Gems of TCS

Easy and Hard Problems

Sasha Golovnev

January 26, 2020

Theoretical Computer Science

Theoretical Computer Science

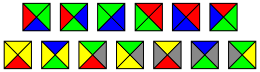
$P \implies Q$

Mathematical logic

Theoretical Computer Science

$$P \implies Q$$

Mathematical logic

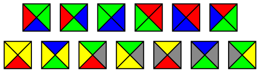


Computability theory

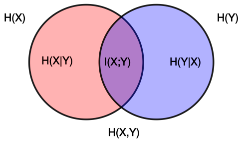
Theoretical Computer Science

$$P \implies Q$$

Mathematical logic



Computability theory

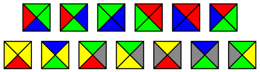


Information theory

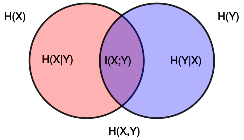
Theoretical Computer Science

$$P \implies Q$$

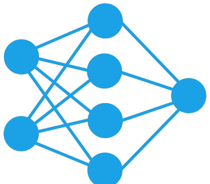
Mathematical
logic



Computability
theory



Information
theory

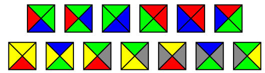


Learning,
neural nets

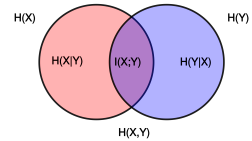
Theoretical Computer Science

$P \implies Q$

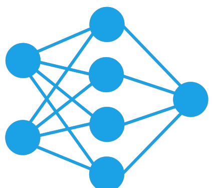
Mathematical logic



Computability theory



Information theory



Learning, neural nets

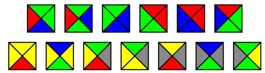
$P = NP?$

Computational complexity

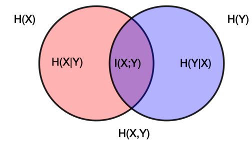
Theoretical Computer Science

$$P \implies Q$$

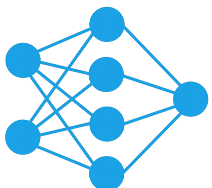
Mathematical logic



Computability theory



Information theory



Learning, neural nets

$$P = NP?$$

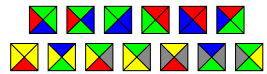
Computational complexity

Cryptography

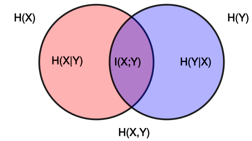
Theoretical Computer Science

$P \implies Q$

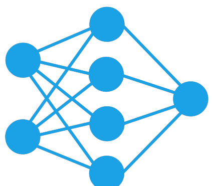
Mathematical logic



Computability theory



Information theory

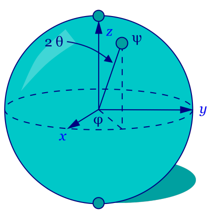


Learning, neural nets

$P = NP?$

Computational complexity

Cryptography

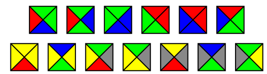


Quantum Algorithms

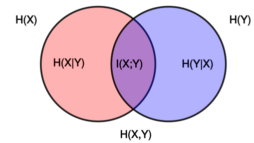
Theoretical Computer Science

$P \implies Q$

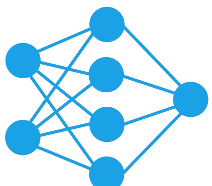
Mathematical logic



Computability theory



Information theory

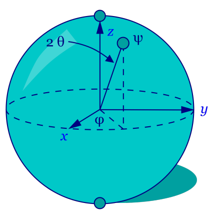


Learning, neural nets

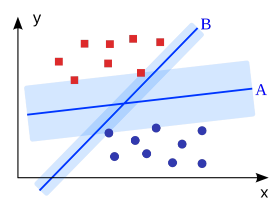
$P = NP?$

Computational complexity

Cryptography



Quantum Algorithms

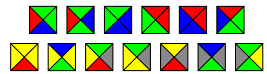


Machine learning

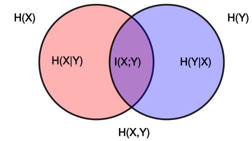
Theoretical Computer Science

$$P \implies Q$$

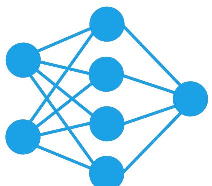
Mathematical logic



Computability theory



Information theory

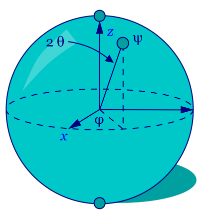


Learning, neural nets

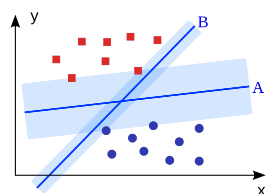
$$P = NP?$$

Computational complexity

Cryptography



Quantum Algorithms



Machine learning

Data Science

This Course

- Theoretical/Mathematical viewpoint

This Course

- Theoretical/Mathematical viewpoint
- Topic overview

This Course

- Theoretical/Mathematical viewpoint
- Topic overview
    - Algorithms

This Course

- Theoretical/Mathematical viewpoint
- Topic overview
    - Algorithms
    - Computational Complexity

This Course

- Theoretical/Mathematical viewpoint
- Topic overview
  - Algorithms
  - Computational Complexity
  - Cryptography

This Course

- Theoretical/Mathematical viewpoint
- Topic overview
    - Algorithms
    - Computational Complexity
    - Cryptography
    - Learning

