 anybody who ...

... wants to work with real-world scale computer systems

... would like to learn how to analyse and design operational and robust systems

... would like to understand more about the existing trade-off between theory, the real-world, traditions, and pragmatism in computer science

... would like to understand why concurrent systems are an essential basis for most contemporary devices and systems
Many algorithms and concepts for the course are in there
– but not all!

References for specific aspects of the course are provided
during the course and are found on our web-site.
**Organization & Contents**

**Topics**

1. **Concurrency** [3]
   - 1.1. Forms of concurrency [1]
     - Coupled dynamical systems
   - 1.2. Models and terminology [1]
     - Abstractions
     - Interleaving
     - Atomicity
     - Proofs in concurrent and distributed systems
   - 1.3. Processes & threads [1]
     - Basic definitions
     - Process states
     - Implementations

2. **Mutual exclusion** [2]
3. **Communication & Synchronization** [4]
4. **Non-determinism** [2]
5. **Data Parallelism** [1]
6. **Scheduling** [2]
7. **Safety and liveness** [2]
8. **Distributed systems** [4]
9. **Architectures** [1]

---

**Organization & Contents**

**Topics**

1. **Concurrency** [3]
   - 2.1. by shared variables [1]
     - Failure possibilities
     - Dekker's algorithm
   - 2.2. by test-and-set hardware support [0.5]
     - Minimal hardware support
   - 2.3. by semaphores [0.5]
     - Dijkstra definition
     - OS semaphores

2. **Mutual exclusion** [2]
3. **Non-determinism** [2]
4. **Data Parallelism** [1]
5. **Scheduling** [2]
6. **Safety and liveness** [2]
7. **Distributed systems** [4]
8. **Architectures** [1]
Organization & Contents

Topics

1. Concurrency [3]
5. Data Parallelism [1]
7. Safety and liveness [2]
8. Distributed systems [4]
9. Architectures [1]

9.1. Hardware architecture
- From switches to registers and adders
- CPU architecture
- Hardware concurrency

9.2. Language architecture
- Chapel
- Occam
- Ada
- C++

Organization & Contents

Laboratories & Assignments

- Data structures
- Data flow diagrams
- Control flow diagrams
- Abstract types

2. Tasking [1]
- Tasking
- Task lifetimes

3. Distribution [1]
- Multithreading
- Multithreading Synchronization

4. Communication Tasks [1]
- Communication
- Implicit Concurrency

5. Distributed Server [1]
- Distributed server

6. Communication [1]
- Communication
- Synchronization

Assignments [2]

1. Concurrent programming (15%)
- Programming task involving
  - Mutual exclusion
  - Synchronization
  - Mutual exclusion

2. Concurrent programming in multi-core systems (15%)
- Programming task involving
  - Multi-core programming
  - Inter-thread communication

3. Final exam (50%)

Examinations [3]

1. Hurdle check (5%)
- Task structure

2. Midsemester check (15%)
- Task structure

3. Final exam (50%)
- Task structure

The final mark is based on the assignments (10%)

© 2020 Uwe R. Zimmer, The Australian National University