

Foundations of Complexity Theory

Lecture 1: Standard Model & Complexity Classes

October 14, 2026



Veranstaltung für
MINT-Studierende

Semesterauftakt- Tischtennis

Für Snacks und Getränke ist gesorgt

Jetzt
anmelden!



23.10, ab 18 Uhr
Inselstraße 27, 04103 Leipzig

Contents

- ➊ Basic (classic) complexity classes & their relations
(L, NL, P, NP, PSpace, NPSpace, ExpTime, NExpTime, etc.)
- ➋ Reductions & complete problems
(graph reachability, satisfiability, emptiness of context-free languages, equivalence of regular expressions, etc.)
- ➌ Circuit complexity & Razborov's theorem
(Boolean circuits, monotone circuits, AC^i , NC^i , TC^i , etc.)
- ➍ Randomization & interactive proof systems
(BPP, PP, IP, zero-knowledge, etc.)
- ➎ Polynomial primality testing (with complete proof)

- Announcements, lecture slides & exercise sheets located on

► Moodle

- Announcements, lecture slides & exercise sheets located on [Moodle](#)
- Literature for self-study & additional detail (hardcovers in library)



Christos Papadimitriou

► Computational Complexity

Addison-Wesley, 1995



Steven Homer & Alan L. Selman

► Computability and Complexity Theory

Springer-Verlag, 2nd ed., 2011



Ingo Wegener

► Komplexitätstheorie: Grenzen der Effizienz von Algorithmen

Springer-Verlag, 2003

Next Dates

LECTURE (WED 1:15PM)	
Oct 14th	Lecture 1: Standard model (multi-tape TM)
21st	Lecture 2: Fundamental relations I
28th	Lecture 3: Fundamental relations II
Nov 4th	Lecture 4: Fundamental relations III
11th	Lecture 5: Reductions & complete problems I
18th	
	(public holiday)
25th	Lecture 6: Complete problems II
Dec 2nd	
	(dies academicus)
9th	Lecture 7: NP-complete problems

LECTURE (WED 1:15PM)	
Dec 16th	Lecture 8: Complete problems III
23rd	
30th	
Jan 6th	Lecture 9: Boolean circuits I
13th	Lecture 10: Boolean circuits II
20th	Lecture 11: Razborov's theorem I
27th	
Feb 3rd	Lecture 12: Razborov's theorem II

- Cancellation of exam registration up to 11:59am on January 11, 2027

- Cancellation of exam registration up to 11:59am on January 11, 2027

Exam

- 30 min oral

Contents

- 1 Basic complexity classes & their relations
- 2 Reductions & complete problems
- 3 Boolean circuits & Razborov's theorem
- 4 Randomization & interactive proof systems
- 5 Polynomial primality test

Basic Complexity Classes

Complexity

- Based on standard model
(multi-tape Turing machine with in- & output)
- Most alternatives yield same results
(limited memory cell content essential)

Basic Complexity Classes

Complexity

- Based on standard model
(multi-tape Turing machine with in- & output)
- Most alternatives yield same results
(limited memory cell content essential)
- Complexity classes are implementation-independent
(i.e. stable under changes of computation model)

Basic Complexity Classes

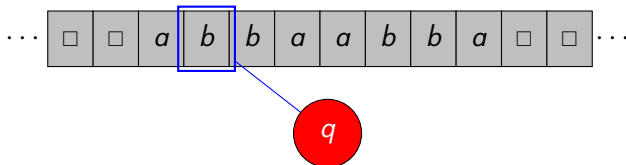
Complexity

- Based on standard model
(multi-tape Turing machine with in- & output)
- Most alternatives yield same results
(limited memory cell content essential)
- Complexity classes are implementation-independent
(i.e. stable under changes of computation model)

Notation

- Nonnegative integers $\mathbb{N} = \{0, 1, 2, 3, \dots\}$ and $\mathbb{N}_+ = \mathbb{N} \setminus \{0\}$
- $\mathcal{P}_{\neq \emptyset}(A) = \{A' \subseteq A \mid A' \neq \emptyset\}$ set of all nonempty subsets of set A
- Standard notation