Administrative Info

- Classes: TR 9:30am–10:45am, Zoom

Administrative Info

- Classes: TR 9:30am–10:45am, Zoom
- Office Hours: T 10:45am–11:45am, Zoom

Administrative Info

- Classes: TR 9:30am–10:45am, Zoom
- Office Hours: T 10:45am–11:45am, Zoom
- Prerequisites: Algorithms or Theory of Computation

Administrative Info

- Classes: TR 9:30am–10:45am, Zoom
- Office Hours: T 10:45am–11:45am, Zoom
- Prerequisites: Algorithms or Theory of Computation
- Webpage: https://golovnev.org/gems

Administrative Info

- Classes: TR 9:30am–10:45am, Zoom
- Office Hours: T 10:45am–11:45am, Zoom
- Prerequisites: Algorithms or Theory of Computation
- Webpage: https://golovnev.org/gems
- Slides and Videos in Canvas and Piazza

Administrative Info

- Classes: TR 9:30am–10:45am, Zoom
- Office Hours: T 10:45am–11:45am, Zoom
- Prerequisites: Algorithms or Theory of Computation
- Webpage: https://golovnev.org/gems
- Slides and Videos in Canvas and Piazza
- Grading: 5-6 Problem Sets

Administrative Info

- Classes: TR 9:30am–10:45am, Zoom
- Office Hours: T 10:45am–11:45am, Zoom
- Prerequisites: Algorithms or Theory of Computation
- Webpage: https://golovnev.org/gems
- Slides and Videos in Canvas and Piazza
- Grading: 5-6 Problem Sets
- email: `alexgolovnev+gems@gmail.com`

Course Begins

- Running time of an algorithm

Course Begins

- Running time of an algorithm
  - $100n^2$ vs $n^3/10$

Course Begins

- Running time of an algorithm
    - $100n^2$ vs $n^3/10$
    - $100n^2$ vs $2^n/100$

Course Begins

- Running time of an algorithm
    - $100n^2$ vs $n^3/10$
    - $100n^2$ vs $2^n/100$
- Complexity class **P** = problems solvable in time poly(n)

Course Begins

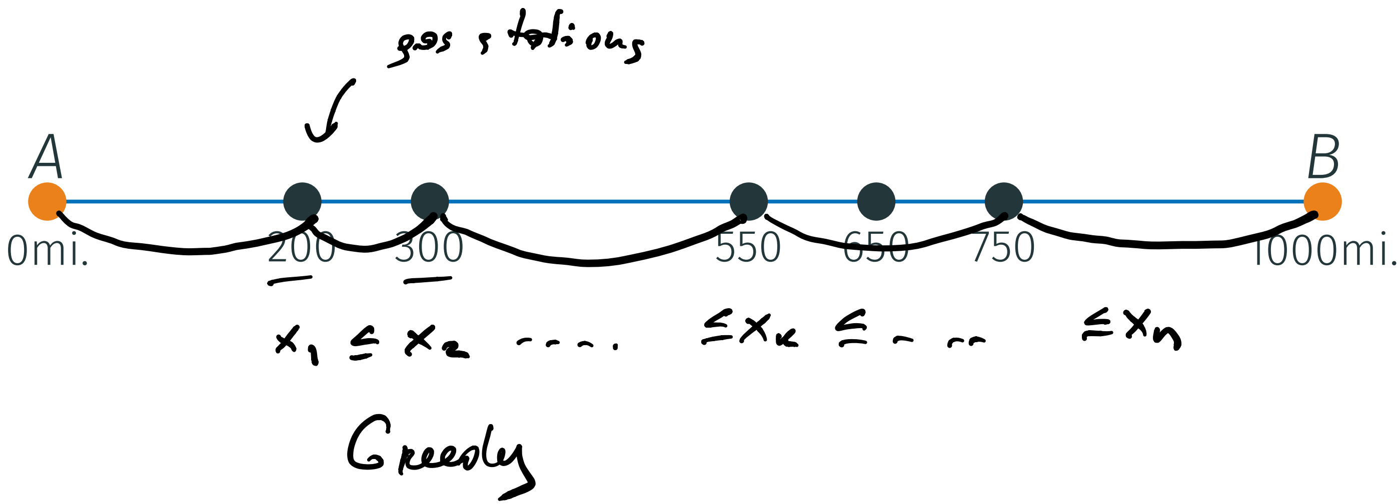
- Running time of an algorithm
  - $100n^2$ vs $n^3/10$
  - $100n^2$ vs $2^n/100$
- Complexity class **P** easy
- Complexity class **NP**-hard hard

Car Fueling

Car Fueling

Distance with full tank 300 mi.

