1

IE5600 Applied Programming | Lecture 9

# TSP & CVRP

Andrew Lim

1

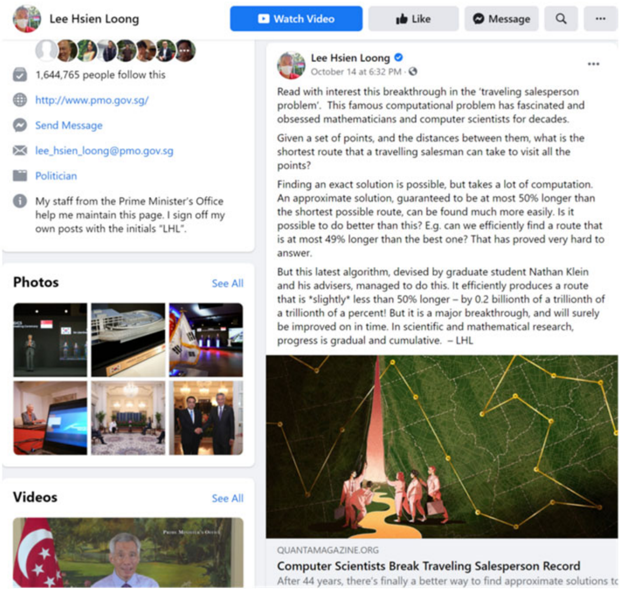
2

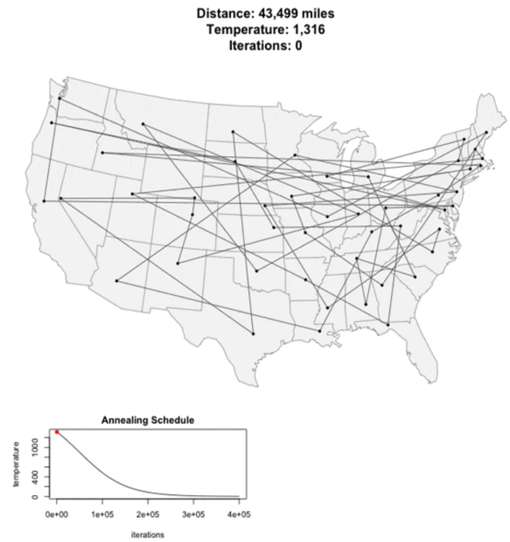
# 01 Travelling Salesman Problem (TSP)

2

# TSP even got the attention of Singapore’s Prime Minister Lee HL

3



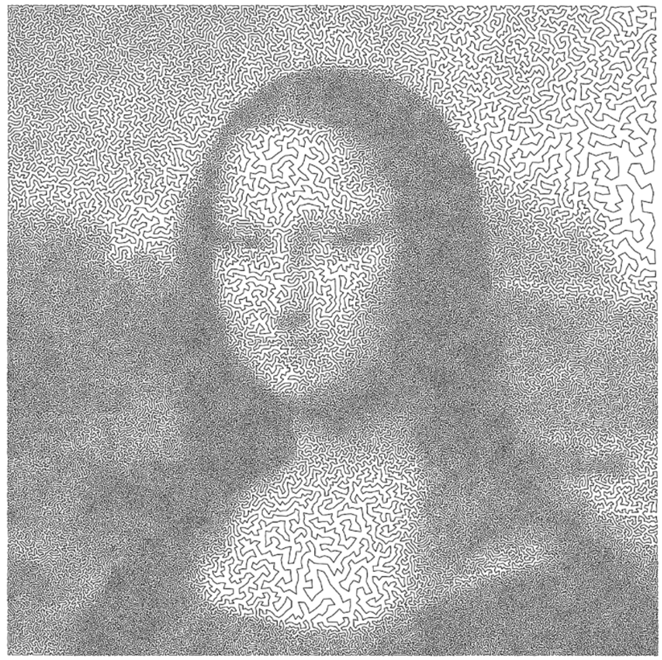



4

Optimal Solution of 15,112-city problem in Germany

In February 2009, Robert Bosch created a 100,000-city instance of the traveling salesman problem (TSP) that provides a representation of Leonardo da Vinci's Mona Lisa as a continuous-line drawing. Techniques for developing such point sets have evolved over the past several years through work of Bosch and Craig Kaplan. $1000 Prize Offered

http://www.math.uwaterloo.ca/tsp/data/ml/monalisa.html

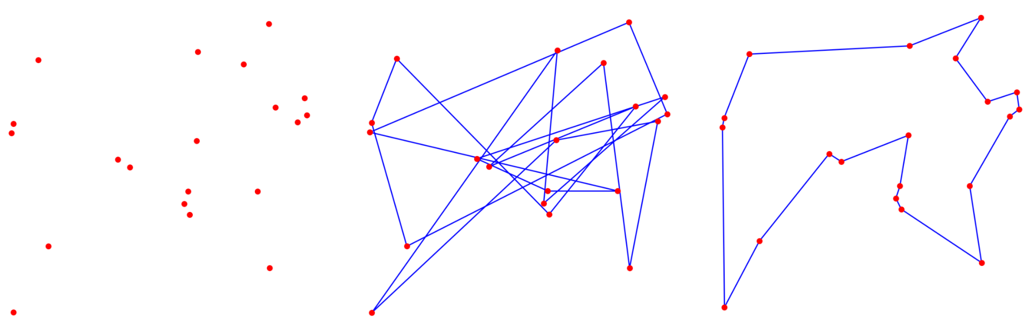




## Travelling Salesman Problem (TSP)



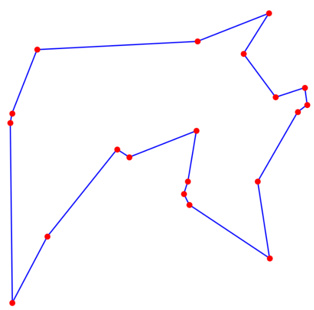
Given a list of cities and their locations (usually specified as Cartesian co-ordinates on a plane), what is the shortest itinerary which will visit every city exactly once and return to the point of origin?



