Foundations of Distributed Systems

Introduction

Summertime Semester 2012

TU Berlin/ T-Labs

Administrivia

- Module: MINF-VS-FDS.S12
- Lectures: Tuesdays, 12:00-14:00, TEL Auditorium 2
- Tutorials: Mondays, 16:00-18:00, TEL 1118/19
  ✓ First: April 9
- Mailing list/web page: ISIS

Administrivia (contd.)

- Office hours:
  ✓ Petr Kuznetsov: Monday, 14:00-16:00, FRA 5508
  ✓ Stefan Schmid: Tuesday: 14:00-16:00, FRA 5025
- Credit = (Homework)x(Oral Exam)
  ✓ Homework is binary (>50% done)
  ✓ Bonus for lecture notes bugs
- Please register soon!

What is computing?
What is done by a Turing machine

Alan Turing
23 June 1912 – 7 June 1954

Not well adjusted to concurrency?

Computation as interaction

Robin Milner
1934-2010

This course is about distributed computing:
independent sequential processes that communicate

Concurrency is everywhere!

- Multi-core processors
- Sensor networks
- Internet
- Basically everything related to computing
Communication models

- Shared memory
  - Processes apply (read–write) operations on shared variables
  - Failures and asynchrony
- Message passing
  - Processes send and receive messages
  - Communication graphs
  - Message delays

Clock speed deadend

- Memory wall
  - Performance gap between memory and CPU
- ILP wall
  - Not enough work to spend the cycles
- Power wall
  - Thermal problems caused by higher clock speeds

The case against the “washing machine science”

- Single-processor performance does not improve
- But we can add more cores
- Run concurrent code on multiple processors

Can we expect a proportional speedup?
(ratio between sequential time and parallel time for executing a job)
Example: painting in parallel

- 5 friends want to paint 5 equal-size rooms, one friend per room
  - Speedup = 5

- What if one room is twice as big?

Amdahl’s Law

- \( p \) – fraction of the work that can be done in parallel (no synchronization)
- \( n \) - the number of processors
- Time one processor needs to complete the job = 1

\[
S = \frac{1}{1 - p + \frac{p}{n}}
\]

Painting in parallel

- Assigning one painter to one room, 5/6 of the work can be performed in parallel.
- Parallel execution time = \( 1 - \frac{5}{6} + \frac{1}{6} = \frac{1}{6} + \frac{1}{6} = \frac{1}{3} \)
  \( S = \frac{1}{\frac{1}{3}} = 3 \)

- Can be worse: 10 rooms, 10 painters, one room twice bigger
  \( S = \frac{1}{1 - \frac{10}{11} + \frac{1}{11}} = \frac{11}{2} = 5.5 \)

- But >90% of the work can be parallelized!

A better solution

- When done, help the others
  - All 5 paint the remaining half-room in parallel
- Communication and agreement is required!
- This is a hard task

- And this is exactly what synchronization algorithms try to achieve!
Challenges

- What is a correct implementation?
  - Safety and liveness
- What is the cost of synchronization?
  - Time and space lower bounds
- Failures/asynchrony
  - Fault-tolerant concurrency?
- How to distinguish possible from impossible?
  - Impossibility results

Distributed ≠ Parallel

- The main challenge is synchronization

“you know you have a distributed system when the crash of a computer you’ve never heard of stops you from getting any work done” (Lamport)

History

- Dining philosophers, mutual exclusion (Dijkstra) ~60’s
- Distributed computing, logical clocks (Lamport), distributed transactions (Gray) ~70’s
- Consensus (Lynch) ~80’s
- Distributed programming models, since ~90’s
- Multicores now

Why theory of distributed systems?

- Every computing system is distributed
- Computing getting mission-critical
  - Understanding fundamentals is crucial
- Intellectual challenge
  - A distinct math domain?
Course outline:
Part 1: shared-memory computing

- Read-write shared memory: basics
  - Safe, regular, atomic
  - Shared-memory transformations
  - Atomic snapshots

- Shared-memory implementations
  - Consensus impossibility
  - BG simulation
  - Universal construction