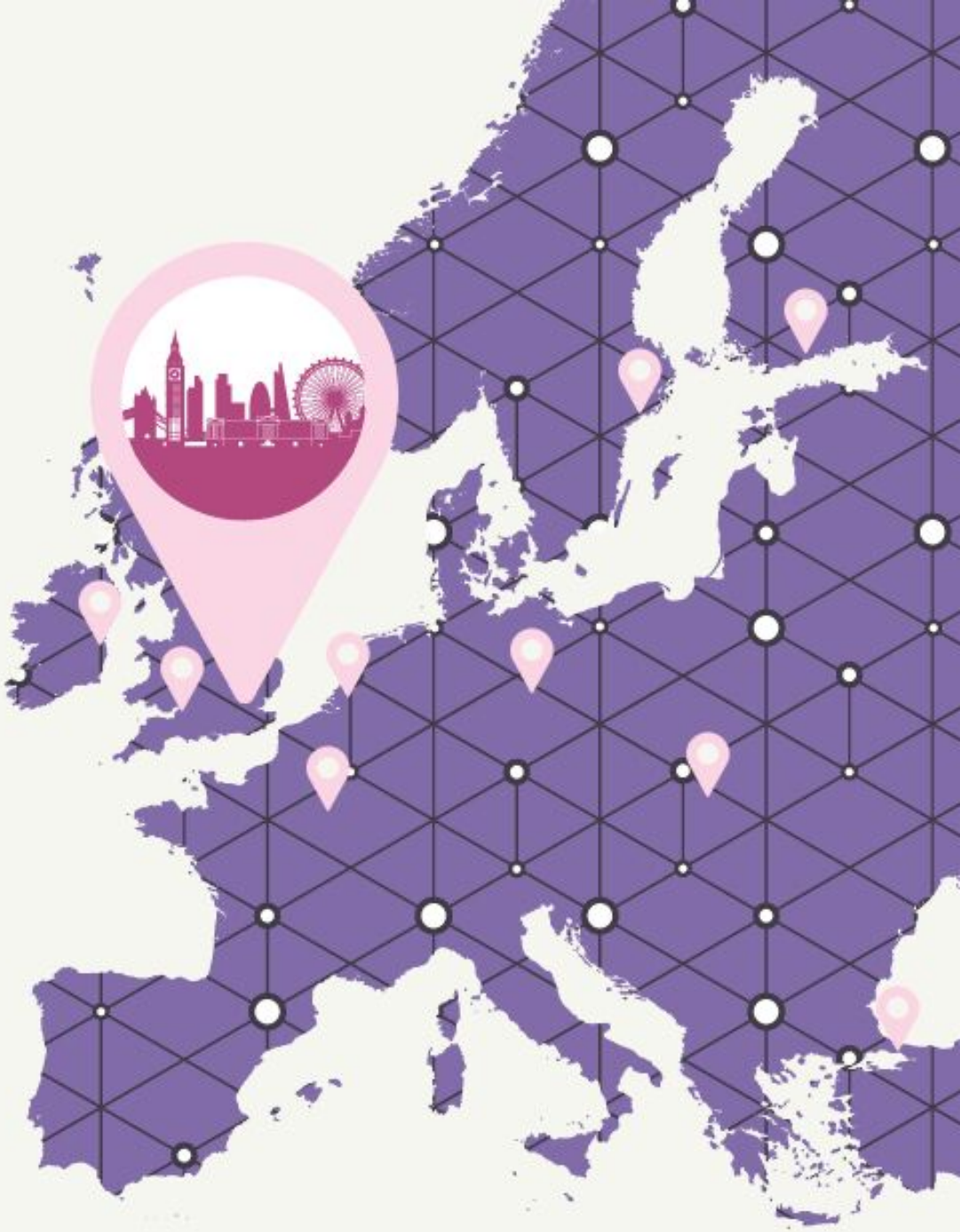
DATA SCIENCE FESTIVAL

LONDON

WELCOME TO
THE DATA SCIENCE FESTIVAL.

GOOD TIMES
GOOD TALKS
GOOD TECH

#DATASCIENCEFEST

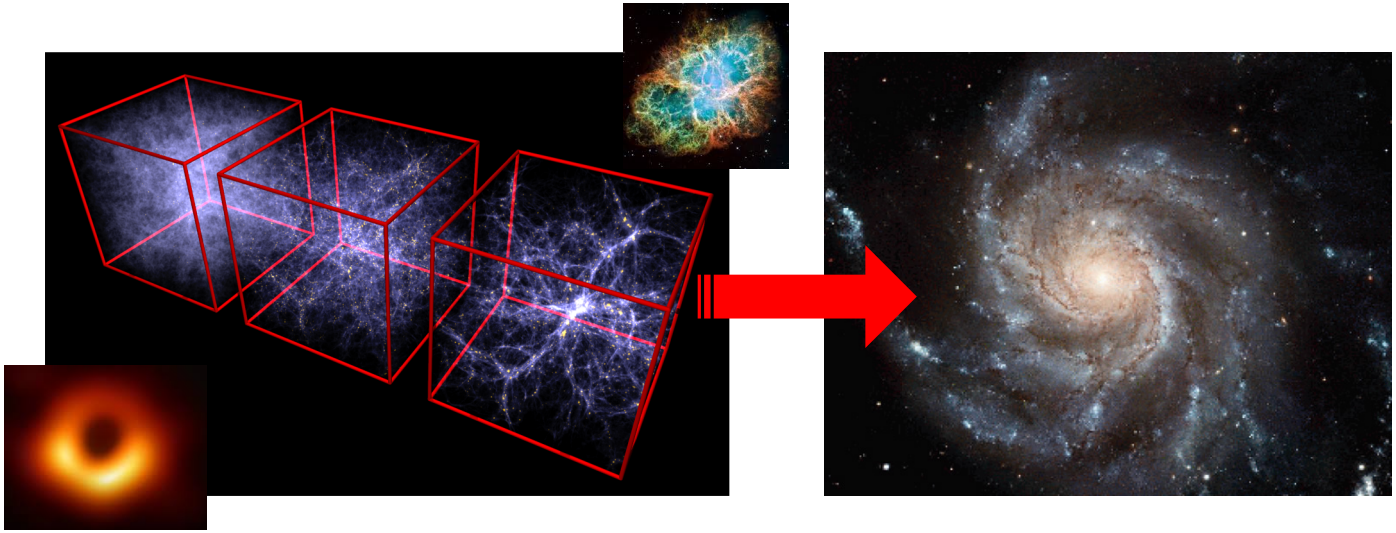


"Efficient Route Optimisation, the Future of Parcel Delivery"

Dr. Fabrice Durier

Credentials

From Computational Cosmology to Data Science



After 13 years modelling Galaxy Formation…

- PhD in Astrophysics (2009)
- Research positions in 4 different countries

Credentials

From Computational Cosmology to Data Science

… I got into Business!

