2. Fundamental Notions and Results

(Short Recapitulation from “Formale Methoden der Informatik”)

Reinhard Pichler

Institut für Informationsysteme
Arbeitsbereich DBAI
Technische Universität Wien

06 March, 2018

Computation and Computability

- Ability to read and formulate decision/optimization problems
- Several kinds of problems (decision p., function p., optimization p., enumeration p., counting p.)
- Problem vs. problem instance
- Problem vs. algorithm vs. program
- Church-Turing thesis
- Halting problem
- Decidability vs. undecidability vs. semi-decidability
- Complement of a decision problem
- Properties of complementation

Complexity of Problems and Algorithms

- Asymptotic, worst-case complexity vs. other notions of complexity
- Basic understanding of growth rates (polynomial vs. exponential)
- The class P
- The class NP
- Tractability vs. intractability
- Optimization vs. decision problem
Reductions

- Two motivations for reducing one problem (or language) to another.
- Two kinds of reductions (Turing, many-one).
- Limiting the resources used by reductions.
- Cook / Karp reductions.
- Proving the correctness of problem reductions.
- The definitions of $C$-hard and $C$-complete problems for a complexity class $C$.
- Understanding the role of complete problems in complexity theory.
- Proving undecidability by reduction from the HALTING problem.
- Proving non-semi-decidability by reduction from the NON-HALTING problem.

NP-Completeness

- You should now be familiar with the intuition of NP-completeness (and recognize NP-complete problems).
- Two fundamental NP-complete problems: **SAT** and **3-SAT**.
- Difference between logical equivalence and sat-equivalence.
- Many more examples of NP-complete problems, e.g.: **CLIQUE**, **INDEPENDENT SET**, **VERTEX COVER**, **3-COLORABILITY**, **HAMILTON-PATH**, **HAMILTON-CYCLE**, **TSP(D)**, etc.
- Usefulness of reductions to **SAT**.

Other Important Complexity Classes

- Understanding the definitions of L, PSPACE and EXPTIME
- Being aware of the main inclusions between P, NP, and the three classes above.

Turing Machines

- Definition of Turing machines.
- Turing machines as a reasonable model of computation.
- Formal definition of “deterministic” complexity classes P, EXPTIME, L, PSPACE, EXPSPACE.
- Solving problems with Turing machines.
  (Decision problems can be considered as languages!)
- (Strengthening of) the Church-Turing Thesis
- Nondeterministic Turing machines. Differences between deterministic and nondeterministic TMs
- Nondeterminism as “guess and check” algorithms
- Definitions of NL, NP, NEXPTIME via nondeterministic TMs.
- The definition of complementary problems.
- Summary of important complexity classes: L, NL, co-NL, P, NP, co-NP, PSPACE, EXPTIME, NEXPTIME, co-NEXPTIME, EXPSPACE.
Overview of Complexity Classes

Recursive Problems

Inside PSPACE