Capacitated Vehicle Routing Problem with Time Windows

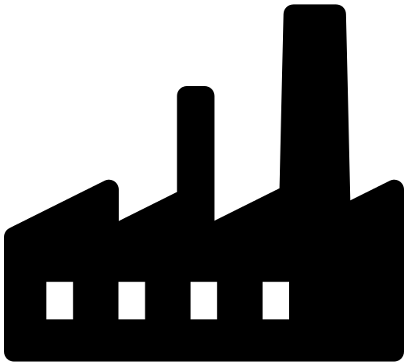
W. David Fröhlingsdorf

University of Glasgow | School of Computing Science

Capacitated Vehicle Routing Problem *with* Time Windows

CVRPTW

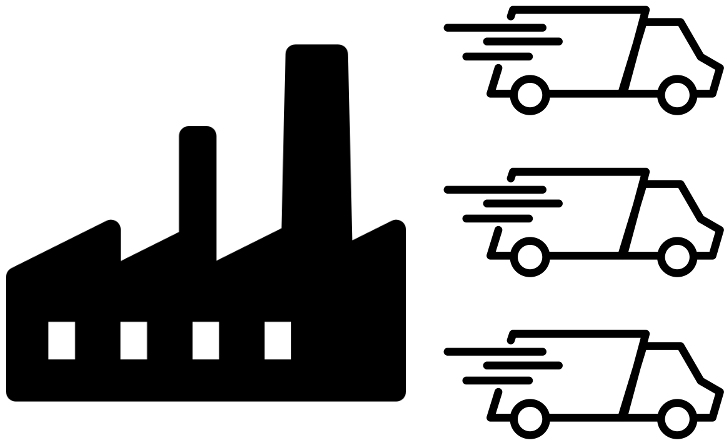
Central Depot



CVRPTW

Central Depot

Vehicle Fleet



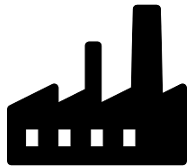
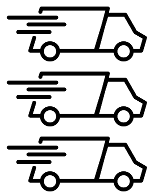
CVRPTW

