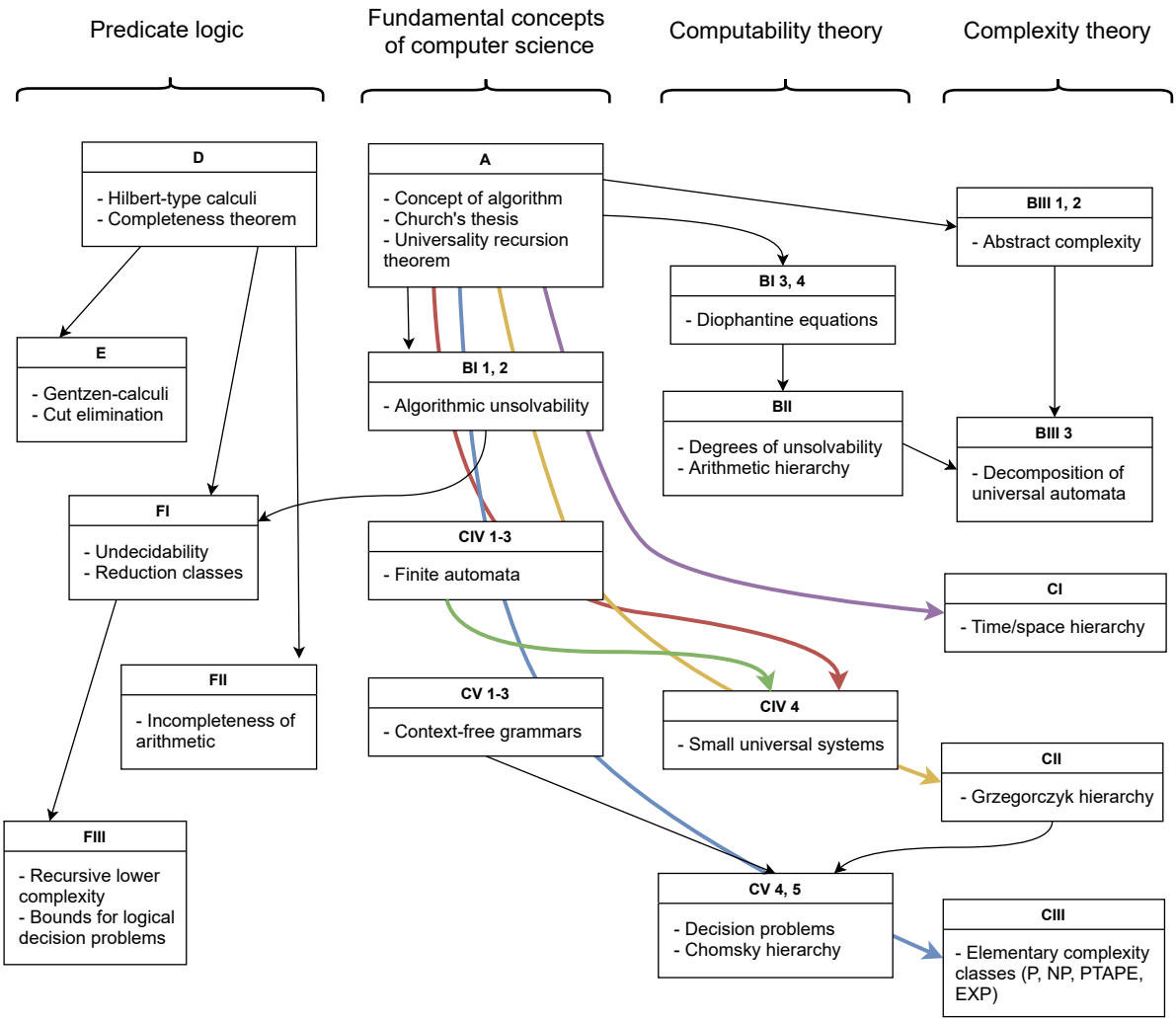
> From Egon Börger, Computability, Complexity, Logic - North-Holland, 1946. Translated into English by J. C. Harvey.