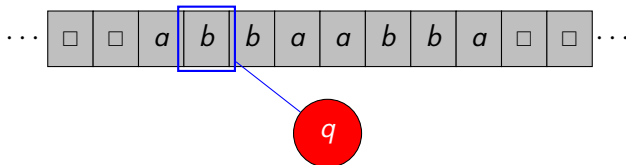
Turing Machine



Notes

- Flexible (both-sided infinite) working tapes with read & write head that can move in **both** directions
($B[i]$ with $i \in \mathbb{N}_+$ is contents of cell i on tape B)
- Finite control (state-driven; program-driven)

Turing Machine



Notes

- Flexible (both-sided infinite) working tapes with read & write head that can move in **both** directions
($B[i]$ with $i \in \mathbb{N}_+$ is contents of cell i on tape B)
- Finite control (state-driven; program-driven)
- Read head on tape with delimited input $\vdash w \vdash$

Alan Turing (* 1912; † 1954)

- Brit. computer scientist
- Broke German Enigma encryption
- Convicted for homosexuality; accepted sterilization royal pardon in 2013



§1.1 Definition (k -tape Turing machine)

Let $k \in \mathbb{N}_+$. k -tape Turing machine (short: k -TM) is tuple

$$\mathcal{M} = (Q, \Sigma, \Gamma, \Sigma', \delta, \square, q_0, q_+, q_-)$$

- Finite set Q of **states**; $q_0 \in Q$ **start state**, $q_+ \in Q$ **accepting state** and $q_- \in Q \setminus \{q_+\}$ **rejecting state**
- Alphabet Σ of **input symbols**, alphabet Γ of **working symbols**, alphabet Σ' of **output symbols**
- **Transition function**

$$\delta: (Q \setminus \{q_+, q_-\}) \times (\Sigma \cup \{\vdash, \dashv\}) \times \Gamma^k \rightarrow \mathcal{B}$$

with restrictions on next slide

- **Blank symbol** $\square \in \Gamma$

Valid transition functions δ

$$\mathcal{B} = \mathcal{P}_{\neq \emptyset}(Q \times \Gamma^k \times \{-1, 0, 1\}^{k+1} \times (\Sigma' \cup \{\varepsilon\}))$$

(‘-1’ = go left; ‘0’ = stay; ‘1’ = go right)

For all $(q', \gamma'_1, \dots, \gamma'_k, d_0, \dots, d_k, \sigma') \in \delta(q, \sigma, \gamma_1, \dots, \gamma_k)$

- $d_0 \neq -1$ if $\sigma = \vdash$ (no left movement on left delimiter of input)
- $d_k \neq 1$ if $\sigma = \dashv$ (no right movement on right delimiter of input)

Valid transition functions δ

$$\mathcal{B} = \mathcal{P}_{\neq \emptyset}(Q \times \Gamma^k \times \{-1, 0, 1\}^{k+1} \times (\Sigma' \cup \{\varepsilon\}))$$

(‘-1’ = go left; ‘0’ = stay; ‘1’ = go right)

For all $(q', \gamma'_1, \dots, \gamma'_k, d_0, \dots, d_k, \sigma') \in \delta(q, \sigma, \gamma_1, \dots, \gamma_k)$

- $d_0 \neq -1$ if $\sigma = \vdash$ (no left movement on left delimiter of input)
- $d_k \neq 1$ if $\sigma = \dashv$ (no right movement on right delimiter of input)

§1.2 Definition (Determinism)

k -TM $\mathcal{M} = (Q, \Sigma, \Gamma, \Sigma', \delta, \square, q_0, q_+, q_-)$ is **deterministic** if

$$|\delta(q, \sigma, \gamma_1, \dots, \gamma_k)| = 1$$

for all $(q, \sigma, \gamma_1, \dots, \gamma_k) \in (Q \setminus \{q_+, q_-\}) \times (\Sigma \cup \{\vdash, \dashv\}) \times \Gamma^k$

§1.3 Definition (Configuration)

Let $M = (Q, \Sigma, \Gamma, \Sigma', \delta, \square, q_0, q_+, q_-)$ k -TM and $w \in \Sigma^*$ input.