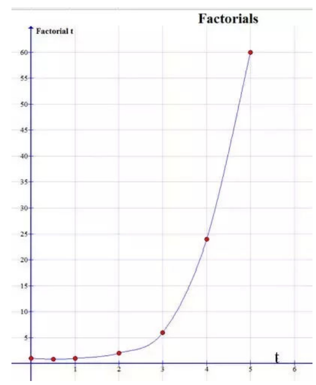
## How to solve?



- Brute force Search： write down all of the possible sequences in which the cities could be visited, compute the distance of each path, and then choose the smallest.
- Time complexity is “n!”.



| n | n! |
|---|---|
| 0 | 1 |
| 1 | 1 |
| 2 | 2 |
| 3 | 6 |
| 4 | 24 |
| 5 | 120 |
| 6 | 720 |
| 7 | 5040 |
| 8 | 40320 |
| … | … |
| 30 | ≈ 2.6525×10^32 |






## TSP — Approach and Solving Methods



- Understand the complexity
- Build the Model
- Problem Solving Methods
  - Constructive Heuristics
  - Branch and Bound, Backtracking
  - Optimization solvers
    1. Cplex
    2. Gurobi
    3. OR-tools
  - Math Programming
  - Meta-heuristics Algorithms

## TSP — Dantzig–Fulkerson–Johnson Formulation

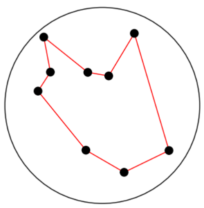
9

Label the cities with the numbers $1, \dots, n$.
Parameters:

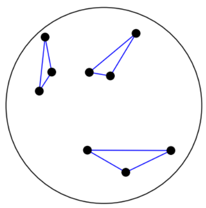
- $c_{ij}$ , the distance from city $i$ to city $j$

Variable:

- $x_{ij}$ , binary variable, is 1 if there is a path from city $i$ to city $j$, otherwise 0.



With Subtour elimination



Without Subtour elimination

For a subset $Q \subsetneq \{1, \dots, n\}$, there must be an edge enter and exit this subset in case of subtour.

$$\min \sum_{i=1}^{n} \sum_{j \neq i, j=1}^{n} c_{ij} x_{ij}:$$

$$x_{ij} \in \{0,1\} \qquad i,j = 1,\dots,n;$$

$$\sum_{i=1, i\neq j}^{n} x_{ij} = 1 \qquad j = 1,\dots,n;$$ Enter each city exactly once

$$\sum_{j=1, j\neq i}^{n} x_{ij} = 1 \qquad i = 1,\dots,n;$$ Leave each city exactly once

$$\sum_{i\in Q} \sum_{j\neq i, j\in Q} x_{ij} \le |Q| - 1 \qquad \forall Q \subsetneq \{1,\dots,n\}, |Q| \ge 2$$

**Subtour elimination constraints**: ensures that there are no sub-tours among the non-starting vertices. Because this leads to an exponential number of possible constraints, in practice it is solved with delayed column generation.

9

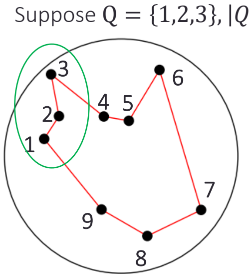
## TSP — Dantzig–Fulkerson–Johnson Formulation

10

Subtour elimination constraints:

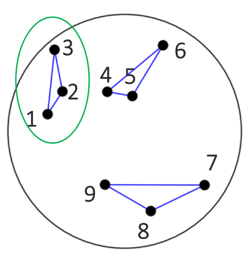
$$\sum_{i\in Q} \sum_{j\neq i, j\in Q} x_{ij} \le \boxed{|Q| - 1} \qquad \forall \boxed{Q \subsetneq \{1,\dots,n\},} |Q| \ge 2$$

Suppose $\mathrm{Q} = \{1,2,3\}, |Q| = 3$




With Subtour elimination

$x_{12} + x_{13} + x_{23} = 2 = |Q|$-1

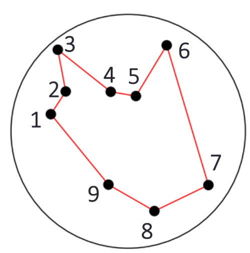



Without Subtour elimination

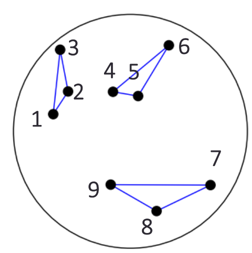
$x_{12} + x_{13} + x_{23} = 3 = |Q|$

If $\sum_{i\in Q} \sum_{j \neq i, j\in Q} x_{ij} = |Q|$ , there must be a subtour.

