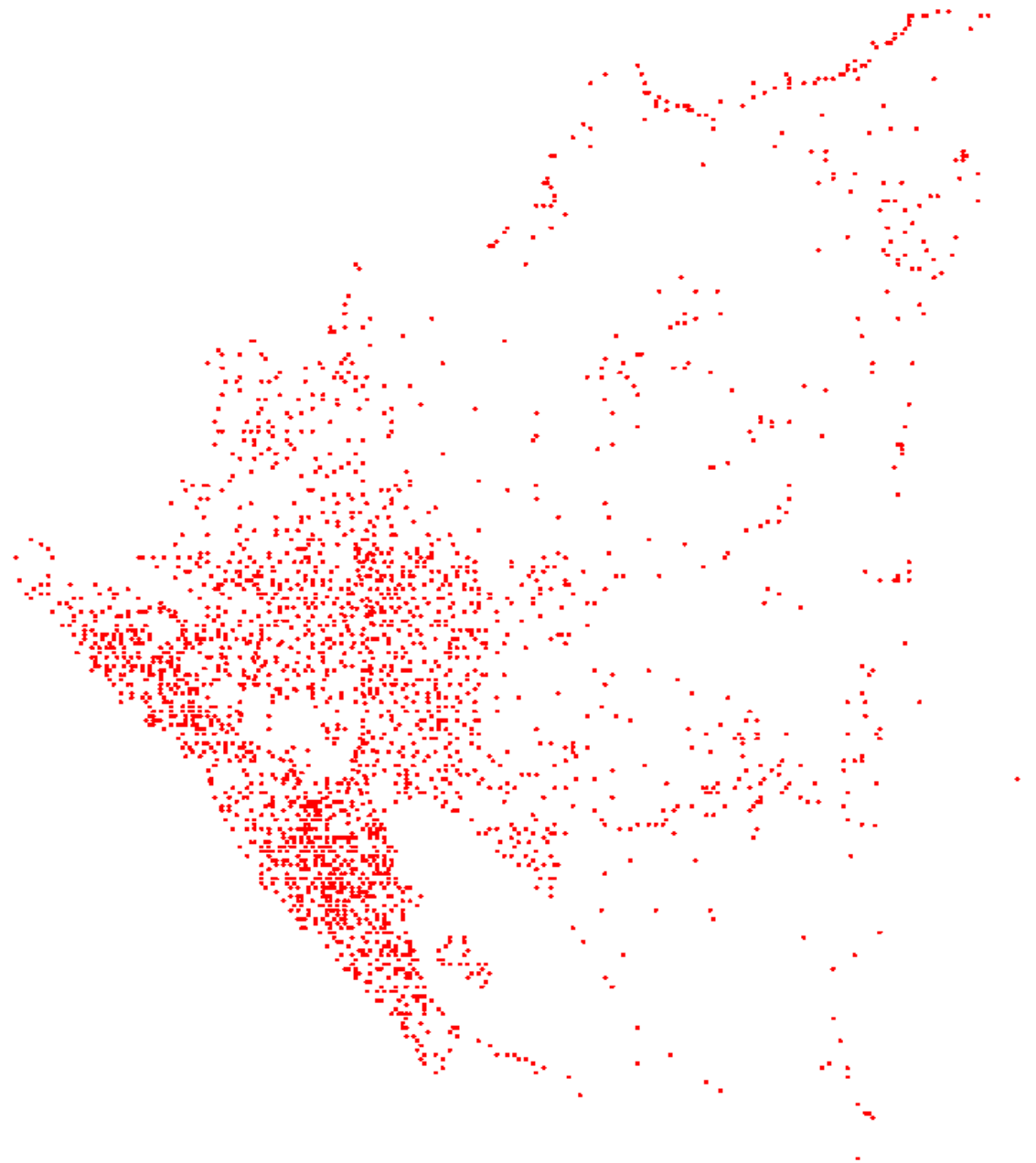


Coursework #2: TSP

CS454 AI-Based Software Engineering
Shin Yoo

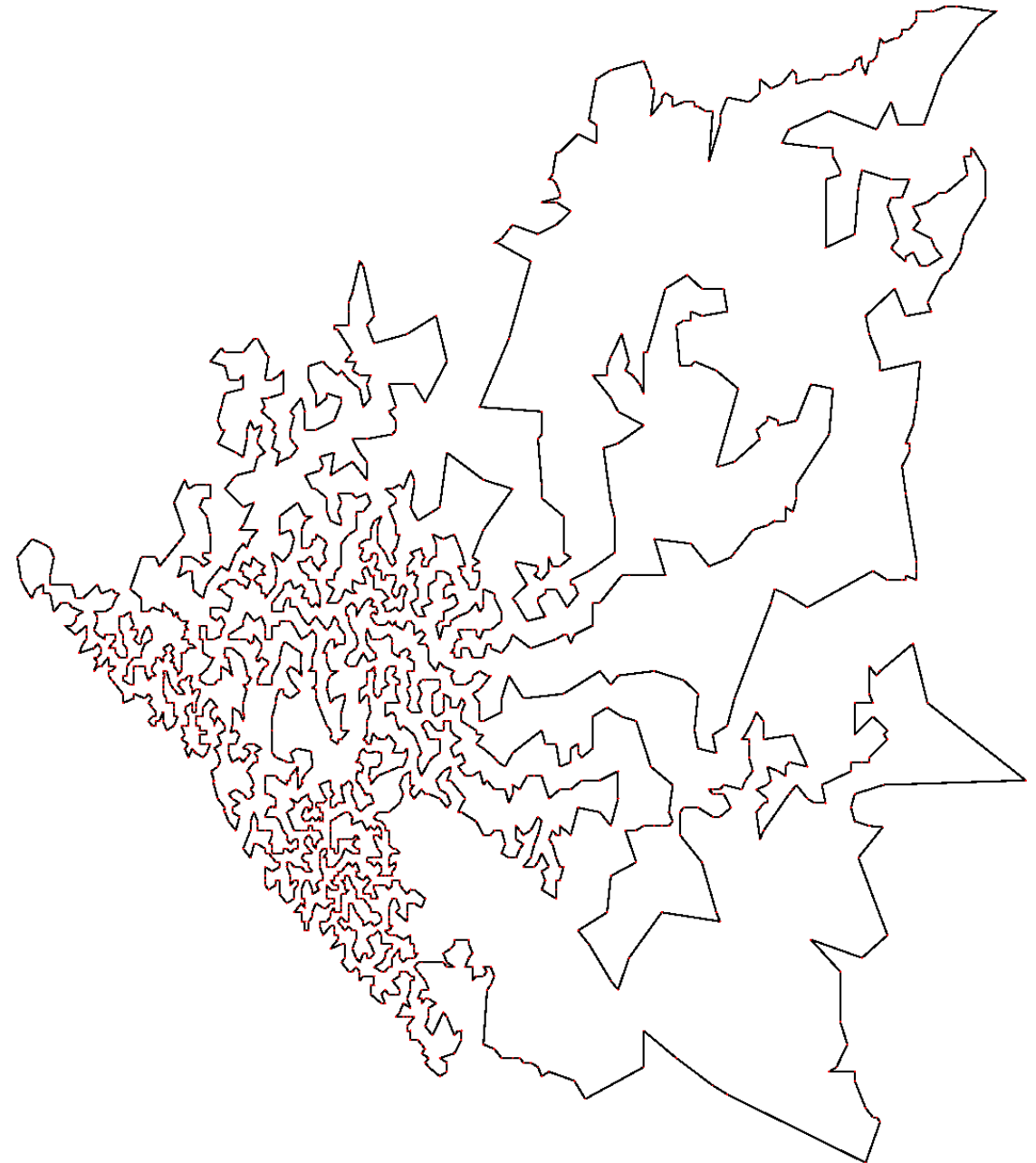
Travelling Salesman Problem

- Given **N** points in space (usually 2D surface)
- Find the shortest tour of all points.
- Search space: **N!**
- Computational complexity: NP-complete
- Brute Force: **$O(N!)$**



Travelling Salesman Problem

- Given **N** points in space (usually 2D surface)
- Find the shortest tour of all points.
- Search space: **N!**
- Computational complexity: NP-complete
- Brute Force: **O(N!)**



Exact Algorithms

- Early dynamic programming
 - Held-Karp algorithm: $O(n^2 2^n)$
- Linear Programming
 - 15,112 German cities: 22.6 CPU years on 500MHz Alpha, 2001
 - 33,810 points on a circuit board: 15.7 CPU years, 2005
 - 85,900 points: 136 CPU years

Heuristic Approach

- Many specific genetic operators have been designed.
- Use domain knowledge. For example:
 - Euclidean TSP observes triangular inequality.
- We're still introducing new algorithms: you can apply them as we go.

Leaderboard

- <http://coinse.kaist.ac.kr/leaderboard>
- System has just been developed for this course: we expect initial problems - report bugs, and be kind to T/As and me :)

☰ Coinse Leaderboard

Hi, coinse Sign out

SCORE SUBMIT

Sept. 12, 2016, 12:14 p.m.

CS492B Coursework 1: TSP

The first coursework is to solve a [Travelling Salesman Problem](#). This is one of the classical NP-hard problems.

MY HOBBY:
EMBEDDING NP-COMPLETE PROBLEMS IN RESTAURANT ORDERS

CHOTCHKIES RESTAURANT

APPETIZERS

MIXED FRUIT	2.15
FRENCH FRIES	2.75
SIDE SALAD	3.35
HOT WINGS	3.55
MOZZARELLA STICKS	4.20
SAMPLER PLATE	5.80

SANDWICHES

BARBECUE	6.55
----------	------

WE'D LIKE EXACTLY \$15.05 WORTH OF APPETIZERS, PLEASE.

... EXACTLY? UHM...

