb{M}$ is turned into $\overline{q}$ and $\underline{q}$.

Church-Turing Thesis

Church-Turing Thesis.

The functions definable in all computation models are the same. They are precisely the computable functions.

Turing Machine in Complexity

J. Hartmanis and R. Stearns. On the Computational Complexity of Algorithms. *Transactions of the American Mathematical Society*, Vol.117, May, 285-306, 1965.

Alphabet and Language

An alphabet Σ is a finite set of symbols

A language L is a subset of the set of all finite length strings of Σ, denoted by Σ^*.

Language and Decision Problem

Problems can be encoded as languages.

Language and Machine

Some languages can be accepted by machines.

k-Tape Turing Machine

A (nondeterministic) k-tape Turing machine is a 6-tuple $M = (S, \Sigma, \Gamma, \delta, p_0, p_f)$, where

- S is a finite set of states,
- Γ is a finite set of tape symbols which includes the special symbol B,
- $\Sigma \subseteq \Gamma \setminus \{B\}$, the set of input symbols,
- δ, the transition function, is a function that maps elements of $S \times \Gamma^k$ into the finite subsets of $S \times ((\Gamma \setminus \{B\}) \times \{L, P, R\})^k$,
- $p_0 \in S$, the initial state,
- $p_f \in S$, the final or accepting state.

k-Tape Turing Machine

A (nondeterministic) k-tape Turing machine is a 6-tuple $M = (S, \Sigma, \Gamma, \delta, p_0, p_f)$, where

- S is a finite set of states,
- Γ is a finite set of tape symbols which includes the special symbol B,
- $\Sigma \subseteq \Gamma \setminus \{B\}$, the set of input symbols,
- δ, the transition function, is a function that maps elements of $S \times \Gamma^k$ into the finite subsets of $S \times ((\Gamma \setminus \{B\}) \times \{L, P, R\})^k$,
- $p_0 \in S$, the initial state,
- $p_f \in S$, the final or accepting state.

A k-tape Turing machine $M = (S, \Sigma, \Gamma, \delta, p_0, p_f)$ is deterministic if for every $p \in S$ and every $a_1, \ldots, a_k \in \Gamma$, the set $\delta(s, a_1, \ldots, a_k)$ contains at most one element.

Configuration

Let $M = (S, \Sigma, \Gamma, \delta, p_0, p_f)$ be a k-tape Turing machine. A configuration of M is a $(k+1)$-tuple

$$K = (p, w_{11} \uparrow w_{12}, w_{21} \uparrow w_{22}, \ldots, w_{k1} \uparrow w_{k2})$$

where $p \in S$ and $w_{j1} \uparrow w_{j2}$ is the content of the j-th tape of M, $1 \leq j \leq k$.

Configuration

Let $M = (S, \Sigma, \Gamma, \delta, p_0, p_f)$ be a k-tape Turing machine. A configuration of M is a $(k+1)$-tuple

$$K = (p, w_{11} \uparrow w_{12}, w_{21} \uparrow w_{22}, \ldots, w_{k1} \uparrow w_{k2})$$

where $p \in S$ and $w_{j1} \uparrow w_{j2}$ is the content of the j-th tape of M, $1 \leq j \leq k$.

Initial configuration

$$(p_0, \uparrow x, \uparrow B, \ldots, \uparrow B)$$

Final configuration

$$(p_f, w_{11} \uparrow w_{12}, w_{21} \uparrow w_{22}, \ldots, w_{k1} \uparrow w_{k2})$$

Computation

A *computation* by a Turing machine M on input x is a sequence of configurations $K_1, \ldots, K_t$, for some $t \geq 1$, where K_1 is the initial configuration, and for all i, $2 \leq i \leq t$, K_i results from K_{I-1} in one move of M. Here t is referred to as the length of the computation. If K_t is the final configuration, then the computation is called an accepting computation.

Time Complexity

The *time taken by a Turing machine M on input x*, denoted by $T_M(x)$, is defined by:

1. If there is an accepting computation of M on input x, then $T_M(x)$ is the length of the shortest accepting computation, and
2. If there is no accepting computation of M on input x, then $T_M(x) = \infty$.