Configuration of M for w is

$$(q, B_1, \dots, B_k, i_0, \dots, i_k) \in Q \times (\Gamma^+)^k \times \mathbb{N}^{k+1}$$

with $1 \leq i_0 \leq |w| + 2$ and for all $1 \leq \ell \leq k$ let $n_\ell = |B_\ell|$ and

- $1 \leq i_\ell \leq n_\ell$ (valid working tape positions)
- $B_\ell[1] \neq \square$ if $1 < i_\ell$ (no unnecessary $\square$ on left end of tapes)
- $B_\ell[n_\ell] \neq \square$ if $i_\ell < n_\ell$ (no unnecessary $\square$ on right end of tapes)

§1.3 Definition (Configuration)

Let $M = (Q, \Sigma, \Gamma, \Sigma', \delta, \square, q_0, q_+, q_-)$ k -TM and $w \in \Sigma^*$ input.

Configuration of M for w is

$$(q, B_1, \dots, B_k, i_0, \dots, i_k) \in Q \times (\Gamma^+)^k \times \mathbb{N}^{k+1}$$

with $1 \leq i_0 \leq |w| + 2$ and for all $1 \leq \ell \leq k$ let $n_\ell = |B_\ell|$ and

- $1 \leq i_\ell \leq n_\ell$ (valid working tape positions)
- $B_\ell[1] \neq \square$ if $1 < i_\ell$ (no unnecessary $\square$ on left end of tapes)
- $B_\ell[n_\ell] \neq \square$ if $i_\ell < n_\ell$ (no unnecessary $\square$ on right end of tapes)

Length of configuration $\kappa = (q, B_1, \dots, B_k, i_0, \dots, i_k)$ is

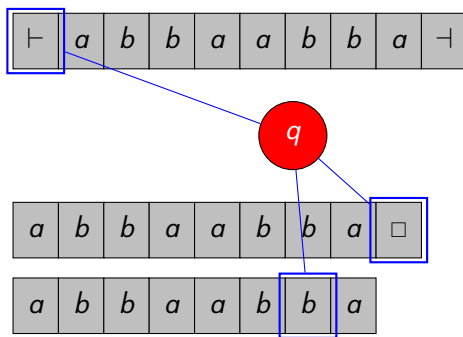
$$|\kappa| = \max \{|B_\ell| \mid 1 \leq \ell \leq k\}$$

(Length of configuration = length of longest working tape)

Turing Machine

Configuration $(q, B_1, \dots, B_k, i_0, \dots, i_k)$

- Current state q ; relevant contents $B_1, \dots, B_k$ of k working tapes
- Position i_0 of head on delimited input $\vdash w \vdash$
- Positions $i_1, \dots, i_k$ of heads on working tapes
- Input & output are not part of configuration



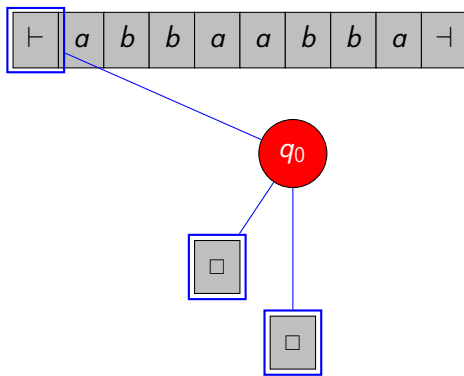
Tapes $B_1 = abbaabba\square$ and $B_2 = abbaabba$

Turing Machine

Initial configuration for input $w \in \Sigma^*$ is

$$l_w = (q_0, \square, \dots, \square, 1, \dots, 1)$$

Hence $|l_w| = 1$



Tapes $B_1 = \square$ and $B_2 = \square$

Notation: For all tapes $B \in \Gamma^*$, positions $1 \leq \ell \leq |B|$ and symbols $\gamma' \in \Gamma$

$$B[\ell \rightarrow \gamma'] = B[1] \cdots B[\ell - 1] \gamma' B[\ell + 1] \cdots B[|B|]$$

with tape contents γ' at position ℓ

Illustration of $abbaabba[4 \rightarrow b]$

<i>a</i>	<i>b</i>	<i>b</i>	<i>a</i>	<i>a</i>	<i>b</i>	<i>b</i>	<i>a</i>
----------	----------	----------	----------	----------	----------	----------	----------