Suppose $\mathrm{Q} = \{1, \dots, 9\}, |Q| = 9$




With Subtour elimination




Without Subtour elimination

$$\sum_{i\in Q} \sum_{j \neq i, j\in Q} x_{ij} = 9 = |Q|$$

10

## TSP — Miller-Tucker-Zemlin formulation

11

Label the cities with the numbers $1, \dots, n$.

Parameters:
- $c_{ij}$ , the distance from city $i$ to city $j$

Variable:
- $x_{ij}$ , binary variable, is 1 if there is a path from city $i$ to city $j$, otherwise 0.
- $\mu$, dummy variable, representing the times at which a city is visited.

$$\min \sum_{i=1}^{n} \sum_{j \neq i, j=1}^{n} c_{ij} x_{ij}:$$

$$x_{ij} \in \{0,1\} \qquad i,j = 1,\dots,n;$$

$$u_i \in \mathbf{Z} \qquad i = 2,\dots,n;$$

$$\sum_{i=1, i \neq j}^{n} x_{ij} = 1 \qquad j = 1,\dots,n;$$

Enter each city exactly once

$$\sum_{j=1, j \neq i}^{n} x_{ij} = 1 \qquad i = 1,\dots,n;$$

Leave each city exactly once

$$u_i - u_j + n x_{ij} \leq n - 1 \qquad 2 \leq i \neq j \leq n;$$

$$0 \leq u_i \leq n - 1 \qquad 2 \leq i \leq n.$$

Subtour elimination constraints: Limit the sequence of the nodes.



## TSP — Miller-Tucker-Zemlin formulation

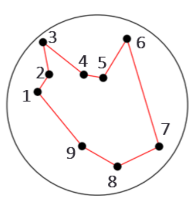
12

Subtour elimination constraints:

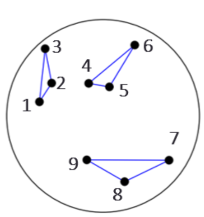
$$u_i - u_j + n x_{ij} \leq n - 1 \qquad 2 \leq i \neq j \leq n;$$

$$0 \leq u_i \leq n - 1 \qquad 2 \leq i \leq n.$$

n=9




| i | j | $u_i - u_j + n * x_{ij} \leq n - 1$ | $x_{ij}$ | $u_i - u_j$ |
|---|---|---|---|---|
| 1 | 2 | $u_1 - u_2 + n * x_{12} \leq n - 1$ | 1 | $u_1 - u_2 \leq -1$ |
| 2 | 3 | $u_2 - u_3 + n * x_{23} \leq n - 1$ | 1 | $u_2 - u_3 \leq -1$ |
| 3 | 1 | $u_3 - u_1 + n * x_{31} \leq n - 1$ | 0 | $u_3 - u_1 \leq 8$ |




| i | j | $u_i - u_j + n * x_{ij} \leq n - 1$ | $x_{ij}$ | $u_i - u_j$ |
|---|---|---|---|---|
| 1 | 2 | $u_1 - u_2 + n * x_{12} \leq n - 1$ | 1 | $u_1 - u_2 \leq -1$ |
| 2 | 3 | $u_2 - u_3 + n * x_{23} \leq n - 1$ | 1 | $u_2 - u_3 \leq -1$ |
| 3 | 1 | $u_3 - u_1 + n * x_{31} \leq n - 1$ | 1 | $u_3 - u_1 \leq -1$ |

Contradiction!

## Constructive Heuristics

13

- Nearest Neighbour

Start from the depot, construct the solution by extending the node one by one. Each time, find the nearest un-visited node run in polynomial time $O(n^2)$. May be further improved: Look ahead more points, better evaluation etc.

- Nearest Insertion

Construct TSP tour for a subset of nodes. Starting from the depot, insert node one by one with nearest insertion cost. Firstly determine the insert node. Then determine the insert position.

- Minimum Spanning Tree

Find Minimum Spanning Tree of the graph (Prims algorithm, Kruskal algorithm). Traverse the tree and skip the visited node.



14

# M1 Simulated Annealing

## Simulated Annealing



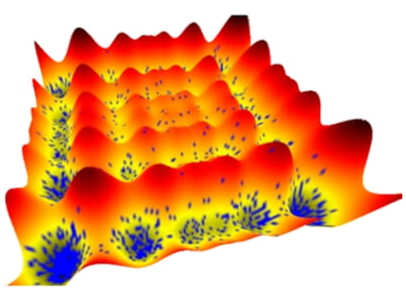
The simulated annealing algorithm was originally inspired from the process of annealing in metal work. Annealing involves heating and cooling a material to alter its physical properties due to the changes in its internal structure. As the metal cools its new structure becomes fixed, consequently causing the metal to retain its newly obtained properties.