Time Complexity Class

Let L be a language and f a function on natural numbers. We say that L is in $DTIME(f)$, respectively $NTIME(f)$, if there exists a deterministic, respectively nondeterministic, Turing machine M that behaves as follows: On input x, if $x \in L$ then $T_M(x) \leq f(|x|)$; otherwise $T_M(x) = \infty$.

$$P = DTIME(n) \cup DTIME(n^2) \cup DTIME(n^3) \cup \ldots \cup DTIME(n^k) \cup \ldots$$

$$NP = NTIME(n) \cup NTIME(n^2) \cup NTIME(n^3) \cup \ldots \cup NTIME(n^k) \cup \ldots$$

Time Complexity Class

$$
\begin{aligned}
DEXT &= \bigcup_{c \geq 0} DTIME(2^{cn}) \\
NEXT &= \bigcup_{c \geq 0} NTIME(2^{cn}) \\
EXPTIME &= \bigcup_{c \geq 0} DTIME(2^{n^c}) \\
NEXPTIME &= \bigcup_{c \geq 0} NTIME(2^{n^c})
\end{aligned}
$$

An Example

The 1-tape Turing machine that recognizes the language $L = \{a^n b^n \mid n \geq 1\}$ is in $DTIME(n^2)$.

Off-Line Turing Machine

A (nondeterministic) *off-line* Turing machine is a 6-tuple $M = (S, \Sigma, \Gamma, \delta, p_0, p_f)$, where

- S is a finite set of states,
- Γ is a finite set of tape symbols including B,
- $\Sigma \subseteq \Gamma \setminus \{B\}$, the set of input symbols; it contains two special symbols $\#$ and $\$$ (left endmarker and right endmarker).
- δ is the transition function that maps elements of $S \times \Gamma$ into finite subsets of $S \times \{L, P, R\} \times (\Gamma \setminus \{B\}) \times \{L, P, R\}$,
- $p_0 \in S$, the initial state,
- $p_f \in S$, the final or accepting state.

Configuration of Off-Line TM

$$K = (p, i, w_1 \uparrow w_2)$$

Space Complexity

The *space used by an off-line Turing machine M on input x*, denoted by $S_M(x)$, is defined by:

1. If there is an accepting computation of M on input x, then $S_M(x)$ is the number of worktape cells used in an accepting computation that uses the least number of worktape cells, and
2. If there is no accepting computation of M on input x, then $S_M(x) = \infty$.

Example Revisited

The off-line Turing machine that recognizes the language $L = \{a^n b^n \mid n \geq 1\}$ using $\lceil \log(n/2) + 1 \rceil$ worktape cells.

Space Complexity Class

Let L be a language and f a function on natural numbers. We say that L is in $DSPACE(f)$, respectively $NSPACE(f)$, if there exists a deterministic, respectively nondeterministic, Turing machine M that behaves as follows: On input x, if $x \in L$ then $S_M(x) \leq f(|x|)$; otherwise $S_M(x) = \infty$.

Time Complexity Class

$$PSPACE = DSPACE(n) \cup DSPACE(n^2) \cup \ldots \cup DSPACE(n^k) \cup \ldots$$

$$NSPACE = NSPACE(n) \cup NSPACE(n^2) \cup \ldots \cup NSPACE(n^k) \cup \ldots$$

$$LOGSPACE = DSPACE(\log n)$$

$$NLOGSPACE = NSPACE(\log n)$$

Graph Accessibility Problem (GAP)

Given a finite directed graph $G = (V, E)$, where $V = \{1, 2, \ldots, n\}$ is there a path from 1 to n.

Complexity of GAP

Theorem. GAP is in *NLOGSPACE*.

Complexity of GAP

Theorem. GAP is in *NLOGSPACE*.

Proof: The worktape record the node most recently visited.

Linear Speed-Up

Linear Speed-Up

Theorem

If a language L is accepted by a $T(n)$ time-bounded Turing machine M with k-tapes ($k > 1$), then for any constant c, $0 < c < 1$, L is accepted by a $cT(n) + n + 2$ time-bounded Turing machine M'.