Central Depot

Vehicle Fleet

Customers



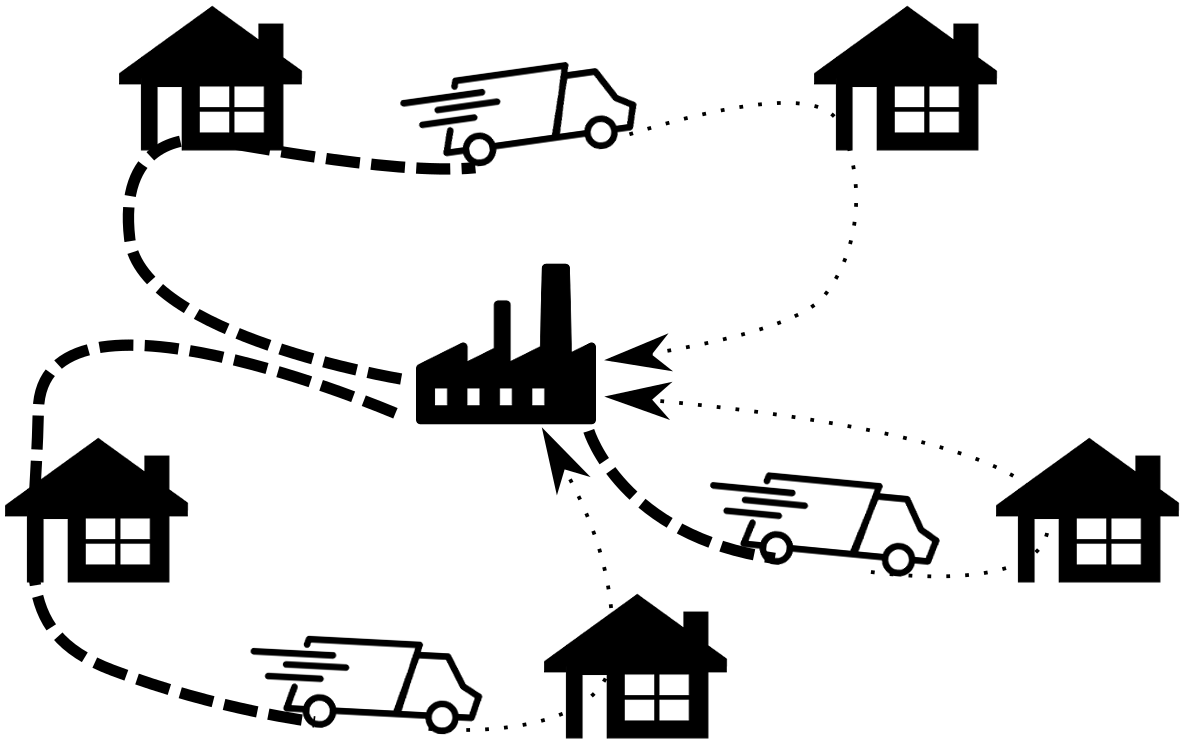




CVRPTW

Central Depot

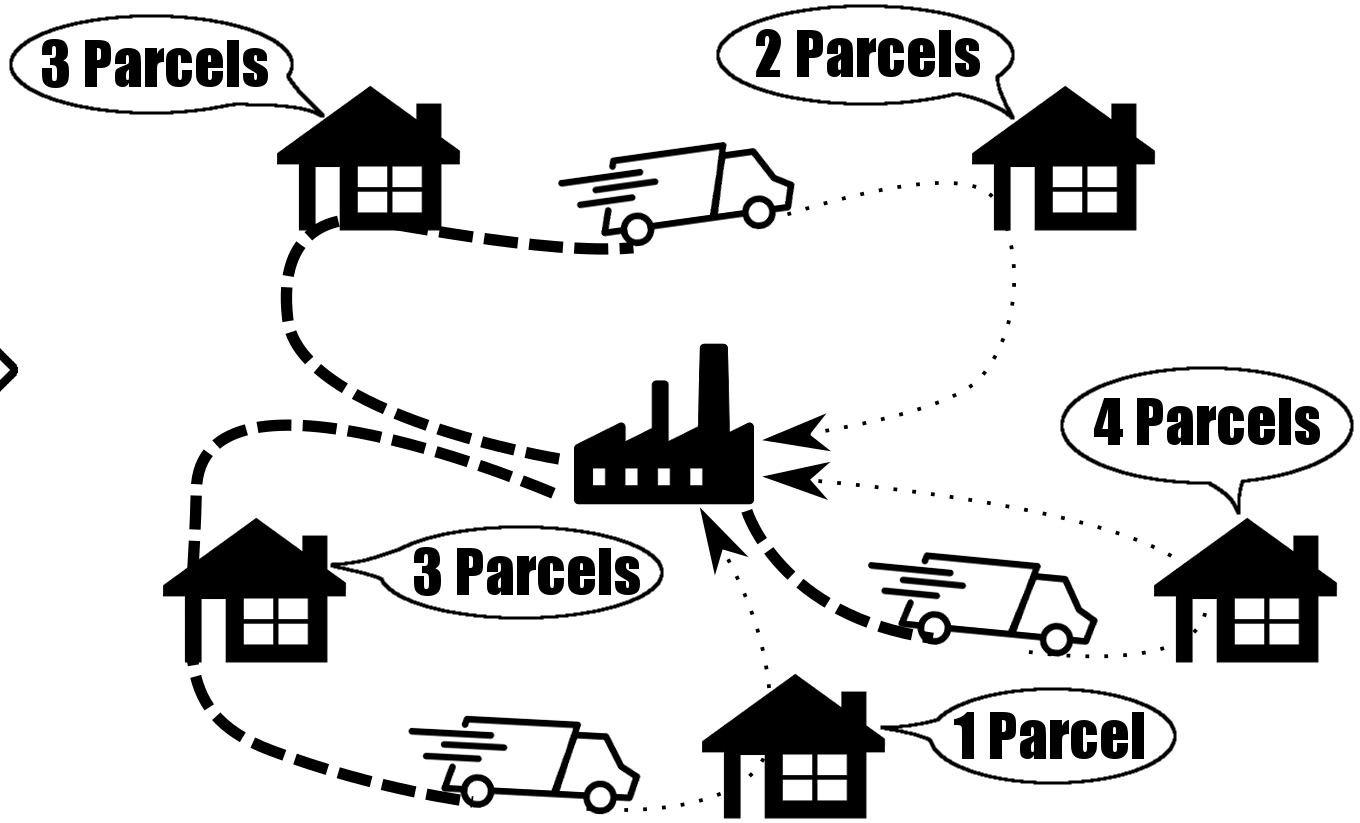
Vehicle Fleet

Customers



CVRPTW

- Central Depot
- Vehicle Fleet
- Customers
- Demand



CVRPTW

- Central Depot