Minimize the number of stops at gas stations

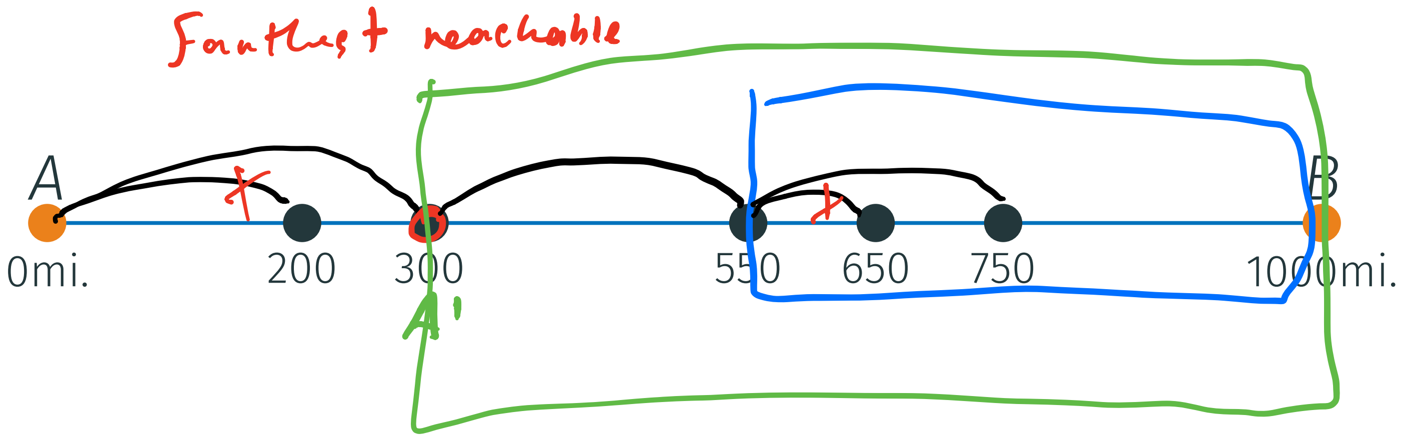


Break http://bit.ly/carfueling

Example

Distance with full tank 300 mi.

Minimize the number of stops at gas stations



Car Fueling. Solution

- “Greedy” algorithm

Car Fueling. Solution

- "Greedy" algorithm

- Runs in linear time $O(n)$, where n is the size of the input (# of gas stations)

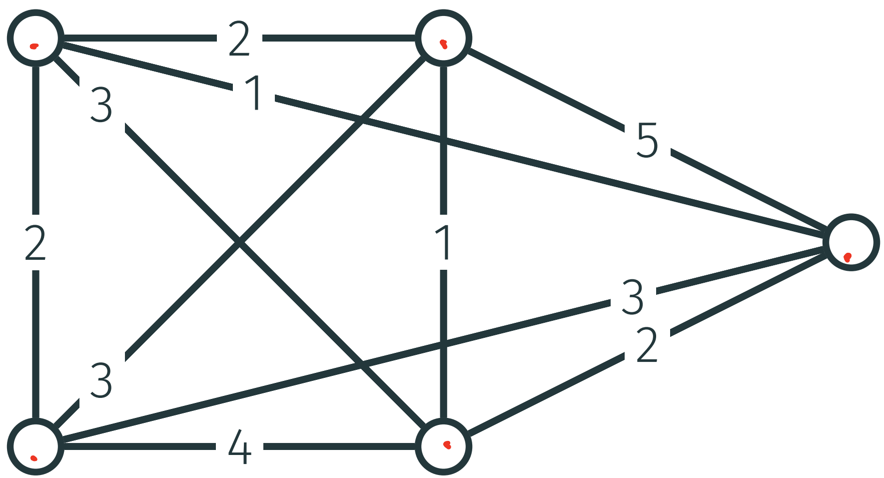
Car Fueling. Solution

- "Greedy" algorithm
- Runs in linear time $O(n)$, where n is the size of the input (# of gas stations)
- Easy problem

Traveling Salesman Problem (TSP)

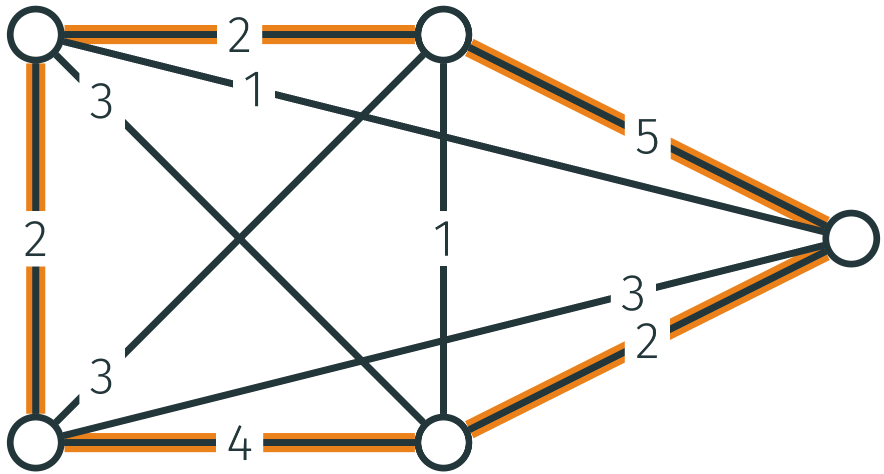
Traveling Salesman Problem

Given a complete weighted graph, find a cycle (or a path) of minimum total weight (length) visiting each node exactly once



Traveling Salesman Problem

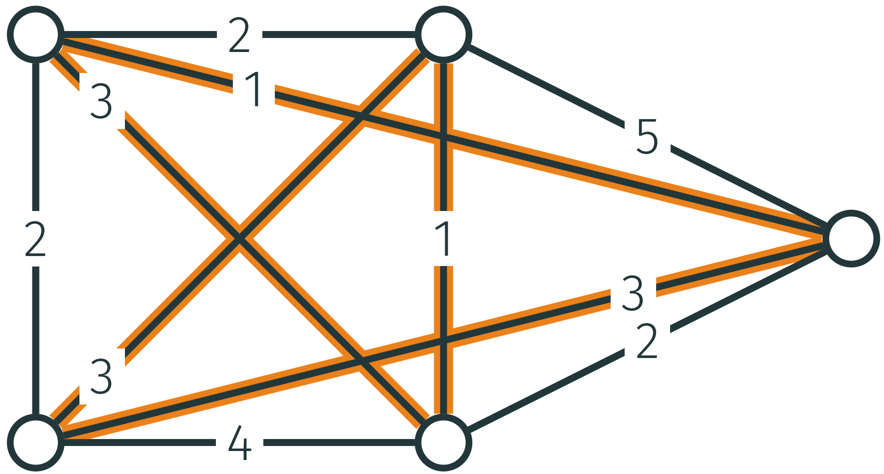
Given a complete weighted graph, find a cycle (or a path) of minimum total weight (length) visiting each node exactly once