Preface

The theme of this book is a pair of concepts, already recognized as belonging together by Leibniz, whose mathematical development from Frege to Turing has laid the theoretical foundation of computer science: the concept of formal language as carrier of the precise expression of meant, facts, problems, and the concept of algorithm or calculus, that is, formally, operating procedure for the solution of precisely described questions and problems. The book gives a unified introduction to the modern theory of these concepts, to the way in which they developed first in mathematical logic and computability theory and later in automata theory, the theory of formal languages and complexity theory. Apart from considering the fundamental themes, and nowadays classical aspects of these areas, the subject matter has been selected to give priority throughout to the new aspects of traditional questions, results and methods which have developed from the needs or knowledge of computer science and particularly of complexity theory.

Book 1 - Elementary computation theory

The first book has the concept of algorithm as its object, particularly its precise mathematical definition, and the study of its basic properties.

Historically speaking, the posing of metamathematical questions has provided the impulse for the growth of computability theory into a recognized branch of knowledge today. Here, especially in the first third of our century, there developed the consciousness of the need to find a mathematically precise and sufficiently general explication of the intuitive concept of algorithmic processes in order, generally, to lead to rigorous proofs of the impossibility of algorithmic solutions of particular problems. The significant achievement of such an explication, particularly in the form given by Turing in 1937, decisively influenced the development of the first electronic computing machines.

With the progressive development of computers, considering the general investigations into the complexity of design and implementation of algorithms, it has even proved to be practically relevant to have available a mathematically flexible formulation of the concept of algorithm which is, to the greatest extent, **machine-independent**.

Accordingly, in this book we present a unified approach to the classical themes of computability theory and to the foundations of complexity theory.

Thus, in Ch. AI we introduce a **general model of transformation and computation systems** in whose form there appear the concepts of Turing- and register- machines, for the explication of algorithmic procedures. The model also shapes the concepts representing special classes of algorithms such as **finite automata** (Ch. CIV) and **context-free grammars** (Ch. CV), which are central for the construction, implementation and study of properties of programs, particularly of high level programming languages.

The study of the basic properties of the general concept of algorithm presented here has a twofold orientation: the general methods developed in Ch. AII for showing a universal algorithmic procedure in Ch. AIII and justification of recursive programming techniques lead on the one hand to concrete problems which prove to be algorithmically unsolvable (Ch. BI) and to a hierarchical determination of the relative degree of difficulty of algorithmically unsolvable problems (Ch. BII); on the other hand they yield, for a natural machine-independent concept of complexity, the demonstration of solvable problems which prove to have no optimal solution, to admit no algorithmic optimization process. These methods also achieve a precise determination of the complexity of the main components of a universal program (namely, the input, output, transition and stop mechanisms — Ch. BIII).

Also for concrete complexity concepts supported by measures of time and space requirements of Turing machines or register machines, the reduction and diagonalization methods developed in Ch. AII, Ch. BII for the analysis of the complexity of algorithmically unsolvable problems, yield analogous hierarchical determinations of complexity for algorithmically solvable problems and computable functions (Ch. CI). In particular, we introduce the now classical example of the complexity aspects of primitive recursive functions — a historically and mathematically significant subclass of algorithmically computable functions, for which complexity measures based on **rate of growth**, **recursion depth**, **program nesting depth** and **computing time** can be proved to be equivalent. The considerations posed by this lead to the opposite investigation (Ch. CIII) of problems which are solvable with exponential or polynomial bounds on computation — a theme which lies at the center of current research interests. Exactly the above mentioned investigation into low complexities, in the tension between the theoretical and actual solvability and computability of problems, finds application and correspondence in the analysis of the complexity of the logical decision problems in the second book (Ch. F).

