CS580: Algorithm Design and Analysis

Simina Branzei

Part I – Logistics

Instructor: Simina Branzei

Course website: Piazza.com (CS580) Sign up TODAY!

Send all your comments/emails via this site. We will use Gradescope for homework submissions and grading.

Teaching assistants: Shoaib Amjad Khan, Anuj Singh, Lu Yan, Zheng Zhong

Lectures: 3:00-4:15PM Tue/Thu, Location BRNG 2280.

A few lectures (about 4) will be virtual. Zoom link will be provided.

Part I – Logistics (Study Groups)

Peer learning: Everyone must have a **study group** throughout the course. The purpose of the group is to help with studying, solving homework problems, asking questions.

Any questions should be discussed first with your study group before bringing them to the course staff.

Submit your group members (each group of size 2-3) by next Tuesday at this link:
https://docs.google.com/document/d/1WEQkOqq7R07Oj4pdzxtm073825bp2Iw
tjipbX62RxJ8/edit?usp=sharing

Part I – Logistics (Books)

Required textbook: “Algorithm Design”, by Kleinberg and Tardos.

Recommended:

- “Introduction to Algorithms”, by Cormen, Leisserson, Rivest, Stein.
- Other materials will be used as needed, e.g. “The Art of Computer Programming”, by Knuth.

Part I – Logistics (Grading)

Homework policy:

- Problem sets due every one to three weeks on Gradescope.
- You can collaborate to discuss proof strategies.
- You cannot copy solutions. Solutions must be written individually and typed in Latex.
- Late homework: you can submit one homework (3 days)

SEE SYLLABUS FOR DETAILS ON SUBMISSION AND COLLABORATION

Grading:

- 20% for homework
- 2 x 17.5% for 2 midterms
- 40% for the final (no makeup exams)
- 5% for class participation (working with your study group; also giving good answers on Piazza).

Part I – Logistics (Exams)

Exams are meant to incentivize students to learn the material well and measure the learning.

Part I – Logistics (Exams)

Exams are meant to incentivize students to learn the material well and measure the learning.

Thus in this class:

- **No** cheatsheets, phones, computers, books, notebooks, or other materials are allowed on exams. **Bring only pen, paper, eraser, calculator to any exam** (midterm or final).

Part I – Logistics (Exams)

Exams are meant to incentivize students to learn the material well and measure the learning.

Thus in this class:

- **No** cheatsheets, phones, computers, books, notebooks, or other materials are allowed on exams. **Bring only pen, paper, eraser, calculator to any exam** (midterm or final).
- There will be questions of the form:

"Write a one page/half a page proof plan for the next statement/theorem [...]".

Part I – Logistics (Exams)

Exams are meant to incentivize students to learn the material well and measure the learning.

Thus in this class:

- **No** cheatsheets, phones, computers, books, notebooks, or other materials are allowed on exams. **Bring only pen, paper, eraser, calculator to any exam** (midterm or final).
- There will be questions of the form:

"Write a one page/half a page proof plan for the next statement/theorem [...]".

A **proof plan** should contain the main logical steps that allow a reader to get the main structure of the proof, while possibly omitting some calculations/details that are easier to reconstruct (or may even obstruct the main idea). Then the exam will ask you to prove one of these steps in detail.

I recommend starting to practice this now as we learn new material.