Big Up to *"Loïc le Geek", "Dan the Man"*

& "Fantastic Nick"

Efficient Route Optimisation

DATA SCIENCE FESTIVAL

I. Data Science @ Royal Mail

- The People
- The Projects

II. Business Case

- Challenges and Benefits
- An Example

III. Science & Technology

- A known Problem
- An Agile approach

IV. Demonstration & Questions

I. Data Science Team

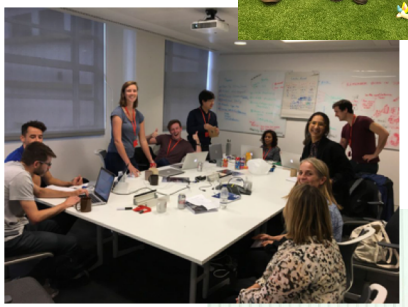
Diversity is Key

13 countries

7 16

PhD	12	52%
MSc	6	26%
BSc	3	13%
App.	2	8%

Many backgrounds



I. Data Science Projects

Wide Range of Problems

Estimated Delivery Window

Geo-location Data

Delivery Time Prediction

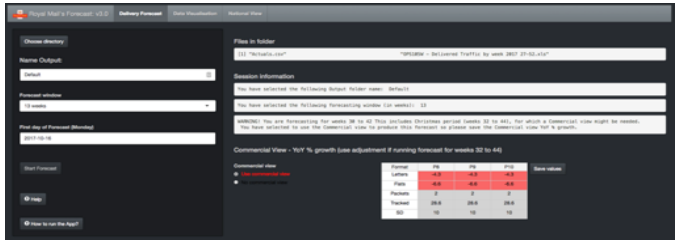
Fleet Management

Telemetry Data

Capacity Optimisation

Mail Forecast

Time-Series Analysis



Fraud Detection

NLP

Anomaly Detection

II. Business Case

Parcel Delivery Market

- ✓ Traffic of **1.9 billion items**, up by 9%
- ✓ Royal Mail handled **52% of Market Share** in 2017-18
- ✓ Delivering to over **29 million addresses**, 6 days a week
- ✓ **Growing expectations** for service quality in a highly competitive market…

II. Business Goal

Motivation

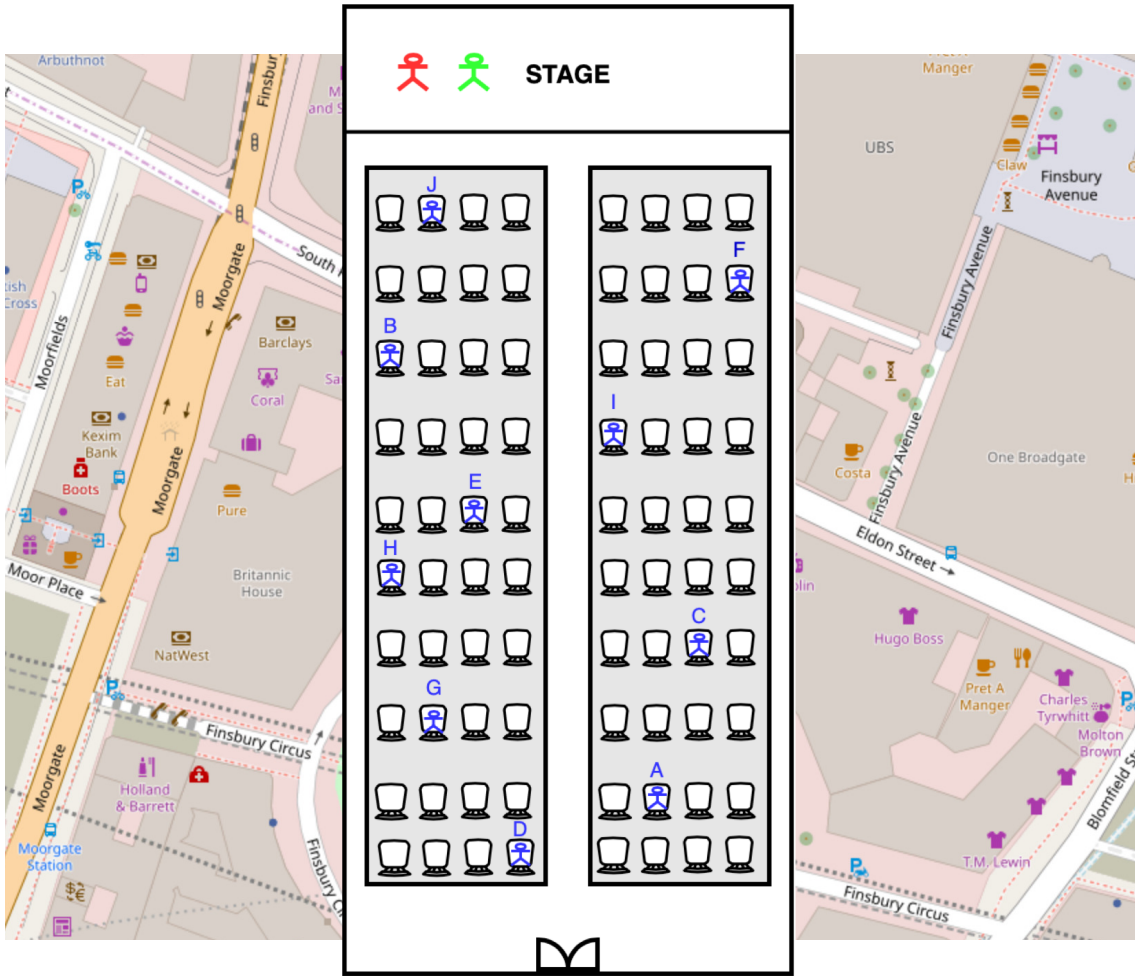
" Utilise our resources efficiently given the daily variation of demand."

Requirements

1. Deliver high quality service
2. Rely on Network of Delivery Offices
3. Account for Fleet Constraints
4. Account for Workforce Constraints

