T–79.148 Introduction to Theoretical Computer Science

Harri Haanpää

Laboratory for Theoretical Computer Science, HUT

This is a *very* quick and dirty English translation of the slides from Spring 2004. There may be errors and some of the terminology may be nonstandard at places.

L0: Introduction of the subject
L1: Basic mathematical concepts
L2: Strings and languages; countable and uncountable sets
L3: Finite state automata
L4: Minimization of FSA; nondeterministic FSA
L5: Regular expressions and FSA
L6: Pumping lemma of regular languages; context-free gramma
L7: Parse trees; parsing LL(1) grammars
L8: Chomsky normal form; CYK-algorithm; pushdown automata
L9: Pumping lemma of context-free languages; Turing machine
L10: Extensions of the Turing machine
L11: Language classes R and RE; the universal Turing machine
L12: Halting problem, unsolvability, Rice's theorem

T–79.148 Tietojenkäsittelyteorian perusteet
Introduction to Theoretical Computer Science

What can one do with a computer?

This course divides roughly into three parts. We examine three models of a computer, their power and limitations, i.e., the classes of languages that they can decide.

It turns out that each class of automata corresponds to a class of grammars. The classes of automata and grammars are:

1. Finite state automata & regular expressions
2. Pushdown automata & context-free grammars
3. Turing machine & unrestricted grammars

Formal methods are used to *study* the models and in the exercises the knowledge is *applied*.

Practical arrangements

Registration: obligatory, by TOPI

Lectures: Thu 14–16 T1, in Finnish by Harri Haanpää

Exercises: Tue 12–13, Tue 13–14, Tue 16–17 in English, Tue 17–18, Wed 12–13, Wed 13–14, Wed 14–15, Wed 15–16, Wed 16–17, Wed 17–18 — choose one and register by TOPI

Computer exercises: obligatory, using a WWW-based system

Course home page: `http://www.tcs.hut.fi/Studies/T-79.148/`

Course newsgroup: `opinnot.tik.tkt` on `news.tky.hut.fi`

- Please ask (and answer!) your questions there!

To pass the course

1. Pass the computer exercises *before taking the exam*.
2. *After passing the computer exercises* pass the exam by scoring at least 30/60 (exams likely in May, Aug, Oct, Dec, Feb)
   - You may earn up to 6 points on the exam by participating in the exercises.
     - 3 homework problems a week, 1 exam point for every 5 homework problems prepared for exercise session
   - You may earn 2 points for the exam by finishing the computer exercises quickly.

Material

Lecture notes (in Finnish) and solutions of *demonstration* exercises (in Finnish) are distributed through Edita.

Recommended textbook Michael Sipser, Introduction to the Theory of Computation, PWS Publishing 1997. (supplementary for Finnish-speaking students; likely necessary for non-Finnish-speaking students)

Some additional material on the course WWW page.

Feedback

I hope a group of students would give regular feedback during the course. Ideally, I hope for feedback from both those who attend lectures and those who do not.

- What was easy? What was difficult?
- How much effort does the course require?
- Does the course communication work?
- Other?

Interested students should send e-mail to `Harri.Haanpaa@hut.fi` soon.

Theoretical computer science

- A mathematical study of what can be done with a computer and how efficiently.
- Offers mathematical concepts and methods for modeling and analysing computing systems as well as for designing clear and effective solutions.

Branches of theoretical computer science

Computability theory

What can be done by a computer in principle?

- Turing, Gödel, Church, Post (1930's); Kleene, Markov (1950's).

Theory of computational complexity

What can be done by a computer in practice?

- Hartmanis, Stearns (1960's); Cook, Levin, Karp (1970's); Papadimitriou, Sipser, Håstad, Razborov etc. (1980's).

Automata and grammar theory

Basic properties of computing systems and formalisms for describing them.

- Chomsky (1950's); Ginsburg, Greibach, Rabin, Salomaa, Schützenberger etc. (1960's)

Correctness of programs

Mathematically exact specification and verification of programs.

- Dijkstra, Hoare (1960's); Manna, Pnueli, Scott etc. (1970's).

Other

- Design and analysis of algorithms (Knuth, Hopcroft, Tarjan etc.)
- cryptology (Rivest, Shamir, Adleman etc.)
- theory of parallel and distributed systems (Lamport, Lynch, Milner, Valiant etc.)
- machine learning theory (Valiant etc.)
- etc.