Piazza Rules of Conduct

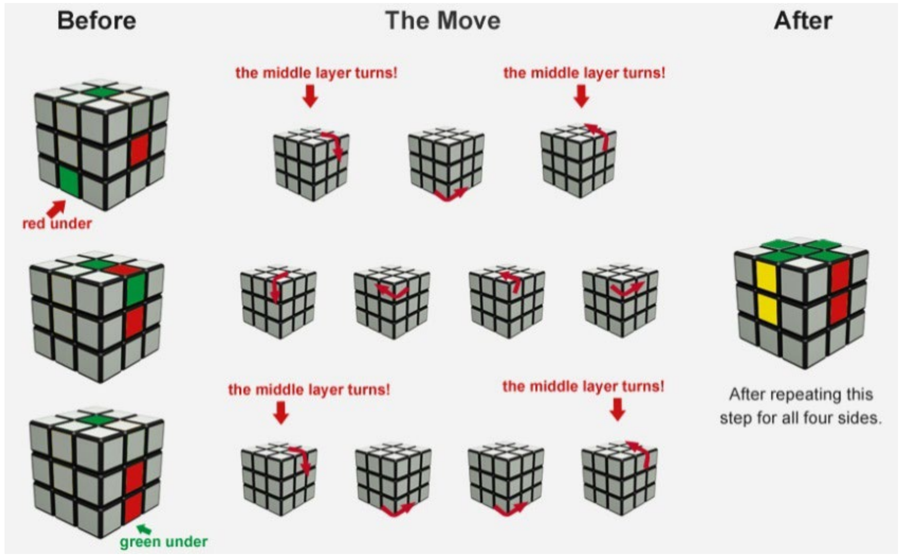
- Public posts on Piazza are intended for clarification of questions of <u>general</u> interest.
- Piazza cannot be used to post answers to assignments, detailed descriptions of solutions, or hints.
- Piazza is not the forum for complaints about an assignment, exam, or the class.
  - Any concerns should be brought to the attention of the instructor.
- Be courteous and professional when posting and use appropriate language.
- If you are not sure whether a post is appropriate, make sure it is made <u>private</u>.

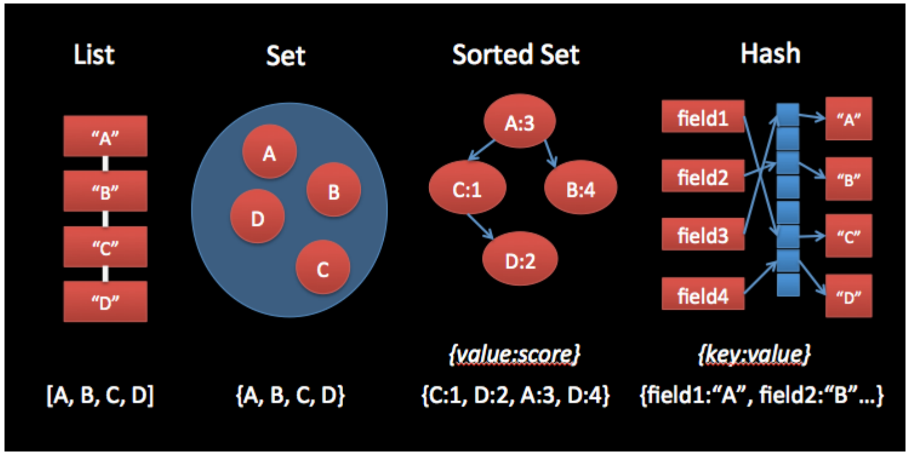
What is this course about?

- Algorithms

And their analysis

- Data structures





What aspects are important?

Computational aspects:

- Efficiency (time and space complexity)
- Feasibility

  Is the problem decidable? is it NP-complete?

Design aspects:

- Robustness
- Scalability
- Modularity
- Simplicity

Etymology of word Algorithm

Muḥammad ibn Mūsā al-Khwārizmī

- Persian mathematician, astronomer, geographer
- Lived in Baghdad (c. 780 AD – c. 850 AD)
- Introduced decimal system to Western world
- Father of algebra - presented the first systematic solution of linear and quadratic equations

Source: Wikipedia

Prerequisites

- **Being able to write mathematical proofs is essential (e.g. induction, contradiction, deduction),** as we will be proving the correctness of algorithms.
- Asymptotic notation (O(n), o(n), etc.)
- Basic data structures: linked lists, stacks, arrays, trees
- Basic algorithmic paradigms: divide and conquer, greedy algorithms, dynamic programming
- Basic algorithms: depth- and breadth- first search, basic sorting algorithms
- Notions of computability and complexity: Turing machines, P, NP

