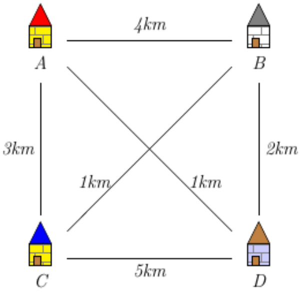
Introduction — Problems!

Dr Kamal Bentahar

School of Computing, Electronics and Mathematics
Coventry University

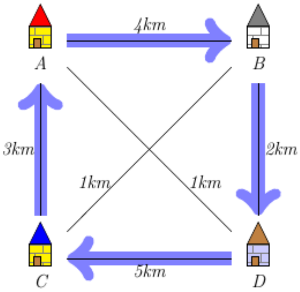
Lecture 1

Travelling Salesman Problem



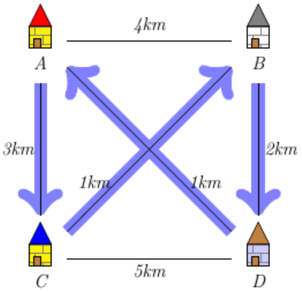
Shortest tour?

Travelling Salesman Problem



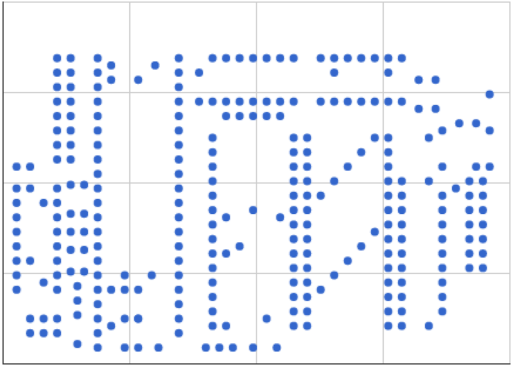
$$4 + 2 + 5 + 3 = 14$$

Travelling Salesman Problem

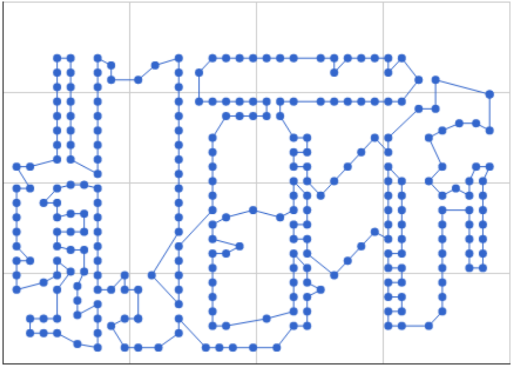


$$3 + 1 + 2 + 1 = 7$$

Travelling Salesman Problem



Travelling Salesman Problem



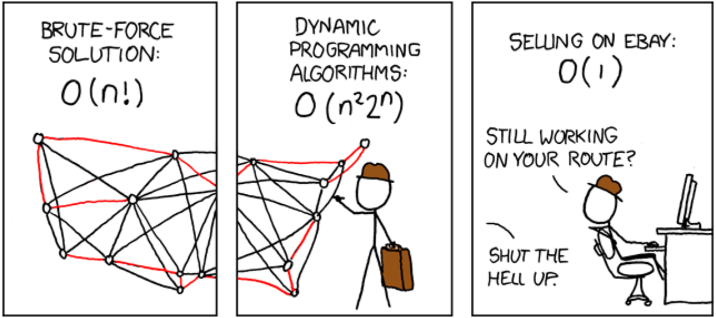
Travelling Salesman Problem

- One of the most famous problems in CS.
- Given a **list of cities** and the **distances between each pair of cities**, what is the shortest possible route that visits each city and returns to the origin city?
- **NP-hard** problem!

Travelling Salesman Problem – what is the issue?

Number of cities n	Number of paths $(n-1)!/2$
3	1
4	3
5	12
6	60
7	360
8	$2,520$
9	$20,160$
10	$181,440$
15	$43,589,145,600$
20	6.082×10^{16}
71	5.989×10^{99}

Travelling Salesman Problem – what is the issue?