➡ Need of a tool that can manage the **sorting** and **ordering** of parcels

II. Live Simulation



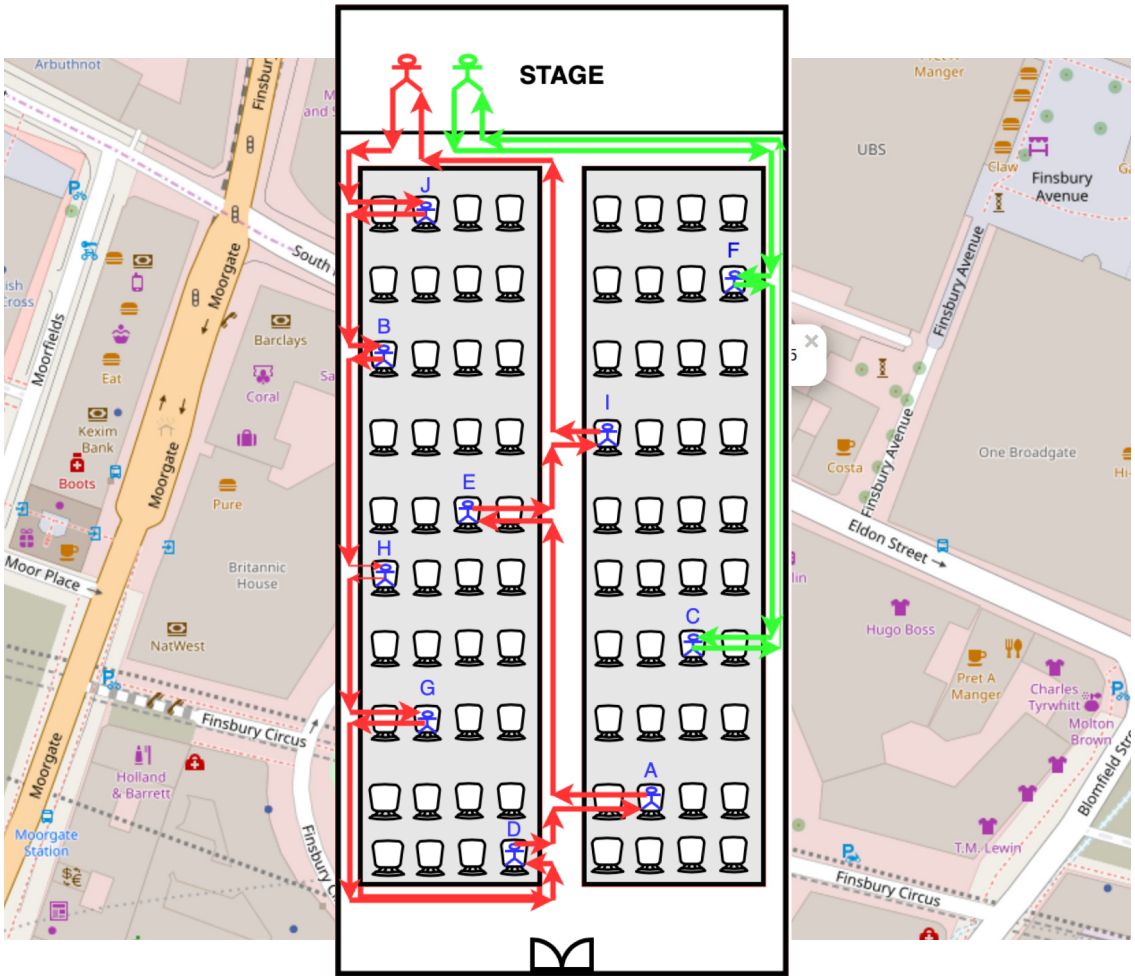
Rules of the Game

- ✓ 2 Posties
- ✓ 10 Customers
- ✓ one minute for Sorting
- ✓ one minute for Delivery

Objective

Minimise the total delivery time

II. Live Simulation



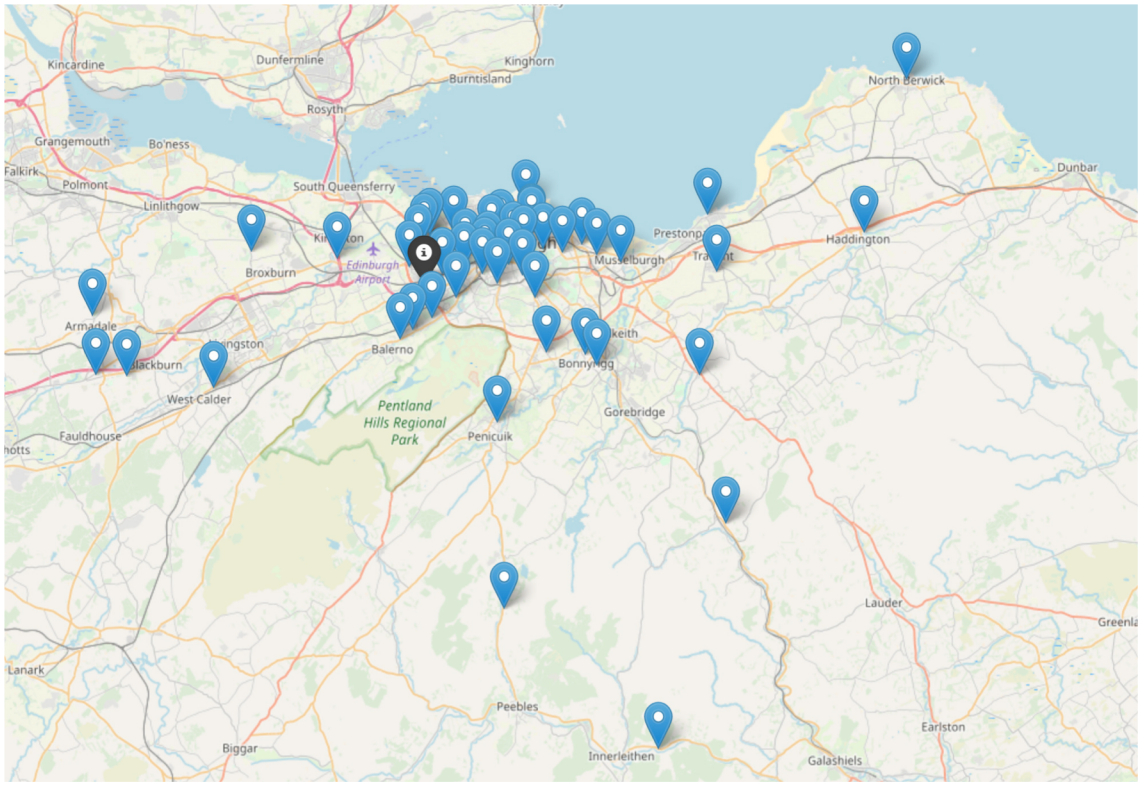
Results

Postie	#1	#2
Drop-1	F	J
Drop-2	C	B
Drop-3		H
Drop-4		G
Drop-5		D
Drop-6		A
Drop-7		E
Drop-8		I
Time	**22 sec**	**46 sec**