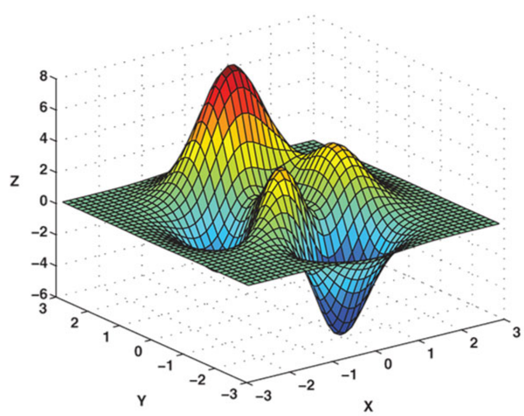


## Simulated Annealing



In simulated annealing, we keep a temperature variable to simulate this heating process. We initially set it high and then allow it to slowly 'cool' as the algorithm runs.

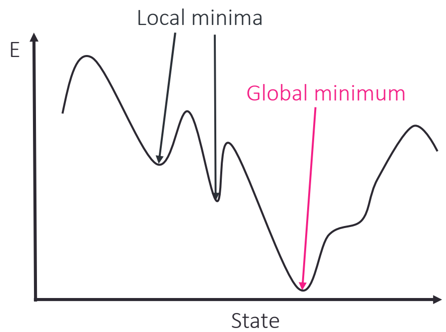


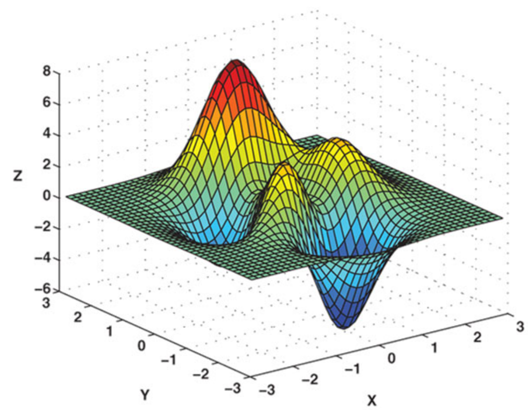


## Simulated Annealing

17

Local optimum VS. Global optimum



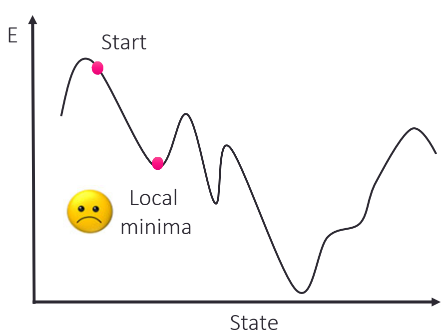


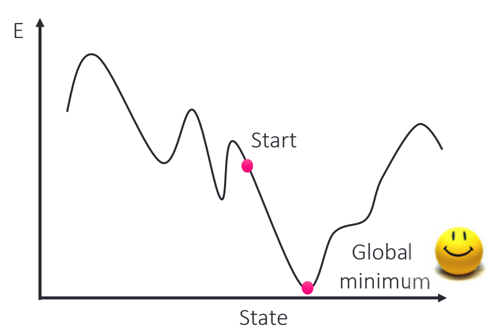

## Simulated Annealing

18

Greedy search:

Initial solution ->always go downhill -> local optimal



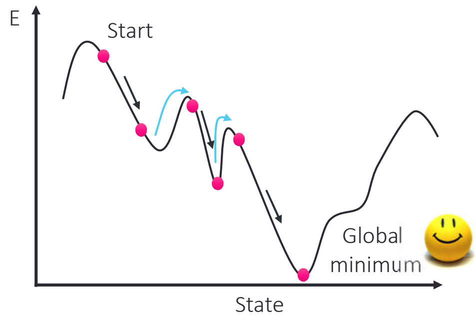



## Simulated Annealing

19

Simulated annealing will move to a new point where the cost is worse with a probability.

Initial solution -> sometimes go uphill, and then go downhill -> Global optimal



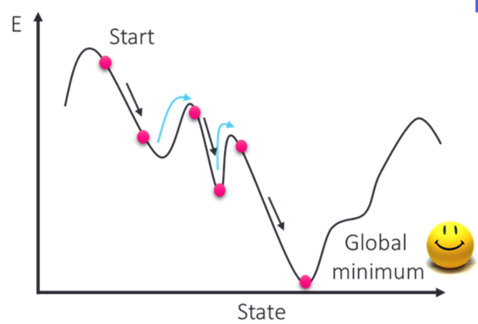



## Simulated Annealing

20

- Probability of accepting that uphill

$$P = exp(-\frac{\Delta Cost}{k_B T})$$




Where $\Delta Cost$ is the change of the energy level, $k_B$ is the Boltzmann's constant, and for simplicity, we can set $k_B$ =1, $T$ is the temperature for controlling the annealing process.

## Simulated Annealing

21

In a minimization problem,

- any better moves or changes that decrease the value of the objective function f will be accepted;
- some changes that increase f will also be accepted with a probability p.

$$P = exp(-\frac{\Delta Cost}{k_B T})$$
$$\Delta Cost = f(\text{NewS})\text{-}f(\text{CurS})$$

- Where $f$(CurS) is the value of current solution, and $f$(NewS) is the value of the new solution.