In-class Quiz

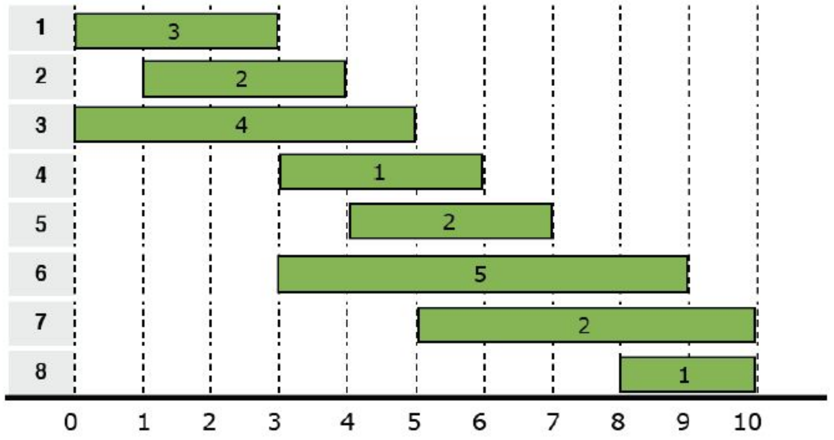
1. How many edges can there be in a tree with n vertices? Justify your answer.
2. Prove that the sum of degrees in an undirected graph is even.
3. What is the worst case time complexity of an insert operation in a stack? What about a delete operation in a linked list?
4. Prove by induction the following statement: $\frac{n^2}{2} + 3n - 5$ is $\Theta(n^2)$.
5. True/False/I don't know questions:
   a. An algorithm with runtime Theta(n!) is polynomial time.
   b. Finding an element in a linked list is O(n * log(n)).
   c. The function sin(1/n) * n is o(n).
   d. An independent set of size $\sqrt{n}$ in a graph can be found in $O(n^4)$ if it exists.
6. Show that any comparison-based sorting algorithm takes $\Omega(n \cdot \log(n))$ time in the worst case. A proof sketch will suffice.

Goals of the course

- Learn how to formulate real-world questions using mathematical abstractions
- Learn about algorithms for classical problems
- Learn how to analyze algorithms formally
- Develop/practice problem-solving, technical-writing, and analytical-thinking skills

Topics

- Scheduling problems: e.g. interval scheduling (tasks to be executed, each within an interval of time → select largest subset of compatible tasks)



- Minimum spanning trees

  applications in network design, approximation algorithms, reducing data storage, cluster analysis

- Data compression: how can we encode data succinctly (in bits)?

Topics (contd)

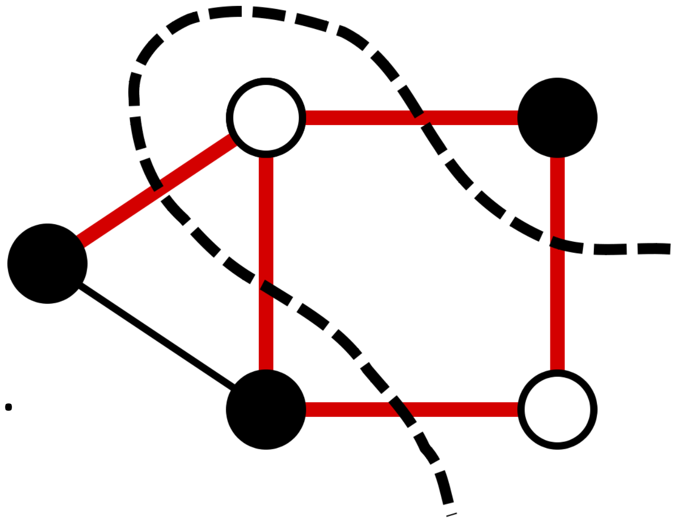
- Max flow, min cut

  applications to data mining, image segmentation, security of statistical data

- Approximation algorithms

- Randomized algorithms. Analysis tools.

- Sublinear algorithms



Techniques

- Greedy algorithms
- Divide and conquer
- Dynamic programming
- Probabilistic tools

Part II
A general perspective on algorithms and their complexity

Slides inspired from Sanjeev Arora,
http://www.cs.princeton.edu/courses/archive/spring06/cos116/

