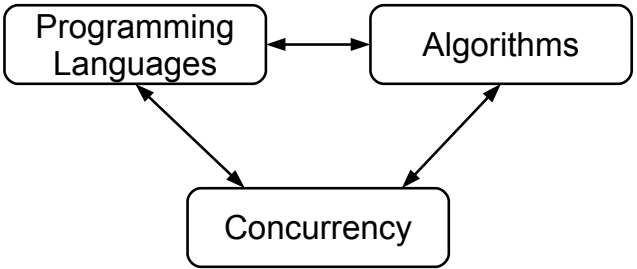
159.341 Programming Languages, Algorithms & Concurrency

Course Introduction

Daniel Playne
d.p.playne@massey.ac.nz

159.341 Course Introduction

This course explores *programming languages*, *algorithms* and *concurrency* and looks at the relationships between them.



What is a Programming Language?

Programming Language - a vehicle for the user to tell the computer what to do to solve the user's problem.

Enables the programmer to bridge the gap between the **problem domain** and the **machine domain**.

- **Problem domain** - the application area of the program.
- **Machine domain** - the machine: CPU, memory, peripherals etc

Why Study Programming Languages?

Increased capacity to express ideas - depth of thought is influenced by the expressive power of the language in which those thoughts are communicated.

Likewise the solutions that programmers develop are influenced by the languages they use - in terms of control structures, data structures, abstractions etc.

Even if the language used does not directly support a particular capability, the wider knowledge of programming languages can still influence the solution.

Why Study Programming Languages?

Improved choice of language - many programmers have little training in Computer Science and either learned to program by themselves or through small training programs. These approaches tend to focus very strongly on one or two particular languages.

As a result these programmers will tend towards using the language they are familiar with for every project they work on. The solutions they design also tend to be very strongly influenced by their favourite language.

Why Study Programming Languages?

Ability to learn new language - computer languages are continuously evolving and improving. This makes computer science and software development an interesting profession but it also means you must continuously adapt new languages and technologies.

Knowledge of the fundamentals of programming languages and computation greatly aides in learning other new languages. It is much easier to learn Java when you are already familiar with concepts of Object-Oriented programming.

Why Study Programming Languages?

Better understanding of implementation - studying the fundamentals of programming languages and computational models can help you understand the implementation issues surrounding those languages.

It allows you to understand the decisions the language developers had to make (and why) which leads to a deeper understanding of the language and allows you to use it more intelligently.

What is an Algorithm?

Algorithm - a well-defined, computational procedure for solving problems.

Algorithms have a finite sequence of unambiguous steps that, when executed, will perform some task.

The execution of an algorithm does not require any knowledge of the problem domain.

Reasons to Study Algorithms

Improved problem-solving - studying algorithms exposes us to different problem-solving techniques and gives us ideas on how to solve new problems when we encounter them.

Many algorithms are based on general approaches that can be applied to many different types of problems, a good understanding of algorithms can give us a library of ideas to try when we encounter a new problem.

Reasons to Study Algorithms

Ability to evaluate algorithms - run-time and memory complexity are two of the most important metrics we use to evaluate algorithms.

A good understanding of algorithm analysis gives us the tools to compare different algorithms and select the most appropriate choice for our application.

Reasons to Study Algorithms

Identify uncomputable or intractable problems - some problems are fundamentally uncomputable and trying to write a program to solve them is simply not possible with classical computing.

Other problems are computable but have a complexity that makes them infeasible to solve in a reasonable amount of time - these problems are often called intractable.

Identify these cases can help to save you from trying to achieve the impossible.

What is Concurrency?

Concurrency - a program where multiple tasks are *in progress* at one time.

A program no longer has one single sequence of instructions but multiple sequences of instructions that may be being executed at once (more on this later).

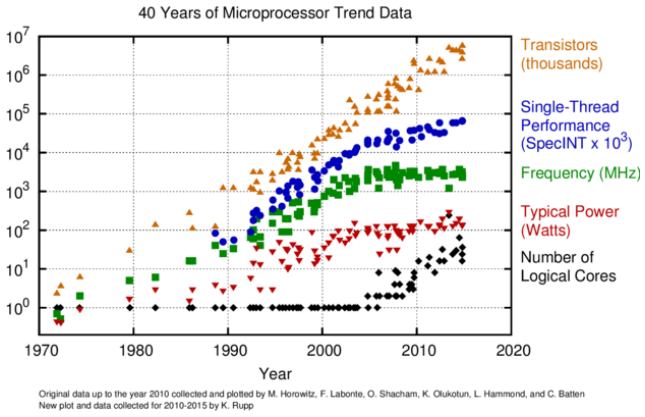
Reasons to Study Concurrency

Hardware is increasingly parallel - for many years computers increased in performance exponentially.

- Moores Law - number of transistors double every 18-24 months (often misquoted as doubles in performance every 18-24 months)
- Dennard Scaling - voltage/current proportional to transistor dimensions.

Reasons to Study Concurrency

Hardware is increasingly parallel - instead of increasing clock speeds and single-core performance, manufacturers are increasing the number of processing cores.



Reasons to Study Concurrency

Hardware is increasingly parallel - CPUs with multiple cores can execute a concurrent program's multiple sequences of instructions at the same time.

If we want to make full use of new hardware, writing concurrent programs is increasingly important.

Reasons to Study Concurrency

Sometimes concurrent solutions are easier or better - some problems are just well-suited to concurrent solutions.

Certain problems have multiple tasks that must be completed, writing concurrent solutions can often be much easier than non-concurrent program.

e.g. GUI's need to respond to user input quickly even when some computation may be occurring in the background.

Summary

This course covers various concepts in programming languages, models of computation, concurrency, algorithms and the relationships between them.