Tape Compression

Theorem

If a language L is accepted by an $S(n)$ space-bounded off-line Turing machine M, then for any constant c, $0 < c < 1$, L is accepted by a $cS(n)$ space-bounded off-line Turing machine M'.

Relationship Between Complexity Classes

Time and Space Constructible Functions

A total function T on natural numbers is said to be *time constructible* if and only if there is a Turing machine which on *every* input of length n halts in exactly $T(n)$ steps.

A total function S on natural numbers is said to be *space constructible* if and only if there is a Turing machine which on *every* input of length n halts in a configuration in which exactly $S(n)$ tape cells of its work space are non-blank.

Almost all familiar functions are time and space constructible.

Relationship between Complexity Classes

- $DTIME(f(n)) \subseteq NTIME(f(n))$
- $DSPACE(f(n)) \subseteq NSPACE(f(n))$.
- $DTIME(f(n)) \subseteq DSPACE(f(n))$.
- $NTIME(f(n)) \subseteq NSPACE(f(n))$.

Relationship between Complexity Classes

If $S(n)$ is a space constructible function and $S(n) \geq \log n$, then $NSPACE(S(n)) \subseteq DTIME(c^{S(n)})$, $c \geq 2$.

$NSPACE(S(n)) \subseteq DTIME(c^{S(n)})$

Proof

- M, a nondeterministic off-line Turing machine whose work space is bounded by $S(n) \geq \log n$
- s, the number of states
- t, the number of worktape symbols
- the number of possible configurations on input x of length n is

$$s(n+2)S(n)t^{S(n)}$$

- which is bounded by $d^{S(n)}$ for some $d \geq 2$
- so M may not make more than $d^{S(n)}$ moves
- w.l.o.g, we assume that if M accepts, it erases both of its tapes and brings its tape heads to the first cells.

$NSPACE(S(n)) \subseteq DTIME(c^{S(n)})$

Proof

- a deterministic Turing machine M': on input x of length n, generates the transition graph that simulates the nondeterministic Turing machine M
- the length of the graph is $O(e^{S(n)})$ for some $e > d$
- apply the Dijkstra algorithm to find a path from the initial configuration to the accepting configuration, which can be done in $O(e^{2S(n)}) = O(c^{S(n)})$ for some $c \geq 2$
- therefore the language is in $DTIME(c^{S(n)})$

Relationship between Complexity Classes

$$LOGSPACE \subseteq NLOGSPACE \subseteq P$$

Relationship between Complexity Classes

Fact

- If $S(n)$ is a space constructible function and $S(n) \geq \log n$, then $NSPACE(S(n)) \subseteq DSPACE(S^2(n))$.

$NSPACE(S(n)) \subseteq DSPACE(S^2(n))$

Proof

- M, a nondeterministic off-line Turing machine whose work space is bounded by $S(n) \geq \log n$
- s, the number of states
- t, the number of worktape symbols
- the number of possible configurations on input x of length n is

$$s(n+2)S(n)t^{S(n)}$$

- which is bounded by $d^{S(n)}$ for some $d \geq 2$
- so M may not make more than $d^{S(n)}$ moves
- a configuration takes $O(S(n))$ space

$NSPACE(S(n)) \subseteq DSPACE(S^2(n))$

Proof

- w.l.o.g, we assume that if M accepts, it erases both of its tapes and brings its tape heads to the first cells
- a deterministic off-line Turing machine M' checks the reachability from the initial configuration C_i to the final configuration C_f in a divide and conquer fashion (**depth first**); the depth of this recursive call is $O(S(n))$
- hence the language is in $DSPACE(S^2(n))$

Savitch Theorem. $NSPACE = PSPACE$.

Theorem. GAP is $O(\log^2(n))$.

Space Hierarchy Theorem

Let $S(n)$ and $S(n')$ be two space constructible space bounds and $S'(n)$ is $o(S(n))$. Then $DSPACE(S(n))$ contains a language that is not in $DSPACE(S'(n))$.

