Complexity Theory

Jan Křetínský

Chair for Foundations of Software Reliability and Theoretical Computer Science
Technical University of Munich

Summer 2016

Based on slides by Jörg Kreiker

Lecture 3

Basic Complexity Classes

Agenda

- universal Turing machine
- decision vs. search
- computability, halting problem
- basic complexity classes
    - time and space
    - deterministic and non-deterministic

Universal TM

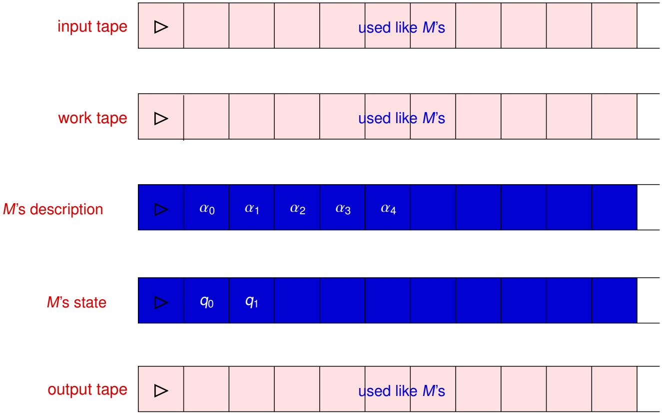
- TMs can be represented as strings (over $\{0, 1\}$) by encoding their transition function (can you?)
  - write M_α for TM represented by string α
  - every string α represents some TM
  - every TM has infinitely many representations
- if TM M computes f, universal TM $\mathcal{U}$ takes representation α of TM M and input x and computes $f(x)$
- like general purpose computer loaded with software
- like interpreter for a language written in same language
- $\mathcal{U}$ has bounded alphabet, rules, tapes; simulates much larger machines efficiently

Efficient simulation

Theorem (Universal TM)

There exists a TM $\mathcal{U}$ such that for every $x, \alpha \in \{0,1\}^*$, $\mathcal{U}(x,\alpha) = M_\alpha(x)$. If M_α holds on x within T steps, then $\mathcal{U}(x,\alpha)$ holds within $\mathcal{O}(T \log T)$ steps.

Construction of $\mathcal{U}$



Simulating another TM

How does $\mathcal{U}$ execute TM M?

Simulating another TM

How does $\mathcal{U}$ execute TM M?

1. transform M into M' with one input, one work, and one output tape computing the same function — quadratic overhead
2. write M''s description α onto third tape — $|M'|$
3. write encoding of M' start state on fourth tape — $|Q'|$
4. for each step of M'
   - 4.1 depending on state and tapes of M' scan δ' tape — $|\delta'|$
   - 4.2 update — constant

Simulation can be done with logarithmic slowdown using clever encoding of k tapes in one.

Decision vs. Search

- often one is interested in functions $f : \{0,1\}^* \to \{0,1\}$
- f can be identified with the language $L_f = \{x \in \{0,1\}^* \mid f(x) = 1\}$
- TM that computes f is said to decide L_f (and vice versa)

Example (Indset)

Consider the independent set problem.

Search What is the largest independent set of a graph?

Decision $\mathrm{Indset} = \{\langle G, k\rangle \mid G \text{ has independent set of size } k\}$

Often decision plus binary search can solve search problems.

Halting Problem

There are languages that cannot be decided by any TM regardless time and space.

Example

The halting problem is the set of pairs of TM representations and inputs, such that the TMs eventually halt on the given input.

$$\text{Halt} = \{\langle \alpha, x \rangle \mid M_\alpha \text{ halts on } x\}$$

Theorem

Halt *is not decidable by any TM.*

Proof: diagonalization and reduction

Agenda

- universal Turing machine ✓
- decision vs. search ✓
- computability, halting problem ✓
- basic complexity classes
    - time and space
    - deterministic and non-deterministic

Time complexity

Definition (DTIME)

Let $T : \mathbb{N} \to \mathbb{N}$ be a function. $L \subseteq \{0,1\}^*$ is in **DTIME**(T) if there exists a TM deciding L in time T' for $T' \in \mathcal{O}(T)$.

