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 7

Hierarchies

Regular Expression Equivalence – Recap

A regular expression over $\{0, 1\}$ is defined by

$$r ::= 0 \mid 1 \mid rr \mid r|r \mid r \cap r \mid r^*$$

The language defined by r is written $\mathcal{L}(r)$.

Regular Expression Equivalence – Recap

A regular expression over $\{0, 1\}$ is defined by

$$r ::= 0 \mid 1 \mid rr \mid r|r \mid r \cap r \mid r^*$$

The language defined by r is written $\mathcal{L}(r)$.

- let $\varphi = C_1 \wedge \ldots \wedge C_m$ be a Boolean formula in 3CNF over variables $x_1, \ldots, x_n$

Regular Expression Equivalence – Recap

A regular expression over $\{0, 1\}$ is defined by

$$r ::= 0 \mid 1 \mid rr \mid r|r \mid r \cap r \mid r^*$$

The language defined by r is written $\mathcal{L}(r)$.

- let $\varphi = C_1 \wedge \ldots \wedge C_m$ be a Boolean formula in 3CNF over variables $x_1, \ldots, x_n$
- compute from φ a regular expression: $f(\varphi)$=$(\alpha_1|\alpha_2|\ldots|\alpha_m)$

Regular Expression Equivalence – Recap

A regular expression over $\{0, 1\}$ is defined by

$$r ::= 0 \mid 1 \mid rr \mid r|r \mid r \cap r \mid r^*$$

The language defined by r is written $\mathcal{L}(r)$.

- let $\varphi = C_1 \wedge \ldots \wedge C_m$ be a Boolean formula in 3CNF over variables $x_1, \ldots, x_n$
- compute from φ a regular expression: $f(\varphi)$=$(\alpha_1|\alpha_2|\ldots|\alpha_m)$
- $\alpha_i = \gamma_{i1} \ldots \gamma_{in}$

Regular Expression Equivalence – Recap

A regular expression over $\{0, 1\}$ is defined by

$$r ::= 0 \mid 1 \mid rr \mid r|r \mid r \cap r \mid r^*$$

The language defined by r is written $\mathcal{L}(r)$.

- let $\varphi = C_1 \wedge \ldots \wedge C_m$ be a Boolean formula in 3CNF over variables $x_1, \ldots, x_n$
- compute from φ a regular expression: $f(\varphi)$=$(\alpha_1|\alpha_2|\ldots|\alpha_m)$
- $\alpha_i = \gamma_{i1} \ldots \gamma_{in}$
- $\gamma_{ij} = \begin{cases} 0 & x_j \in C_i \\ 1 & \overline{x_j} \in C_i \\ (0|1) & \text{otherwise} \end{cases}$

Regular Expression Equivalence – Recap

A regular expression over $\{0, 1\}$ is defined by

$$r ::= 0 \mid 1 \mid rr \mid r|r \mid r \cap r \mid r^*$$

The language defined by r is written $\mathcal{L}(r)$.

- let $\varphi = C_1 \wedge \ldots \wedge C_m$ be a Boolean formula in 3CNF over variables $x_1, \ldots, x_n$
- compute from φ a regular expression: $f(\varphi)$=$(\alpha_1|\alpha_2|\ldots|\alpha_m)$
- $\alpha_i = \gamma_{i1} \ldots \gamma_{in}$
- $\gamma_{ij} = \begin{cases} 0 & x_j \in C_i \\ 1 & \overline{x_j} \in C_i \\ (0|1) & \text{otherwise} \end{cases}$
- example: $(x \vee y \vee \overline{z}) \wedge (\overline{y} \vee z \vee w)$ transformed to $(001(0|1)) \mid ((0|1)100)$

Regular Expression Equivalence – Recap

A regular expression over $\{0, 1\}$ is defined by

$$r ::= 0 \mid 1 \mid rr \mid r|r \mid r \cap r \mid r^*$$

The language defined by r is written $\mathcal{L}(r)$.

