

3515ICT: Theory of Computation

Preliminaries

See (H, 1) and (S, 0).

Definition of a *set*, the *empty set* \emptyset , extensional and intensional set definitions, *set membership* ($x \in S$), *set inclusion* ($S \subseteq T$), common set operations (union, intersection, difference, complement), Cartesian product ($S \times T = \{ \langle s, t \rangle \mid s \in S, t \in T \}$), power set ($2^S = \{ s \mid s \subseteq S \}$).

Specific sets: Natural numbers $N = \{0, 1, 2, \dots\}$, integers $Z = \{\dots - 2, -1, 0, 1, 2, \dots\}$, rational numbers $Q = \{p/q \mid p \in Z, q \in N \setminus \{0\}, (p, q) = 1\}$, set of strings Σ^* over a finite alphabet Σ .

Functions and relations. A *relation* over sets S and T is a subset of $S \times T$ (sets S and T may be identical). If $\langle s, t \rangle \in R$, we also write sRt . Relations may be *reflexive*, *symmetric*, *transitive*, *antisymmetric*, etc. A relation may be an *equivalence relation* (a reflexive, symmetric, transitive relation) or a *partial order* (a reflexive, transitive, antisymmetric relation).

A *function* from set S to set T is a relation R over S and T such that for every $s \in S$ there exists at most one $t \in T$ such that sRt . Functions may be *total*, *one-to-one (injective)* and/or *onto (surjective)*.

Sets may be *finite* ($\{a_1, \dots, a_n\}$), *countably infinite* (there exists a one-to-one, onto function from N to the set), or *uncountably infinite*. We say a set is *countable* if it is finite or countably infinite.

A *graph* is a structure consisting of a set of *vertices* or *nodes* and a set of *edges* connecting pairs of (normally distinct) vertices. If the edges are directed, the graph is called a *directed graph* or *digraph*. If the edges are weighted, the graph is called a *weighted graph*. Graphs have many properties and there are many algorithms on graphs! Too many to summarise here.

We need to know the definitions and basic properties of *trees*, *binary trees*, and several variants of trees.

In the theory of computation, we often model problems as languages. We'll describe this equivalence later. An *alphabet* (Σ) is a finite set of symbols a, b, c, \dots . A *string* is a finite sequence of symbols $a_1 \dots a_n$. The empty string is denoted ε . The main operations on strings are *concatenation* and *reversal*.

The (countably infinite) set of all strings composed of symbols from an alphabet Σ is denoted Σ^* . A *language* over an alphabet Σ is a subset of Σ^* , i.e., a set of strings of symbols from Σ . The set of languages over any nonempty alphabet is uncountably infinite. (Proof later, by diagonalisation.) The main operations on languages are *union* ($L_1 \cup L_2$), *concatenation* ($L_1 L_2 = \{s_1 s_2 \mid s_1 \in L_1, s_2 \in L_2\}$), and *closure* ($L^* = \{s_1 \dots s_n \mid s_i \in L, 1 \leq i \leq n, n \geq 0\}$).

We're interested in universally true statements about languages (and other mathematical structures). If we can *prove* such a statement, we call it a *theorem*. There are several main *proof techniques* commonly used:

1. Construction, e.g., Sipser, Theorem 0.22.
2. Chain of equivalences (or inclusions, etc.), e.g., $(a - b)(a + b) = a^2 - b^2$,
3. Contradiction, e.g., Sipser, Theorem 0.24.

4. Induction (mathematical or structural), *e.g.*, for all $n \geq 0$, $n(n^2 + 5)$ is divisible by 6.
5. Case analysis, *e.g.*, same result.
6. Diagonalisation, *e.g.*,
 - (a) The set of real numbers between 0 and 1 is not countable (Cantor).
 - (b) The set of languages over any finite, nonempty alphabet is not countable.

‘H’ or ‘IALC’ or ‘Hopcroft *et al.*’ refers to *Introduction to Automata Theory, Languages and Computation, Second Edition*, by J.E. Hopcroft,, R. Motwani and J.D. Ullman, Addison-Wesley, 2001.

‘S’ or ‘Sipser’ refers to *Introduction to the Theory of Computation, Second Edition*, by M. Sipser, Thomson, 2006.