Algorithms and Programming IV
Recap: Concepts of Programming

Summer Term 2021 | 12.04.2021
Barry Linnert
Our Approach for an Efficient Solution for Problems

Problem

Algorithm

Software in a Programming Language

Essential part of the solution of a problem. Very creative phase.

Programming or coding the algorithm in a specific programming language. Easier phase.
Need for Higher Programming Languages

\[ a = b + c \]

<table>
<thead>
<tr>
<th>Assembly Code</th>
<th>Machine Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>mov -0x8(%rbp),%eax</code></td>
<td><code>8b 45 f8</code></td>
</tr>
<tr>
<td><code>mov -0x4(%rbp),%edx</code></td>
<td><code>8b 55 fc</code></td>
</tr>
<tr>
<td><code>add %edx,%eax</code></td>
<td><code>01 d0</code></td>
</tr>
<tr>
<td><code>mov %eax,-0xc(%rbp)</code></td>
<td><code>89 45 f4</code></td>
</tr>
<tr>
<td><code>ldr r2, [fp, #-8]</code></td>
<td><code>e5 1b 20 08</code></td>
</tr>
<tr>
<td><code>ldr r3, [fp, #-12]</code></td>
<td><code>e5 1b 30 0c</code></td>
</tr>
<tr>
<td><code>add r3, r2, r3</code></td>
<td><code>e0 82 30 03</code></td>
</tr>
<tr>
<td><code>str r3, [fp, #-16]</code></td>
<td><code>e5 0b 30 10</code></td>
</tr>
</tbody>
</table>
From Machine Languages to Higher Programming Languages

Machine Languages
  Assembler Languages
  Higher Programming Languages
  Declarative Languages
  Imperative Languages
Classification According to Programming Paradigm

Higher Programming Languages

Declarative Languages
- Functional Languages
- Logical Languages
  
in ALP 1

Imperative Languages
- Object-oriented Languages
  
in ALP 2
Higher Programming Languages

We translated an algorithm in a sequential order.

Individual operations are executed one after the other in the sequence of instructions clearly defined in the source code. You defined thread.

Functional Languages

Imperative Languages

Declarative Languages

Logical Languages

in ALP 1

in ALP 2
What is a different Approach for Implementing an Algorithm?

• We can implement an algorithm non-sequentially.

• An algorithm is called non-sequential when the linear order of its operations is replaced by a non-linear order. We create multiple threads at the same time.

• However, these threads can have
  – A shared memory in one CPU -> concurrent algorithms
  – Shared or distributed memory in more than one CPU -> parallel algorithms
  – Neither shared memory nor CPU -> distributed algorithms
Our Course Perspective

• The programming model is derived from the machine model. We explain existing mechanisms and algorithms depending on the programming model. For these, we discuss existing problems and present possible solutions.

• The course is divided into three major areas of non-sequential programming:
  − Concurrent programming: Machine with one CPU, but with a common memory
  − Parallel programming: Machine with several CPUs with or without shared memory machine, which requires message exchange
  − Distributed programming: Different machines