Space Hierarchy Theorem

Let $S(n)$ and $S(n')$ be two space constructible space bounds and $S'(n)$ is $o(S(n))$. Then $DSPACE(S(n))$ contains a language that is not in $DSPACE(S'(n))$.

This technique is often necessary for diagonalization argument.

Time Hierarchy Theorem

Let $T(n)$ and $T(n')$ be two time constructible time bounds and $T'(n)\log T'(n)$ is $o(T(n))$. Then $DSTIME(T(n))$ contains a language that is not in $DTIME(T'(n))$

Padding

Padding Arguments

For a particular problem Π, we can create a version of Π that has lower complexity by *padding* each instance of Π with a long sequence of extra symbols. This technique is called *padding*.

Basic Idea of Padding

Let $L \subseteq \Sigma^*$ be a language in $DTIME(n^2)$, where Σ does not contain the symbol 0. Define the language

$$L' \stackrel{\text{def}}{=} \{x0^k \mid x \in L \text{ and } k = |x|^2 - |x|\}$$

L' is called the *padded* version of L.

Basic Idea of Padding

Let $L \subseteq \Sigma^*$ be a language in $DTIME(n^2)$, where Σ does not contain the symbol 0. Define the language

$$L' \stackrel{\text{def}}{=} \{x0^k \mid x \in L \text{ and } k = |x|^2 - |x|\}$$

L' is called the *padded* version of L.

Fact: L' is $DTIME(n)$.

Basic Idea of Padding

Let $L \subseteq \Sigma^*$ be a language in $DTIME(n^2)$, where Σ does not contain the symbol 0. Define the language

$$L' \stackrel{\text{def}}{=} \{x0^k \mid x \in L \text{ and } k = |x|^2 - |x|\}$$

L' is called the *padded* version of L.

Fact: L' is $DTIME(n)$.

Proof: Suppose M recognizes L. Construct M' that recognizes L' as follows:

- M' checks if the input x' is of the form $x0^k$;
- M' then simulates M.

Application of Padding (I)

Fact: If $DSPACE(n) \subseteq P$ then $PSPACE = P$.

Application of Padding (I)

Fact: If $DSPACE(n) \subseteq P$ then $PSPACE = P$.

Proof: Let $L \subseteq \Sigma^*$ be in $PSPACE$, where $0 \notin \Sigma$. Suppose M accepts L in the polynomial space $p(n)$. Consider the padded version of L:

$$L' \stackrel{\text{def}}{=} \{x0^k \mid x \in L \text{ and } k = p(|x|) - |x|\}$$

Then

- There is a DTM M' that recognizes L' in linear space;
- So there is a DTM M'' that recognizes L' in polynomial time;
- Construct DTM M''' as follows: upon input x, construct $x0^k$; then simulates M''. Clearly M''' recognizes L in polynomial time.

Fact

$$DSPACE(n) \neq P$$

Application of Padding (II)

Fact: If $NTIME(n) \subseteq P$ then $NEXT = DEXT$.

Application of Padding (II)

Fact: If $\mathit{NTIME}(n) \subseteq P$ then $\mathit{NEXT} = \mathit{DEXT}$.

Proof: Let $L \subseteq \Sigma^*$ be in $\mathit{NTIME}(2^{cn})$, where $0 \notin \Sigma$. Suppose M is a Nondeterministic Turing Machine that accepts L in time 2^{cn}. Consider the padded version of L:

$$L' \stackrel{\text{def}}{=} \{x0^k \mid x \in L \text{ and } k = 2^{cn} - |x|\}$$

Then

- There is an NTM M' that recognizes L' in linear time;
- So there is a DTM M'' that recognizes L' in polynomial time;
- Construct DTM M''' as follows: upon input x, construct $x0^k$; then simulates M''. Clearly M''' recognizes L in time 2^{cn}.

Reduction

Reduction

Let $A \subseteq \Sigma^*$ and $B \subseteq \Delta^*$. A **function** $f : \Sigma^* \to \Delta^*$ is a transformation if the following property is satisfied:

$$\forall x \in \Sigma^*.(x \in A \Leftrightarrow f(x) \in B)$$

Making Use of Transformation

A can be solved using the transformation f and an algorithm for B:

- Transform x into $f(x)$
- Decide whether $f(x) \in B$ or not
- If $f(x) \in B$ then answer yes; otherwise answer no.