length: 15

Traveling Salesman Problem

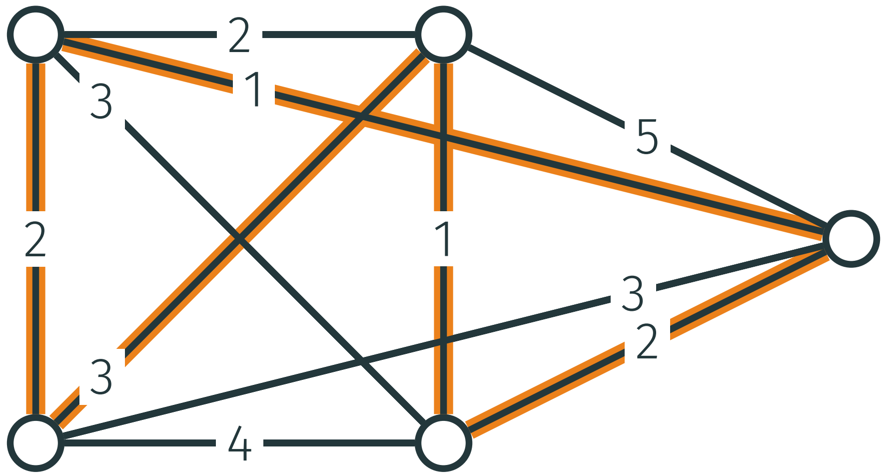
Given a complete weighted graph, find a cycle (or a path) of minimum total weight (length) visiting each node exactly once



length: 11

Traveling Salesman Problem

Given a complete weighted graph, find a cycle (or a path) of minimum total weight (length) visiting each node exactly once



length: 9

Status

- Classical optimization problem with countless number of real life applications (we'll see soon)

Status

- Classical optimization problem with countless number of real life applications (we'll see soon)
- No polynomial time algorithms known

Status

- Classical optimization problem with countless number of real life applications (we'll see soon)
- No polynomial time algorithms known
- The best known algorithm runs in time 2^n

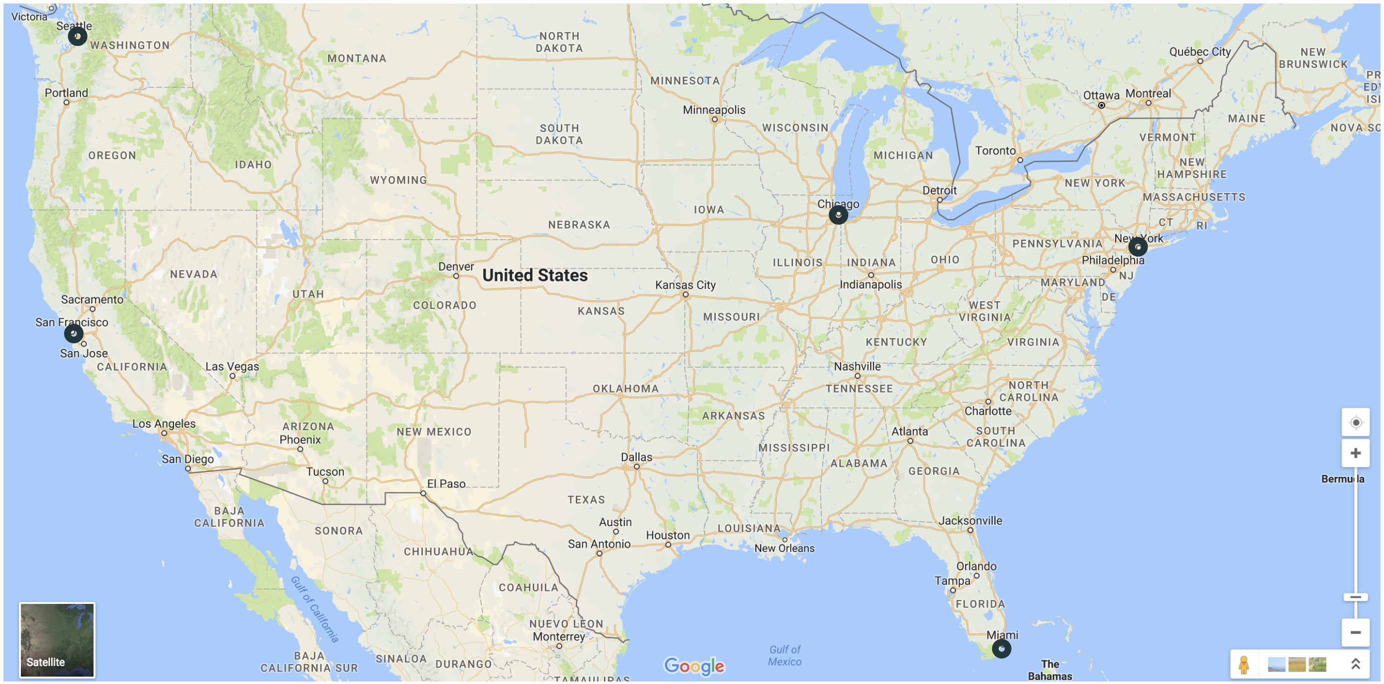
Delivering Goods

Need to visit several points. What is the optimal order of visiting them?

