CSC460 (Spring 2005) Theory of Computation
List of Definitions, Notations, and Concepts

Feb. 4, 2005 [revised]

Note: The section and page numbers refer to the text (Hopcroft et al., 2001). For other references, see the reference list at the end of this document.

Problems

Practical problem [definition]: phenomenon (generally perceived negatively) that calls for some *action* [ref. Booth et al.]

Research problems [definition]: question (yes-no or *wh*-question) that is generally crucial to solve a practical problem and calls for an *answer* [ref. Booth et al.]

Computational problems [definition]: (informally) research problem that rends to computational some computational processing/analysis; (formally, within the traditional Theory) membership problem with respect to a certain set (thus, can be identified with the **set**); ~ language Sec 1.5.4

Cost/significance (of a problem) [definition]: The "cost" of a problem is what one would suffer if the problem is not solved. The "significance" of a problem is what one would benefit if the problem is solved. Thus, these two notions are the opposite side of the same idea (i.e., you only need to identify one, not both). While the process of solving a problem would be *objective* and can be purely technical ("value-free"), the cost/significance of the problem is *subjective* ("value-laden"). [ref. Booth et al.]

Theory [ref. Enderton]

Axioms [definition]: the very basic principles consistent with our experience; the starting point of a theory (Note that there is no way to "logically" justify these; these are assumed and generally obtained intuitively.)

Rules of inference [definition]: the basic rules of obtaining additional information (statements) from the axioms

Theorem [definition]: statement justifiable from the axioms and the rules of inference

Proof [definition]: The "mechanical" process of justifying a theorem

Logical consequence [definition]: If a statement follows another by analyzing their "meanings," the former is called a "logical consequence" of the latter. (Note that the notion of "logical consequence" is stronger than that of "proof," and thus, there may be a logical consequence of the axioms that may not be proven.)

Theory [definition]: (informally) potentially infinite, consistent body of knowledge which can be systematically derived from a small number of abstract principles; (formally) the collection of all the statements that are logical consequences of the axioms

Computability

Turing machine (**TM**) [definition] Sec 8.2

Effective procedure [description]: well-defined, step-by-step procedure Sec 8.2

Algorithm [description]: effective procedure that always terminates ~ recursive Sec. 8.2.6, 9.2.1, p. 367

(Computational) **Problem** [definition]: = set ~ property ~ predicate ~ characteristic function ~ language Sec 1.5.4

Decidable [definition]: = **recursive** = TM-decidable = computable = solvable Sec. 1.1.3, 8.2.6, 9.2.1, p. 374

Undecidable [definition]: = unsolvable p. 302, 310, Ch 9

TM-recognizable [definition]: = **recursively enumerable** (**RE**) Sec 8.2.5, p. 374

Non-TM-recognizable [definition]: = **non-RE** Sec 9.1, p. 374

Semi-decidable [definition]: = TM-recognizable (but not decidable)

co-*X* (where *X* is some property) [definition]: The complement of the problem has property *X*. E.g., co-TM-recognizable. [ref. Sipser]

Diagonalization [description] Sec 9.1

Rice's Theorem Sec 9.3.3

Church-Turing thesis Sec 8.2.1

Formal Languages and Automata

Language [definition]: set (often, of strings) Sec 1.5.4

Finite-state automaton (**FSA**; finite automaton, **FA**) [definition] Sec 2.1
Deterministic finite-state automaton (**DFA**) [definition] Sec 2.2
Nondeterministic finite-state automaton (**NFA**) [definition] Sec 2.3
Regular expression [definition] Sec 3.1
Regular language [definition] = **regular set** Sec 3.1
Pumping lemma [description] Sec 4.1
Context-free grammar (**CFG**) [definition] Sec 5.1
Context-free language (**CFL**) [definition] Sec 5.1
Deterministic context-free language (**DCFL**) [definition] Sec 6.4
Push-down automaton (**PDA**) [definition] Sec 6.1
Deterministic push-down automaton (**DPDA**) [definition] Sec 6.4
Pumping lemma for CFLs [description] Sec 7.2

Complexity

Complexity (within Theory of Computation) [description]: the quantitative effects of the input size on computational performance
Time complexity [description] p. 414
Space complexity [description]: performance measure based on space (~ memory size) [ref. Sipser 1997]
Worst-case analysis [description]: the performance analysis for the worst input pattern
Average-case analysis [description]: the (estimated) average performance for all possible input patterns
'***O***' (asymptotic upper bound) [description] [ref. Sipser 1997]
Informal idea: Behaves roughly as bad as a certain function
Definition #1: $f(n) \in O(g(n))$ if there are constants $c > 0$ and $n_0 \geq 1$ such that $f(n) \leq c\, g(n)$ for every integer $n \geq n_0$.
Definition #2: $f(n) \in O(g(n))$ if $\lim_{n\to\infty} \frac{f(n)}{g(n)} = c$ for some $c \geq 0$

Class P (**tractable**) [definition] p. 361, Sec 10.1
Class Exp (EXPTIME) [description]: $O(c^n)$ for any constant $c > 1$
Intractable [definition] p. 1, p. 361, Sec 10.1
Class NP [definition] Sec 10.1
Class NP-complete [definition] Sec 10.2
Cook-Levin theorem [description] Sec 10.2.3
Class NP-hard [definition] p. 423

General

Power set (of a set) [definition]: given set A, the set of all the subsets of A, i.e., $\{x \mid x \subseteq A\}$
Deterministic [definition]: at most one transition/option for a unique situation
Nondeterministic [definition]: possibly multiple transitions/options for a unique situation
Reduction, **reducibility** [definition] Sec 8.1.3
Polynomial time reducibility [definition] Sec 10.1.5

References [all available in the library]

Booth, Wayne C., Colomb, Gregory G., and Williams, Joseph M. 2003. *The craft of research*, 2nd ed. University of Chicago press. [excellent book on writing a research paper]

Enderton, Herbert B. 2001. *A mathematical introduction to logic*, 2nd ed. San Diego, CA: Harcourt. [classic introduction; only the 1st edition (1972) is available in the library]

Hopcroft, John E., Motwani, Rajeev, and Ullman, Jeffrey D. 2001. *Introduction to Automata Theory, Languages, and Computation*, 2nd ed. Addison-Wesley.

Hopcroft, John E. and Ullman, Jeffrey D. 1979 Introduction to automata theory, languages, and computation. Addison-Wesley. [the first edition of our text; contains some more advanced materials]

Sipser, Michael. 1997. *Introduction to the theory of computation*. PWS. [used as a textbook in the past]

// End