Mostly automata and grammars, and some elementary theory of computability are considered in this course. Other subjects are considered in other courses of the Laboratory for Theoretical Computer Science.

1. Basic mathematical concepts

1.1 Sets

A set is a collection of elements. The elements may be given by listing them, e.g.,

$$S = \{2, 3, 5, 7, 11, 13, 17, 19\}$$

or by some rule, e.g.,

$$S = \{p \mid p \text{ is prime}, 2 \leq p \leq 20\}.$$

If the element a is a member of A, we write $a \in A$, otherwise $a \notin A$. (E.g. $3 \in S$, $8 \notin S$.)

An empty set $\emptyset$ is an important special case; it contains no elements.

If all elements of the set A are also members of the set B, A is said to be a *subset* of B and we may write $A \subseteq B$ (sometimes the notation $A \subset B$ is also used). If A is *not* a subset of B we write $A \not\subseteq B$. Thus e.g.

$$\{2, 3\} \subseteq S, \qquad \{1, 2, 3\} \not\subseteq S.$$

Trivially $\emptyset \subseteq A$ for all A.

The sets A ja B are the same, if they contain the same elements, that is if $A \subseteq B$ and $B \subseteq A$. If $A \subseteq B$, but $A \neq B$, we say that A is a *proper subset* of B, denoted with $A \subsetneq B$. Above it could have been written $\{2, 3\} \subsetneq S$ and $\emptyset \subsetneq A$ if $A \neq \emptyset$.

A set may consist of other sets. Such a set may be called a "family of sets". For example

$$X = \{\emptyset, \{1\}, \{2\}, \{1,2\}\}.$$

The set formed by all subsets of a set A is called the *powerset* of A and denoted with $\mathcal{P}(A)$, e.g., on $X = \mathcal{P}(\{1,2\})$. [Since the powerset of an n-element set has 2^n elements, the notation 2^A is also used.]

Note that $A \subseteq B$ if and only if $A \in \mathcal{P}(B)$.

Be careful with the empty set:

$$\emptyset \neq \{\emptyset\}, \quad \mathcal{P}(\emptyset) = \{\emptyset\}, \quad \mathcal{P}(\{\emptyset\}) = \{\emptyset, \{\emptyset\}\}.$$

Sets may be combined by *set operations*, of which the most important are:

union

$$A \cup B = \{x \mid x \in A \text{ or } x \in B\},$$

e.g. $\{1,2,3\} \cup \{1,4\} = \{1,2,3,4\}$.

intersection

$$A \cap B = \{x \mid x \in A \text{ and } x \in B\},$$

e.g. $\{1,2,3\} \cap \{1,4\} = \{1\}$.

difference

$$A - B = \{x \mid x \in A \text{ and } x \notin B\},$$

e.g. $\{1,2,3\} - \{1,4\} = \{2,3\}$.
[The notation $A \setminus B$ is also sometimes used for difference.]

Certain rules govern the set operations. The most important of these are the *associativity* of union and intersection:

$$A \cup (B \cup C) = (A \cup B) \cup C, \quad A \cap (B \cap C) = (A \cap B) \cap C$$

their *commutativity*:

$$A \cup B = B \cup A, \quad A \cap B = B \cap A$$

and their *distributivity*:

$$A \cap (B \cup C) = (A \cap B) \cup (A \cap C),$$

$$A \cup (B \cap C) = (A \cup B) \cap (A \cup C).$$

If all sets under consideration are subsets of a common "universal set" U, the difference $U - A$ is called the complement of A (with regard to U) and denoted with $\bar{A}$:lla.

Observe the important *de Morgan laws*:

$$\overline{A \cup B} = \bar{A} \cap \bar{B},$$

$$\overline{A \cap B} = \bar{A} \cup \bar{B}.$$

Difference of sets may also be expressed as follows in terms of intersection and complement:

$$A - B = A \cap \bar{B}.$$

If the elements of a set family $\mathcal{A}$ have been *indexed*, e.g.,

$$\mathcal{A} = \{A_1, A_2, A_3, \dots\},$$

then for union and intersection the following notation can be used:

$$\bigcup_{i \geq 1} A_i = A_1 \cup A_2 \cup A_3 \cdots \qquad \text{ja} \qquad \bigcap_{i \geq 1} A_i = A_1 \cap A_2 \cap A_3 \cdots$$

The indices need not be natural numbers; rather, any set I may serve as the *index set*. The notation

$$\mathcal{A} = \{A_i \mid i \in I\}$$

and

$$\bigcup_{i \in I} A_i, \qquad \bigcap_{i \in I} A_i$$

may be used.

