General Approach to Undecidability and Computational Complexity

Martin Čermák, Jiří Koutný and Alexander Meduna

Deparment of Information Systems
Faculty of Informatin Technology
Brno University of Technology, Faculty of Information Technology
Božetěchova 2, Brno 612 00, Czech Republic

Advanced Topics of Theoretical Computer Science

FRVŠ MŠMT FR2581/2010/G1

Definition

Let $K, L \subseteq \triangle^*$ be two languages. A total computable function f over $\triangle^*$ is a reduction of K to L, symbolically written as $K {}_f\angle L$, if for all $w \in \triangle^*$, $w \in K$ iff $f(w) \in L$.

Convention

Let $K, L \subseteq \triangle^*$ be two languages. We write $K \angle L$ to express that there exists a reduction of K to L.

Theorem

Let $K, L \subseteq \triangle^*$ be two languages. If $K \angle L$ and $L \in {}_{TM}\Phi$, then $K \in {}_{TM}\Phi$.

Proof:

- Let $K, L \subseteq \triangle^*$ be two languages.
- As $L \in {}_{TM}\Phi$, there is a Turing machine $M \in {}_{TM}\Psi$ satisfying $L = L(M)$.
- Construct a new Turing machine $N \in {}_{TM}\Psi$ that works on every input $w \in \triangle^*$ as follows:
  - N computes $f(w)$;
  - N runs M on $f(w)$;
  - if M accepts, so does N.

Corollary

Let $K, L \subseteq \triangle^*$ be two languages. If $K \angle L$ and $L \notin {}_{TM}\Phi$, then $K \notin {}_{TM}\Phi$.

Theorem

$_{TM-Equivalence}L \notin {}_{TM}\Phi$.

Theorem

$_{Non-TM-Equivalence}L \notin {}_{TM}\Phi$.

TM–Equivalence

Problem: *TM–Equivalence*
Question: Let $M, N \in {}_{TM}\Psi$. Are M and N equivalent?
Language: $_{TM-Equivalence}L = \{\langle M, N\rangle \mid M, N \in {}_{TM}\Psi, L(M) = L(N)\}$.

Non–TM–Equivalence

Problem: *Non–TM–Equivalence*
Question: Let $M, N \in {}_{TM}\Psi$. Are M and N nonequivalent?
Language: $_{TM-Equivalence}L = \{\langle M, N\rangle \mid M, N \in {}_{TM}\Psi, L(M) \neq L(N)\}$.

Corollary

$_{TM-Equivalence}L \notin {}_{TD}\Phi$ and $_{Non-TM-Equivalence}L \notin {}_{TD}\Phi$.

Theorem

Let $K, L \subseteq \triangle^*$ be two languages. If $K \angle L$ and $L \in {}_{TD}\Phi$, then $K \in {}_{TD}\Phi$.

Corollary

Let $K, L \subseteq \triangle^*$ be two languages. If $K \angle L$ and $L \notin {}_{TD}\Phi$, then $K \notin {}_{TD}\Phi$.

Definition

Let $\pi \subseteq {}_{TM}\Phi$. Then, π said to be a property of Turing languages.

I. A language $L \in {}_{TM}\Phi$ satisfies π if $L \in \pi$.
II. Set ${}_{\pi}L = \{\langle M \rangle | M \in {}_{TM}\Psi, L(M) \in \pi\}$. We say that π is decidable if ${}_{\pi}L \in {}_{TD}\Phi$; otherwise, π is undecidable.
III. We say that π is trivial if $\pi = {}_{TM}\Phi$ or $\pi = \emptyset$; otherwise, π is non–trivial.

Rice's Theorem

Every non–trivial property is undecidable.

Definition

Let $M = ({}_M\Sigma, {}_MR)$ be a Turing decider. The time-complexity function of M, denoted by ${}_Mtime$, is defined over ${}_0\mathbb{N}$ so for all $n \in {}_0\mathbb{N}$, ${}_Mtime(n)$ is the maximal number of moves M makes on an input string of length n before halting.

Definition

I. Let f and g be two functions over ${}_0\mathbb{N}$. If there exist $c, d \in \mathbb{N}$ such that for every $n \geq d$, $f(n)$ and $g(n)$ are defined and $f(n) \leq cg(n)$, then g is an upper bound for f, written as $f = O(g)$.

II. If $f = O(g)$ and g is of the form n^m, where $m \in \mathbb{N}$, then g is a polynomial bound for f.

III. Let $M \in {}_{TD}\Phi$. M is polynomially bounded if there is a polynomial bound for ${}_Mtime$.

Definition

Let P be a decidable problem. If P is decided by a polynomially bounded Turing decider, P is tractable; otherwise, P is intractable.

Definition

I Let M be a Turing machine. M is a nondeterministic Turing decider if M halts on every input string.

II Let M be a nondeterministic Turing decider. The timecomplexity of M, ${}_{M}time$, is defined over ${}_{0}\mathbb{N}$ so for all $n \in {}_{0}\mathbb{N}$, ${}_{M}time(n)$ is the maximal number of moves M makes on an input string of length n before halting.

III M is polynomially bounded if there is a polynomial bound for ${}_{M}time$.

Convention

- ${}_{P}\Phi$ denotes the family of languages accepted by polynomially bounded (deterministic) Turing deciders, and
- ${}_{NP}\Phi$ denotes the family of languages accepted by polynomially bounded nondeterministic Turing deciders.

Definition

Let $\triangle$ and ς be two alphabets, $J \subseteq \triangle^*$, and $K \subseteq \varsigma^*$. Then, J is polynomially transformable into K, symbolically $J \propto K$, if there exist a Turing decider M and a total function f from $\triangle^*$ to ς^* so M is polynomially bounded, $L(M) = K$, and $x \in J$ iff $f(x) \in K$.

Definition

Let $L \in {}_{NP}\Phi$. If $J \propto L$ for every $J \in {}_{NP}\Phi$, then L is NP–complete.

Definition

Let $M = ({}_M\Sigma, {}_MR)$ be a Turing decider. A function g over ${}_0\mathbb{N}$ represents the space complexity of M, denoted by ${}_Mspace$, if ${}_Mspace(i)$ equals the minimal number $j \in {}_0\mathbb{N}$ such that for all $x \in {}_M\triangle^i$, $y, v \in \Gamma^*$, $\triangleright {}_Msx \triangleleft$ in M implies $|yv| \leq j$.

Convention

- ${}_{PS}\Phi$ denotes the family of languages accepted by polynomially bounded (deterministic) Turing deciders, and
- ${}_{NPS}\Phi$ denotes the family of languages accepted by polynomially bounded nondeterministic Turing deciders.

References

- Wayne Goddard.
  Introducing the Theory of Computation.
  Jones Bartlett Publishers, 2008.
- Jeffrey D. Ullman John E. Hopcroft, Rajeev Motwani.
  Introduction to Automata Theory, Languages, and Computation.
  Addison Wesley, 2006.
- Dexter C. Kozen.
  Automata and Computability.
  Springer, 2007.
- Dexter C. Kozen.
  Theory of Computation.
  Springer, 2010.
- John C. Martin.
  Introduction to Languages and the Theory of Computation.
  McGraw-Hill Science/Engineering/Math, 2002.

Thank you for your attention!

End