Formal Languages & Compilers

Prof. Luca Breveglieri

What is a Formal Language ?

- **Artificial vs natural language**
  - communication with machines vs with human beings
  - non-verbal vs verbal contents
  - formal vs non-formal structure
- **Formal language**
  - a language is *formal* if its *syntax* (structure) and *semantic* (interpretation) are defined in a precise *algorithmic* way
  - there is an *effective procedure* that verifies the grammatical *correctness* of the language phrases and determines their *meaning*
  - in a more restricted meaning a formal language is
    - a mathematical object built over an *alphabet*
    - by means of some axiomatic rules called *grammars*
    - or by means of mathematical tools called *automata*
- **Theory of formal languages**
  - deals with the structure or *syntax* of the language phrases
  - uses both *grammars* and *automata* to work with *languages*
  - helps to design *compilers* (e.g., for programming languages)

Brief History

- **Years ’50 – foundation**
    - Noam Chomsky proposes the mathematical model of a grammar (1956)
    - initially for the purpose of preserving and studying natural languages
    - but quickly grammars become the tool to model artificial languages
- **Years ‘60 – basic research**
    - definition of the programming language ALGOL and later of many others
    - discovery of the connections between formal languages and automata
    - invention of formal grammars: regular, (context-)free, (context-)sensitive
    - invention or extension of automata: finite, pushdown, and others
    - development of the theory and practice of automated compilation
- **Years ‘70-’80 – consolidation**
    - formal language theory becomes a standard university discipline
    - textbooks appear, as well as compiler SDKs: e.g., Flex and Bison
- **Later until today – extensive application**
    - many new formal languages are designed, for data, media, and communication
    - grammars and automata are widely used in basic and innovative applications

Lecture - Contents

- **Basics on languages and their operations**
- **Generative grammar**
  - regular expression
  - context-free grammar
- **Recognition automaton**
  - finite automaton
  - pushdown automaton
  - syntax analyzer
- **Translation and semantic**
  - syntactic translation
  - attribute grammar
  - static flow analysis

Laboratory - Contents

- **Software tools for designing compilers**
  - lexical analysis – Flex
  - syntax analysis – Bison
  - design of a sample compiler
    - for a pseudo-C language
- **Where**: classroom
- **How long**: 6 sessions
- **When**: approximately from december to january
- A few software tools are available for individual self-training, e.g., Flex, Bison and a compiler-assembler

Textbooks

- **Textbook of the course (in english)**
  - **Stefano Crespi Reghizzi, Luca Breveglieri, Angelo Morzenti, "*Formal Languages and Compilation*", Springer, 2nd ed., 2013, XII, 397 pages, ISBN 978-1-4471-5513-3**
- **Testo del corso (in italian)**
  - **Stefano Crespi Reghizzi, Luca Breveglieri, Angelo Morzenti, «Linguaggi Formali e Compilazione", Società Editrice Esculapio, 2° ed., 2015, 488 pagine, ISBN 978-887488-875-7**
- **The first editions of the text are obsolete and do not match the current program of the course**
- **Other texts may be suggested on the website:**

beep.metid.polimi.it

Other Didactic Material

- **Lecture slides (in english)**
  - on the website beep.metid.polimi.it (more may be added or changed)
- **Laboratory slides and other material (in english)**
  - on the website beep.metid.polimi.it (more may be added or changed)
- **Exam texts and solutions (in english):**
  - on the website beep.metid.polimi.it
  - an english exercise book is online
- **Please request registration to the website !**
  - click on the registration button and wait for authorization to be released in one or two days

More Litterature

- the classical reference text on formal languages
  - A. Salomaa, *Formal Languages*, Academic Press, 1973
- another classical text with a special attention to the regular and free languages, and to their automata
  - J. Hopcroft, J. Ullman, *Formal Languages and their Relation to Automata*, Addison Wesley, 1969
- a basic text on theoretical computer science, which deals with the theory of computation and complexity, but with a special attention to formal languages
  - D. Mandrioli, C. Ghezzi, *Theoretical Foundations of Computer Science*, John Wiley & Sons, 1987

Course & Exam Structure

- **Lecture**
  - theory of FLC with a few examples and exercises
- **Laboratory**
  - design of compilers by means of Flex and Bison
- **Exam**
  - written classwork with exercises
  - calls from the end of the course
  - open-book exam: textbook and notes admitted
  - two independent parts: theory and laboratory
  - parts can be done separately within four calls
  - refuse the final grade ⇒ do a part again (or both)
- **No oral exam !**

Contacts

- **Lecture**
  - prof. Luca Breveglieri
  - office: building 20 (DEIB) 1st floor, Via Ponzio 34/A, MI
  - visit: office hours or appointment by email / phone call
  - email: luca.breveglieri@polimi.it
  - tel: 02 2399 3653 (office) 3400 (DEIB reception)
  - web: http://www.deib.polimi.it
- **Practice**
  - dr. Alessandro Barenghi
  - see web: http://www.deib.polimi.it