1.2 Relations and functions

Let A and B be sets. The *ordered pair* of the elements $a \in A$ and $b \in B$ is denoted by (a, b). Note that for sets always $\{a, b\} = \{b, a\}$, but if $a \neq b$, then for the ordered pairs $(a, b) \neq (b, a)$.

The Cartesian product of the sets A and B is defined as

$$A \times B = \{(a, b) \mid a \in A \text{ ja } b \in B\},$$

e.g.

$$\begin{aligned} &\{1, 2, 3\} \times \{1, 4\} \\ &= \{(1, 1), (1, 4), (2, 1), (2, 4), (3, 1), (3, 4)\}. \end{aligned}$$

A relation R from the set A to the set B is a subset of the Cartesian product $A \times B$

$$R \subseteq A \times B.$$

For $(a, b) \in R$ the notation aRb may also be used and we may say that a is *in relation R* with b. This *infix*-notation is used especially when the relation is named with a special symbol such as $\leq$, $\prec$, $\equiv$, $\sim$. [It is more convenient to write "$a \prec b$" than "$(a, b) \in \prec$".] If the *domain* A and the *range* of the relation $R \subseteq A \times B$ are the same, i.e., $A = B$, then R is said to be a relation in A.

Let $R \subseteq A \times B$. The *image* of $A' \subseteq A$ in R is

$$R(A') = \{b \in B \mid \exists a' \in A' \text{ s.t. } (a', b) \in R\}$$

and the *inverse image* of $B' \subseteq B$ is

$$R^{-1}(B') = \{a \in A \mid \exists b' \in B' \text{ s.t. } (a, b') \in R\}.$$

The inverse relation of $R \subseteq A \times B$ is the relation $R^{-1} \subseteq B \times A$,

$$R^{-1} = \{(b, a) \mid (a, b) \in R\}.$$

If $R \subseteq A \times B$ and $S \subseteq B \times C$ are relations, then their *composite relation* $R \circ S \subseteq A \times C$ is defined by

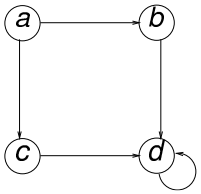
$$R \circ S = \{(a, c) \mid \exists b \in B \text{ s.t. } (a, b) \in R, (b, c) \in S\}.$$

Especially when A and B are finite, the relation $R \subseteq A \times B$ may be described as a *directed graph*, whose *vertices* are the elements of A and B and there is an *edge* from vertex $a \in A$ to vertex b if and only if $(a, b) \in R$.

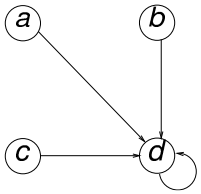
For example, let $A = \{a, b, c, d\}$ and $R \subseteq A \times A$ a relation in A,

$$R = \{(a, b), (a, c), (b, d), (c, d), (d, d)\}.$$

The graph presentation of R is



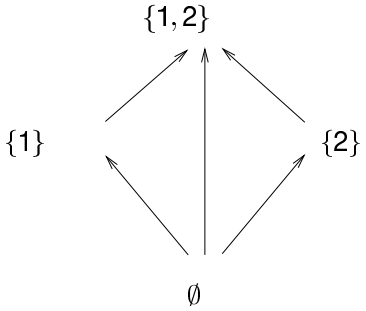
The graph presentation of $R \circ R$ is



Example: Over the set $X = \mathcal{P}(\{1, 2\})$, define the relation $S \subseteq X \times X$,

$$S = \{(A, B) \mid A \subseteq B\}.$$

The graph presentation of S is



The relation $f \subseteq A \times B$ is a *function*, if every $a \in A$ is in relation f with exactly one $b \in B$ kanssa. Instead of the notation for relations, usually the notation $f : A \to B$ ja $f(a) = b$ is used. Everything that has been said of relations goes for functions as well, but for historical reasons a different order is used for composite functions: if $f : A \to B$ and $g : B \to C$ are functions, then their composite function is defined by $(g \circ f)(a) = g(f(a))$, that is, as relations

$$g \circ f = \{(a, c) \mid \exists b \in B \text{ s.e. } f(a) = b, g(b) = c\}.$$

A function $f : A \to B$ is a *surjection* (onto map), if every $b \in B$ is the image of some $a \in A$ that is, if $f(A) = B$, and an *injection* (one-to-one map), if all $a \in A$ map to distinct elements, that is, if $a \neq a' \Rightarrow f(a) \neq f(a')$. A function f is a *bijection*, if it is both an injection and a surjection (one-to-one and onto) that is, if every $b \in B$ is the image of one and only one $a \in A$.