Traveling

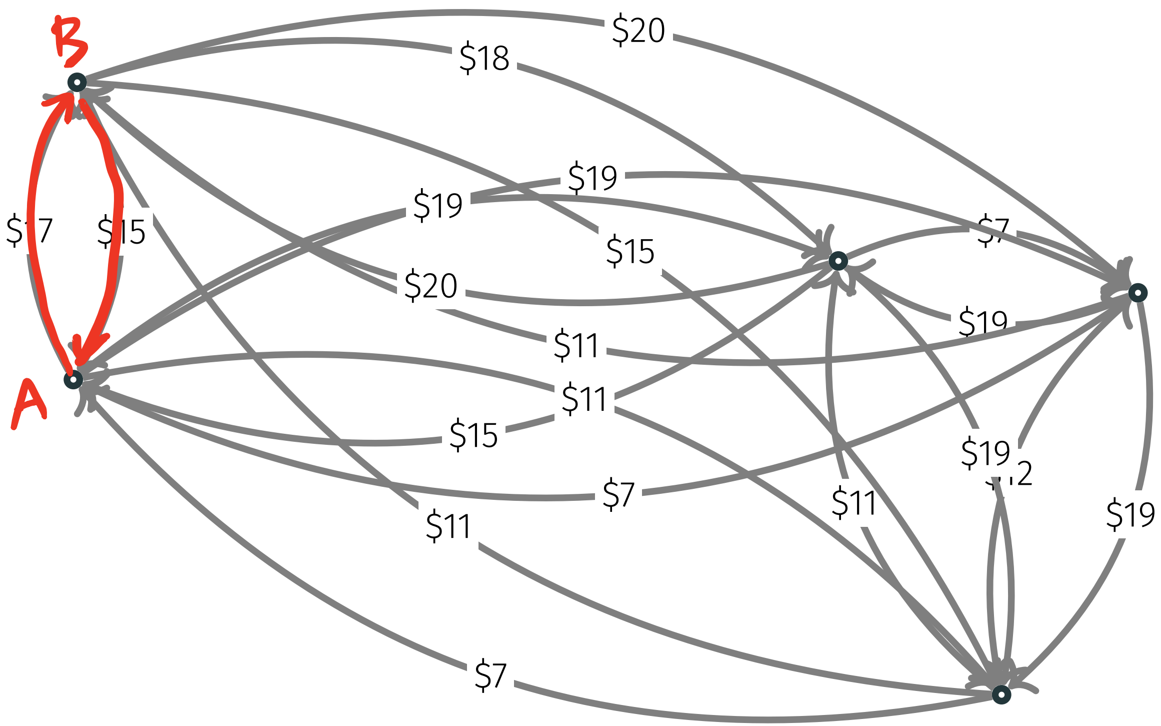

Victoria
Seattle
WASHINGTON
MONTANA
NORTH DAKOTA
MINNESOTA
Québec City
NEW BRUNSWICK
Portland
Minneapolis
Ottawa
Montreal
MAINE
NOVA SC
OREGON
IDAHO
SOUTH DAKOTA
WISCONSIN
VERMONT
MICHIGAN
Toronto
NEW HAMPSHIRE
WYOMING
Detroit
NEW YORK
MASSACHUSETTS
IOWA
Chicago
CT
RI
NEBRASKA
PENNSYLVANIA
New York
NEVADA
Denver
United States
ILLINOIS
INDIANA
OHIO
Philadelphia
NJ
Kansas City
Indianapolis
MARYLAND
DE
UTAH
Sacramento
COLORADO
WEST VIRGINIA
KANSAS
MISSOURI
San Francisco
San Jose
KENTUCKY
VIRGINIA
CALIFORNIA
Las Vegas
Nashville
OKLAHOMA
TENNESSEE
NORTH CAROLINA
Charlotte
Los Angeles
ARIZONA
NEW MEXICO
ARKANSAS
Atlanta
SOUTH CAROLINA
Phoenix
San Diego
MISSISSIPPI
Dallas
ALABAMA
GEORGIA
Tucson
El Paso
TEXAS
Bermuda
BAJA CALIFORNIA
SONORA
Austin
LOUISIANA
Jacksonville
CHIHUAHUA
San Antonio
Houston
New Orleans
Orlando
Tampa
COAHUILA
FLORIDA
Gulf of California
NUEVO LEON
Monterrey
Miami
BAJA CALIFORNIA SUR
SINALOA
DURANGO
Google
Gulf of Mexico
The Bahamas
Satellite

Traveling

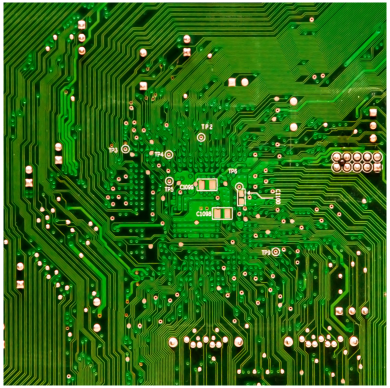

United States
Victoria
Seattle
WASHINGTON
Portland
OREGON
IDAHO
MONTANA
WYOMING
NORTH DAKOTA
SOUTH DAKOTA
NEBRASKA
MINNESOTA
Minneapolis
WISCONSIN
MICHIGAN
IOWA
Chicago
Detroit
Toronto
Ottawa
Montreal
Québec City
NEW BRUNSWICK
MAINE
NOVA SC
VERMONT
NEW HAMPSHIRE
NEW YORK
MASSACHUSETTS
CT
RI
New York
PENNSYLVANIA
Philadelphia
NJ
MARYLAND
DE
OHIO
INDIANA
Indianapolis
ILLINOIS
WEST VIRGINIA
VIRGINIA
KENTUCKY
Nashville
TENNESSEE
NORTH CAROLINA
Charlotte
SOUTH CAROLINA
Atlanta
GEORGIA
ALABAMA
MISSISSIPPI
ARKANSAS
MISSOURI
Kansas City
KANSAS
Denver
COLORADO
UTAH
NEVADA
Sacramento
San Francisco
San Jose
CALIFORNIA
Las Vegas
Los Angeles
San Diego
ARIZONA
Phoenix
Tucson
NEW MEXICO
El Paso
OKLAHOMA
Dallas
TEXAS
Austin
San Antonio
Houston
LOUISIANA
New Orleans
Jacksonville
Orlando
Tampa
FLORIDA
Miami
The Bahamas
Bermuda
Gulf of Mexico
BAJA CALIFORNIA
SONORA
CHIHUAHUA
COAHUILA
NUEVO LEON
Monterrey
SINALOA
DURANGO
BAJA CALIFORNIA SUR
Gulf of California
Satellite
Google