Notation: For all tapes $B \in \Gamma^*$, positions $1 \leq \ell \leq |B|$ and symbols $\gamma' \in \Gamma$

$$B[\ell \rightarrow \gamma'] = B[1] \cdots B[\ell - 1] \gamma' B[\ell + 1] \cdots B[|B|]$$

with tape contents γ' at position ℓ

Illustration of $abbaabba[4 \rightarrow b]$

<i>a</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>a</i>	<i>b</i>	<i>b</i>	<i>a</i>
----------	----------	----------	----------	----------	----------	----------	----------

§1.4 Definition (Derivation step)

Let $M = (Q, \Sigma, \Gamma, \Sigma', \delta, \square, q_0, q_+, q_-)$ k -TM,
($q, B_1, \dots, B_k, i_0, \dots, i_k$) configuration of M for $w \in \Sigma^*$ and

$$\rho = (q', \gamma'_1, \dots, \gamma'_k, d_0, \dots, d_k, \sigma') \in \delta(q, (\vdash w \dashv)[i_0], B_1[i_1], \dots, B_k[i_k])$$

transition. Then

$$(q, B_1, \dots, B_k, i_0, \dots, i_k) \Rightarrow_{\rho} (q', B'_1, \dots, B'_k, i_0 + d_0, i'_1, \dots, i'_k)$$

where $(B'_\ell, i'_\ell) = \text{trim}(\square B_\ell[i_\ell \rightarrow \gamma'_\ell] \square, i_\ell + d_\ell + 1)$ for all $1 \leq \ell \leq k$

Exercise: Define 'trim' formally

Derivation step

- $\kappa' = \kappa''$ for $\kappa \Rightarrow_{\rho} \kappa'$ and $\kappa \Rightarrow_{\rho} \kappa''$
(next configuration determined by configuration κ and transition ρ)

Derivation step

- $\kappa' = \kappa''$ for $\kappa \Rightarrow_{\rho} \kappa'$ and $\kappa \Rightarrow_{\rho} \kappa''$
(next configuration determined by configuration κ and transition ρ)
- $|\kappa| - |\kappa'| \in \{-1, 0, 1\}$ for all $\kappa \Rightarrow_{\rho} \kappa'$
(length of configuration changes by at most 1 since we can expand each tape by at most one symbol and move at most one step)

Derivation step

- $\kappa' = \kappa''$ for $\kappa \Rightarrow_{\rho} \kappa'$ and $\kappa \Rightarrow_{\rho} \kappa''$
(next configuration determined by configuration κ and transition ρ)
- $|\kappa| - |\kappa'| \in \{-1, 0, 1\}$ for all $\kappa \Rightarrow_{\rho} \kappa'$
(length of configuration changes by at most 1 since we can expand each tape by at most one symbol and move at most one step)

Final configurations

Configuration $(q, B_1, \dots, B_k, i_0, \dots, i_k)$ of M for w is **accepting** (resp. **rejecting**) if $q = q_+$ (resp. $q = q_-$)

§1.5 Definition (Protocol, computation, output, language)

Let $M = (Q, \Sigma, \Gamma, \Sigma', \delta, \square, q_0, q_+, q_-)$ k -TM and $w \in \Sigma^*$ input. Additionally let $\kappa_0, \dots, \kappa_n$ configurations of M for w with $\kappa_0 = I_w$. If there exist transitions $\rho_1, \dots, \rho_n$ such that $\kappa_0 \Rightarrow_{\rho_1} \dots \Rightarrow_{\rho_n} \kappa_n$, then

- $(\kappa_0, \dots, \kappa_n)$ is **computation** (of M on input w) with **protocol** $(\rho_1, \dots, \rho_n)$

§1.5 Definition (Protocol, computation, output, language)

Let $M = (Q, \Sigma, \Gamma, \Sigma', \delta, \square, q_0, q_+, q_-)$ k -TM and $w \in \Sigma^*$ input. Additionally let $\kappa_0, \dots, \kappa_n$ configurations of M for w with $\kappa_0 = I_w$. If there exist transitions $\rho_1, \dots, \rho_n$ such that $\kappa_0 \Rightarrow_{\rho_1} \dots \Rightarrow_{\rho_n} \kappa_n$, then

- $(\kappa_0, \dots, \kappa_n)$ is **computation** (of M on input w) with **protocol** $(\rho_1, \dots, \rho_n)$
- computation $(\kappa_0, \dots, \kappa_n)$ is **accepting** if κ_n is accepting configuration

§1.5 Definition (Protocol, computation, output, language)