## Simulated Annealing

22

'Cooling' process: high temperature -> lower temperature

1. While this temperature variable is high the algorithm will be allowed, with more frequency, to accept solutions that are worse than our current solution. This gives the algorithm the ability to jump out of any local optimums it finds itself in early on in execution.
2. As the temperature is reduced so is the chance of accepting worse solutions, therefore allowing the algorithm to gradually focus in on a area of the search space in which hopefully, a close to optimum solution can be found.

$$\boxed{\begin{array}{c} P = exp(-\frac{\Delta Cost}{k_B T}) \\ \Delta Cost = f(\text{NewS})\text{-}f(\text{CurS}) \end{array}}$$

## Simulated Annealing ——Analogy

23

| Physical System | | Optimization Problem |
|---|---|---|
| State | → | Solution |
| Energy | → | Cost function |
| Ground State | → | Optimal solution |
| Rapid Quenching | → | Iteration improvement |
| Careful Annealing | → | Simulated annealing |



## Simulated Annealing

24

SA

Choose, at random, an initial solution $s$ for the system to be optimized

Initialize the temperature $T$

**while** the stopping criterion is not satisfied **do**

    **repeat**

        Randomly select $s' \in N(s)$

        **if** $f(s') \leq f(s)$ **then**

            $s \leftarrow s'$

        **else**

            $s \leftarrow s'$ with a probability $p(T, f(s'), f(s))$

        **end**

    **until** the "thermodynamic equilibrium" of the system is reached

    Decrease $T$

**end**

**return** the best solution met

## Simulated Annealing—Control Parameters

25

1. Definition of equilibrium
   1. Definition is reached when we cannot yield any significant improvement after certain number of loops
   2. A constant number of loops is assumed to reach the equilibrium
2. Annealing schedule (i.e. How to reduce the temperature )
   1. A constant value is subtracted to get new temperature, $T' = T - T\_d$
   2. A constant scale factor is used to get new temperature, $T' = \alpha T$



## Simple Approach to Parameterizing SA

$\boxed{P = exp(-\frac{\Delta Cost}{k_B T}) \quad \Delta Cost = f(\text{NewS})\text{-}f(\text{CurS})}$ $\boxed{T' = \alpha T}$

26

**Parameter $T_o$**

If we are given $P1$ and $\Delta Cost$, then we can compute $T_o$.

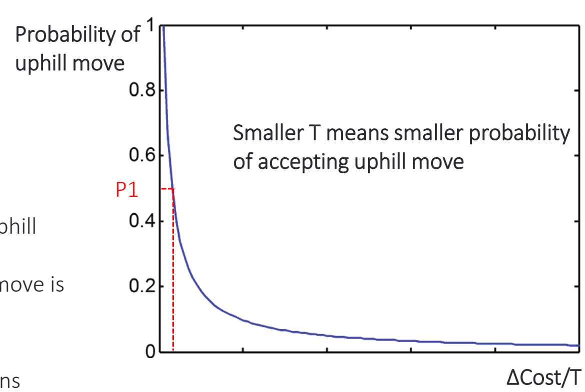
$$T_o = -\frac{\Delta Cost}{k_B \ln P1}$$

**Parameter $\alpha$**

1. How fast do we want the probability of accepting an uphill move to decline?
   The probability $P_k$ in $k_{th}$ iteration of accepting uphill move is
   $$P_k = exp(-\frac{\Delta Cost}{k_B T^k}) = exp(-\frac{\Delta Cost}{k_B T_0 \alpha^k})$$
2. What's the probability $P2$ do you want after $G$ iterations
   $$P_G = exp(-\frac{\Delta Cost}{k_B T_0 \alpha^G})$$

If we are given $P2$ and $G$, then we can compute $\alpha$.

$$\alpha = (-\frac{\Delta Cost}{k_B T_o \ln P2})^{1/G} = (-\frac{\ln P1}{\ln P2})^{1/G}$$




- $P1$ and $P2$ can determined by yourself
- $\Delta Cost$ can be estimated by doing $c$ extra evaluations

$$\Delta Cost = \frac{1}{c}\sum_{j=1}^{c}(Cost(S_j) - Cost(S_0))$$

where $S_0$ is a local optima, and $S_j$ are the uphill for $S_0$.

## Simulated Annealing – Minimizing a function

27

- *Min* $f(s) = (400 - (s/2 - 21)^2) * \sin(s*pi/12)$

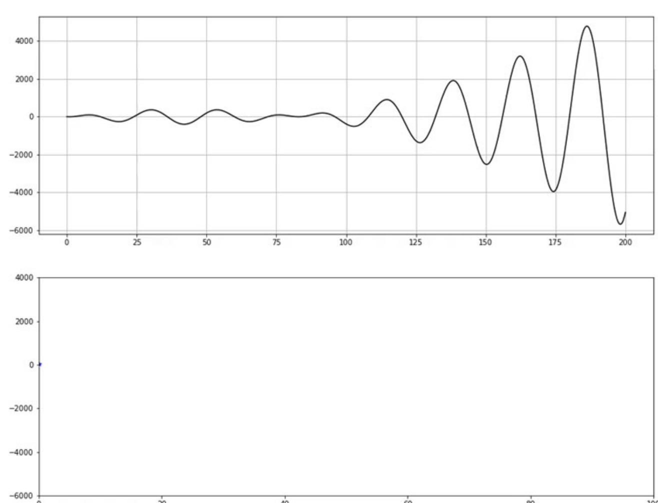
1. Random research
2. Simulated Annealing