Traveling

Traveling

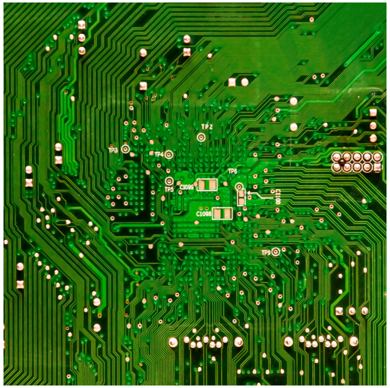


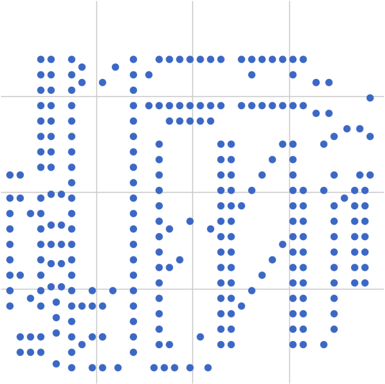
Drilling a Circuit Board



https://developers.google.com/optimization/routing/tsp/tsp

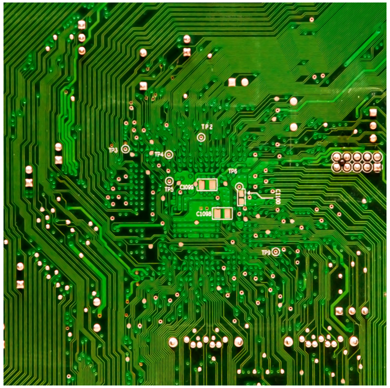
Drilling a Circuit Board

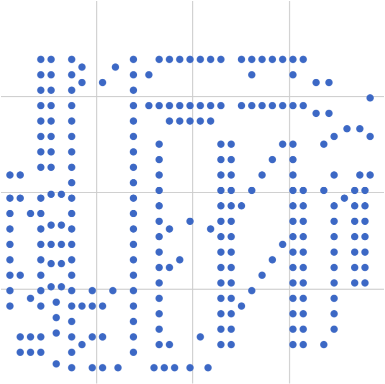


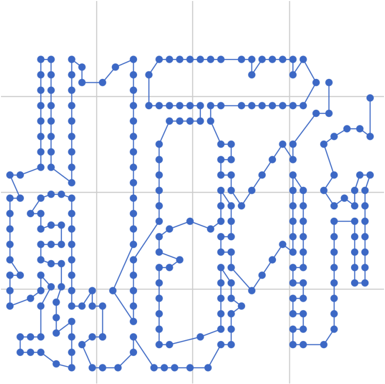


https://developers.google.com/optimization/routing/tsp/tsp

Drilling a Circuit Board



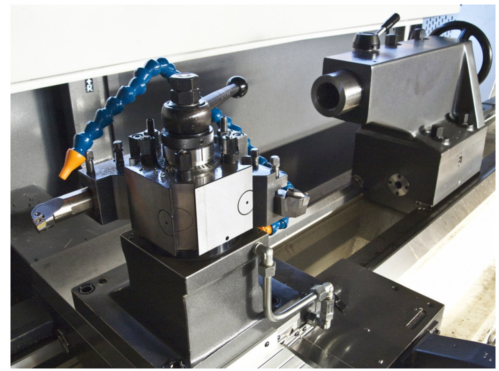




https://developers.google.com/optimization/routing/tsp/tsp

Processing Components

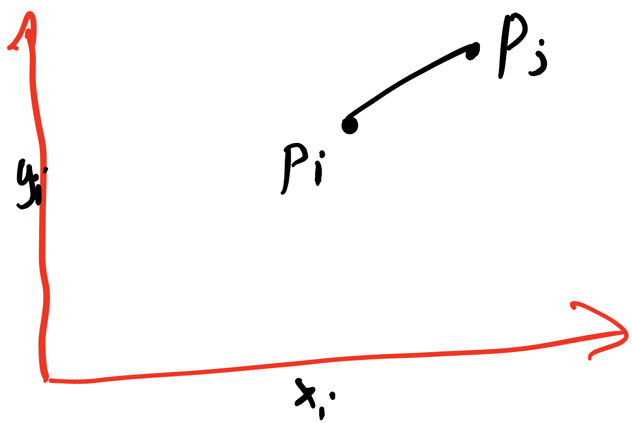
There are n mechanical components to
be processed on a complex machine. After processing the i-th component, it takes

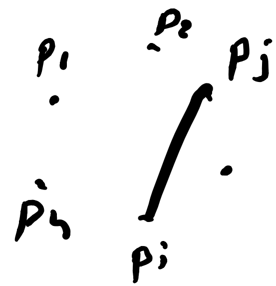


t_{ij} units of time to reconfigure the machine so that it is able to process the j-th component. What is the minimum processing cost?