Let $M = (Q, \Sigma, \Gamma, \Sigma', \delta, \square, q_0, q_+, q_-)$ k -TM and $w \in \Sigma^*$ input. Additionally let $\kappa_0, \dots, \kappa_n$ configurations of M for w with $\kappa_0 = I_w$. If there exist transitions $\rho_1, \dots, \rho_n$ such that $\kappa_0 \Rightarrow_{\rho_1} \dots \Rightarrow_{\rho_n} \kappa_n$, then

- $(\kappa_0, \dots, \kappa_n)$ is **computation** (of M on input w) with **protocol** $(\rho_1, \dots, \rho_n)$
- computation $(\kappa_0, \dots, \kappa_n)$ is **accepting** if κ_n is accepting configuration
- $\sigma'_1 \dots \sigma'_n$ is **output** of protocol $(\rho_1, \dots, \rho_n)$,
where $\rho_\ell = (\dots, \sigma'_\ell)$ for all $1 \leq \ell \leq n$

§1.5 Definition (Protocol, computation, output, language)

Let $M = (Q, \Sigma, \Gamma, \Sigma', \delta, \square, q_0, q_+, q_-)$ k -TM and $w \in \Sigma^*$ input. Additionally let $\kappa_0, \dots, \kappa_n$ configurations of M for w with $\kappa_0 = I_w$. If there exist transitions $\rho_1, \dots, \rho_n$ such that $\kappa_0 \Rightarrow_{\rho_1} \dots \Rightarrow_{\rho_n} \kappa_n$, then

- $(\kappa_0, \dots, \kappa_n)$ is **computation** (of M on input w) with **protocol** $(\rho_1, \dots, \rho_n)$
- computation $(\kappa_0, \dots, \kappa_n)$ is **accepting** if κ_n is accepting configuration
- $\sigma'_1 \dots \sigma'_n$ is **output** of protocol $(\rho_1, \dots, \rho_n)$,
where $\rho_\ell = (\dots, \sigma'_\ell)$ for all $1 \leq \ell \leq n$

Language accepted by M is

$$L(M) = \{w \in \Sigma^* \mid \exists \text{ accepting computation of } M \text{ on input } w\}$$

§1.6 Definition (Time & space consumption)

Let $M = (Q, \Sigma, \Gamma, \Sigma', \delta, \square, q_0, q_+, q_-)$ k -TM and $w \in \Sigma^*$ input.

- Time resp. space consumption of computation $(\kappa_0, \dots, \kappa_n)$ is

$$n \quad \text{resp.} \quad \max \{ |\kappa_\ell| \mid 0 \leq \ell \leq n \}$$

§1.6 Definition (Time & space consumption)

Let $M = (Q, \Sigma, \Gamma, \Sigma', \delta, \square, q_0, q_+, q_-)$ k -TM and $w \in \Sigma^*$ input.

- Time resp. space consumption of computation $(\kappa_0, \dots, \kappa_n)$ is

$$n \quad \text{resp.} \quad \max \{ |\kappa_\ell| \mid 0 \leq \ell \leq n \}$$

- Time resp. space consumption of M on input w is $\ell \in \mathbb{N}$
if time resp. space consumption of each computation of M on input w
is at most ℓ

§1.6 Definition (Time & space consumption)

Let $M = (Q, \Sigma, \Gamma, \Sigma', \delta, \square, q_0, q_+, q_-)$ k -TM and $w \in \Sigma^*$ input.

- Time resp. space consumption of computation $(\kappa_0, \dots, \kappa_n)$ is

$$n \quad \text{resp.} \quad \max \{|\kappa_\ell| \mid 0 \leq \ell \leq n\}$$

- Time resp. space consumption of M on input w is $\ell \in \mathbb{N}$
if time resp. space consumption of each computation of M on input w
is at most ℓ
- M is f -time-bounded resp. f -space-bounded
for monotonic increasing function $f: \mathbb{N} \rightarrow \mathbb{N}$
if time resp. space consumption of M is at most $f(|w'| + 2)$
for each input $w' \in \Sigma^*$

Recall: Landau-Notation

For simplicity we restrict ourselves to functions $f: \mathbb{N} \rightarrow \mathbb{N}$
(with understanding that $\log n$ is actually $\lceil \log n \rceil$)