- D refers to deterministic
- constants are ignored since TM can be sped up by arbitrary constants

Space complexity

Definition (SPACE)

Let $S : \mathbb{N} \to \mathbb{N}$ and $L \subseteq \{0,1\}^*$. Define $L \in \mathbf{SPACE}(S)$ iff

- there exists a TM M deciding L
- no more than $S'(n)$ locations on M's work tapes ever visited during computations on every input of length n for $S' \in \mathcal{O}(S)$

Remarks

- more detailed definition (cf. exercises): count non-$\square$ symbols, where $\square$ must not be written
- constants do not matter
- as for time complexity, require space-constructible bounds
  - S is space-constructible: there is TM M computing $S(|x|)$ in $\mathcal{O}(S(|x|))$ space on input x
  - TM knows its bounds
- work tape restrictions: allows to store input
- space bounds $< n$ make sense (as opposed to time)
- require space $\log n$ to remember positions in input

Non-deterministic TMs

Definition (NDTM)

A non-deterministic TM (NDTM) is a triple (Γ, Q, δ) like a deterministic TM except

- Q contains a distinguished state q_{accept}
- δ is a pair (δ_0, δ_1) of transition functions

- in each step, NDTM non-deterministically chooses to apply either δ_0 or δ_1
- NDTM M accepts x, $M(x) = 1$ if there exists a sequence of choices s.t. M reaches q_{accept}
- $M(x) = 0$ if every sequence of choices makes M halt without reaching q_{accept}

On non-determinism

- not supposed to model realistic devices
- remember impact of non-determinism finite state machines, pushdown automata
- NDTM compute the same functions as DTM (why?)
- non-determinism $\sim$ guessing

Non-deterministic complexity
Define **NTIME**(T) and **NSPACE**(S) such that T and S are bounds regardless of non-deterministic choices.

Basic complexity classes

deterministic — time — non-deterministic

P	=	$\bigcup_{p\geq 1}$ **DTIME**(n^p)	**NP**	=	$\bigcup_{p\geq 1}$ **NTIME**(n^p)
EXP	=	$\bigcup_{p\geq 1}$ **DTIME**(2^{n^p})	**NEXP**	=	$\bigcup_{p\geq 1}$ **NTIME**(2^{n^p})

space

L	=	**SPACE**$(\log n)$	**NL**	=	**NSPACE**$(\log n)$
PSPACE	=	$\bigcup_{p>0}$ **SPACE**(n^p)	**NPSPACE**	=	$\bigcup_{p>0}$ **NSPACE**(n^p)

Interesting examples

Most examples are the hardest within a given complexity class. They are complete for the class (wrt suitable reductions).

Interesting examples

Most examples are the hardest within a given complexity class. They are complete for the class (wrt suitable reductions).

L: essentially constant number of pointers into input plus logarithmically many boolean flags

- UPath $= \{\langle G, s, t\rangle \mid \exists$a path from s to t in **undirected** graph $G\}$ [Reingold 2004]
- Even $= \{x \mid x$ has an even number of 1s$\}$

Interesting examples

Most examples are the hardest within a given complexity class. They are complete for the class (wrt suitable reductions).

L: essentially constant number of pointers into input plus logarithmically many boolean flags

- UPath $= \{\langle G, s, t\rangle \mid \exists$a path from s to t in **undirected** graph $G\}$ [Reingold 2004]
- Even $= \{x \mid x$ has an even number of 1s$\}$

NL: **L** plus guessing, read-once certificates

- Path $= \{\langle G, s, t\rangle \mid \exists$a path from s to t in **directed** graph $G\}$
- 2SAT $= \{\varphi \mid$
  φ satisfiable Boolean formula in CNF with two literals per clause $\}$

Interesting examples

P: polynomial time, tractable, low-level choices of TM definitions are immaterial to **P**

- Circuit – Eval $= \{\langle C, x\rangle \mid C$ is a $n-$in$/1-$out circuit, $x$satisfying signals$\}$
- Primes $= \{x \mid x$ prime$\}$ [AKS 2004]
- many graph problems like DFS and BFS

Interesting examples

P: polynomial time, tractable, low-level choices of TM definitions are immaterial to **P**

- $\text{Circuit} - \text{Eval} = \{\langle C, x\rangle \mid C$ is a $n-$in$/1-$out circuit, $x$satisfying signals$\}$
- $\text{Primes} = \{x \mid x \text{ prime}\}$ [AKS 2004]
- many graph problems like DFS and BFS

NP: polynomially verifiable certificates, puzzles

- $\text{Indset} = \{\langle G, k\rangle \mid G$ has an independent set of size $k\}$
- $3-\text{Coloring} = \{G \mid G$ is 3-colorable$\}$
- $3\text{SAT} = \{\varphi \mid \varphi$ satisfiable Boolean formula in CNF with three literals per clause $\}$

Interesting examples

P: polynomial time, tractable, low-level choices of TM definitions are immaterial to **P**

- $\text{Circuit} - \text{Eval} = \{\langle C, x\rangle \mid C \text{ is a } n{-}\text{in}/1{-}\text{out circuit, } x \text{satisfying signals}\}$
- $\text{Primes} = \{x \mid x \text{ prime}\}$ [AKS 2004]
- many graph problems like DFS and BFS