- Vehicle Fleet
- Customers
- Demand
- Capacity

CVRPTW

Central Depot

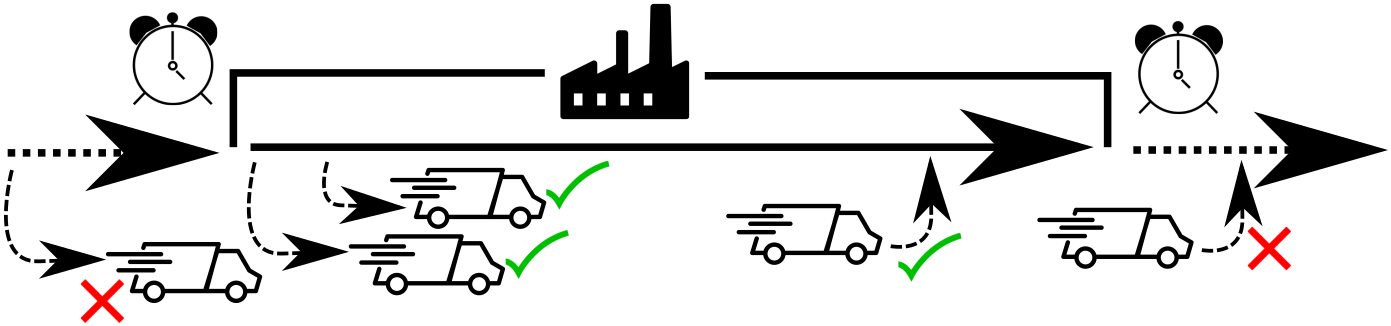
Vehicle Fleet

Customers

Demand

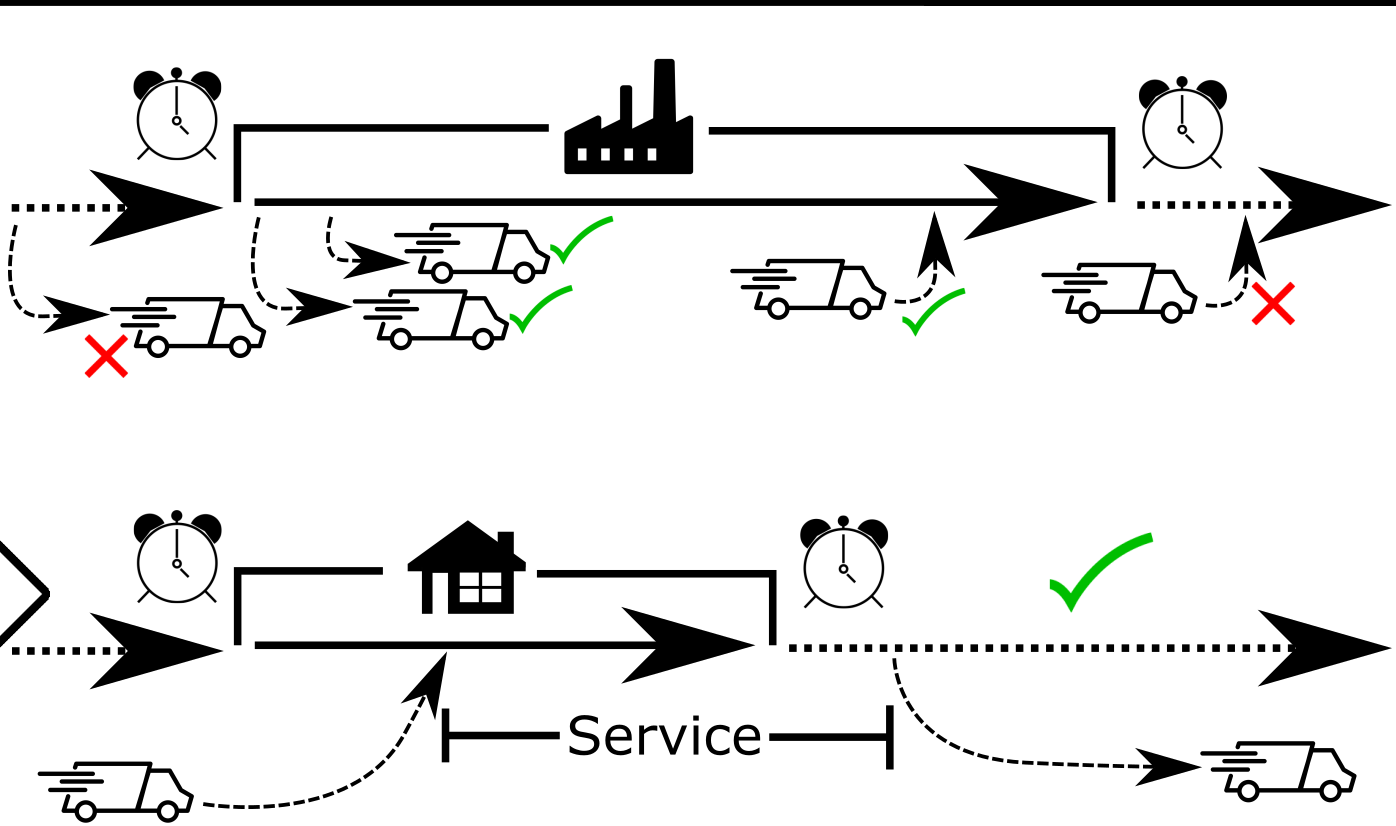
Capacity

Time Windows



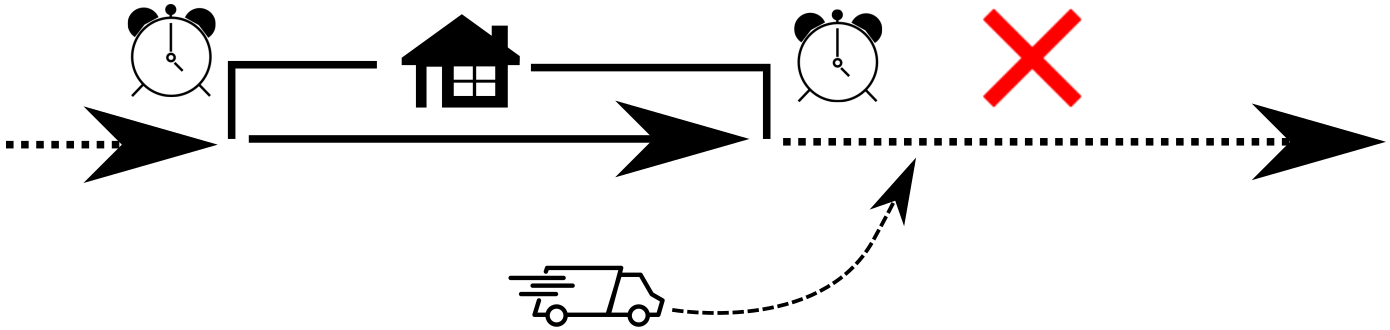
CVRPTW