Relevant Notions

Let $f, g: \mathbb{N} \rightarrow \mathbb{N}$

- $f \in O(g)$ if there exist $c, n_0 \in \mathbb{N}$ such that $f(n) \leq c \cdot g(n)$ for all $n > n_0$

$$\limsup_{n \rightarrow \infty} \frac{f(n)}{g(n)} < \infty \quad (\text{some multiple of } g \text{ eventually upper bounds } f)$$

Recall: Landau-Notation

For simplicity we restrict ourselves to functions $f: \mathbb{N} \rightarrow \mathbb{N}$
(with understanding that $\log n$ is actually $\lceil \log n \rceil$)

Relevant Notions

Let $f, g: \mathbb{N} \rightarrow \mathbb{N}$

- $f \in O(g)$ if there exist $c, n_0 \in \mathbb{N}$ such that $f(n) \leq c \cdot g(n)$ for all $n > n_0$

$$\limsup_{n \rightarrow \infty} \frac{f(n)}{g(n)} < \infty \quad (\text{some multiple of } g \text{ eventually upper bounds } f)$$

- $f \in o(g)$ if for each $\epsilon > 0$ there exists $n_0 \in \mathbb{N}$ such that $f(n) < \epsilon \cdot g(n)$ for all $n > n_0$

$$\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = 0 \quad (\text{every multiple of } g \text{ eventually dominates } f)$$

Recall: Landau-Notation

For simplicity we restrict ourselves to functions $f: \mathbb{N} \rightarrow \mathbb{N}$
(with understanding that $\log n$ is actually $\lceil \log n \rceil$)

Relevant Notions

Let $f, g: \mathbb{N} \rightarrow \mathbb{N}$

- $f \in O(g)$ if there exist $c, n_0 \in \mathbb{N}$ such that $f(n) \leq c \cdot g(n)$ for all $n > n_0$

$$\limsup_{n \rightarrow \infty} \frac{f(n)}{g(n)} < \infty \quad (\text{some multiple of } g \text{ eventually upper bounds } f)$$

- $f \in o(g)$ if for each $\epsilon > 0$ there exists $n_0 \in \mathbb{N}$ such that $f(n) < \epsilon \cdot g(n)$ for all $n > n_0$

$$\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = 0 \quad (\text{every multiple of } g \text{ eventually dominates } f)$$

- $f \in \Omega(g)$ if $g \in O(f)$ (some multiple of f eventually upper bounds g)

Number of Configurations

§1.7 Lemma (Approximation of number of configurations)

Let $M = (Q, \Sigma, \Gamma, \Sigma', \delta, \square, q_0, q_+, q_-)$ k -TM.

There exist constants $a, b \in \mathbb{N}$ such that for all inputs $w \in \Sigma^+$

- There exists at most $a \cdot |w| \cdot b^n$ configurations of M on input w that have length at most $n \in \mathbb{N}$

Number of Configurations

§1.7 Lemma (Approximation of number of configurations)

Let $M = (Q, \Sigma, \Gamma, \Sigma', \delta, \square, q_0, q_+, q_-)$ k -TM.

There exist constants $a, b \in \mathbb{N}$ such that for all inputs $w \in \Sigma^+$

- There exists at most $a \cdot |w| \cdot b^n$ configurations of M on input w that have length at most $n \in \mathbb{N}$
- If M is f -space-bounded, then number of configurations reachable from initial configuration I_w is at most $a \cdot |w| \cdot b^{f(|w|+2)}$

For $f \in \Omega(\log x)$ (i.e., $\log x \in O(f)$) we thus have $g \in 2^{O(f)}$ such that there exist at most $g(|w| + 2)$ such configurations

Note

- Inside Landau's $O(\cdot)$, $\Omega(\cdot)$, $\Theta(\cdot)$ we use x as anonymous variable
e.g. $f \in O(x^2 \log x)$ means $f \in O(g)$ where $g(x) = x^2 \log x$

Number of Configurations

Proof.

Obviously there are at most $|Q| \cdot (\sum_{\ell=1}^n |\Gamma|^\ell)^k \cdot (|w| + 2) \cdot n^k$ configurations of length at most n .

Number of Configurations

Proof.