Some more examples

Problem (Cliques)

Given a graph and an integer n, decide if it contains a clique with k vertices.

A clique in a graph is a set of vertices for which any two are connected.

Problem (Subset-Sum Problem)

Given a set $S = \{x_1, x_2, \ldots, x_n\}$ of integers, and an integer t (called target) decide if there is a subset of S whose sum is equal to t.

Problem (A Diophantine quadratic equation)

Given three positive integers a, b, c, decide if the equation $ax^2 + by = c$ has a solution in positive integers.

Problem (Satisfiability)

Given a Boolean expression, decide if there is a way of assigning the values true and false to the variables so that the expression is true.

A needle in a haystack — a search problem

Problem:

Given any (finite) haystack H, decide whether H contains a needle.

Exhaustive Search

Search every location within the haystack, in some order, and terminate answering **yes** if a needle is found.
If the search is *completed* with no needle found then terminate answering **no**.

This problem is a **decision problem**: given some data (the haystack) decide if the data has a certain property (needle containment).

We may divide all possible instances of the problem into **yes-instances** and **no-instances** using our process.

Types of problems

- Decision
- Search
- Computation/Construction
- Counting
- Optimization
- ...

Important observation

As far as "Can these problems be solved at all using computation?", they can be reduced to **decision problems**.

Problems vs Problem Instances

1. What is $1 + 1$?
   $\rightarrow$ instance of the problem called *addition*,
2. What is the shortest route across the rail network from Coventry to London?
   $\rightarrow$ instance of the *shortest path problem*,
3. What is the shortest tour around all the universities in the UK and back to your starting point (by car say)?
   $\rightarrow$ instance of the *travelling salesman problem*.

Problems vs Problem Instances

Problems: Generalization of a problem instance.

- e.g. not interested in just $1 + 1$, but $x + y$ in general.

For a specific *problem instance*, we could measure exactly the amount of processor time and memory capacity required to solve it, using some suitable process.
However, when solving a general problem, we cannot always say exactly what resources will be used.

- We express resource usage as a function of the **instance's size.**
- When we ask questions about whether a problem is solvable by some machine, we allow the machine unlimited memory and time — all problems become unsolvable at some point if these are finite.
- It is for this reason, among others, that theoretical machines are used in classifying hardness.

Review: O-notation scale

In increasing order:

- Constant: $O(1)$
- Polynomial: $O(n^k)$ for $k \geq 1$. $[O(n), O(n^2), \ldots]$
- Exponential $O(c^n)$ for $c > 1$. $[O(2^n), O(3^n), \ldots]$
- Factorial: $O(n!)$
- Combinatorial: $O(n^n)$

"Tricks":

$$n^k \log n \sim n^{k+\varepsilon} \qquad (\varepsilon \text{ small})$$

$$n^n > n! \quad \text{because } n^n = \underbrace{n \times n \times n \times \cdots \times n \times n \times n}$$

$$> n(n-1)(n-2)\cdots 3 \times 2 \times 1 = n!$$

Review: Big-O notation cheat-sheet

1. 1 — constant, does not depend on n
2. $\log n$ — think of this as n^ε for a "small" ε
3. n
4. $n \log n$ — think $n \times n^\varepsilon = n^{1+\varepsilon}$
5. n^2
6. $n^2 \log n$ — think $n^{2+\varepsilon}$
7. n^3
8. 2^n
9. 3^n
10. $n!$
11. n^n

Review: Big-O notation

Intuitively, $f(n) = O(g(n))$ means that $f(n)$ is bounded above by $g(n)$ (up to constant factor) for sufficiently large n.