Efficiency aspects

- We'll focus on **computable problems**
- Restricted resources:
  1. time

  2. memory

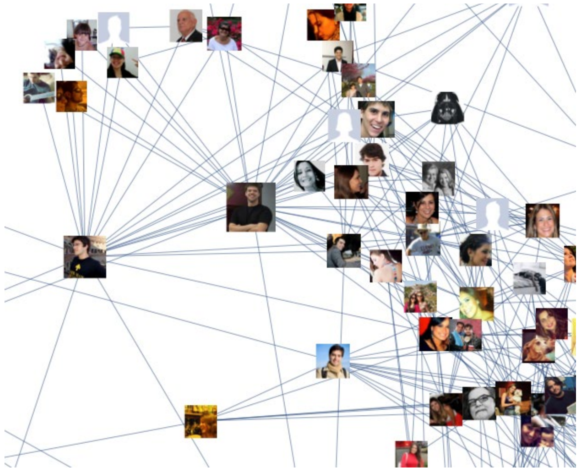
Question 1: What problems are **easy**?

Question 2: What problems are **hard**?

Rumor spreading

Pam starts a rumor on Facebook.

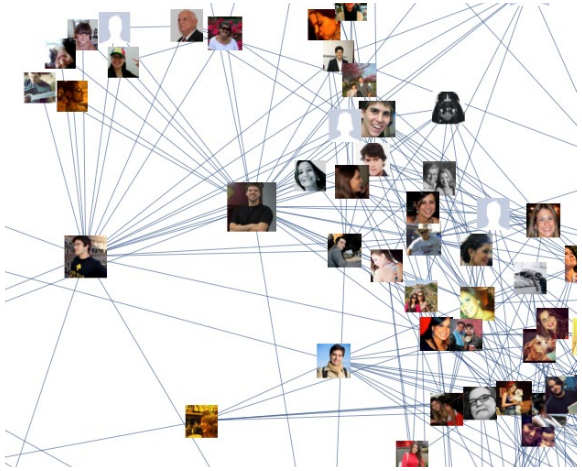
Question: Will it reach Terry?



Rumor spreading

Pam starts a rumor on Facebook.

Question: Will it reach Terry?

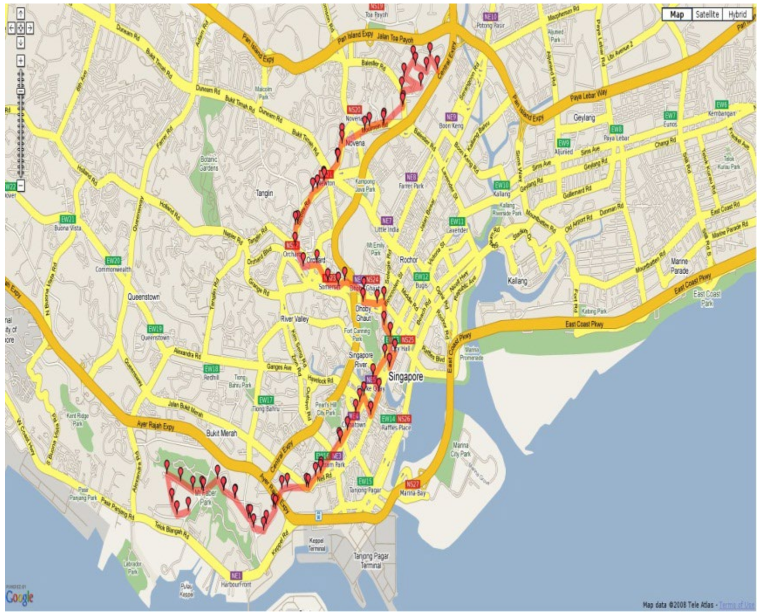


Answer: This is easy!

If 10 people are represented, can explore the graph by checking 100 edges.

More generally, **n** nodes (people) in **n^2** steps (edge checks)