## Using Random Search

28

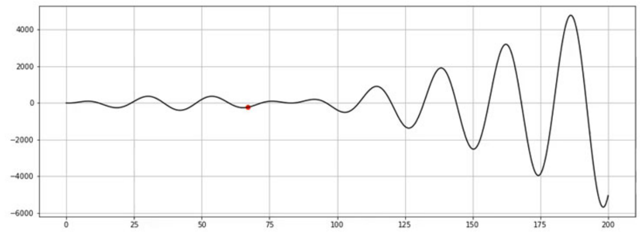
Min linear function use Random Search

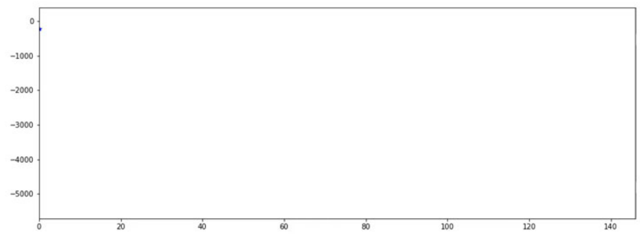


## Using Simulated Annealing

29

Min linear function
use SA



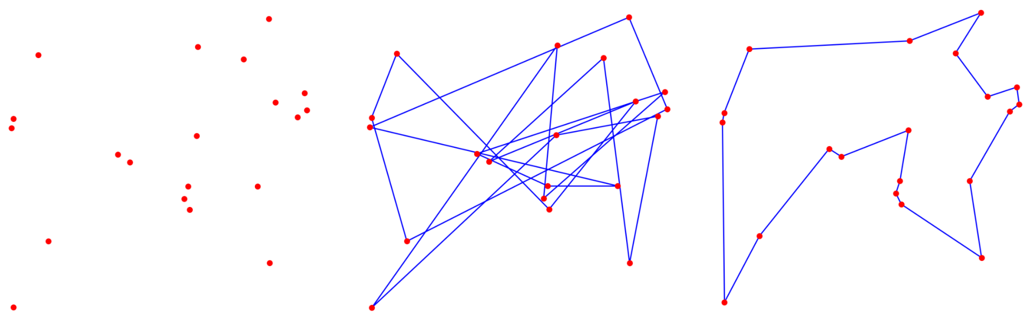



## Simulated Annealing— TSP

30

### Travelling Salesman Problem (TSP)

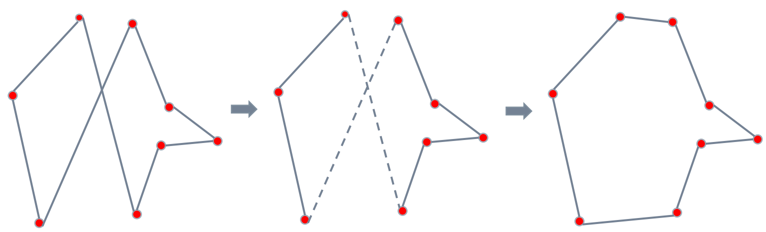
Given a list of cities and their locations (usually specified as Cartesian co-ordinates on a plane), what is the shortest itinerary which will visit every city exactly once and return to the point of origin?



## Simulated Annealing——Example 2 TSP

31

- Local search operators: 2-opt, …

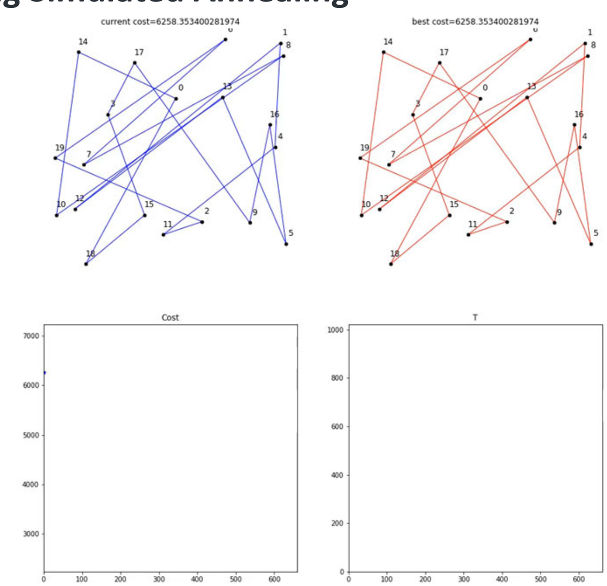


Current solution Select two edge New solution



## Solving TSP using Simulated Annealing

32



## TSP — Data

33

- Traveling Salesman Problem

These pages are devoted to the history, applications, and current research of this challenge of finding the shortest route visiting each member of a collection of locations and returning to your starting point.

http://www.math.uwaterloo.ca/tsp/data/index.html

- TSPLIB

A library of sample instances for the TSP (and related problems) from various sources and of various types.

http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/index.html



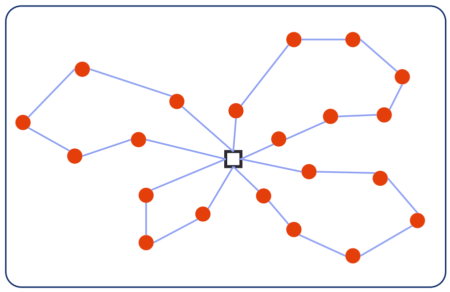
34

# 02 Capacitated Vehicle Routing Problem

## Capacitated Vehicle Routing Problem

35

The *capacitated vehicle routing problem* (CVRP) is a VRP in which vehicles with limited carrying capacity need to pick up or deliver items at various locations. The items have a quantity, such as weight or volume, and the vehicles have a maximum *capacity* that they can carry. The problem is to pick up or deliver the items for the least cost, while never exceeding the capacity of the vehicles.