Book 2 - Elementary predicate logic

G. W. Leibniz recognized as a necessary condition for the realization of the old dream of a general method for the solution of arbitrary scientific problems the development of a *characteristica universalis*, a universal language of sufficient mathematical precision, in which the totality of facts can be uniquely expressed. Leibniz' efforts to build such a language, in which in addition the structure of things and facts should be mirrored in the structure of the linguistic terms and expressions representing them, first achieved a success in Frege's work which gave precision to it. Frege developed a universal language for the formalization of all (then known) mathematical facts with the aim of separating the "logical" from the historical, psychological and similar contents of mathematical thought-processes and to reduce all legitimate mathematical deductions and concepts to a few clear basic principles. The explication and separation of a universal logical language into the aspect concerned with content **(semantics)** and the formal-logical **(syntactical)** aspect assumed its form, still valid today, in the work of Tarski 1936 and Gödel 1930 and passed its final fundamental theoretical test in Gödel's completeness theorem which

yielded a syntactical (read: **algorithmic**, depending only on the external form of the signs occurring, and not on their meaning) characterization of the semantic concept of logical universal validity (truth).

Ch. D is devoted to this theme. As direct consequences of the given proof of completeness we deal, in Ch. DIII, with some simple problems of characterizing mathematical concepts and structures by first (or higher) order formulas which very clearly illustrate the role of the formal logical language underlying numerous mathematical problems. As an application of a simple model construction of Skølem we also show here the Herbrand reduction of the predicate calculus to the sentence-logic subcalculus and deduce from it the completeness of the resolution calculus, which lies at the basis of many computer implementations of predicate logic in automatic proof procedures or programming languages such as Prolog.

The successful precise definition and separation of the syntactical and semantic aspects of formal logical languages represent a simple prototype for the necessary distinction between the syntactical and semantical phases of programming languages. In relation to this, (first- or higher order-) logic languages are suitable for expressing many problems connected with programming languages and have found well known applications there (for instance the database theory). Many logic-linguistic elements even enter in many higher level programming languages; logic programming raises logical (construction- and proof-) methods almost to a principle. **Stimulated by Hoare, all kinds of formal rules for proving the correctness of structured programs were set up which were adapted from Getzen's logical analysis of the first order proof concept and correspond to the basic of mathematical proofs processes.** We follow this Gentzen approach a little distance in chapter Ch. E and give there at the same time an elementary view of the character of proof-theoretic investigations. In particular, we present the normal form of derivations from Gentzen's **Hauptsatz** which clearly and explicitly shows the interplay of proposition- and quantifier- logic in derivations, clearer than in the resolution calculus and from which can be seen the close connection between logical proof systems (enumeration processes) and logical test systems (recognition procedures). As an application example we deduce from the methods of Gentzen's normal form some model-theoretic theorems which play a role in the more recent investigations in complexity theory (see Ch. F).

With the realization of Leibniz' idea of a *characteristica universalis* as the language of first order predicate logic, the dream, mentioned at the beginning, of an all-embracing method of solving problems, which already found expression in the *Ars Magna* of R. Lullus, took the form of a mathematical task: **to look for an algorithm which for each predicate logic expression decides whether or not it is universally valid**. Hilbert called this problem the decision problem per se; the historical background was the Hilbert Programme which was formulated for the purpose of providing the whole of Mathematics with a foundation proof against the occurrence of paradoxes and to characterize the different branches of Mathematics by first order axioms so that each mathematical proof would yield a logical derivation of the assertion from the given axioms. Church and Turing in 1935 were able to show, by giving mathematical precision to the intuitive concept of algorithm (see Ch. A), that **no such algorithm exists and thus the first order concept of universal logical validity is really calculable in the sense of recursive enumerability but not effectively decidable**. Refinements and applications of Church's and Turing's methods of proof have led to corresponding assertions concerning the complexity of restricted (decidable as well as undecidable) logical decision problems, which arise from the formalizability of suitable decision problems for special classes of algorithms (see complexity classes in Ch. C); as representative of this name here the Prolog-definability of all computable functions and the NP-completeness of decision problems of propositional logic (read: that decidable part of the predicate logic which is described by the theory of Boolean functions). In content as well as in method there also belongs here the impossibility in principle, first brought to light in the first Gödel incompleteness theorem, of encompassing the truth-concept of a first order theory in a calculus in which all numerical equations $\underline{n} + \underline{m} = \underline{n + m}$ and $\underline{n} \cdot \underline{m} = \underline{n \cdot m}$ for numerical term $\underline{r}$ representing the number r are derivable. This and related matters is the content of Ch. F.