- let $\varphi = C_1 \wedge \ldots \wedge C_m$ be a Boolean formula in 3CNF over variables $x_1, \ldots, x_n$
- compute from φ a regular expression: $f(\varphi)$=$(\alpha_1|\alpha_2|\ldots|\alpha_m)$
- $\alpha_i = \gamma_{i1} \ldots \gamma_{in}$
- $\gamma_{ij} = \begin{cases} 0 & x_j \in C_i \\ 1 & \overline{x_j} \in C_i \\ (0|1) & \text{otherwise} \end{cases}$
- example: $(x \vee y \vee \overline{z}) \wedge (\overline{y} \vee z \vee w)$ transformed to $(001(0|1)) \mid (0|1)100)$
- observe: φ is unsatisfiable iff $f(\varphi) = \{0, 1\}^n$

Agenda

- proof of Ladner's theorem
- deterministic time hierarchy theorem
- non-deterministic time hierarchy theorem
- space hierarchy theorem
- relation between space and time

Ladner's Theorem

NP-intermediate languages do exist!

Theorem (Ladner)

If **P** $\neq$ **NP** *then there exists a language* $L \subseteq$ **NP** $\setminus$ **P** *that is not* **NP***-complete.*

Proof Roadmap

1. **P** $\neq$ **NP** implies SAT $\notin$ **P**
2. construct language $L \in$ **NP** such that
   - 2.1 $L \notin$ **P**
   - 2.2 L not **NP**-complete
3. $L = \{\varphi 0 1^{f(n)-n-1} \mid \varphi \in \text{SAT}, |\varphi| = n\}$ padding SAT
4. f and L constructed by diagonalization by enumerating all languages in **P**
5. show that $L \in$ **P** implies SAT $\in$ **P** (contradiction!)
6. assume L is **NP**-complete, then there is a polynomial reduction from SAT, which yields a polynomial algorithm to decide SAT (contradiction!)

Agenda

- proof of Ladner's theorem ✓
- deterministic time hierarchy theorem
- non-deterministic time hierarchy theorem
- space hierarchy theorem
- relation between space and time

Time Hierarchy Theorem

Theorem (Time Hierarchy)

Let $f, g : \mathbb{N} \to \mathbb{N}$ be time-constructible such that $f \cdot \log f \in o(g)$. Then $\mathbf{DTIME}(f(n)) \subset \mathbf{DTIME}(g(n))$.

Time Hierarchy Theorem

Theorem (Time Hierarchy)

Let $f, g : \mathbb{N} \to \mathbb{N}$ be time-constructible such that $f \cdot \log f \in o(g)$. Then **DTIME**$(f(n)) \subset$ **DTIME**$(g(n))$.

- inclusion is strict
- proof: diagonalization, simulate M_x on x for $g(|x|)$ steps
- shows that **P** does not collapse to level k
- logarithmic factor due to slowdown in universal simulation
- corollary: **P** $\subset$ **EXP**

Non-deterministic versions

Theorem (Time Hierarchy (non-det))

Let $f, g : \mathbb{N} \to \mathbb{N}$ be time-constructible such that $f(n+1) \in o(g(n))$. Then $\mathbf{NTIME}(f(n)) \subset \mathbf{NTIME}(g(n))$.

Non-deterministic versions

Theorem (Time Hierarchy (non-det))

Let $f, g : \mathbb{N} \to \mathbb{N}$ be time-constructible such that $f(n+1) \in o(g(n))$. Then $\mathbf{NTIME}(f(n)) \subset \mathbf{NTIME}(g(n))$.

- inclusion is strict
- proof by lazy diagonalization (see: AB Th. 3.2)
- note: proof of deterministic theorem does not carry over

Space Hierarchy Theorem

Theorem (Space Hierarchy)

Let $f, g : \mathbb{N} \to \mathbb{N}$ be space-constructible such that $f \in o(g)$. Then $\mathbf{SPACE}(f(n)) \subset \mathbf{SPACE}(g(n))$.

Space Hierarchy Theorem

Theorem (Space Hierarchy)

Let $f, g : \mathbb{N} \to \mathbb{N}$ be space-constructible such that $f \in o(g)$. Then $\mathbf{SPACE}(f(n)) \subset \mathbf{SPACE}(g(n))$.

