Formal Language & Finite Automata

Introduction

Dr Noleen Koehler

University of Leeds

Overview

1. Module Summary
2. Logistics
3. Formal Language & Finite Automata

Module Summary

Module summary

Programming and programming languages are essential tools for computer science practitioners. The theory and technology that underpins the development of programming languages, in the past and the present, is that of formal languages and finite automata. This module exists to provide a solid foundation of formal languages and finite automata which will be built on in subsequent modules. This module focuses on the formal specification of languages, the corresponding hierarchy of abstract machines and contributes to developing an understanding of design considerations of programming languages.

Learning outcomes

On successful completion of this module a student will have demonstrated the ability to:

- recall definitions and theorems relating to formal languages and finite automata and apply them appropriately.
- describe and compare different models of computation.
- articulate the existence of undecidable problems.
- illustrate the applications of formal languages and finite automata in the fields of programming language design and compilers.

Syllabus

This module covers the following 3 topic areas:

- Formal languages : regular languages, context-free languages, recursive languages, recursively enumerable languages, grammars, Backnus Naur form (BNF), parsing and parse trees.
- Abstract models of computation : finite automata and Turing machines (both deterministic and non-deterministic).
- Computability : computable and uncomputable functions, computable and uncomputable sets, undecidable problems, the Halting problem, Rice's theorem.

Logistics

Delivery

1. Lectures (20 hours)
    - Monday 17:00-18:00 (RBLT)
    - Tuesday 16:00-17:00 (Conference Auditorium 1)
2. Tutorials (10 hours)
    - Weekly from week 2 (see timetable)
3. Office hours (on request)
    - Email: n.koehler@leeds.ac.uk
4. Math Lunch Hour
    - Tuesday 12:00-13:00 (Bragg GR.25)
    - Wednesday 13:00-14:00 (Bragg GR.25)

Your responsibility

1. Private study (70 hours)
    - Approximately 6 hours per week
    - Additional time for revision

Assessment and additional resources

1. **Summative:** 100% exam
2. **Formative:** weakly worksheets $\rightarrow$ feedback in tutorials

Additional resources:

[1] Michael Sipser. **Introduction to the Theory of Computation.** Cengage learning, 2012.

[2] **JFLAP.** URL: www.jflap.org.

Teaching approach

- We will study as set of types of languages and their associated machine models and relationships.
- The lectures will introduce the topics and present proofs of any theorems required. The details of the proofs can be found on the module website.

Formal Language & Finite Automata

Strings

In this module we define module of computation for strings, as we progress through the module you will hopefully see that this is not unrealistics.

- A string is a finite sequence over some finite set of symbol, called the **alphabet** (usually denoted Σ).
- A **language** over an alphabet Σ is a set of strings over Σ (usually denoted L).
- If w is a string over Σ then the length of w, denoted $|w|$, is the number of symbols in the sequence.
- The symbol ϵ denotes the empty string, which has length 0.
- $w = w_1 w_2 w_3 \dots w_n$ where $w_i \in \Sigma$ for all $1 \leq i \leq n$ is a string of length n over the alphabet Σ whose first element is w_1, whose second element is w_2, etc. .

Deterministic finite state automata (DFA)

Definition (Finite automaton)
A **deterministic finite automaton** is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$, where

- Q is a finite set called the **states**,
- Σ is a finite set called the **alphabet**,
- $\delta : Q \times \Sigma \to Q$ is the **transition function**,
- $q_0 \in Q$ is the **start state**, and
- $F \subseteq Q$ is the set of **accepting states**.

Graphical representations of DFA's

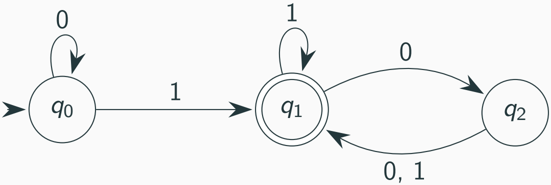

Figure 1: A simple state diagram, M_1, with 3 states.

Computation

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a finite automaton and let $w = w_1 \dots w_n$ be a string where $w_i \in \Sigma$. The automaton M **accepts** w if a sequence of states $r_0, \dots, r_n \in Q$ exists such that:

1. $r_0 = q_0$
2. $\delta(r_i, w_{i+1}) = r_{i+1}$ for $0 \leq i < n$
3. $r_n \in F$