Euclidean TSP

- Euclidean TSP: instead of a complete graph, the input consists of n points $p_1 = (x_1, y_1), \ldots, p_n = (x_n, y_n)$ on the plane





Euclidean TSP

- Euclidean TSP: instead of a complete graph, the input consists of n points $p_1 = (x_1, y_1), \ldots, p_n = (x_n, y_n)$ on the plane
- Weights are given implicitly:

$$d(p_i, p_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

Euclidean TSP

- Euclidean TSP: instead of a complete graph, the input consists of n points $p_1 = (x_1, y_1), \ldots, p_n = (x_n, y_n)$ on the plane
- Weights are given implicitly:

$$d(p_i, p_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

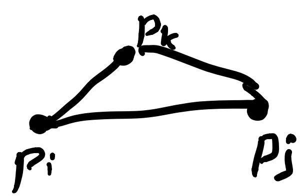
- Weights are symmetric: $d(p_i, p_j) = d(p_j, p_i)$

Euclidean TSP

- Euclidean TSP: instead of a complete graph, the input consists of n points $p_1 = (x_1, y_1), \ldots, p_n = (x_n, y_n)$ on the plane
- Weights are given implicitly:

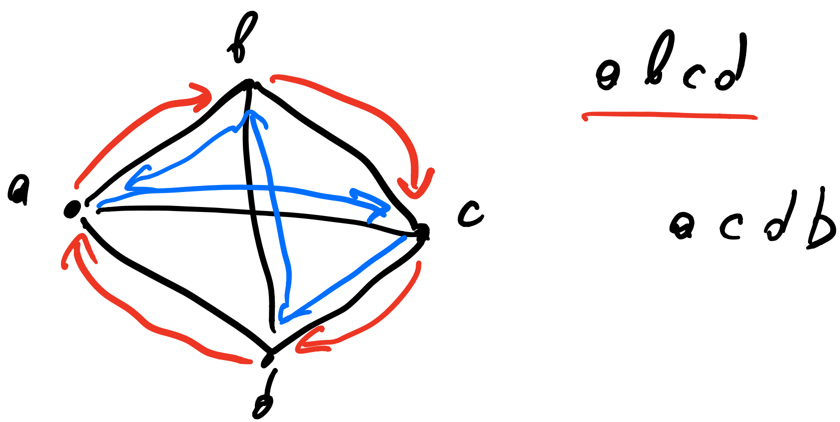
$$d(p_i, p_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

- Weights are symmetric: $d(p_i, p_j) = d(p_j, p_i)$
- Weights satisfy the triangle inequality: $d(p_i, p_j) \leq d(p_i, p_k) + d(p_k, p_j)$



Brute Force Search

- Finding the best permutation is easy: simply iterate through all of them and select the best one



Brute Force Search

- Finding the best permutation is easy: simply iterate through all of them and select the best one
- But the number of permutations of n objects is $n!$

$n!$: Growth Rate

n	$n!$
5	120
8	40320
10	3628800
13	6227020800
20	2432902008176640000
30	265252859812191058636308480000000

$\approx n!$ —later→ 2^n

Satisfiability Problem (SAT)

SAT

$x_1 \ldots x_n$ Boolean

x_i: True 1, False 0

$x_1 = 1 \quad x_2 = 1 \quad x_3 = 0$ doesn't satisfy formula

$$(x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \neg x_2) \wedge (\neg x_1 \vee x_3) \wedge (x_2 \vee \neg x_3)$$

negation; clauses; OR; AND

each clause = OR vars or their negations

"1 / True"

Satisfy all clauses

SAT = does there exist assignment to $x_1, \ldots, x_n$ that satisfies all clauses?

SAT

YES: satisfiable

$x_1 = 1 \quad x_2 = 1 \quad x_3 = 1$

$$(x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \neg x_2) \wedge (\neg x_1 \vee x_3) \wedge (x_2 \vee \neg x_3)$$

0 0 0 0 1 0 1

$$\rightarrow (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \neg x_2) \wedge (\neg x_1 \vee x_3) \wedge (x_2 \vee \neg x_3) \wedge (\neg x_1 \vee \neg x_2 \vee \neg x_3)$$

NO: unsatisfiable

Case 1. $x_1 = 1 \Rightarrow x_3 = 1 \Rightarrow x_2 = 1$

Case 2. $x_1 = 0 \Rightarrow x_2 = 0 \Rightarrow x_3 = 0$

Applications of SAT

- Software Engineering
- Chip testing
- Circuit design
- Automatic theorem provers
- Image analysis
- ...

k-SAT

$$\phi(x_1, \ldots, x_n) = (x_1 \vee \neg x_2 \vee \ldots \vee x_k) \wedge$$
$$\ldots \wedge$$
$$(x_2 \vee \neg x_3 \vee \ldots \vee x_8)$$

k-SAT

$$\phi(x_1, \ldots, x_n) = (x_1 \vee \neg x_2 \vee \ldots \vee x_k) \quad \wedge$$
$$\ldots \quad \wedge$$
$$(x_2 \vee \neg x_3 \vee \ldots \vee x_8)$$

ϕ is satisfiable if

$$\exists x \in \{0,1\}^n \colon \phi(x) = 1\,.$$

Otherwise, ϕ is unsatisfiable

k-SAT

$$\begin{aligned}\phi(x_1,\ldots,x_n) = &(x_1 \vee \neg x_2 \vee \ldots \vee x_k) \quad \wedge \\ &\qquad \ldots \qquad\qquad \wedge \\ &(x_2 \vee \neg x_3 \vee \ldots \vee x_8)\end{aligned}$$