## CVRP — Formulation

36

Set partitioning formulation

Label the cities with the numbers $1, \ldots, n$, and the depot node with $0$.

Parameters:

- $c_{ij}$ , the distance from city $i$ to city $j$
- $K$, the number of available vehicles

Variable:

- $x_{ij}$ , binary variable, is 1 if there is a path from city $i$ to city $j$, otherwise 0.

$r(S)$ corresponds to the minimum number of vehicles needed to serve set $S$.
The last constraints are the capacity cut constraints, which impose that the routes must be connected and that the demand on each route must not exceed the vehicle capacity

$$\min \sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij}$$

subject to

$$\sum_{i \in V} x_{ij} = 1 \quad \forall j \in V \setminus \{0\}$$

Enter each city exactly once

$$\sum_{j \in V} x_{ij} = 1 \quad \forall i \in V \setminus \{0\}$$

Leave each city exactly once

$$\sum_{i \in V} x_{i0} = K$$

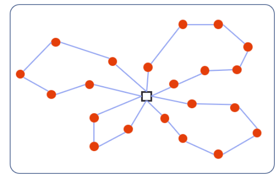
$$\sum_{j \in V} x_{0j} = K$$

$K$ vehicles enter and leave the depot

$$\sum_{i \notin S} \sum_{j \in S} x_{ij} \geq r(S), \quad \forall S \subseteq V \setminus \{0\}, S \neq \emptyset$$

**Subtour elimination constraints**: ensures that there are no sub-tours among the non-starting vertices.

## CVRP – Handling subtour and capacity constraints



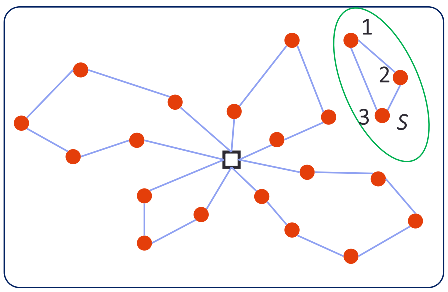
37

Subtour elimination constraints:

$$\sum_{i \notin S} \sum_{j \in S} x_{ij} \geq r(S), \quad \forall S \subseteq V \setminus \{0\}, S \neq \emptyset$$

$r(S)$ corresponds to the minimum number of vehicles needed to serve any subset *S* of customers that does not include the depot.

The Subtour elimination constraints are the capacity cut constraints, which impose that the routes must be connected and that the demand on each route must not exceed the vehicle capacity




$Q = 30, q_1 = 5, q_2 = 4, q_3 = 3$

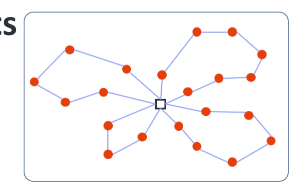
To satisfy capacity of node in subset *S={1,2,3}*, at least one vehicle is needed, that is, $r(S) = 1$.

However, no edges connect the nodes inside *S*, that is, $\sum_{i \notin S} \sum_{j \in S} x_{ij} = 0 < 1 = r(S)$

Therefore, the Subtour elimination constraint is violated.



## CVRP - Handling subtour and capacity constraints



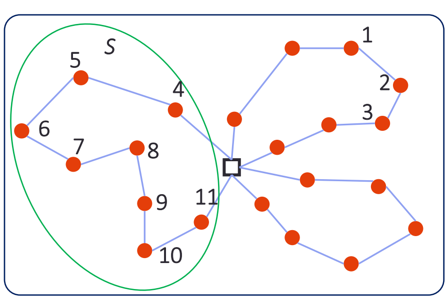
38

Subtour elimination constraints:

$$\sum_{i \notin S} \sum_{j \in S} x_{ij} \geq r(S), \quad \forall S \subseteq V \setminus \{0\}, S \neq \emptyset$$

$r(S)$ corresponds to the minimum number of vehicles needed to serve any subset *S* of customers that does not include the depot.

The Subtour elimination constraints are the capacity cut constraints, which impose that the routes must be connected and that the demand on each route must not exceed the vehicle capacity




$Q = 30, q_4 = 5, q_5 = 4, q_6 = 3, q_7 = 5, q_8 = 10, q_9 = 3, q_{10} = 4, q_{11} = 3$

To satisfy capacity of node in subset, *S={4,5,6,7,8,9,10,11}*, $q(S) = 37$, at least two vehicle is needed, that is, $r(S) = 2$.

However, only one edge enter *S*, that is, $\sum_{i \notin S} \sum_{j \in S} x_{ij} = 1 < 2 = r(S)$

Therefore, the Subtour elimination constraint is violated.

## CVRP — Formulation

39

Two-index vehicle flow formulation

Label the cities with the numbers $1, \dots, n$, and the depot node with $0$ and $n+1$, where all routes must start on $0$ and return to $n+1$, and their positions are the same.