Table of contents

2. Post's correspondence problem
3. Domino problem
4. Rödding's path problem

3. Exponential diophantine equations
    1. Simulation of RO
4. $\lambda x, y, z.\, x = y*$ is diophantine. Pell equations.

- BII. The arithmetical hierarchy and degrees of unsolvability

1. Recursively enumerable predicates
    1. Representation theorem
    2. Universality
    3. Arithmetical hierarchy
2. Enumeration- and hierarchy- theorems
    1. Representation theorem
    2. Determination of complexity
        1. Infinity and cardinality statements
        2. Arithmetical truth-concept
3. Reduction concepts and degrees of unsolvability
    1. Reduction concepts
        1. Theorem of Post
    2. Index sets
        1. Theorem of Rice and Shapiro
        2. $\sum_n$-complete program properties
    3. Creativity and $\sum_1$-completeness
        1. Theorem of Myhill
    4. Simple sets
        1. $\equiv_1$ versus $\equiv_m$ versus $\equiv_{t\,t}$
        2. Theorem of Dekker and Yates
    5. Priority method
        1. Theorem of Friedberg and Muchnik
    6. Complexity of the arithmetical truth concept

- BIII. Abstract complexity of computation

1. Speed-up phenomena
    1. Abstract measures of complexity
    2. Blum's speed-up theorem
    3. Impossibility of effective speed-up
2. Functions of arbitrary complexity
    1. Theorem of Rabin-Blum-Meyer on functions of arbitrarily large program- or computing- time complexity
    2. Blum's program-shortening theorem
    3. Gap theorem
    4. Union theorem
3. Decomposition theory for universal automata
    1. Characterization of the run-time-, input- output-transition- and stop- functions of universal automata
    2. Impossibility of uniform recursive simulation bounds on universal automata

C. Recursiveness and complexity

- CI. Complexity classes of recursive functions

1. The k-tape Turing-machine model
    1. Tape reduction
    2. Tape- and time- compression
    3. Simulation complexity of a universal program

2. Time- and place- hierarchy theorems
    1. Theorem of Fürer
3. Complexity of non-deterministic programs
    1. Theorem of Savitch

- CII. Complexity classes of primitive recursive functions

1. Grzegorczyk hierarchy theorem
    1. Equivalence of the characterization by growth-rate
        1. Limited recursion
        2. Excursus on Ackermann branches
    2. Recursion- and loop- depth
    3. Computing-time complexity from Kleene normal form with polynomially bounded or R_3-coding functions
2. E_n-Basis and E_n-computing time hierarchy theorem
3. Ackermann function and Goodstein sequences
    1. Theorem of Goodstein, Kirby and Paris

- CIII. Polynomially- and exponentially- bounded complexity classes

1. NP-complete problems (programming problems)
    1. Halting
    2. Domino
    3. Partition
    4. Knapsack
    5. Clique
    6. Hamiltonian cycle
    7. Traveling salesman
    8. Integer
2. Complete problems for PTAPE and exponential classes

- CIV. Finite automata

1. Characterization by (non-)deterministic acceptors and regular expressions
    1. Theorem of Rabin and Scott
    2. Theorem of Kleene
2. Characterization by indistinguishability congruence relation
    1. Theorem of Myhill and Nerode with corollaries
        1. State minimization
        2. Examples of non-regular languages
        3. Loop lemma
        4. 2-way automata
3. Decomposition theorems
    1. Product decomposition
    2. Modular decomposition
        1. Rödding normal form in sequential and parallel signal processing
4. Small universal programs
    1. 2-dimensional Turing machine with 2 states and 4 letters
    2. 2-dimensional Thue-system with 2 rules and 3 letters
    3. PTAPE-complete Loop problem

- CV. Context-free languages

1. Normal forms of Chomsky and Greibach
    1. Derivation trees
2. Periodicity properties
    1. Loop lemma
    2. Parikh's theorem
    3. Inductive characterization through substitution iteration

3. Characterization by machines
    1. Push-down automata
    2. Closure properties
4. Decision problems
    1. Decidability theorem for context-free and regular grammars
    2. Complexity of the equivalence problem for regular expressions
    3. Undecidability theorem for context-free grammars
    4. Impossibility of effective minimization
5. Comparison with the Chomsky hierarchy classes
    1. Intersection of regular with bracket languages
    2. L-R derivation restrictions of type-0 grammars
    3. Context-dependent languages
        1. Space-requirement theorem
        2. LBA problem