NP: polynomially verifiable certificates, puzzles

- $\text{Indset} = \{\langle G, k\rangle \mid G \text{ has an independent set of size } k\}$
- $3{-}\text{Coloring} = \{G \mid G \text{ is 3-colorable}\}$
- $\text{3SAT} = \{\varphi \mid \varphi$ satisfiable Boolean formula in CNF with three literals per clause $\}$

EXP: exponential-time, for instance the language
$\text{Halt}_{\text{k}} = \{\langle M, x, k\rangle \mid \text{DTM } M \text{ stops on input } x \text{ within } k \text{ steps}\}$

Interesting examples

P: polynomial time, tractable, low-level choices of TM definitions are immaterial to **P**

- $\mathrm{Circuit-Eval} = \{\langle C, x\rangle \mid C$ is a $n-$in$/1-$out circuit, $x$satisfying signals$\}$
- $\mathrm{Primes} = \{x \mid x$ prime$\}$ [AKS 2004]
- many graph problems like DFS and BFS

NP: polynomially verifiable certificates, puzzles

- $\mathrm{Indset} = \{\langle G, k\rangle \mid G$ has an independent set of size $k\}$
- $3-\mathrm{Coloring} = \{G \mid G$ is 3-colorable$\}$
- $\mathrm{3SAT} = \{\varphi \mid \varphi$ satisfiable Boolean formula in CNF with three literals per clause $\}$

EXP: exponential-time, for instance the language
$\mathrm{Halt}_{\mathrm{k}} = \{\langle M, x, k\rangle \mid$ DTM M stops on input x within k steps $\}$

PSPACE: polynomial space, games, for instance
$\mathrm{TQBF} = \{Q_1 x_1 \ldots Q_k x_k \varphi \mid k \geq 0, Q_i \in \{\forall, \exists\}, \varphi$ Boolean formula over x_i such that whole formula is true $\}$

Complements

Definition (Complement classes)

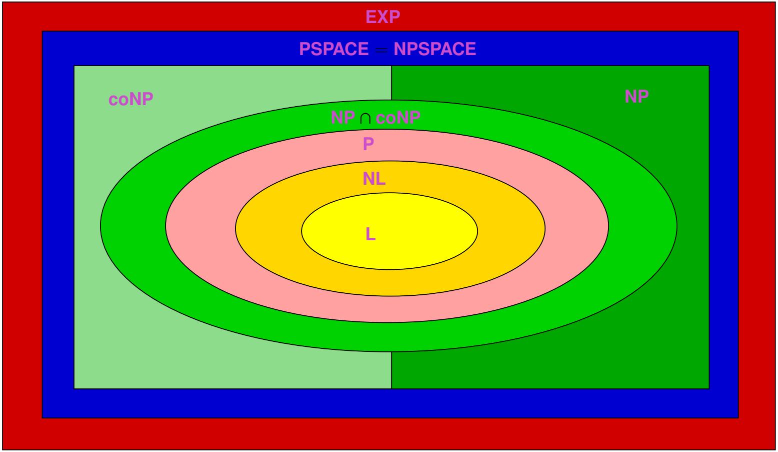
Let $\mathcal{C} \subseteq \mathcal{P}(\{0,1\}^*)$ be a complexity class. We define $co\mathcal{C} = \{\overline{L} \mid L \in \mathcal{C}\}$ to be the complement class of $\mathcal{C}$, where $\overline{L} = \{0,1\}^* \setminus L$ is the complement of L.

- important class **coNP**
- **coNP** is not the complement of **P**
- example: Tautology $\in$ **coNP**, where a tautology is Boolean formula that is true for every assignment
- reminder: closure under complement wrt expressiveness and conciseness
  - finite state machines
  - pushdown automata
  - DTM, NDTM
- note: **P** $\subseteq$ **NP** $\cap$ **coNP**

Agenda

- universal Turing machine ✓
- decision vs. search ✓
- computability, halting problem ✓
- basic complexity classes ✓

Relation between classes



What have we learnt?

- TM can be represented as strings; universal TM can simulate any TM given its representations with polynomial overhead only
- uncomputable functions do exist (halting problem): diagonalization and reductions
- non-deterministic TMs
- space, time, deterministic, non-deterministic, complement complexity classes
- **L**, **NL**, **P**, **NP**, **EXP**, **PSPACE**
- 2SAT, 3SAT, Path, UPath, TQBF, Primes, Indset, 3−Coloring
- big picture
- up next: justify and explore the big picture