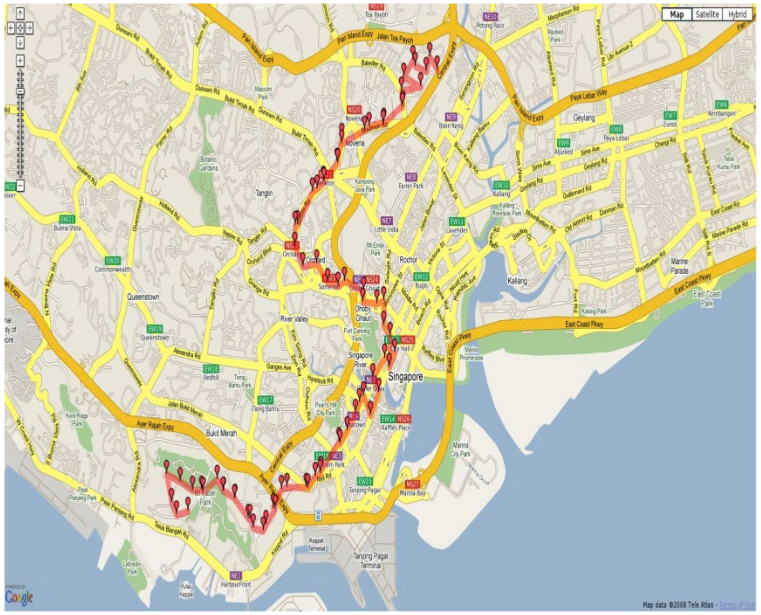
Shortest path



Input: 1000 cities, and pairwise distances
Want to travel from Purdue
to Boston fast.

Problem: Compute shortest route.

Shortest path

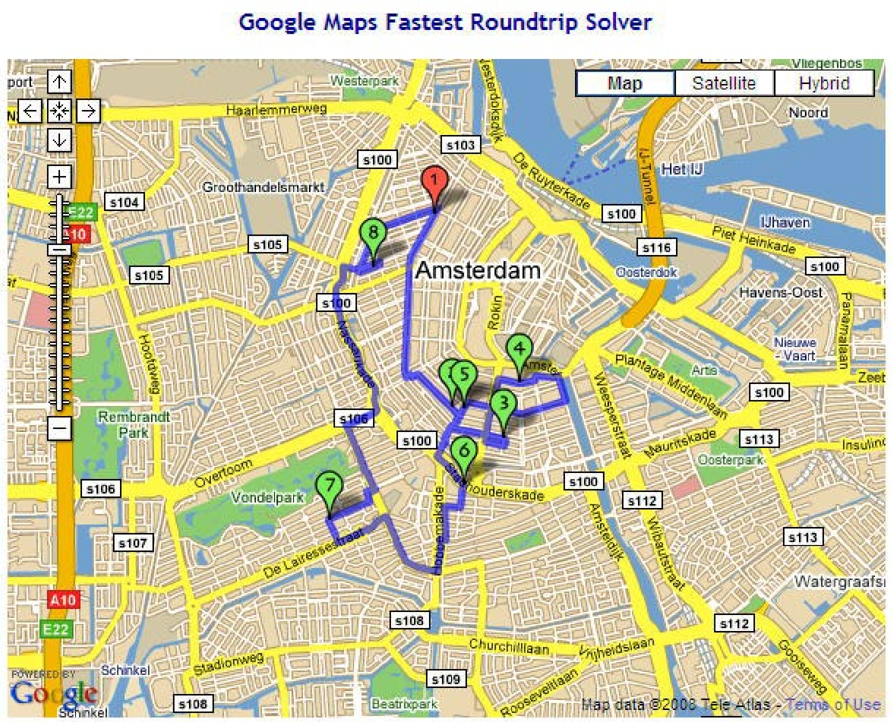


Input: 1000 cities, and pairwise distances
Want to travel from Purdue
to Boston fast.

Problem: Compute shortest route.