1.3 Equivalence relations

Equivalence relations are a mathematical way of formulating the idea that mathematical objects are *similar* with respect to some property X suhteen. An equivalence relation based on X partitions the set of objects into *equivalence classes*, which correspond to distinct values of the property X. (Conversely, every partition Π of the objects Π defines an abstract similarity property: two objects may be considered similar if they are members of the same class in Π.)

The notion of similarity can be captured by the following three conditions:

Definition 1.1 A relation $R \subseteq A \times A$ is

1. *reflexive*, if $aRa \; \forall a \in A$;
2. *symmetric*, if $aRb \Rightarrow bRa \; \forall a, b \in A$;
3. *transitive*, if $aRb, bRc \Rightarrow aRc \; \forall a, b, c \in A$.

Definition 1.2 A relation $R \subseteq A \times A$, that satisfies 1–3 above is an *equivalence relation*. The equivalence class of $a \in A$ *ekvivalenssiluokka* (with regard to R) is

$$R[a] = \{x \in A \mid aRx\}.$$

Instead of R equivalence relations are usually denoted by special symbols such as $\sim$, $\equiv$, $\simeq$ etc.

Example. Let

$$A = \{\text{the people born in the 20th century}\}$$

and aRb if a and b were born in the same year. Clearly R is an equivalence relation, whose equivalence classes consist of people born in a given year. There are 100 classes which correspond to the years of the 20th century 1901,...,2000.

Lemma 1.3 Let $R \subseteq A \times A$ be an equivalence relation. Then for $a, b \in A$ we have:

$$R[a] = R[b] \quad \text{iff} \quad aRb.$$

Proof Easy, omitted. □

Lemma 1.4 Let $R \subseteq A \times A$ be an equivalence relation. Then the equivalence classes of R:n *partition* A into distinct nonempty subsets, viz.

- $R[a] \neq \emptyset$ for all $a \in A$;
- $A = \bigcup_{a \in A} R[a]$;
- if $R[a] \neq R[b]$, then $R[a] \cap R[b] = \emptyset$, for all $a, b \in A$.

Proof Easy, omitted. □

Conversely every partition of the set A into distinct nonempty classes A_i, $i \in I$, defines the corresponding equivalence relation

$$a \sim b \quad \Leftrightarrow \quad a \text{ and } b \text{ belong to the same } A_i.$$

*1.4 Order relations

The various ideas of "order" may be captured as follows:

Definition 1.5 A relation $R \subseteq A \times A$ is *antisymmetric*, if for all $a, b \in A$ it holds that: $aRb, bRa \Rightarrow a = b$.

Definition 1.6 A relation $R \subseteq A \times A$, that is reflexive, antisymmetric and transitive, is a *(partial) order* on A.

Order relations are usually denoted with symbols such as $\leq$, $\preceq$.

If for $a, b \in A$ we have aRb or bRa, we say that a and b are *comparable*.

If all elements of A are (pairwise) comparable, R is a *complete (total, linear)* order.

An element $a \in A$ of an ordered set $(A, \preceq)$ is:

- *maximal*, if $a \preceq x \Rightarrow a = x \quad \forall x \in A$;
- *minimal*, if $x \preceq a \Rightarrow a = x \quad \forall x \in A$;
- *maximum*, if $x \preceq a \quad \forall x \in A$;
- *minimum*, if $a \preceq x \quad \forall x \in A$.

An ordered set $(A, \preceq)$ is *well-ordered* if every nonempty $B \subseteq A$ contains a minimum element with regard to $\preceq$.

Lemma. Every well-order is perfect.□

Examples

- $(\mathbb{N}, \leq)$ — well-ordered.
- $(\mathbb{Z}, \leq)$ — perfect, but not well-ordered.
- Let X be a set. Then $(\mathcal{P}(X), \subseteq)$ is ordered, but not perfect if $|X| \geq 2$.
- Let $m, n \in \mathbb{N}$. Use the notation

$$m \mid n \;\Leftrightarrow m \text{ is a factor of } n.$$

The set $(\mathbb{N}, \mid)$ is an order, but not perfect.