➢ **Total Time:** 68 sec

III. Optimisation Problem

Mock Example



Setup

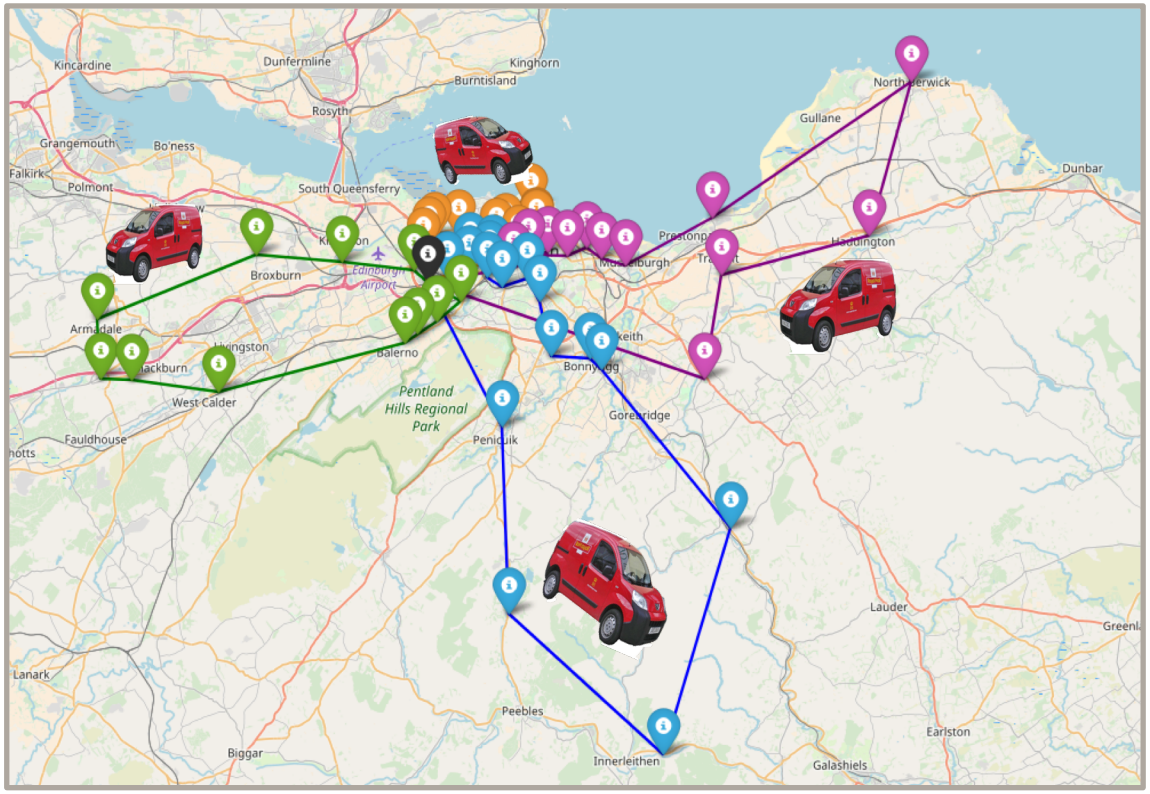
- ✓ **Delivery:** 50 parcels
- ✓ **Max. Vans:** 5
- ✓ **Time Window:** 4h

Goal

- ✓ **Visit every DP only once**
- ✓ **Minimise time & distance**

III. Optimisation Problem

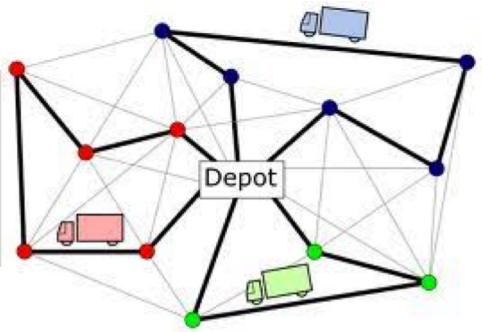
Mock Example



Solution

- ✓ **4 Vans only**
- ✓ **Total Time:** 756 min
- ✓ **Total Distance:** 383 km

III. Vehicle Routing Problems



Solving Vehicle Routing Problem Using Ant Colony

NEURAL COMBINATORIAL OPTIMIZATION WITH REINFORCEMENT LEARNING

2017

Irwan Bello*, Hieu Pham*, Quoc V. Le, Mohammad Norouzi, Samy Bengio
Google Brain

Learning Combinatorial Optimization Algorithms over Graphs

2017

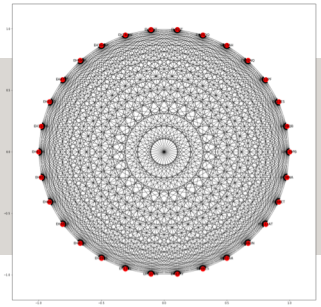
Hanjun Dai*, Elias B. Khalil*, Yuyu Zhang, Bistra Dilkina, Le Song
College of Computing, Georgia Institute of Technology

Deep Reinforcement Learning for Solving the Vehicle Routing Problem

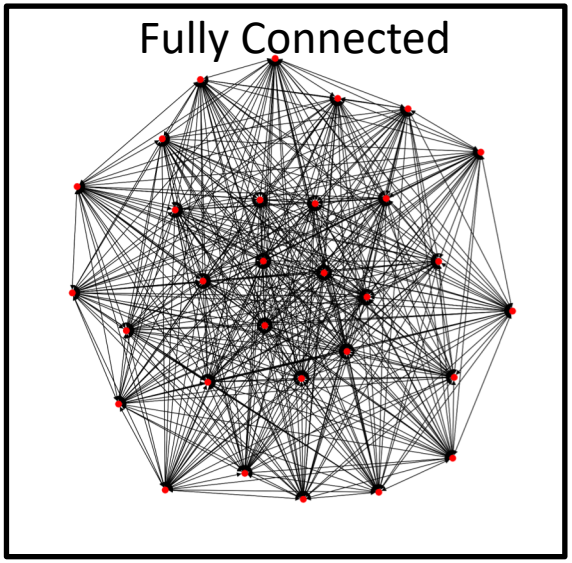
2018

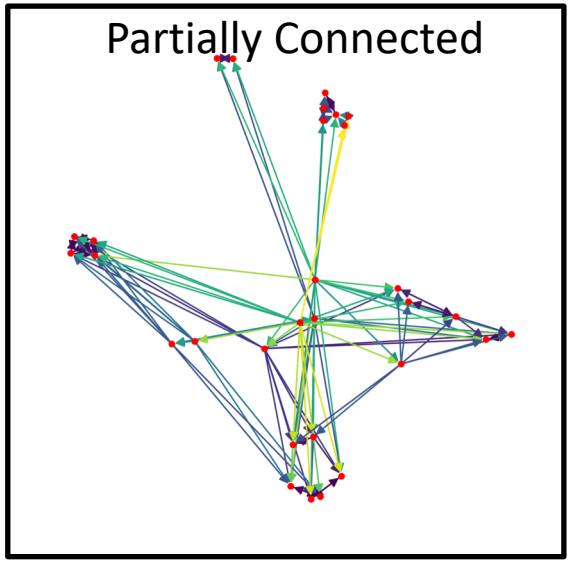
Mohammadreza Nazari, [1] Afshin Oroojlooy, [1] Lawrence V. Snyder, [1] Martin Takáč [1]

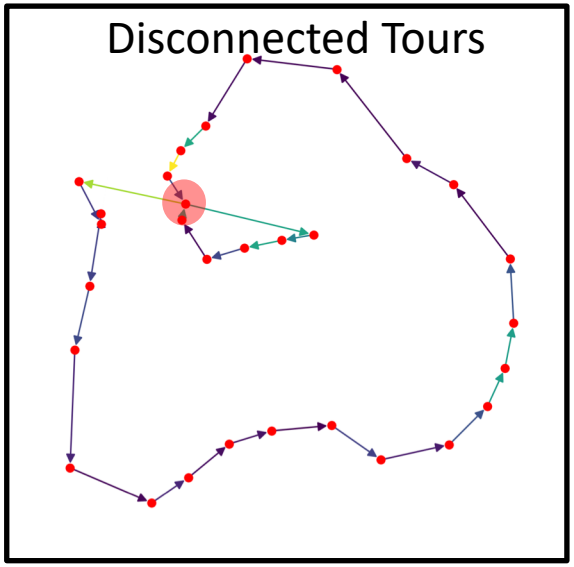
III. Scientific Illustration



From Graph to Routes







- **Heuristics**: Operators on Nodes & Edges
- **Oriented Search** through Solution space

(Graphs taken from solving a real problem)