Easy: Dijkstra's algorithm takes n^2 steps
If total number of cities is n

Traveling salesman's problem



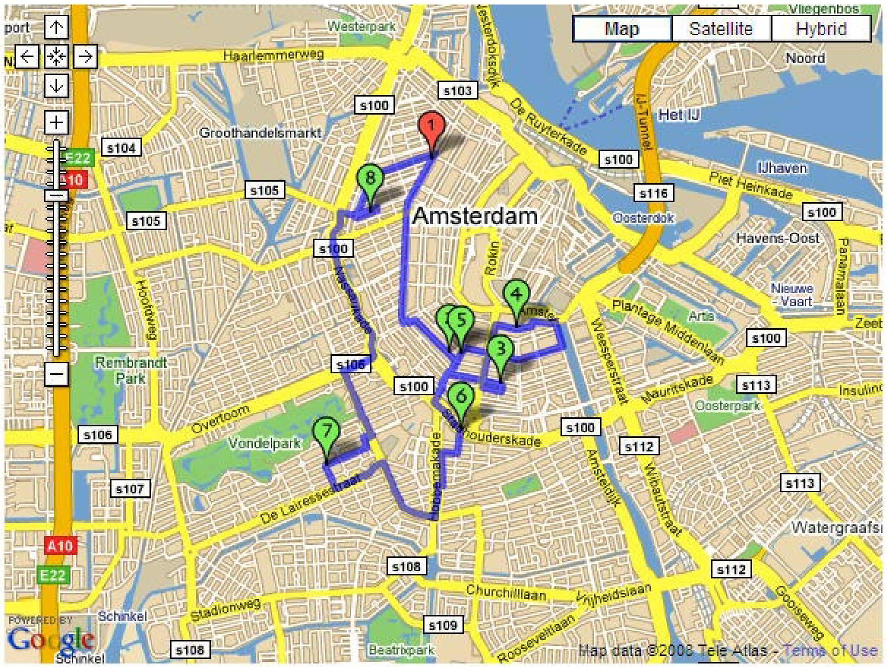
Input: **n** cities,
pairwise distances,
a number **k**

Output: Yes/No: Decide if exists a tour that visits all cities once and has length **< k**.

Trivial solution: check all tours

Traveling salesman's problem

Google Maps Fastest Roundtrip Solver



TSP is hard!

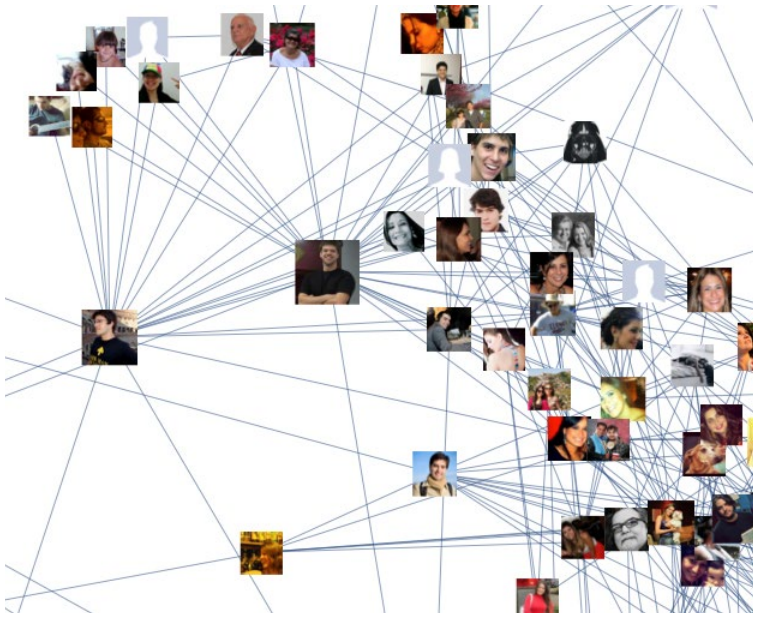
Input: **n** cities,
pairwise distances,
a number **k**

Output: Yes/No: Decide if exists a tour that visits all cities once and has length **< k**.

Trivial solution: check all tours
Takes time (# steps) > 2^n

If n = 1000, the sun would have died by the time the algorithm terminates.

