

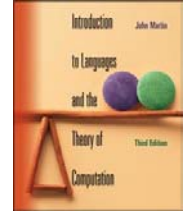
INF210 Datamaskint teori (Models of Computation)

Fedor V. Fomin

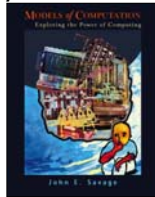


Textbook:

John Martin, Introduction to Languages and the Theory of Computation, McGraw Hill, Third Edition.



- Some parts from John Savage, Models of Computation, Addison-Wesley, will be used. Booklets with selected parts from Savage's book will be available at Studia for approximately 80 NOK



For more information see

- www.ii.uib.no/~fomin/inf210/



What this course is about?



What this course is about?



Model of computation



What this course is about?



*What a computing machine can do
and what it could not do?*



Surprisingly, the question is much older than computers are.



*1936: an abstract machine that
has all the capabilities of today's
computers*

Alan Turing (1912-1954)



- In 1940's and 1950's simpler kinds of machines 'finite automata' were studied by a number of researchers.



Study of formal 'grammars'

*Naom Chomsky
(1928-)*



*What could and what could
not be computed*

Stephen Cook



Type of machines

- Very restricted (finite automata)
- More complicated (push-down automata)
- Turing machine
- Logical circuits



Subject to study

- What these machines can do and what they can not.



Part I. Mathematical Notations and Techniques

- Sets operations Complement A' ,
union $A \cup B$,
intersection $A \cap B$,
difference $A - B$



Venn diagram, example

- De Morgan laws

$$(A \cup B)' = A' \cap B'$$

$$(A \cap B)' = A' \cup B'$$



- Symmetric difference

$$A \oplus B = (A - B) \cup (B - A)$$



Power set

- The set of all subsets of the set A : 2^A
- If A has n elements then 2^A has 2^n elements (proof by induction)



Cartesian product (A cross B)

- $A \times B$



Logic

- Proposition,
- compound proposition,
- logical connectives:
 - Conjunction \wedge
 - Disjunction \vee
 - Negation \neg



Logic

- Conditional connective: if p then q :

$$p \rightarrow q$$



Logical Implication and Equivalence

$$P \Rightarrow Q$$

(P logically implies Q)

$$P \Leftrightarrow Q$$

(P and Q are logically equivalent)



Quantifiers

\exists Existential

\forall Universal



Question: What is that?

- $(\exists A, B, C)(\forall v)[(A(v) \vee B(v) \vee C(v)) \wedge (\forall w)(E(v, w) \rightarrow \neg(A(v) \wedge A(w)) \wedge \neg(B(v) \wedge B(w)) \wedge \neg(C(v) \wedge C(w)))]$



Answer:

- Graph 3-coloring



Functions

$f: \text{Domain} \rightarrow \text{Codomain}$



Functions

- Onto (surjection)
- One-to-one (injection)
- Bijection
- Compositions and Inverses



Relations

- A relation on a set A is a subset of $A \times A$



Relations

1. Reflexive $\forall a \in A (aRa)$
2. Symmetric $\forall a, b \in A (aRb \rightarrow bRa)$
3. Transitive $\forall a, b, c \in A (aRb \wedge bRc \rightarrow aRc)$

Equivalence relation 1+2+3



Examples

- Relatives
- Subsets



Languages

- Alphabet Σ
- A string over an alphabet
- The length of a string x over Σ , $|x|$
- Null string Λ
- Σ^*



- Language L over alphabet Σ is a subset of Σ^*



Examples of languages (maybe not very sophisticated)

- The set of strings of 0's and 1's with an equal number of each:
 - $\{\Lambda, 01, 10, 0011, 0101, 1001, \dots\}$
- The set of binary numbers whose value is even and positive
 - $\{10, 100, 1000, \dots\}$
- $\{\Lambda\}$



Strings

- Concatenation
- Substring



Concatenation

- Languages
 - $\{\text{para, huma}\}\{\text{normal, noid}\} = \{\text{paranoral, paranoid, humanormal, humanoid}\}$
- $L\{\Lambda\} = L$



Power

- $a^k = aa \dots a$, ($a \in \Sigma$)
- $x^k = xx \dots x$, ($x \in \Sigma^*$)
- $\Sigma^k = \Sigma \Sigma \dots \Sigma = \{x \in \Sigma^* \mid |x| = k\}$
- $L^k = LL \dots L$ ($L \subseteq \Sigma^*$)



Important

- $a^0 = x^0 = \Lambda$
- $\Sigma^0 = \Lambda^0 = \{\Lambda\}$
- $L^k \not\subseteq \Sigma^k$



Example

If $\Sigma = \{0, 1\}$, $\Sigma^1 = \{0, 1\}$

$\Sigma^2 = \{00, 01, 10, 11\}$

$\Sigma^3 = \{000, 001, 010, 011, 100, 101, 110, 111\}$



Example

If $\Sigma = \{0, 1\}$, $\Sigma^1 = \{0, 1\}$

alphabet

Set of strings



Kleene star



Stephen Cole Kleene
1909-1994

$$L^* = \bigcup_{i=0}^{\infty} L^i$$

$$L^+ = \bigcup_{i=1}^{\infty} L^i$$



How to describe language???



Problem

Given a string w in Σ^* ,
decide whether or not w is in L .



Example

- Primality test: given a string of 0's and 1's to say 'yes' if the string is a binary representation of a prime or say 'no' if not.



If it is hard to decide whether a given string s of the language L_x of valid strings in programming language X



Compiling programs in X is hard



If it is hard to decide whether a given string s of the language L_x of valid strings in programming language X



Compiling programs in X is hard

Proof: If it is easy to generate code we can run translator, and conclude that the s is a valid member of L_x when translator producing object code



Main problems

- Generate languages
- Recognize languages



Summary

- An **alphabet** is any finite set of symbols
- A **string** is a finite-length sequence of symbols
- A **language** is a set (possibly infinite) set of strings, all of which choose their symbols from some one alphabet.