Polynomial Time and Log Space Reducibilities

Let $A \subseteq \Sigma^*$ and $B \subseteq \Delta^*$. If there is a transformation $f : \Sigma^* \to \Delta^*$, then we say that A is *reducible* to B, denoted by $A \propto B$.

- A is *polynomial time reducible to* B, denoted by $A \propto_{poly} B$, if $f(x)$ can be computed in polynomial time.
- A is *log space reducible to* B, denoted by $A \propto_{log} B$, if $f(x)$ can be computed using $O(\log |x|)$ space.

Closure

Let $\propto$ be a reducibility relation. Let $\mathcal{L}$ be a family of languages. Define the *closure of $\mathcal{L}$ under the reducibility $\propto$* by

$$closure_{\propto}(\mathcal{L}) \stackrel{\text{def}}{=} \{L \mid \exists L' \in \mathcal{L}. L \propto L'\}$$

We say that $\mathcal{L}$ is *closed under the reducibility relation* $\propto$ if

$$closure_{\propto}(\mathcal{L}) \subseteq \mathcal{L}$$

Comparing $\propto_{log}$ to $\propto_{poly}$

Fact: If $A \propto_{log} B$ then $A \propto_{poly} B$.

Comparing $\propto_{log}$ to $\propto_{poly}$

Fact: If $A \propto_{log} B$ then $A \propto_{poly} B$.

Proof: Let M be in $LOGSPACE$. Let s be the number of states and t be the number of tape symbols. The number of distinct configurations is bounded by

$$s(n+2)(\log n)t^{\log n} = s(n+2)(\log n)n^{\log t}$$

which is polynomial.

Closed Results

Fact: P is closed under polynomial time reductions.

Closed Results

Fact: P is closed under polynomial time reductions.

Proof: Suppose $L \propto_{poly} L' \in P$. If the transformation takes $O(n^l)$ and the decision algorithm for L' takes $O(n^k)$, then the decision algorithm for L takes $O(n^{lk})$. Hence $L \in P$.

Closed Results

Fact: *LOGSPACE* is closed under log space reductions.

Completeness

Completeness

Let $\propto$ be a reducibility relation, and $\mathcal{L}$ be a family of languages. A language L is complete for $\mathcal{L}$ with respect to the reducibility relation $\propto$ if L is in the class $\mathcal{L}$ and every language in $\mathcal{L}$ is reducible to the language L by the relation $\propto$, that is, $\mathcal{L} \subseteq \mathit{closure}_{\propto}(L)$.

LOGSPACE-Complete Problems

Every set in *LOGSPACE* is log space reducible to a set with just one element. Given a set $S \subseteq \Sigma^*$ in *LOGSPACE*, we define the function f_S by

$$f_S(x) \stackrel{\text{def}}{=} \begin{cases} 1 & \text{if } x \in S \\ 0 & \text{otherwise} \end{cases}$$

LOGSPACE-Complete Problems

Every set in *LOGSPACE* is log space reducible to a set with just one element. Given a set $S \subseteq \Sigma^*$ in *LOGSPACE*, we define the function f_S by

$$f_S(x) \stackrel{\text{def}}{=} \begin{cases} 1 & \text{if } x \in S \\ 0 & \text{otherwise} \end{cases}$$

In fact every problem in *LOGSPACE* is *LOGSPACE* complete with respect to log space reduction.

NLOGSPACE-Complete Problems

GAP: Given a finite directed graph $G = (V, E)$, where $V = \{1, 2, \ldots, n\}$, is there a path from 1 to n.

NLOGSPACE-Complete Problems

GAP: Given a finite directed graph $G = (V, E)$, where $V = \{1, 2, \ldots, n\}$, is there a path from 1 to n.

Fact: GAP is log space complete for the class *NLOGSPACE*.

NLOGSPACE-Complete Problems

GAP: Given a finite directed graph $G = (V, E)$, where $V = \{1, 2, \ldots, n\}$, is there a path from 1 to n.