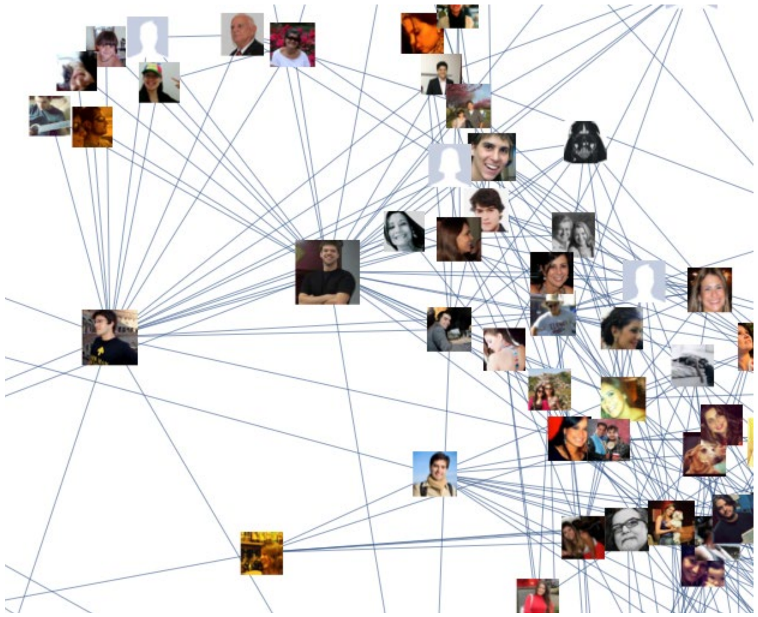
Clique



k-Clique= set of **k** people where every pair gets along with each other

Problem: Given **n** people, and number **k** is there a k-clique?

Clique

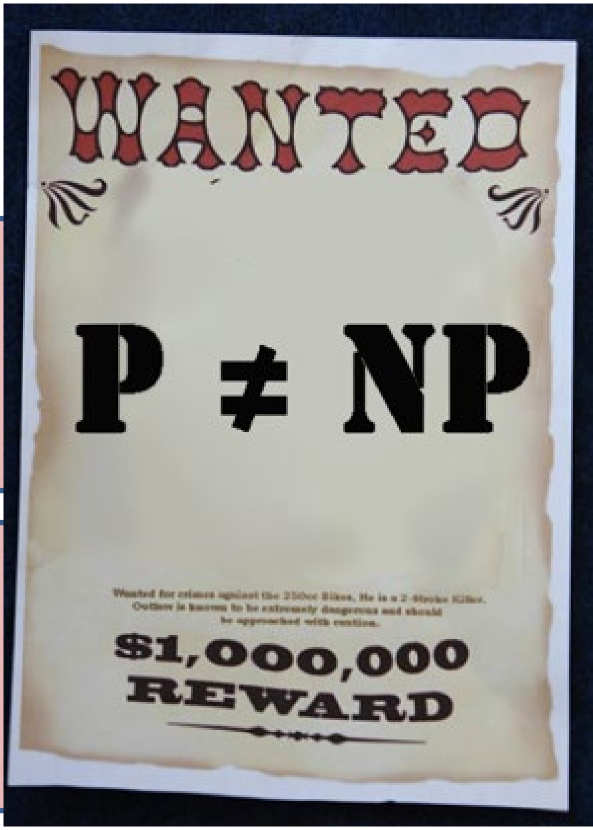


k-Clique= set of **k** people where every pair gets along with each other

Problem: Given **n** people, and number **k** is there a k-clique?

Answer: Best algorithms run in exponential time!

P vs. NP question



- P = set of problems that are easy to solve
(fast algorithms: rumor spreading, shortest path)

- NP = set of problems for which a good solution
(no fast algorithm is known: Clique, TSP)

- Million dollar question: P = NP?

(Is solving a problem just as easy as verifying the solution?)

Why is P vs NP a million dollar question?

- If P=NP everything is easy, all our internet transactions are insecure!
- If P ≠ NP then we learn something fundamental about computers and the world

(akin to: nothing travels faster than light)

How do we deal with NP complete problems?

- Heuristics

(algorithms can produce reasonable approximate solutions in practice)

- Approximation algorithms/optimization

(provable guarantees about the goodness of the approximation)

Cryptography relies on hard problems

Security of encryption (e-mails, signatures), e-commerce rely on the following problem:

Integer Factoring Problem: Given integer N=p*q, find integers p, q.

Believed to be Hard.

If quantum computers can be built then it is easy!

(polynomial time quantum algorithm by Peter Shor)

Randomness

Partisan dominance in state legislatures heading into the 2010 state legislative elections
Both legislative chambers dominated by Democratic Party
Both legislative chambers dominated by Republican Party
Split: one chamber dominated by the Democratic Party, one dominated by the Republican Party
Non-partisan legislature

Often, ability to toss coins helps in computation.

Polling: Pick random 2000 people and take the majority answer.

This guarantees to give majority vote of population of **n** with probability 99%. (regardless of how large **n** is)

Randomness in computation

- If we can fail with some very small probability can use randomness to get fast algorithms for hard problems.