Parameters:

- $c_{ij}$ , the distance from city $i$ to city $j$
- $K$, the number of available vehicles

Variable:

- $x_{ij}$ , binary variable, is 1 if there is a path from city $i$ to city $j$, otherwise 0.
- $y_j$ is a continuous decision variable corresponding to the cumulated demand on the route that visits node $j \in N$ up to this visit.

$$
\begin{aligned}
\min \quad & \sum_{i=0}^{n+1}\sum_{j=0}^{n+1} c_{ij}x_{ij} & \\
\text{s.t.} \quad & \sum_{\substack{j=1\\ j\neq i}}^{n+1} x_{ij} = 1, & i = 1,\dots,n,\\
& \sum_{\substack{i=0\\ i\neq h}}^{n} x_{ih} - \sum_{\substack{j=1\\ j\neq h}}^{n+1} x_{hj} = 0, & h = 1,\dots,n,\\
& \sum_{j=1}^{n} x_{0j} \le K, & \\
& y_j \ge y_i + q_j x_{ij} - Q(1-x_{ij}), & i,j = 0,\dots,n+1,\\
& d_i \le y_i \le Q, & i = 0,\dots,n+1,\\
& x_{ij} \in \{0,1\}, & i,j = 0,\dots,n+1.
\end{aligned}
$$

Munari, P., Dollevoet, T., & Spliet, R. (2016). *A generalized formulation for vehicle routing problems*. (1), 1–19. Retrieved from http://arxiv.org/abs/1606.01935



## CVRP — Two-index vehicle flow formulation

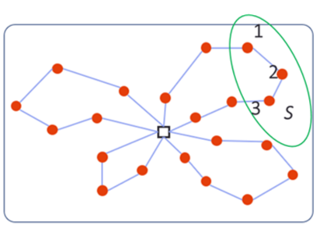
40

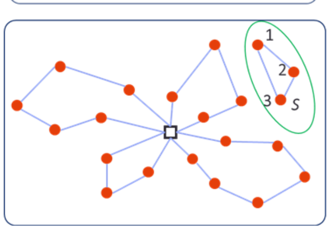
Subtour elimination constraints:

$y_j \ge y_i + q_j x_{ij} - Q(1-x_{ij}),$ $\quad i,j = 0,\dots,n+1,$

$d_i \le y_i \le Q,$ $\quad i = 0,\dots,n+1,$

$Q = 30, q_1 = 5, q_2 = 4, q_3 = 3$



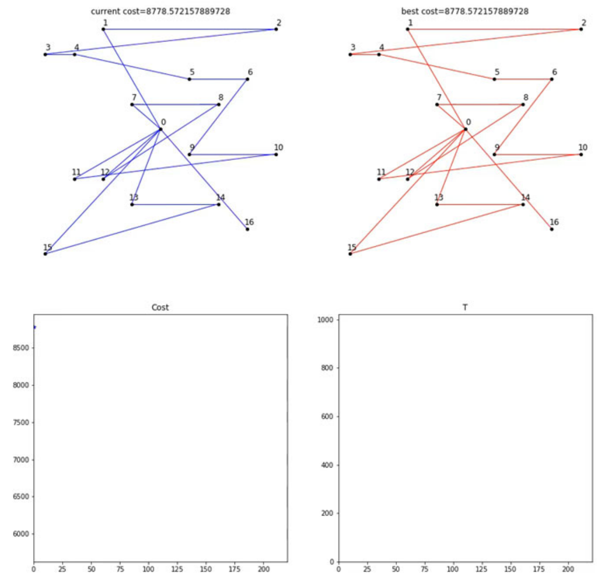

| i | j | $y_j \ge y_i + q_j x_{ij} - Q(1-x_{ij})$ | $x_{ij}$ | |
|---|---|---|---|---|
| 1 | 2 | $y_2 \ge y_1 + q_2 x_{12} - Q(1-x_{12})$ | 1 | $y_2 \ge y_1 + 4$ |
| 2 | 3 | $y_3 \ge y_2 + q_3 x_{23} - Q(1-x_{23})$ | 1 | $y_3 \ge y_2 + 3$ |
| 3 | 1 | $y_1 \ge y_3 + q_1 x_{31} - Q(1-x_{31})$ | 0 | $y_1 \ge y_3 - 25$ |

| i | j | $y_j \ge y_i + q_j x_{ij} - Q(1-x_{ij})$ | $x_{ij}$ | |
|---|---|---|---|---|
| 1 | 2 | $y_2 \ge y_1 + q_2 x_{12} - Q(1-x_{12})$ | 1 | $y_2 \ge y_1 + 4$ |
| 2 | 3 | $y_3 \ge y_2 + q_3 x_{23} - Q(1-x_{23})$ | 1 | $y_3 \ge y_2 + 3$ |
| 3 | 1 | $y_1 \ge y_3 + q_1 x_{31} - Q(1-x_{31})$ | 1 | $y_1 \ge y_3 + 5$ |

Contradiction!

## Solving CVRP use Simulated Annealing








## CVRP—— Data



- Vehicle Routing Data Sets

https://www.coin-or.org/SYMPHONY/branchandcut/VRP/data/index.htm.old

- CVRPLIB

http://vrp.atd-lab.inf.puc-rio.br/index.php/en/

- CVRP Instances

https://neo.lcc.uma.es/vrp/vrp-instances/capacitated-vrp-instances/