Obviously there are at most $|Q| \cdot (\sum_{\ell=1}^n |\Gamma|^\ell)^k \cdot (|w| + 2) \cdot n^k$ configurations of length at most n . Additionally

- $\sum_{\ell=1}^n |\Gamma|^\ell \leq (|\Gamma| + 1)^n$
(expand shorter words with extra symbol to length n)

Number of Configurations

Proof.

Obviously there are at most $|Q| \cdot (\sum_{\ell=1}^n |\Gamma|^\ell)^k \cdot (|w| + 2) \cdot n^k$ configurations of length at most n . Additionally

- $\sum_{\ell=1}^n |\Gamma|^\ell \leq (|\Gamma| + 1)^n$
(expand shorter words with extra symbol to length n)
- $|w| + 2 \leq 3|w|$ since $|w| \geq 1$
- there exists $c \in \mathbb{N}$ such that $n^k \leq c^n$ for all $n \in \mathbb{N}$ (select $c = 2^k$)

Number of Configurations

Proof.

Obviously there are at most $|Q| \cdot (\sum_{\ell=1}^n |\Gamma|^\ell)^k \cdot (|w| + 2) \cdot n^k$ configurations of length at most n . Additionally

- $\sum_{\ell=1}^n |\Gamma|^\ell \leq (|\Gamma| + 1)^n$
(expand shorter words with extra symbol to length n)
- $|w| + 2 \leq 3|w|$ since $|w| \geq 1$
- there exists $c \in \mathbb{N}$ such that $n^k \leq c^n$ for all $n \in \mathbb{N}$ (select $c = 2^k$)

Thus there exist at most $3|Q| \cdot |w| \cdot ((|\Gamma| + 1)^k)^n \cdot c^n$ configurations of length at most n . Setting $a = 3|Q|$ and $b = (|\Gamma| + 1)^k \cdot c$ thus proves the first item.

Number of Configurations

Proof.

Obviously there are at most $|Q| \cdot (\sum_{\ell=1}^n |\Gamma|^\ell)^k \cdot (|w| + 2) \cdot n^k$ configurations of length at most n . Additionally

- $\sum_{\ell=1}^n |\Gamma|^\ell \leq (|\Gamma| + 1)^n$
(expand shorter words with extra symbol to length n)
- $|w| + 2 \leq 3|w|$ since $|w| \geq 1$
- there exists $c \in \mathbb{N}$ such that $n^k \leq c^n$ for all $n \in \mathbb{N}$ (select $c = 2^k$)

Thus there exist at most $3|Q| \cdot |w| \cdot ((|\Gamma| + 1)^k)^n \cdot c^n$ configurations of length at most n . Setting $a = 3|Q|$ and $b = (|\Gamma| + 1)^k \cdot c$ thus proves the first item. For the final claim we note that the function $g: \mathbb{N} \rightarrow \mathbb{N}$ given by $g(n) = a \cdot n \cdot b^{f(n)} = b^{\log_b(a) + \log_b(n) + f(n)}$ for all $n \in \mathbb{N}$ belongs to

$$g \in b^{O(f)} = 2^{O(f)} \quad \text{because } \log_b(a) \text{ is constant and } f \in \Omega(\log x)$$

□

§1.8 Definition (Complexity classes)

Let $f: \mathbb{N} \rightarrow \mathbb{N}$ monotonic increasing function

$$\text{DTIME}(f) = \{L(M) \mid M \text{ deterministic \& } f\text{-time-bounded TM}\}$$

$$\text{NTIME}(f) = \{L(M) \mid M \text{ } f\text{-time-bounded TM}\}$$

$$\text{DSpace}(f) = \{L(M) \mid M \text{ deterministic \& } f\text{-space-bounded TM}\}$$

$$\text{NSpace}(f) = \{L(M) \mid M \text{ } f\text{-space-bounded TM}\}$$

§1.8 Definition (Complexity classes)