Let $(A, \preceq)$ be an ordered set. Let us denote

$$\begin{aligned} a \prec b &\Leftrightarrow a \preceq b, a \neq b \\ a \succeq b &\Leftrightarrow b \preceq a \\ a \succ b &\Leftrightarrow a \succeq b, a \neq b. \end{aligned}$$

The element $a \in A$ is an *immediate predecessor* of $b \in A$ (and b is an *immediate successor* of a), if

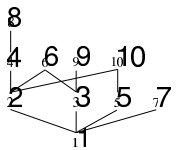
1. $a \prec b$ and
2. $\nexists c \in A$ s.t. $a \prec c \prec b$.

Every finite ordered set $(A, \preceq)$ may be described by a *Hasse diagram*, whose vertices correspond to the elements of A and where from every $a \in A$ there is a line "upwards" to every immediate successor of a.

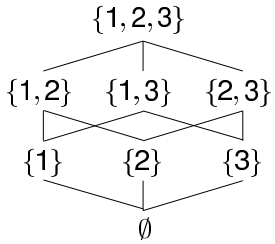
Example

(i) The Hasse diagram for $(\{1, 2, \ldots, 5\}, \leq)$ is

(ii) The Hasse diagram for $(\{1, 2, \ldots, 10\}, \mid)$ is



(iii) The Hasse diagram for $(\mathcal{P}(\{1, 2, 3, \}), \subseteq)$ is



The Hasse diagram of an order relation may be thought of as a "transitive reduction" of the graph presentation of the relation. From the Hasse diagram those edges have been omitted that can be deduced by transitivity and reflexivity from the edges in the diagram.

1.5 Induction principle

Theorem 1.7 Let $(A, \preceq)$ be a well-ordered set and $P(a)$ a proposition that concerns the elements of A. If we can show for all $a \in A$ the induction property

(*) $\quad [P(x) \text{ true for all } x \prec a] \Rightarrow P(a)$ is true,

then $P(a)$ is true for all $a \in A$.

Proof Assume that (*) holds but still the set

$$B = \{a \in A \mid P(a) \text{ is false}\}$$

is nonempty. Since A is well-ordered, the set B contains a minimum element $b \in B$ with regard to the order $\preceq$ But in such a case

$$[P(x) \text{ true for all } x \prec b],$$

so we should also have $P(b)$ true.

By this contradiction we must have $B = \emptyset$, and hence $P(a)$ for all $a \in A$. □

Corollary 1.8 [Strong induction on natural numbers.] Let $P(k)$ be a property of the natural numbers. If we have:

1. $P(0)$ and
2. for all $k \geq 0$:

$$[P(0)\&P(1)\&\ldots\&P(k)] \Rightarrow P(k+1),$$

then $P(n)$ is true for all $n \in \mathbb{N}$. □

Corollary 1.9 [Weak induction on natural numbers.] Let $P(k)$ be a property of the natural numbers. If we have:

1. $P(0)$ and
2. for all $k \geq 0$:

$$P(k) \Rightarrow P(k+1),$$

then $P(n)$ is true for all $n \in \mathbb{N}$. □

Example

Claim For all $n \in \mathbb{N}$ it holds that

$$P(n): \quad (1+2+\cdots+n)^2 = 1^3+2^3+\cdots+n^3.$$

Proof

1. Induction base: $P(0): \quad 0^2 = 0.$

(continues)

2. Induction step: assume that for a given $k \geq 0$ the formula

$$P(k): \quad (1+2+\cdots+k)^2 = 1^3+2^3+\cdots+k^3$$

holds. Then also

$$\begin{aligned}
&(1+2+\cdots+k+(k+1))^2 \\
&= (1+\cdots+k)^2 + 2(1+\cdots k)(k+1) + (k+1)^2 \\
&= 1^3+\cdots+k^3 + 2\cdot\frac{k(k+1)}{2}\cdot(k+1) + (k+1)^2 \\
&= 1^3+\cdots+k^3 + k(k+1)^2 + (k+1)^2 \\
&= 1^3+\cdots+k^3 + (k+1)^3.
\end{aligned}$$

We have seen that from $P(k)$ being true it follows that $P(k+1)$ is true, that is, $P(k) \Rightarrow P(k+1)$ for all $k \geq 0$. By the induction principle for natural numbers 1.9 we conclude that $P(n)$ is true for all $n \in \mathbb{N}$. □