• **Weeks 1–2:** informal introduction  
  • network = path

• **Week 3:** graph theory

• **Weeks 4–7:** models of computing  
  • what can be computed (efficiently)?

• **Weeks 8–11:** lower bounds  
  • what cannot be computed (efficiently)?

• **Week 12:** recap
Week 12

– Conclusions
Recap: Distributed algorithms

Algorithms for computer networks
Recap: 

Distributed algorithms

Identical computers in an unknown network, all running the same algorithm
Recap: Distributed algorithms

Initially each computer only aware of its immediate neighbourhood
Recap:

Distributed algorithms

Nodes can exchange messages with their neighbours to learn more...
Recap: Distributed algorithms

Finally, each computer has to stop and produce its own local output
Recap:
**Distributed algorithms**

Focus on graph problems:
**network topology = input graph**
Recap: Distributed algorithms

Focus on graph problems: local outputs = solution (here: graph colouring)
Recap: Distributed algorithms

Typical research question:

“How fast can we solve graph problem X?”

Time = number of communication rounds
What have we learned?

• Dealing with *unknown systems*
• Dealing with *partial information*
• Dealing with *parallelism*
• Applications beyond distributed computing: fault tolerance, online, streaming, multicore…
Learning objectives

- Models
- Algorithms
- Lower bounds
- Graph theory
Objective 1: Models of computing

- Precisely what is a “distributed algorithm”

- In each of these models:
  - PN, LOCAL, CONGEST
  - deterministic, randomised
Objective 2: Algorithms

- **Colouring paths:** LOCAL, \(O(\log^* n)\)
- **Colouring graphs:** LOCAL, \(O(\log n)\) w.h.p.
- **Gather everything:** LOCAL, \(O(\text{diam}(G))\)
- **Bipartite maximal matching:** PN, \(O(\Delta)\)
- **All-pairs shortest paths:** CONGEST, \(O(n)\)
Algorithm P3CBit: 
Fast colour reduction

\[ c_0 = 123 = 01111011_2 \ (\text{my colour}) \]
\[ c_1 = 47 = 00101111_2 \ (\text{successor’s colour}) \]

\[ i = 2 \ (\text{bits numbered 0, 1, 2, … from right}) \]
\[ b = 0 \ (\text{in my colour bit number } i \text{ was 0}) \]
\[ c = 2 \cdot 2 + 0 = 4 \ (\text{my new colour}) \]

\[ k = 8, \text{ reducing from } 2^8 = 256 \text{ to } 2^4 = 16 \text{ colours} \]
Algorithm P3CBit: Fast colour reduction

\[ c_0 = 123 = 01111 \]
\[ c_1 = 47 = 00101 \]
\[ i = 2 \text{ (bits numbered 0, 1, 2, … from right)} \]
\[ b = 0 \text{ (in my colour bit number } i \text{ was 0)} \]
\[ c = 2 \cdot 2 + 0 = 4 \text{ (my new colour)} \]

\( k = 8 \), reducing from 256 to 2\( \cdot 8 \) = 16 colours

Algorithm idea 3

- Colour palette: \( \{1, 2, \ldots, \Delta + 1\} \)
- Active with probability \( 1/2 \)
- If active, pick a random free colour
  - not used by any neighbour that has stopped
- Try again if conflicts…
Algorithm P3CBit:
Fast colour reduction

$c_0 = 123 = 01111$
$c_1 = 47 = 00101$
$i = 2$ (bits numbered 0, 1, 2, … from right)
$b = 0$ (in my colour bit number $i$ was 0)
$c = 2 \cdot 2 + 0 = 4$

Algorithm idea 3
• Colour palette: \{1, 2, …, $\Delta + 1$\}
• Active with probability 1/2
• If active, pick a random free colour
• not used by any neighbour that has stopped
• Try again if conflicts…

Gathering everything

• Given $E(v, r)$, we can learn $E(v, r + 1)$ in 1 round
  • send $E(v, r)$ to all neighbours, take union

Week 5
Algorithm \( \text{P3CBit} \):
Fast colour reduction
\[
\begin{align*}
c_0 &= 123 = 01111_2 \\
c_1 &= 47 = 00101_2 \\
\end{align*}
\]
(successor's colour)
\[
\begin{align*}
i &= 2 \\
b &= 0 \\
c &= 2 \cdot 2 + 0 = 4 \\
k &= 8, reducing from 2^8 = 256 to 2^4 = 16 colours
\end{align*}
\]