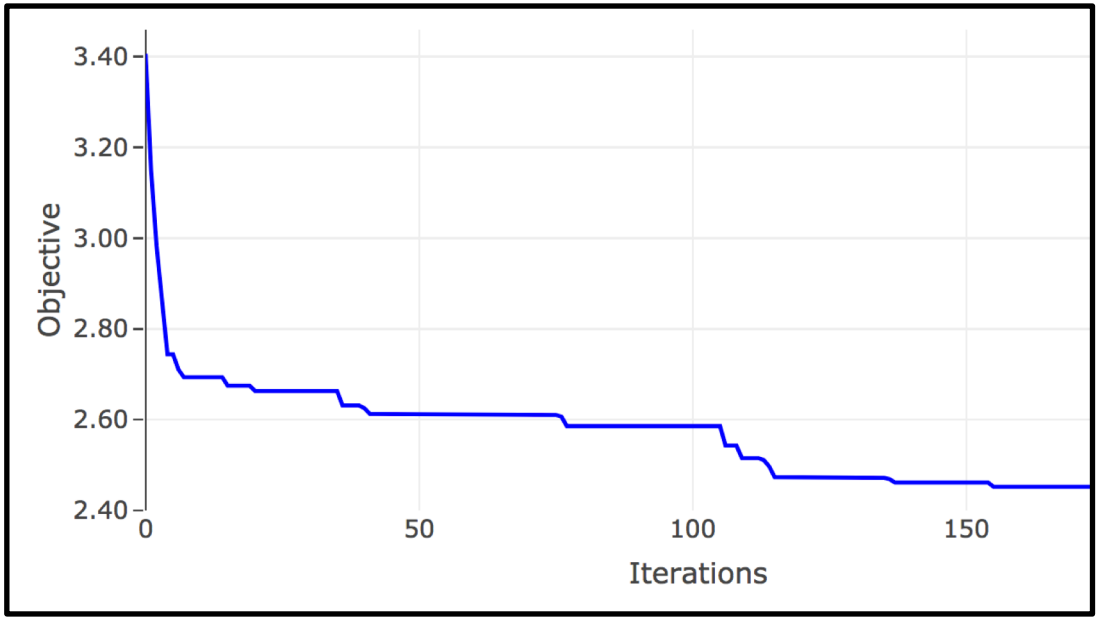
III. Optimisation Objective

Iterative Procedure

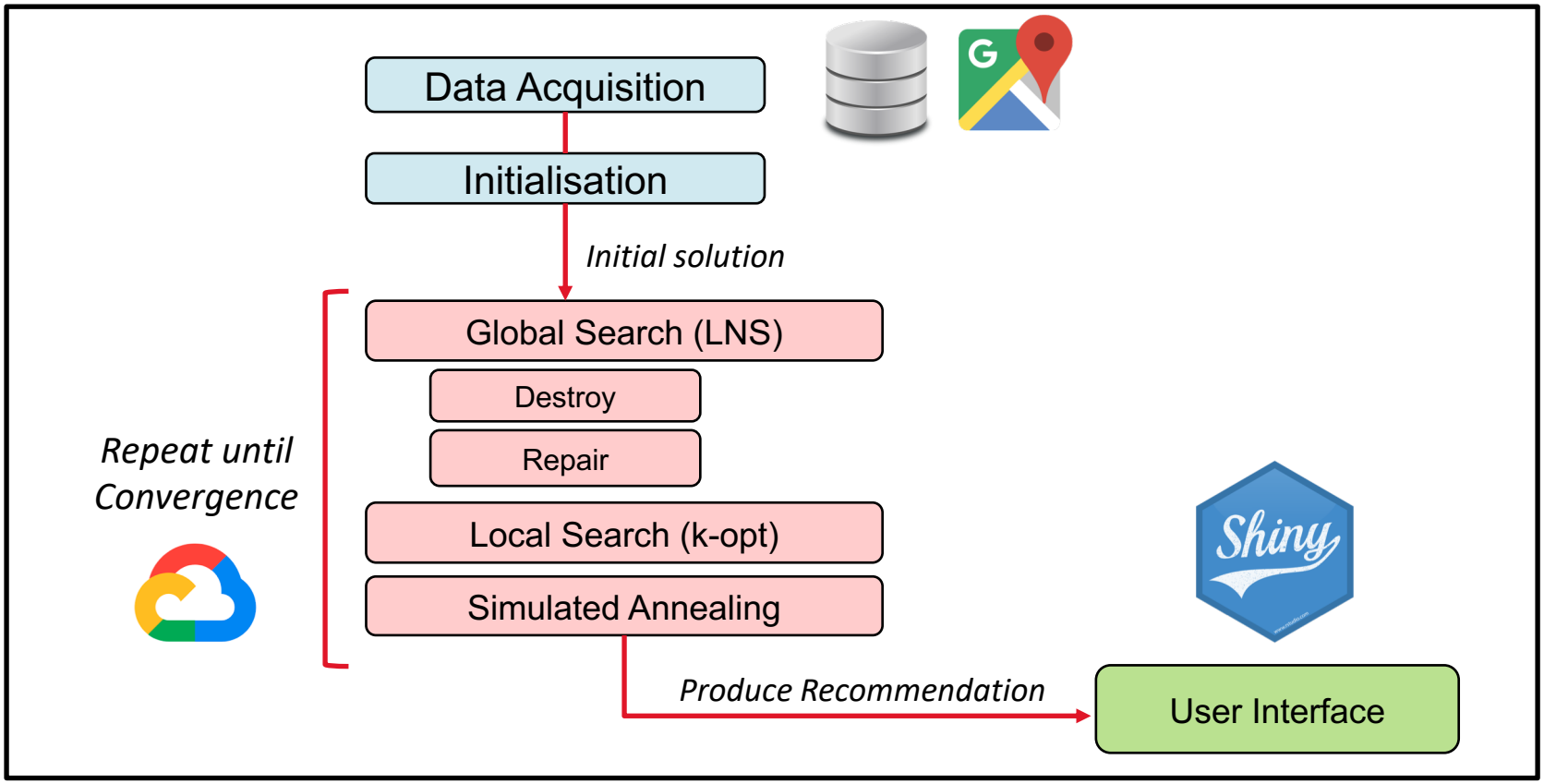
$$Cost = \frac{\omega_T . Time + \omega_D . Distance}{\#Parcels}$$

