Computational Complexity

Lecture 1

February 5, 2019

Universiteit van Amsterdam

Practical Information

- Ronald de Haan (`R.deHaan@uva.nl`)
- Jan Czajkowski (`J.Czajkowski@uva.nl`)
- Course web page:
  `https://staff.science.uva.nl/r.dehaan/complexity2019/`
- Book: *Computational Complexity: A Modern Approach*,
  by Sanjeev Arora & Boaz Barak, 2009.
  (`http://www.cs.princeton.edu/theory/complexity/`)

Practical Information

- Lectures:
  - Tuesday 15:00-17:00 (SP G0.18B)
  - Thursday 11:00-13:00 (SP G0.18B)
  - (There are a few exceptions! See the web page.)
- Lectures will be mostly on the white board
  - For everything we do, there is accompanying explanation/proofs in the book (or in another text)
- Exercise session:
  - Friday 13:00-15:00 (SP A1.04)

Practical Information

- Weekly homeworks
  - To be handed in Wednesday at 17:00
  - Preferably by email to Jan Czajkowski
  - Total of 6 homework sets
  - When computing the average grade for the exercises, the lowest score will be dropped
- Final exam
  - Open book: you can bring the book and any notes
  - No electronic equipment allowed
- Final grade: 50% homeworks, 50% final exam

What is Computational Complexity?

- The study of what you can compute with limited resources
  - Resources, e.g.: time, memory space, random bits
- Computability theory tells us what can be computed in principle

  Computational complexity theory tells us what can be computed realistically
- Central notions:
  - The resource use of an algorithm, in the worst case
  - Computational problems, and algorithms solving these problems
  - How does the resource use of algorithms scale?
  - ...

What is Computational Complexity?

- Distinguish different degrees of computational difficulty:

  different complexity classes

  (a whole zoo of complexity classes:
  `https://complexityzoo.uwaterloo.ca/Complexity_Zoo`)

- Central question: the P versus NP problem
  (one of the $1 Million *Millennium Prize Problems*)

Relation to Other Fields

- Computation plays a role nearly everywhere..
- Therefore, computational complexity is relevant for many areas; for example:
  - Computer science
  - Cryptography
  - Economics, game theory
  - Artificial intelligence
  - Biology
  - Scheduling, vehicle routing
  - ...

Some courses that are related

- Recursion Theory
- Kolmogorov Complexity
- Knowledge Representation
- Quantum Computing

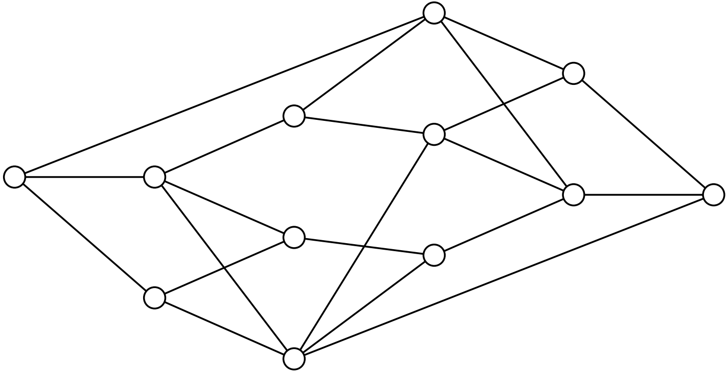
Schedule of the Course

1. Turing machines, foundations
2. P, NP, coNP, reductions
3. NP-completeness, Cook-Levin Theorem
4. Diagonalization, time hierarchy theorems, Ladner's Theorem
5. Relativization, Baker-Gill-Solovay Theorem
6. Space complexity, PSPACE, L, NL
7. Polynomial Hierarchy
8. Circuit complexity, Karp-Lipton Theorem
9. Probabilistic algorithms, BPP
10. Approximation, PCP Theorem
11. Subexponential-time complexity, Exponential Time Hypothesis (ETH)
12. *(t.b.a.)*
13. *(t.b.a.)*

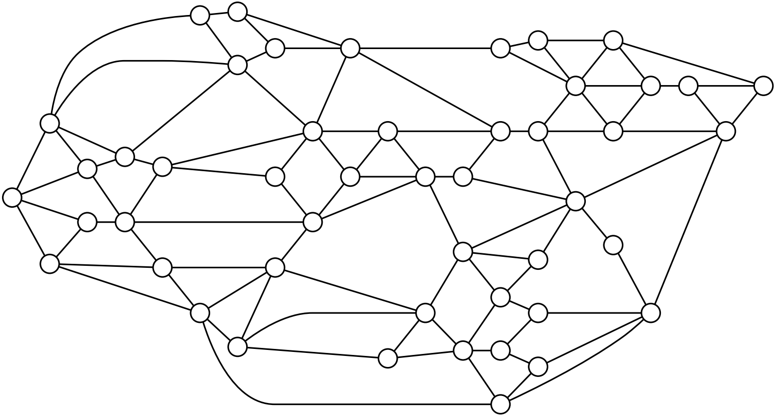
Illustration: Graph k-Coloring

- You are given an undirect graph
  - Nodes / vertices, with edges between them.
- The task is to color each node with a color in $\{1, 2, \ldots, k\}$ so that no two connected nodes have the same color
- This can be used to model many applications
  - For example, nodes are regions with their own radio station, colors are radio frequencies, and two nodes are connected if the regions border each other
  - The task is to assign radio frequencies without conflict (in the border areas)

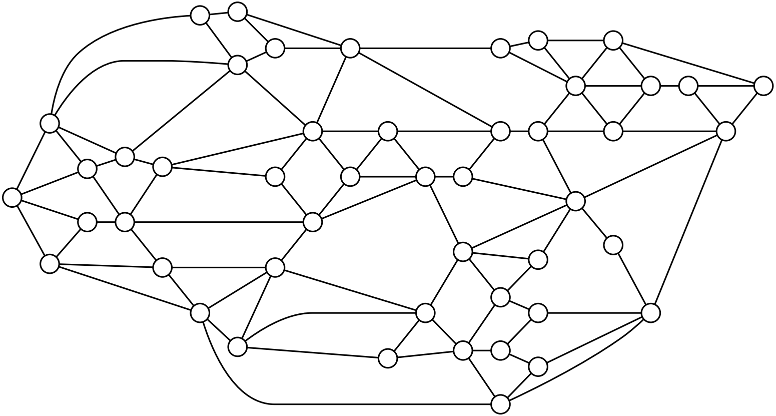
Color this graph with 2 colors!



Color this graph with 2 colors!



Now, color this graph with 3 colors!



(We'll continue on the white board..)

Quadratic vs. Exponential

- Important difference between algorithms that run in time, say, n^2 vs. algorithms that run in time, say, 2^n

- Illustration (time needed for 10^{10} steps per second):

n	n^2 steps	2^n steps
2	0.00000002 msec	0.00000002 msec
5	0.00000015 msec	0.00000019 msec
10	0.00001 msec	0.0001 msec
20	0.00004 msec	0.10 msec
50	0.00025 msec	31.3 hours
100	0.001 msec	9.4×10^{11} years
1000	0.100 msec	7.9×10^{282} years

- # of atoms in universe $\approx 10^{80}$