Fact: GAP is log space complete for the class *NLOGSPACE*.

Proof: Let L be in *NLOGSPACE*. We show that $L \propto_{log} GAP$. Suppose M is a nondeterministic off-line Turing machine that accepts L. We can construct a log space reduction which transforms each input string x into an instance of the problem GAP consisting a directed graph $G = (V, E)$, the initial state s and the accepting state f. The vertices are configuration $(p, i, w_1 \uparrow w_2)$, the initial vertex is the initial configuration and the final vertex is the accepting state.

P-Complete Problems

A problem Π is P-complete if it is in P and all problems in P can be reduced to Π using log space reduction.

P-Complete Problems

The P-complete problems are generally believed to be hard to parallelize.

A P-Complete Problem

Linear Programming: Given an $n \times m$ matrix A of integers, a vector b of n integers, a vector c of m integers, and an integer k, determine whether there exists a vector x of m nonnegative rational numbers such that $Ax \leq b$ and $cx \geq k$.

PSPACE-Complete Problems

A problem Π is *PSPACE*-complete if it is in *PSPACE* and all problems in *PSPACE* can be reduced to Π using polynomial time reduction.

PSPACE-Complete Problems

Quantified Boolean Formula (QBF): Given a boolean expression E on n variables $x_1, x_2, \ldots, x_n$, is the boolean formula

$$F = (Q_1 x_1)(Q_2 x_2) \ldots (Q_n x_n) E$$

true? Here Q_i is either $\forall$ or $\exists$.

PSPACE-Complete Problems

Quantified Boolean Formula (QBF): Given a boolean expression E on n variables $x_1, x_2, \ldots, x_n$, is the boolean formula

$$F = (Q_1 x_1)(Q_2 x_2) \ldots (Q_n x_n) E$$

true? Here Q_i is either $\forall$ or $\exists$.

Fact: QBF is *PSPACE*-complete.

PSPACE-Complete Problems

Quantified Boolean Formula (QBF): Given a boolean expression E on n variables $x_1, x_2, \ldots, x_n$, is the boolean formula

$$F = (Q_1 x_1)(Q_2 x_2)\ldots(Q_n x_n)E$$

true? Here Q_i is either $\forall$ or $\exists$.

Fact: QBF is *PSPACE*-complete.

Proof: We can check whether F is true by trying all possible truth assignments for the variables $x_1, x_2, \ldots, x_n$ and evaluating E for each.

Some Conclusions of Completeness

Fact: Let Π be an NP-complete problem with respect to polynomial time reductions. Then $NP = P$ if and only if $\Pi \in P$.

Fact: Let Π be an NP-complete problem with respect to log space reductions. Then
(1) $NP = P$ if and only if $\Pi \in P$.
(2) $NP = NLOGSPACE$ if and only if $\Pi \in NLOGSPACE$.
(3) $NP = LOGSPACE$ if and only if $\Pi \in LOGSPACE$.

Some Conclusions of Completeness

Fact: Let Π be a problem that is complete for the class $PSPACE$ with respect to log space reductions. Then
(1) $PSPACE = NP$ if and only if $\Pi \in NP$.
(2) $PSPACE = P$ if and only if $\Pi \in P$.

Fact: Let Π be a problem that is complete for the class $NLOGSPACE$ with respect to log space reductions. Then
$NLOGSPACE = LOGSPACE$ if and only if $\Pi \in LOGSPACE$.

Fact: Let Π be a P-complete problem. Then
(1) $P = LOGSPACE$ if and only if $\Pi \in LOGSPACE$.
(2) $P = NLOGSPACE$ if and only if $\Pi \in NLOGSPACE$.

Exercise

1. Show that the language $\{a^n b^n \mid n \geq 1\}$ is in $DTIME(n)$.

2. Show that the language $\{ww \mid w \in \{a,b\}^+\}$ is in $LOGSPACE$.

3. Show that the relation $\propto_{poly}$ is transitive.

4. Show that the relation $\propto_{log}$ is transitive.

5. Show that the problem k-Clique is in $LOGSPACE$. What if k is part of the input?