ϕ is satisfiable if

$$\exists x \in \{0,1\}^n \colon \phi(x) = 1\,.$$

Otherwise, ϕ is unsatisfiable

k-SAT is SAT where clause length $\leq k$

integer k

1-SAT
2-SAT
3-SAT

k-SAT. Examples

3-SAT

$$(x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \neg x_2) \wedge (\neg x_1 \vee x_3) \wedge (x_2 \vee \neg x_3)$$

k-SAT. Examples

2-SAT

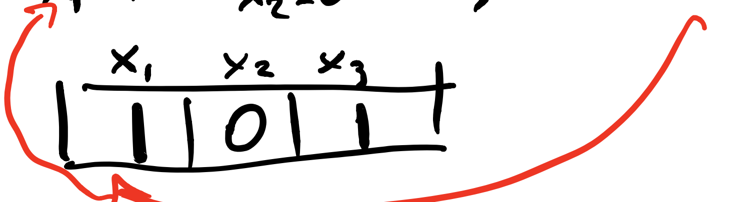
$(x_1 \lor \overline{x_2}) \land (\overline{x_3} \lor x_7)$

$$(x_1 \lor x_2 \lor x_3) \land (x_1 \lor \neg x_2) \land (\neg x_1 \lor x_3) \land (x_2 \lor \neg x_3)$$

1-SAT

$\rightarrow \quad (x_1) \land (\neg x_2) \land (x_3) \land (\neg x_1)$

$x_1 = 1 \qquad x_2 = 0 \qquad x_3 = 1 \qquad x_1 = 0$



Queen of NP-complete problems

- Cook-Levin Theorem [Coo71, Lev73]: SAT can model non-deterministic Turing machine:

  SAT is NP-complete

Queen of NP-complete problems

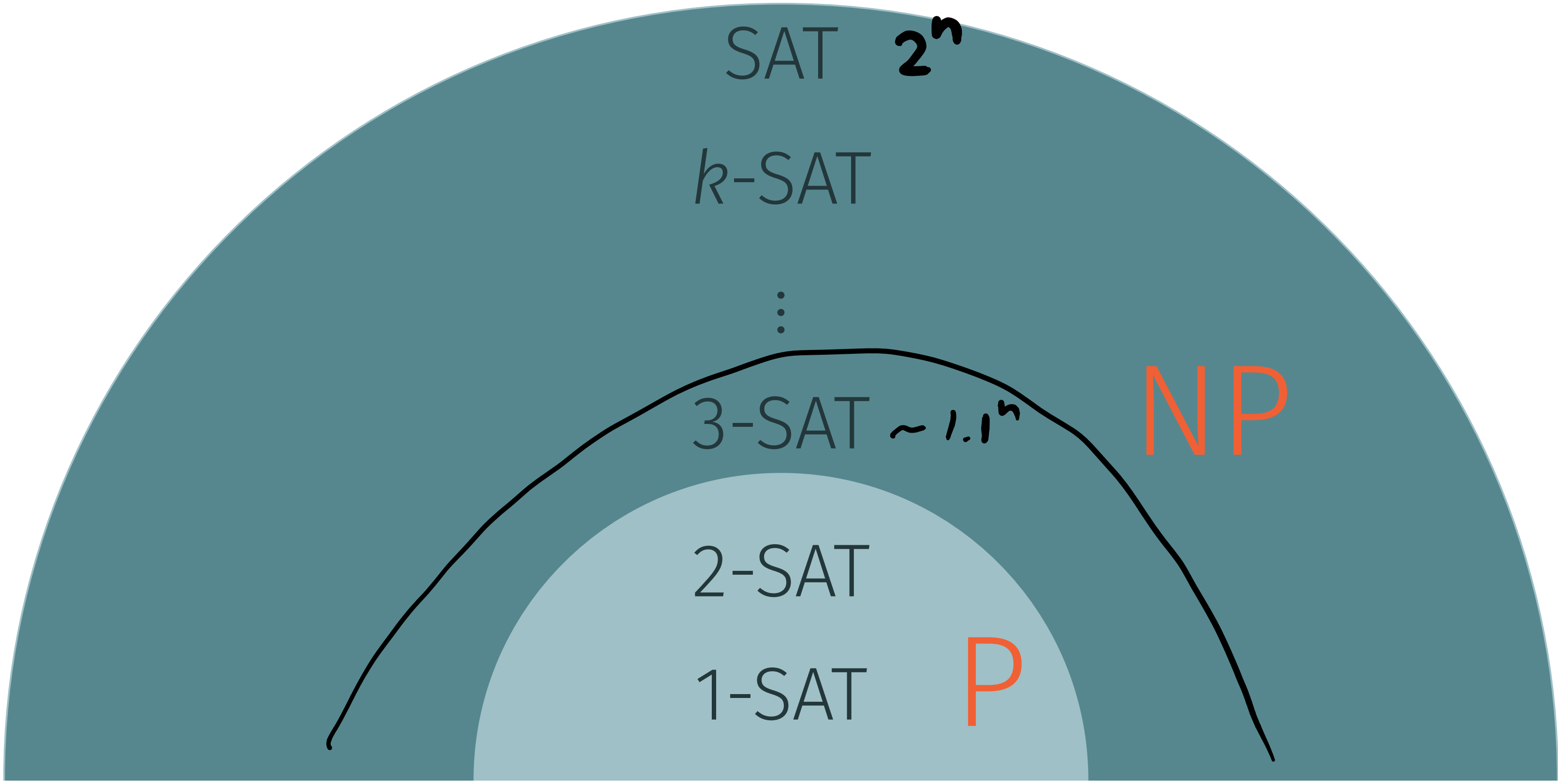
- Cook-Levin Theorem [Coo71, Lev73]: SAT can model non-deterministic Turing machine: SAT is NP-complete
- 3-SAT is NP-complete

Queen of NP-complete problems

- Cook-Levin Theorem [Coo71, Lev73]: SAT can model non-deterministic Turing machine: SAT is NP-complete
- 3-SAT is NP-complete
- 2-SAT is in P
  1-SAT is in P

Complexity of SAT

Complexity of SAT



The SAT game

http://bit.ly/sat-game