- Randomness is crucial for cryptography

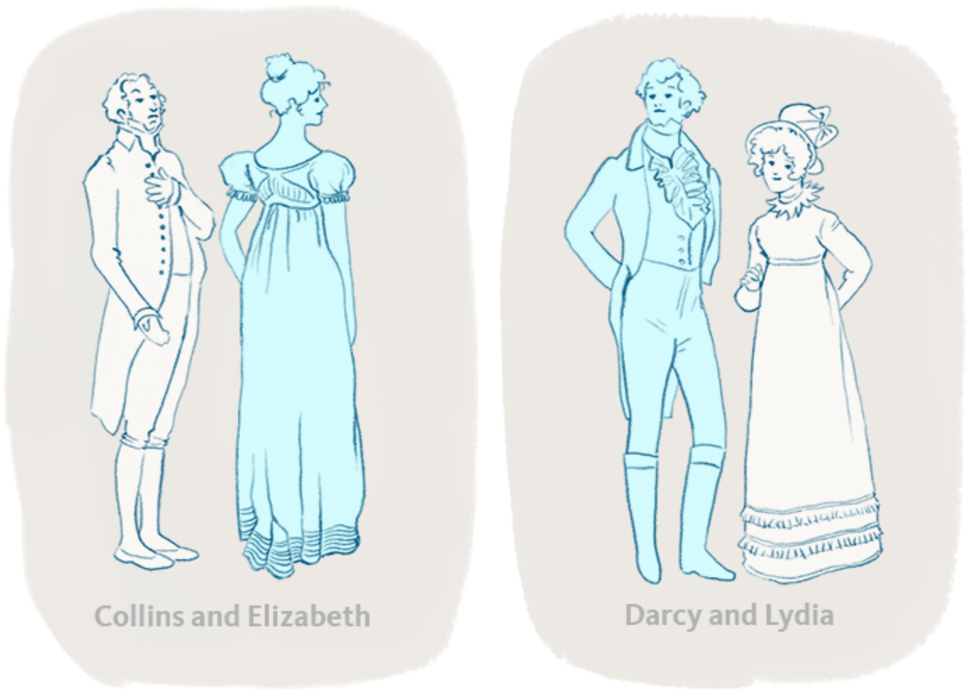
- Useful in the study of massive data sets (genome project, population records)

Part III – A first problem

The Stable Matching Problem:

algorithm and analysis.

(see textbook slides by Kevin Wayne)



An unstable match:
ELIZABETH AND DARCY LIKE EACH OTHER BETTER THAN THEIR PARTNERS

Matching Residents to Hospitals

Goal. Given a set of preferences among hospitals and medical school students, design a self-reinforcing admissions process.

Unstable pair: applicant x and hospital y are unstable if:

- x prefers y to its assigned hospital.
- y prefers x to one of its admitted students.

Stable assignment. Assignment with no unstable pairs.

- Natural and desirable condition.
- Individual self-interest will prevent any applicant/hospital deal from being made.

Stable Matching Problem

Goal. Given n men and n women, find a "suitable" matching.

- Participants rate members of opposite sex.
- Each man lists women in order of preference from best to worst.
- Each woman lists men in order of preference from best to worst.

favorite ↓ … least favorite ↓

	1st	2nd	3rd
Xavier	Amy	Bertha	Clare
Yancey	Bertha	Amy	Clare
Zeus	Amy	Bertha	Clare

Men's Preference Profile

favorite ↓ … least favorite ↓

	1st	2nd	3rd
Amy	Yancey	Xavier	Zeus
Bertha	Xavier	Yancey	Zeus
Clare	Xavier	Yancey	Zeus

Women's Preference Profile

Stable Matching Problem

Perfect matching: everyone is matched 1-1.

- Each man gets exactly one woman.
- Each woman gets exactly one man.

Stability: no incentive for some pair of participants to undermine assignment by joint action.

- In matching M, an unmatched pair m-w is unstable if man m and woman w prefer each other to current partners.
- Unstable pair m-w could each improve by eloping.

Stable matching: perfect matching with no unstable pairs.

Stable matching problem. Given the preference lists of n men and n women, find a stable matching if one exists.