- inclusion is strict
- stronger theorem than corresponding time theorem
  - only constant space overhead
  - f, g can be logarithmic too
- proof analogous to deterministic time hierarchy
- corollary: $\mathbf{L} \subset \mathbf{PSPACE}$

Agenda

- proof of Ladner's theorem ✓
- deterministic time hierarchy theorem ✓
- non-deterministic time hierarchy theorem ✓
- space hierarchy theorem ✓
- relation between space and time

Relation between time and space

Theorem (Time vs. Space)

Let $s : \mathbb{N} \to \mathbb{N}$ be space-constructible. Then

$$\mathbf{DTIME}(s(n)) \subseteq \mathbf{SPACE}(s(n)) \subseteq \mathbf{NSPACE}(s(n)) \subseteq \mathbf{DTIME}(2^{O(s(n))})$$

Relation between time and space

Theorem (Time vs. Space)

Let $s : \mathbb{N} \to \mathbb{N}$ be space-constructible. Then

$$\mathbf{DTIME}(s(n)) \subseteq \mathbf{SPACE}(s(n)) \subseteq \mathbf{NSPACE}(s(n)) \subseteq \mathbf{DTIME}(2^{O(s(n))})$$

- inclusions are non-strict
- first two are obvious
- third inclusion requires notion of configuration graphs
- first inclusion can be strengthened to $\mathbf{DTIME}(s(n)) \subseteq \mathbf{SPACE}(\frac{s(n)}{\log n})$

Configuration Graphs

Let M be a deterministic or non-deterministic TM using $s(n)$ space. Let x be some input.

Configuration Graphs

Let M be a deterministic or non-deterministic TM using $s(n)$ space. Let x be some input.

- this induces a configuration graph $G(M, x)$
- nodes are configuration
  - state
  - content of work tapes
- edges are transitions (steps) that M can take

Properties of configuration graph

- outdegree of $G(M, x)$ is 1 if M is deterministic; 2 if M is non-deterministic
- $G(M, x)$ has at most $|Q| \cdot \Gamma^{c \cdot s(n)}$ nodes (c some constant)
- which is in $2^{\mathcal{O}(s(n))}$
- $G(M, x)$ can be made to have unique source and sink
- acceptance $\sim$ existence of path from source to sink
- which can be checked in time $\mathcal{O}(G(M, x))$

Properties of configuration graph

- outdegree of $G(M, x)$ is 1 if M is deterministic; 2 if M is non-deterministic
- $G(M, x)$ has at most $|Q| \cdot \Gamma^{c \cdot s(n)}$ nodes (c some constant)
- which is in $2^{\mathcal{O}(s(n))}$
- $G(M, x)$ can be made to have unique source and sink
- acceptance $\sim$ existence of path from source to sink
- which can be checked in time $\mathcal{O}(G(M, x))$

$\Rightarrow$ **NSPACE**$(s(n)) \subseteq$ **DTIME**$(2^{\mathcal{O}(s(n))})$ (using BFS)

References

- regular expression inequivalence from *Schöning* Theoretische Informatik – kurzgefasst
- the proof of Ladner's theorem given here follows AB, Th. 3.3
- nice survey, see `blog.computationalcomplexity.org/media/ladner.pdf`
- original proof of time hierarchy by *Hartmanis and Stearns* On the computational complexity of algorithms in Transactions of the American Mathematical Society 117.
- non-det time hierarchy by *Stephen Cook*: A hierarchy for nondeterministic time complexity in 4th annual ACM Symposium on Theory of Computing.
- stronger result on time vs space using pebble games by *Hopcroft, Paul, and Valiant* On time versus space in Journal of the ACM 24(2):332-337, April 1977.

Summary

- a lot of diagonalization
- Ladner: **NP**-intermediate languages exist
- $f \cdot \log f \in o(g)$ implies $\mathbf{DTIME}(f(n)) \subset \mathbf{DTIME}(g(n))$
- $f \in o(g)$ implies $\mathbf{SPACE}(f(n)) \subset \mathbf{SPACE}(g(n))$
- $\mathbf{DTIME}(f(n)) \subseteq \mathbf{SPACE}(s(n)) \subseteq \mathbf{NSPACE}(s(n)) \subseteq \mathbf{DTIME}(2^{O(s(n))})$
- $\mathbf{P} \subset \mathbf{EXP}$ and $\mathbf{L} \subset \mathbf{PSPACE}$

Next time: **PSPACE**