Express the following using O-notation (find the *fastest growing* term)

Examples:

$$
\begin{aligned}
2018 &= O(1) \\
n + 5 &= O(n) \\
543n + n^3 + 13 &= O(n^3) \\
4n^2 + 2784n + 10^{74} &= O(n^2) \\
n^{578} + 4685 + 2^n &= O(2^n) \\
n + n\log n + 35 &= O(n \log n) \\
n^2 + n\log n + 35 &= O(n^2) \\
n^2 + n^2 \log n + 35 &= O(n^2 \log n) \\
n^{86754} + n! &= O(n!)
\end{aligned}
$$

Notation: Greek alphabet

Σ	Sigma	Set of alphabet symbols
Γ	Gamma	Set of stack/tape symbols
α	alpha	
β	beta	
γ	gamma	
δ	delta	Transition function
ε	epsilon	Empty string
σ	sigma	

Numeric

$=$	equals
$\neq$	not equal
$<$	less than
$\leq$	less than or equal
$>$	greater than
$\geq$	greater than or equal
$n!$	Factorial of n: $n \times (n-1) \times (n-2) \times \cdots \times 2 \times 1$

Notation: Logic

Expression	**Meaning**
$a \wedge b$	a and b
$a \vee b$	a or b
$a \oplus b$	a xor b
$\neg a$ (or $\bar{a}$)	not a
$a \implies b$	a implies b, or: if a then b
$a \iff b$	a and b are equivalent, or: "a if and only if b"

Sets

$\{x_1, \dots, x_n\}$	Finite set consisting of the elements x_1 until x_n
$\emptyset$	Empty set, i.e. $\{\}$
$x \in S$	"in", member of a set
$x \notin S$	"not in", not a member of a set
$A \cup B$	Union of two sets
$A \cap B$	Intersection of two sets
$A - B$	Difference of two sets
$A \times B$	Cartesian product of two sets
$A \subset B$	Subset of ...
$\lvert A \rvert$ or $\#A$	Cardinality of the set A, i.e. count of its elements
2^A	Power set of A, i.e. set of all subsets of A
$\mathbb{N}$	Natural numbers: $\{1, 2, 3, \dots\}$
$\mathbb{Z}$	Integers: $\{0, 1, -1, 2, -2, 3, -3, \dots\}$
$\mathbb{Z}_{\geq 0}$	Non-negative integers: $\{0, 1, 2, 3, \dots\}$
S'	A set called "S prime" (a way of making new names)
S'' or S'''	A set called "S double prime" / "S triple prime"

Sets

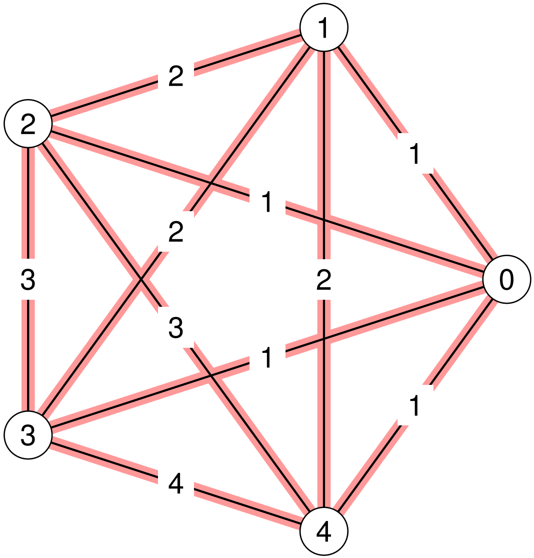
$\{pattern \mid condition\}$ Set of items matching *pattern* and satisfying *condition*.
The $\mid$ symbol is read "such that"

$$
\begin{array}{ll}
A \cup B & \{x \mid x \in A \vee x \in B\} \\
A \cap B & \{x \mid x \in A \wedge x \in B\} \\
A - B & \{x \mid x \in A \wedge x \notin B\} \\
A \times B & \{(a, b) \mid a \in A \wedge b \in B\}
\end{array}
$$

Notation: Strings

w	String made of symbols from Σ
w^R	String obtained by writing w in the reverse order
$\lvert w \rvert$	Length of the string x
xy	String made by concatenating x and y
w^n	String made by concatenating n copies of w: $\underbrace{ww\ldots w}_{n\text{ copies}}$
	In particular: $w^0 = \varepsilon$, $w^1 = w$ and $w^2 = ww$
$\{0,1\}^n$	Binary strings of length exactly n symbols
$\{0,1\}^*$	Binary strings of any length: $\{\varepsilon, 0, 1, 00, 01, 10, 11, 000, \ldots\}$

Notation: Graphs



Graphs

$G = (V, E)$, where

- V: set of **vertices.**
- E: set of edges.