Algorithm idea 3
- Colour palette: \( \{1, 2, \ldots, \Delta + 1\} \)
- Active with probability 1/2
- If active, pick a random free colour
  - not used by any neighbour that has stopped
  - Try again if conflicts…

Gathering everything
- Given \( E(v, r) \), we can learn \( E(v, r+1) \) in 1 round
  - send \( E(v, r) \) to all neighbours, take union

Algorithm \( \text{BMM} \): Maximal matching

- **Blue nodes** send proposals to their orange neighbours one by one
  - using port numbers
- **Orange nodes** accept the first proposal that they get
  - using port numbers to break ties
Algorithm P3CBit:

Fast colour reduction

\[ c_0 = 123 = 01111 \]
\[ c_1 = 47 = 00101 \]

\( i = 2 \) (bits numbered 0, 1, 2, … from right)

\( b = 0 \) (in my colour bit number \( i \) was 0)

\[ c = 2 \cdot 2 + 0 = 4 \]

\( k = 8 \), reducing from 2\(^8\) = 256 to 2\( \cdot 8 \) = 16 colours

Algorithm idea 3

• Colour palette: \( \{1, 2, …, \Delta + 1\} \)
• Active with probability \( 1/2 \)
• If active, pick a random free colour
• not used by any neighbour that has stopped
• Try again if conflicts…

Gathering everything

• Given \( E(v, r) \), we can learn \( E(v, r + 1) \) in 1 round
• send \( E(v, r) \) to all neighbours, take union

Algorithm BMM:

Maximal matching

• Blue nodes send proposals to their orange neighbours one by one
• using port numbers
• Orange nodes accept the first proposal that they get
• using port numbers to break ties

Algorithm APSP

wave
token
Objective 2: Algorithms

- Reductions!
- Graph colouring is a very useful subroutine
Objective 3: Lower bounds

• Covering maps: what cannot be solved at all in PN model

• Local neighbourhoods: what cannot be solved fast in any model

• Ramsey’s theorem: what cannot be solved in $O(1)$ time
Objective 4: Graph theory

- Basic definitions
- Connections between graph problems
  - e.g. maximal matching $\rightarrow$ small vertex covers
- Ramsey’s theorem
  - at least for $c = 2, k = 2$
What else is studied in distributed computing?

- Fault-tolerance
- Asynchrony
- Shared memory
- Physical models
- Robot navigation
- Nondeterminism
- Complexity measures
- High-performance computing
- Practical aspects of networking …
What next?

- ICS-E4020 Programming Parallel Computers
  - 5th period, 5 credits, intensive course
  - programming modern parallel computers: multicore, GPU, memory hierarchies …
  - hands-on programming exercises
  - main goal: make it as fast as you can!
What next?

• Just ask if you want to do more!
  • master’s thesis topics?
  • summer internships?
  • doctoral studies?
Practicalities

- 2nd mid-term exam: **11 December**
  - remember to register on time!

- **Course feedback: deadline 12 December**
  - this is a new course, feedback very important!
  - 1 extra point in grading
What to expect in the exam?

- See the learning objectives!
- Do not think that you can safely forget what we learned during the 1st period!
- Expect both algorithm design and lower bound proofs
Examples of old exam problems

- Prove: no deterministic PN-algorithm that finds a *minimum vertex cover* in cycle graphs, given a *minimal vertex cover*
Examples of old exam problems

• Prove: no deterministic PN-algorithm that finds a 6-colouring in cycle graphs given a maximal independent set
Examples of old exam problems

• Counting problem: all nodes output $|V|$

• Prove: no deterministic PN-algorithm for cycle graphs

• Prove: no $o(n)$-time deterministic LOCAL-algorithm for cycle graphs
Examples of old exam problems

• Prove: no deterministic PN-algorithm for maximal matching in arbitrary graphs
Examples of old exam problems

• Prove: no deterministic $o(n)$-time PN-algorithm for weak 2-colouring in paths of length $\geq 3$
Examples of old exam problems

• Give an elementary proof that any graph with 6 nodes contains a clique with 3 nodes or an independent set with 3 nodes
• **Weeks 1–2:** informal introduction
  • network = path 😄

• **Week 3:** graph theory

• **Weeks 4–7:** models of computing
  • what can be computed (efficiently)?

• **Weeks 8–11:** lower bounds
  • what cannot be computed (efficiently)?

• **Week 12:** recap