HERE, THESE PAPERS ON THE KNAPSACK PROBLEM MIGHT HELP YOU OUT.

LISTEN, I HAVE SIX OTHER TABLES TO GET TO -

- AS FAST AS POSSIBLE, OF COURSE. WANT SOMETHING ON TRAVELING SALESMAN?

NP Complete by xkcd

We start with N cities, each with a coordinate that specifies its location. The aim of TSP is to visit all cities by travelling the shortest distance. In other words, to find a permutation of cities, so that the length of the resulting tour is as short as possible.

Our problem instance is from the [TSP Library](#). It is [r11849](#), which contains 11,849 cities. The file format is self-explanatory: apart from the text header, the important part is the rows of cities, each consisting of index, x - and y -coordinate. This is a symmetric, Euclidean problem (i.e. distance from x to y is the same as from y to x , and is Euclidean).

You should submit your solution as a `.csv` file, which should contain a single

Leaderboard

- Register with your KAIST email and student ID number
- Submit your solutions to CS454 Coursework 2: TSP

☰

Coinse Leaderboard

Hi, coinse

Sign out

SCORE

SUBMIT

Sept. 12, 2016, 12:14 p.m.

CS492B Coursework 1: TSP

The first coursework is to solve a [Travelling Salesman Problem](#). This is one of the classical NP-hard problems.

MY HOBBY:
EMBEDDING NP-COMPLETE PROBLEMS IN RESTAURANT ORDERS

CHOTCHKIES RESTAURANT

APPETIZERS

MIXED FRUIT	2.15
FRENCH FRIES	2.75
SIDE SALAD	3.35
HOT WINGS	3.55
MOZZARELLA STICKS	4.20
SAMPLER PLATE	5.80

SANDWICHES

BARBECUE	6.55
----------	------

WE'D LIKE EXACTLY \$15.05 WORTH OF APPETIZERS, PLEASE.

... EXACTLY? UHM...

HERE, THESE PAPERS ON THE KNAPSACK PROBLEM MIGHT HELP YOU OUT.

LISTEN, I HAVE SIX OTHER TABLES TO GET TO -

- AS FAST AS POSSIBLE, OF COURSE. WANT SOMETHING ON TRAVELING SALESMAN?

NP Complete by xkcd

We start with N cities, each with a coordinate that specifies its location. The aim of TSP is to visit all cities by travelling the shortest distance. In other words, to find a permutation of cities, so that the length of the resulting tour is as short as possible.

Our problem instance is from the [TSP Library](#). It is [r11849](#), which contains 11,849 cities. The file format is self-explanatory: apart from the text header, the important part is the rows of cities, each consisting of index, x - and y -coordinate. This is a symmetric, Euclidean problem (i.e. distance from x to y is the same as from y to x , and is Euclidean).

You should submit your solution as a `.csv` file, which should contain a single

Leaderboard

- Top solution at the end of the coursework period will get a prize :)
- But this is separate from grading, which will also consider the report, the code quality, as well as the novelty in the approach

☰

Coinse Leaderboard

Hi, coinse

Sign out

SCORE

SUBMIT

Sept. 12, 2016, 12:14 p.m.

CS492B Coursework 1: TSP

The first coursework is to solve a [Travelling Salesman Problem](#). This is one of the classical NP-hard problems.

MY HOBBY:
EMBEDDING NP-COMPLETE PROBLEMS IN RESTAURANT ORDERS

CHOTCHKIES RESTAURANT

APPETIZERS

MIXED FRUIT	2.15
FRENCH FRIES	2.75
SIDE SALAD	3.35
HOT WINGS	3.55
MOZZARELLA STICKS	4.20
SAMPLER PLATE	5.80

SANDWICHES

BARBECUE	6.55
----------	------

WE'D LIKE EXACTLY \$15.05 WORTH OF APPETIZERS, PLEASE.

... EXACTLY? UHM...

HERE, THESE PAPERS ON THE KNAPSACK PROBLEM MIGHT HELP YOU OUT.

LISTEN, I HAVE SIX OTHER TABLES TO GET TO -

- AS FAST AS POSSIBLE, OF COURSE. WANT SOMETHING ON TRAVELING SALESMAN?

NP Complete by xkcd

We start with N cities, each with a coordinate that specifies its location. The aim of TSP is to visit all cities by travelling the shortest distance. In other words, to find a permutation of cities, so that the length of the resulting tour is as short as possible.

Our problem instance is from the [TSP Library](#). It is [r11849](#), which contains 11,849 cities. The file format is self-explanatory: apart from the text header, the important part is the rows of cities, each consisting of index, x - and y -coordinate. This is a symmetric, Euclidean problem (i.e. distance from x to y is the same as from y to x , and is Euclidean).

You should submit your solution as a `.csv` file, which should contain a single

Note

- Coursework: to write a TSP solver that can take any problem instance (in the TSPLIB format).
- Competition: to submit a solution to **r111849** instance to the leaderboard using your solver.