Graph can be:

- **directed** or **undirected**.
- **weighted** or **unweighted**.
- **labelled** or **unlabelled**.
- etc.

Properties:

- Is the graph **connected**?
- Does it contain **cycles**?
- etc.

Algorithms:

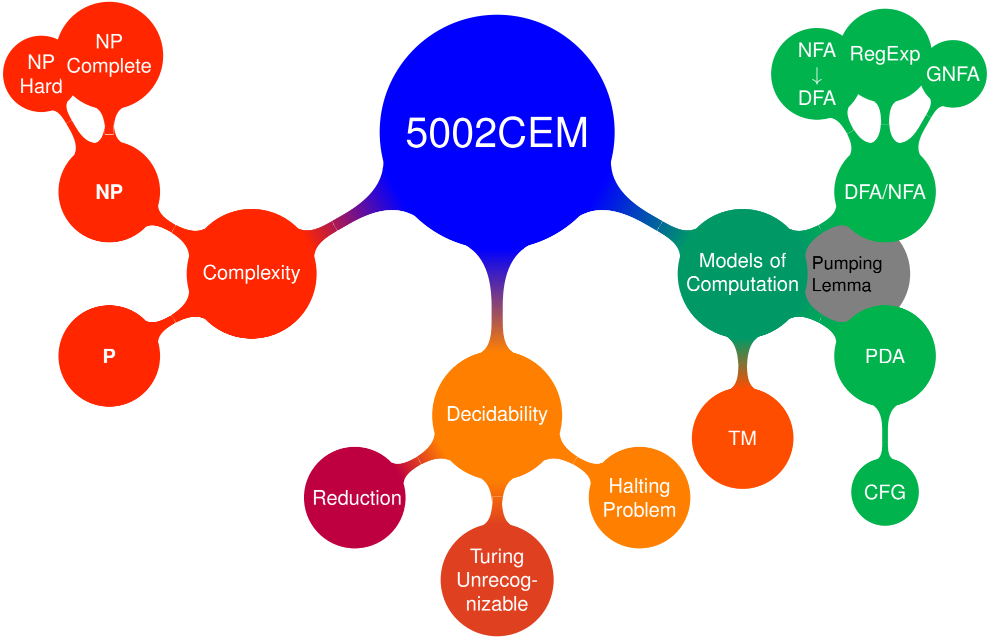
- Traversal, e.g. BFS, DFS.
- Shortest path.
- etc.

Next few weeks. . .

What is a “computer”? What is “computation”?

Questions about this first arose in the context of pure Mathematics:

- Gottlob Frege (1848–1925)
- David Hilbert (1862–1943)
- George Cantor (1845–1918)
- Kurt Gödel (1906–1978)
- 1936:
  - Gödel and Stephen Kleene (1909-1994): **Partial Recursive Functions**
  - Gödel, Kleene and Jacques Herbrand (1908–1931)
  - Alonzo Church (1903–1995): **Lambda Calculus**
  - Alan Turing (1912–1954): **Turing Machine**
- 1943: Emil Post (1897–1954): **Post Systems**
- 1954: A.A. Markov: Theory of Algorithms – **Grammars**
- 1963: Shepherdson and Sturgis: **Universal Register Machines**


5002CEM
Complexity
NP
NP Complete
NP Hard
P
Decidability
Reduction
Turing Unrecog-nizable
Halting Problem
Models of Computation
Pumping Lemma
DFA/NFA
NFA ↓ DFA
RegExp
GNFA
PDA
CFG
TM