D. Logical analysis of the truth concept

- ### DI. Syntax and semantics
    1. Formal languages of the first order
    2. Interpretation of formal languages
    3. Hilbert calculus
- ### DII. Completeness theorem
    1. Derivations and deduction theorem for sentence logic
    2. Completeness of propositional logic
        1. Lindenbaum maximization process
        2. Analytical tables
        3. Resolution
    3. Derivations and deduction theorem of the predicate logic
    4. Completeness of predicate logic
- ### DIII. Consequences of the completeness theorem
    1. Weakness of expressibility of PL 1
        1. Theorem of Skølem
        2. Compactness theorem
        3. Non-characterizability of the concept of infiniteness
        4. Non-standard models of number theory
    2. Second-order predicate logic and type theory
        1. Characterization of finiteness and countability and of $(N; O, +1)$ in second order
        2. Languages of n-th order
    3. Canonical satisfiability
        1. Skølem normal form
        2. (Minimal) Herbrand models
        3. Predicate logic resolution
        4. Procedural interpretation of Horn formulas
        5. Completeness of SLD-resolution

E. Logical analysis of the concept of proof

- ### EI. Gentzen's calculus LK
    1. The calculus LK
    2. Equivalence to the Hilbert calculus
- ### EII. Cut elimination for LK

- EIII. Consequences of the cut elimination theorem
  1. Gentzen's Haptsatz (cor.: Herbrand's theorem)
  2. Interpolation theorem (cor.: definability theorem)
     1. Analysis of interpolation- and definability- theorems in the finite

F. Complexity of logical decision problems

- FI. Undecidability and reduction classes
  1. Theorems of Church & Turing, Trachtenbrot, Aanderaa & Börger
     1. Cor.: Prolog program as axiom of an essentially undecidable theory, respectively, as a satisfiable formula without recursive models
     2. Prolog-definability of all computable functions
     3. Impossibility of recursive interpolation and of recursive explication bounds of implicit definitions
  2. Reduction class of Kahr-Moore-Wang
- FII. Incompleteness of arithmetic
  1. Incompleteness theorem of Gödel
  2. Theorem of Löb
- FIII. Recursive lower complexity bounds
  1. Reduction method
  2. Complexity of Boolean functions
     1. Theorem of Cook
     2. Theorem of Henschen & Wos
     3. Polynomial equivalence of Horn- and network-complexity
     4. Theorem of Stockmeyer
  3. Spectrum problem
     1. Spectrum characterization of the E_3-computation time hierarchy
        1. Theorem of Rödding & Schwitchtenberg
        2. Jones & Selman
        3. Christen
     2. Logical characterization of NP by global existential second order predicate
        1. Theorem of Faing
     3. Logical characterization of P by $\mathrm{PL1} + \mathrm{LFP}$-with-order
        1. Theorem of Immerman & Vardi
     4. Logical characterization of PTAPE by $\mathrm{PL2} + \mathrm{TC}$
        1. Theorem of Immerman
  4. Complete decision problems for polynomial and exponential complexity classes
     1. Theorem of Lewis
        1. $\mathrm{NEXTPTIME}$-completeness of the satisfiability problem of the monadic Gödel-Kalmar-Schütte class $V^\infty \Lambda^2 V^\infty \circ$ monad
     2. Theorem of Plaisted
        1. $\mathrm{EXPTIME}$-completeness of the satisfiability problem of the Bernays-Schönfinkel class $V^\infty \Lambda^\infty$ in Horn formulas
        2. Corollary: $\mathrm{NEXPTIME}$-completeness for $V^\infty \Lambda^\infty$
     3. Theorem of Plaisted and Deneberg & Lewis
        1. PTAPE-completeness of the satistfiability problem of the Bernays-Schönfinkel class in Krom- (and Horn-) formulas
        2. Corollary: PTAPE-completeness of $V^\infty \Lambda^\infty$ in deterministic Krom- and Horn-formulas