Business Parameters

- ✓ Hourly Rate: ω_T
- ✓ Kilometric Rate: ω_D



III. Software Architecture



III. Data Acquisition

Delivery Service

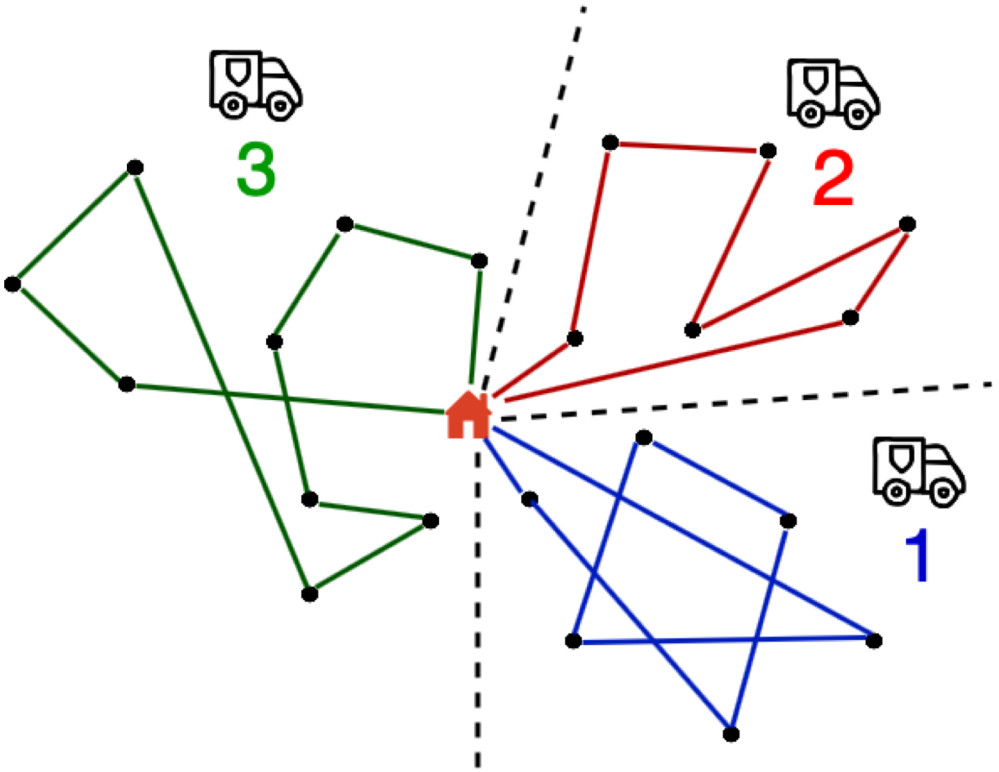
	barcode	postcode	latitude	longitude
0	XXXXX	A	55.939856	-3.0655134
1	XXXXX	B	55.890223	-3.3303288
2	XXXXX	C	55.936345	-3.2533391
3	XXXXX	D	55.960307	-3.1723718
4	XXXXX	E	55.944777	-3.0945816
5	XXXXX	F	55.932246	-3.2797301
6	XXXXX	G	55.896023	-3.3148871
7	XXXXX	H	55.931263	-3.1838993
8	XXXXX	I	55.933932	-2.9493796
9	XXXXX	J	55.946943	-3.509429

Travel Matrix

	Depot	A	B	C	D
Depot			14	9	29
A	27		28	29	22
B	15	25		16	33
C	10	25	16		20
D	31	23	36	22	
E	28	7	30	30	14
F	7	25	15	9	28
G	14	25	3	15	33
H	25	25	26	18	16
I	33	16	35	36	28
J	21	37	27	28	41

III. Initialisation

Ad-hoc Solutions

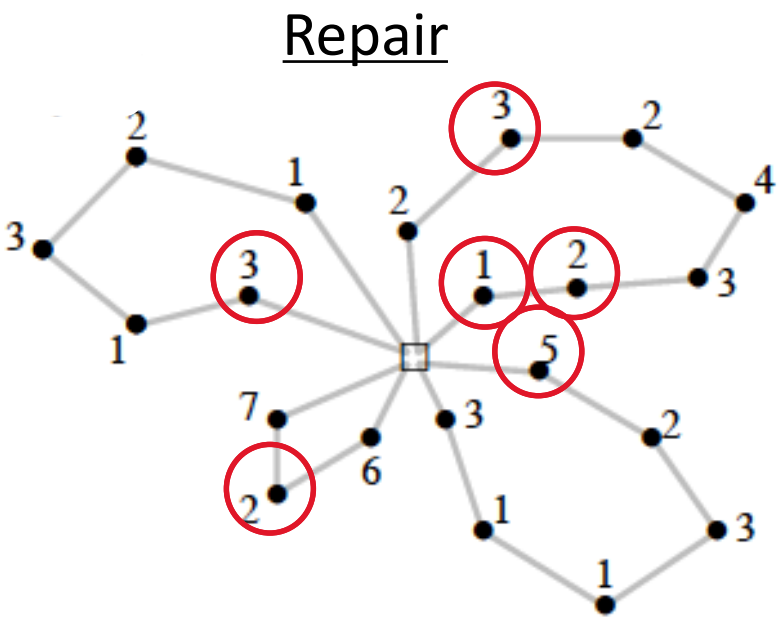


1. ~~**Random Assignment**~~
2. **Round-Robin:** assignment 1 parcel at a time…
3. **Close-First:** assignment 1 van at a time…
4. **Alpha-Num:** assignment per postcode area…