Let $f: \mathbb{N} \rightarrow \mathbb{N}$ monotonic increasing function

$$\text{DTIME}(f) = \{L(M) \mid M \text{ deterministic \& } f\text{-time-bounded TM}\}$$

$$\text{NTIME}(f) = \{L(M) \mid M \text{ } f\text{-time-bounded TM}\}$$

$$\text{DSpace}(f) = \{L(M) \mid M \text{ deterministic \& } f\text{-space-bounded TM}\}$$

$$\text{NSpace}(f) = \{L(M) \mid M \text{ } f\text{-space-bounded TM}\}$$

Additionally for every class $\mathcal{C}$ of languages we let

$$\text{co-}\mathcal{C} = \{\bar{L} \mid L \in \mathcal{C}\}$$

be set of complements $\bar{L}$ of elements L of $\mathcal{C}$

Complexity classes

- $\text{DTIME}(f)$ and $\text{NTIME}(f)$ only considered for functions $f: \mathbb{N} \rightarrow \mathbb{N}$ with $f(n) \geq n$ for all $n \in \mathbb{N}$ (allows to read the whole input)
- $\text{DSpace}(f)$ and $\text{NSpace}(f)$ only considered for functions $f \in \Omega(\log x)$ (allows to store 1 position in input)

Complexity Classes

Standard classes

$$\mathbf{L} = \mathbf{DSpace}(\log x)$$

$$\mathbf{NL} = \mathbf{NSpace}(\log x)$$

$$\mathbf{P} = \bigcup_{i \in \mathbb{N}_+} \mathbf{DTIME}(x^i)$$

$$\mathbf{NP} = \bigcup_{i \in \mathbb{N}_+} \mathbf{NTIME}(x^i)$$

$$\mathbf{PSPACE} = \bigcup_{i \in \mathbb{N}_+} \mathbf{DSpace}(x^i) \stackrel{\dagger}{=} \bigcup_{i \in \mathbb{N}_+} \mathbf{NSpace}(x^i)$$

$$\mathbf{EXPTIME} = \bigcup_{i \in \mathbb{N}_+} \mathbf{DTIME}(2^{x^i})$$

$$\mathbf{NEXPTIME} = \bigcup_{i \in \mathbb{N}_+} \mathbf{NTIME}(2^{x^i})$$

$\dagger$ = Savitch's theorem

Example

$$\{a^n b^n c^n d^n \mid n \in \mathbb{N}_+\} \in \mathbf{L}$$

Working principle

- Read initial sequence a^n and store n in binary on tapes 1, 2, and 3

Example

$$\{a^n b^n c^n d^n \mid n \in \mathbb{N}_+\} \in \mathbf{L}$$

Working principle

- Read initial sequence a^n and store n in binary on tapes 1, 2, and 3
- Read symbols b while decrementing tape 1
- Check count (tape 1 contains 0)

Example

$$\{a^n b^n c^n d^n \mid n \in \mathbb{N}_+\} \in \mathbf{L}$$

Working principle

- Read initial sequence a^n and store n in binary on tapes 1, 2, and 3
- Read symbols b while decrementing tape 1
- Check count (tape 1 contains 0)
- Read symbols c while decrementing tape 2
- Check count (tape 2 contains 0)

Complexity Classes

Example

$$\{a^n b^n c^n d^n \mid n \in \mathbb{N}_+\} \in \mathbf{L}$$

Working principle

- Read initial sequence a^n and store n in binary on tapes 1, 2, and 3
- Read symbols b while decrementing tape 1
- Check count (tape 1 contains 0)
- Read symbols c while decrementing tape 2
- Check count (tape 2 contains 0)
- Read symbols d while decrementing tape 3
- Check count (tape 3 contains 0) and accept
- Otherwise reject

Example

$\text{PRIM} = \{w \in 1\{0,1\}^* \mid w \text{ prime coded in binary}\} \in \text{DSPACE}(x)$

Working principle

- Test all smaller numbers as divisor by iterated subtraction
(copy input as minuend and divisor candidate as subtrahend;
decrement both until subtrahend is 0; then set subtrahend again to
divisor candidate)

Example

$\text{PRIM} = \{w \in \{0,1\}^* \mid w \text{ prime coded in binary}\} \in \text{DSPACE}(x)$

Working principle

- Test all smaller numbers as divisor by iterated subtraction
(copy input as minuend and divisor candidate as subtrahend;
decrement both until subtrahend is 0; then set subtrahend again to
divisor candidate)
- Tape 1: current divisor candidate
- Tape 2: current minuend (for iterated subtraction)
- Tape 3: current subtrahend (for iterated subtraction)

Summary

- Turing machines
- Time & space consumption
- Simple complexity classes

First exercise sheet available in Moodle