➢ **Choose Cheapest One !**

III. Global Search … Large Neighbourhood

Destroy and Repair Operators



1. **Start from Input Solution**

2. **Destroy a fraction of the Route:**

Partial route + Stash of nodes

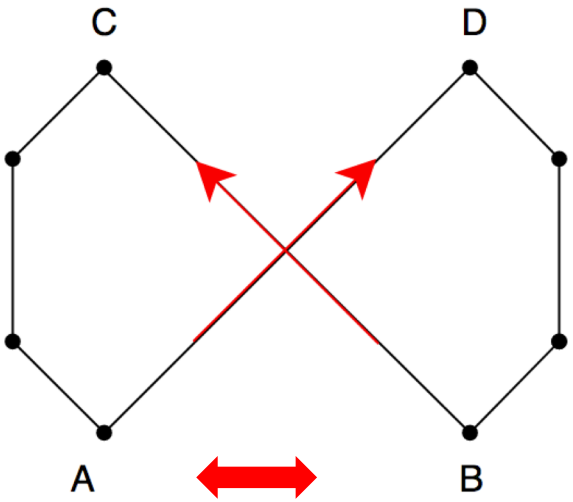
3. **Repair the full Route:**

Re-connect nodes via a Greedy heuristic

III. Local Searches … 2-Opt

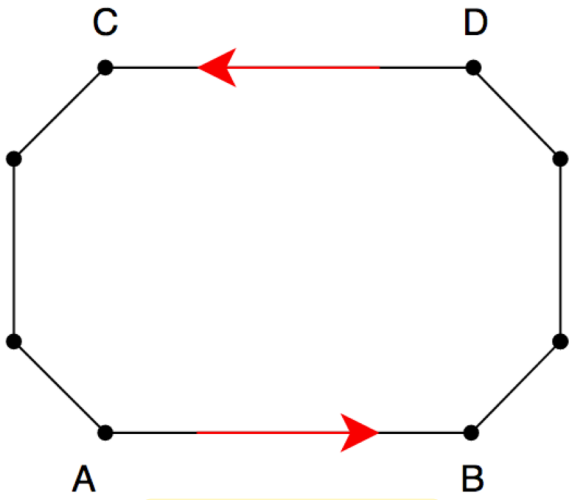
Operators on a Tour

Segment Mirror



A-D-.....-B-C-....A

Cheaper Tour

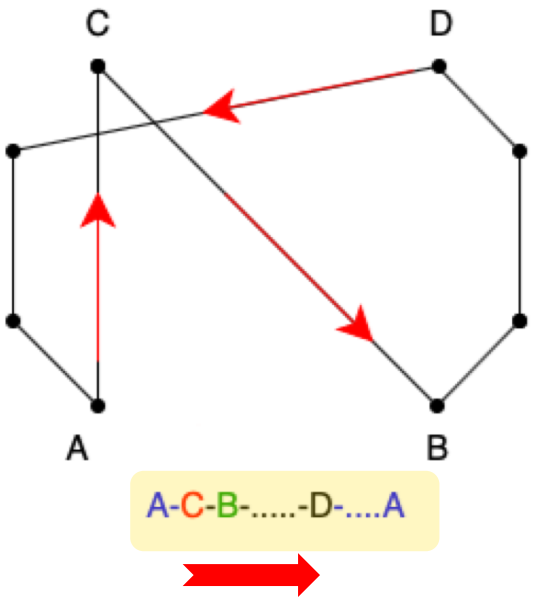


A-B-.....-D-C-....A

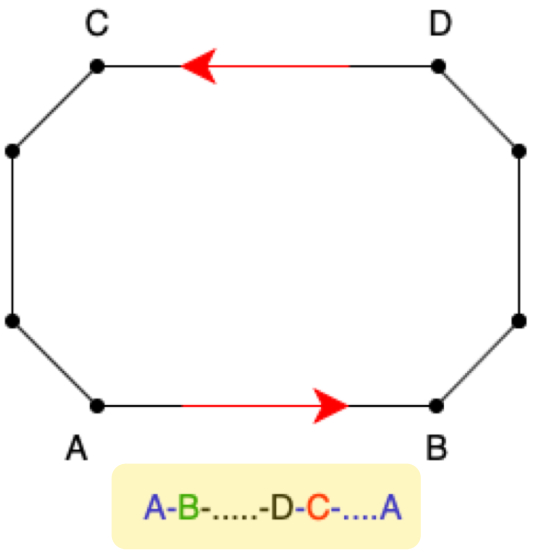
III. Local Searches … 3-Opt

Operators on a Tour

Point Insertion



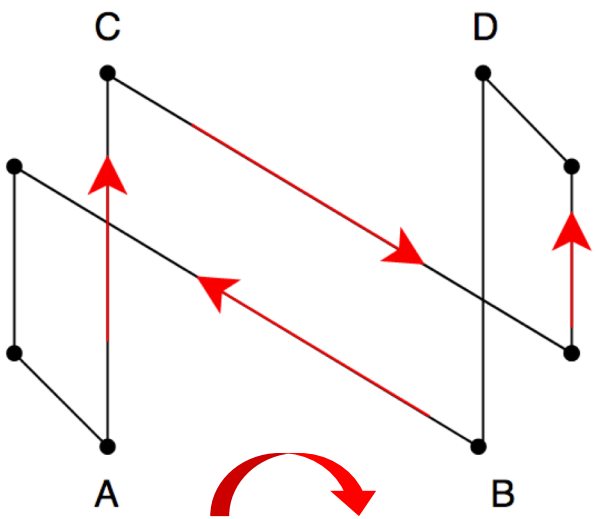
Cheaper Tour



III. Local Searches … 4-Opt

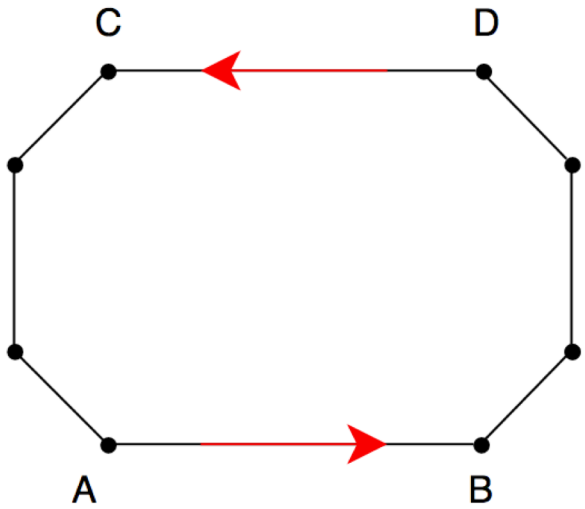
Operators on a Tour

2-Point Swap



A-C-.....-D-B-....A

Cheaper Tour



A-B-.....-D-C-....A

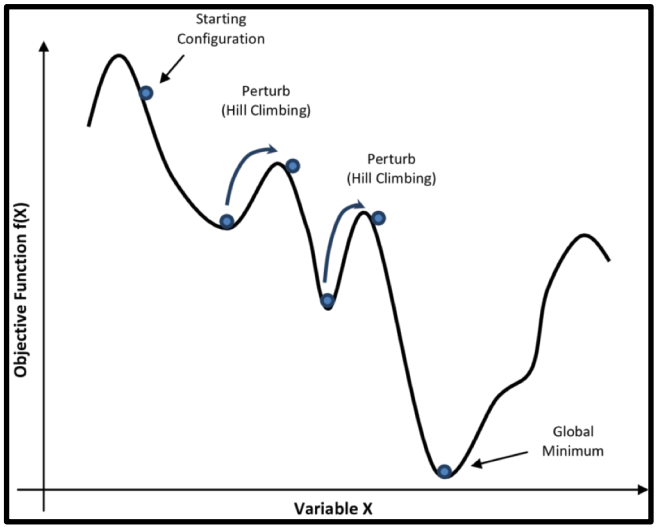
III. Simulated Annealing (SA)

Avoiding Local Optima during Convergence

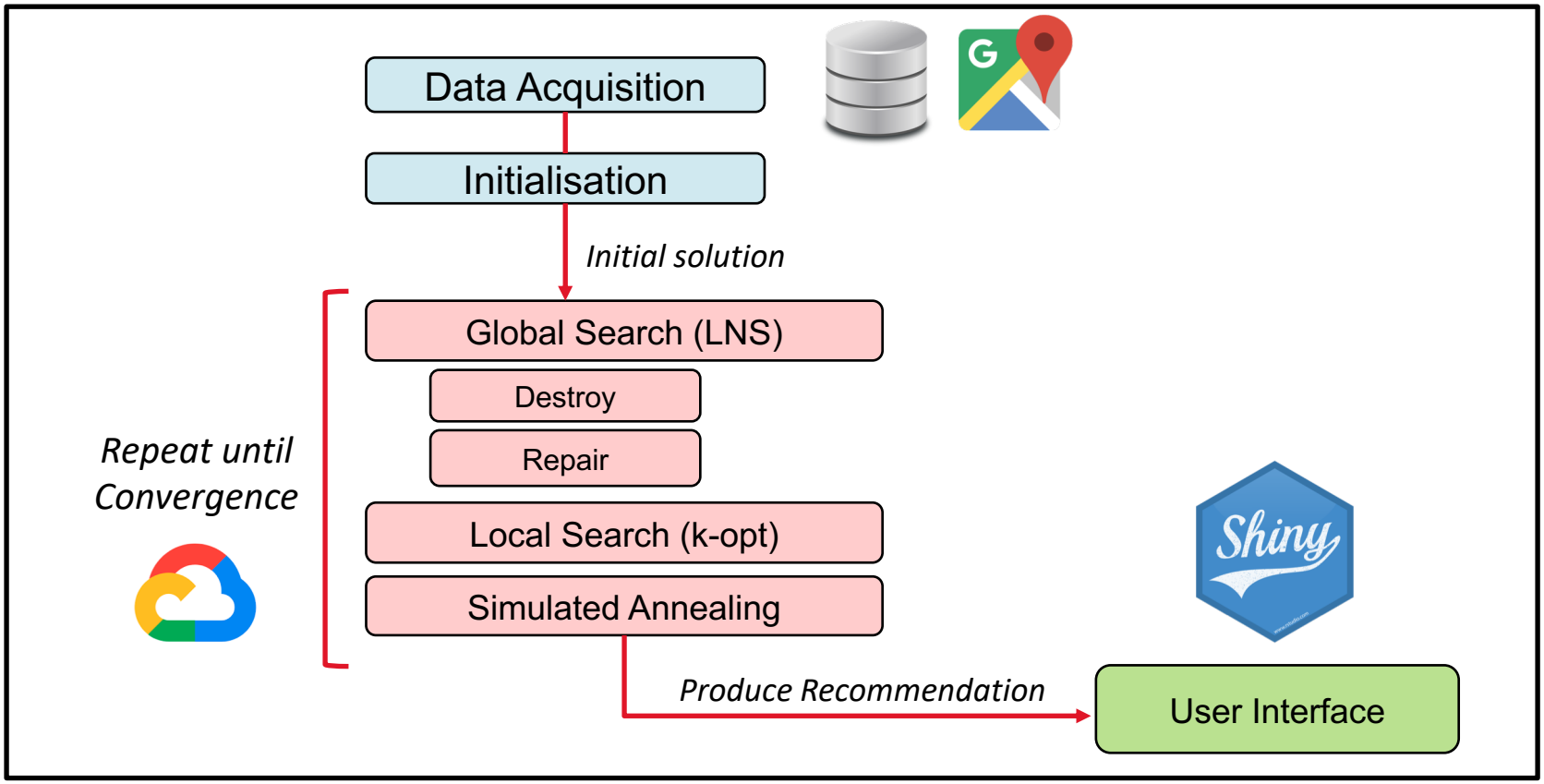
```
apply_annealing(best, input, output):
    update_temperature()
    if cost(output) < cost(best):
        best = output
        input = output
    elif P(out) > rnd(0,1):
        input = output
    return best, input
```

$$Cost = \frac{\omega_T.Time + \omega_D.Distance}{\#Parcels}$$

where, $P(out) = e^{\frac{-[C(out) - C(in)]}{T}}$



III. Software Architecture



III. Deliverable

Sorting Manifest

- Hub level -

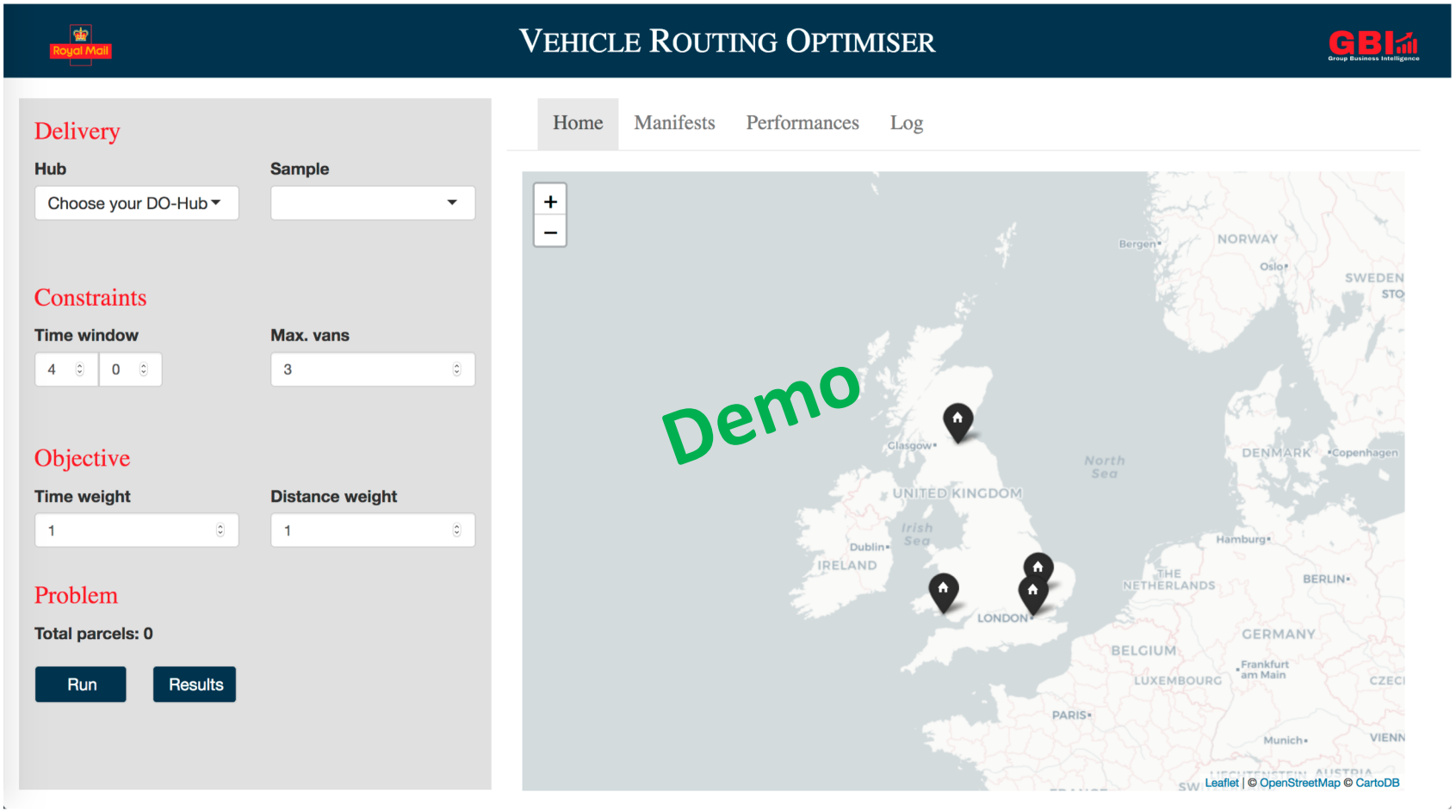
Nodes	Van	Order
A	2	6
B	2	2
C	1	2
D	2	4
E	2	7
F	1	1
G	2	4
H	2	3
I	2	8
J	2	1

Delivery Manifest

- Van level -

Van	#1	#2
Drop - 1	F	J
Drop - 2	C	B
Drop - 3		H
Drop - 4		G
Drop - 5		D
Drop - 6		A
Drop - 7		E
Drop - 8		I

III. Visualisation



IV. Ongoing Investigations

Current Limitations

- Operation Feasibility: needing input from front-line
- Hard *vs* Soft constraints: ensuring stability of service

Future work

- Engineering: reproducibility, parallelisation, scaling…
- Setting-up Parameters: adaptive learning, off-line learning
- Adds-on: local time-window, capacity constraints, collections…

IV. Questions ?

Thank You …