- Central Depot
- Vehicle Fleet
- Customers
- Demand
- Capacity
- Time Windows



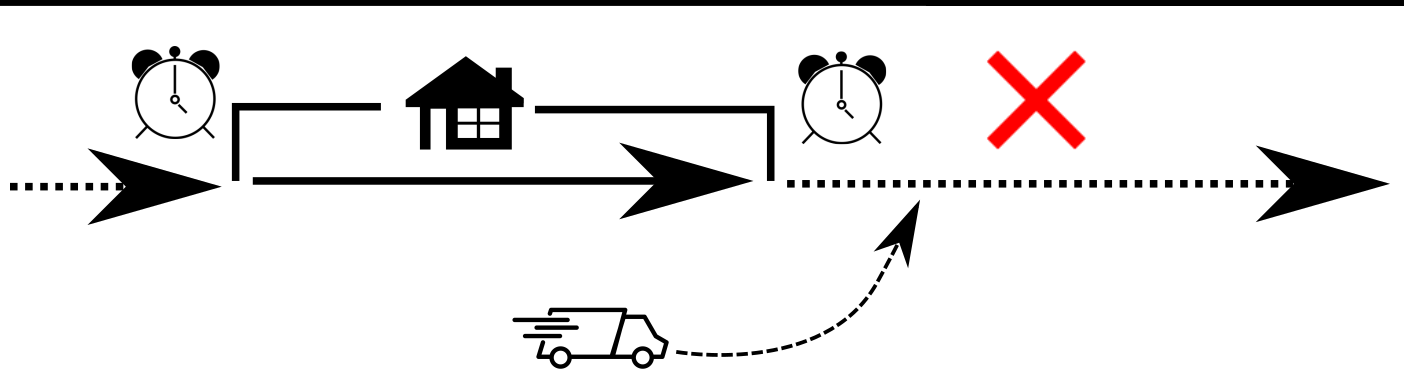
CVRPTW

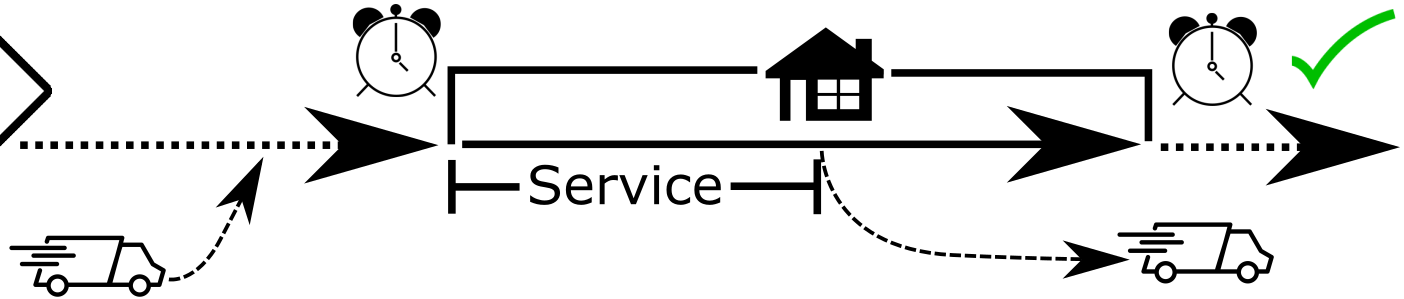
- Central Depot
- Vehicle Fleet
- Customers
- Demand
- Capacity
- Time Windows



CVRPTW

- Central Depot
- Vehicle Fleet
- Customers
- Demand
- Capacity
- Time Windows





CVRPTW

Central Depot
Vehicle Fleet
Customers
Demand
Capacity
Time Windows
Objective

Minimise the total distance travelled by all vehicles combined.


Const✓aint
Pr×gramming

Decision Variables

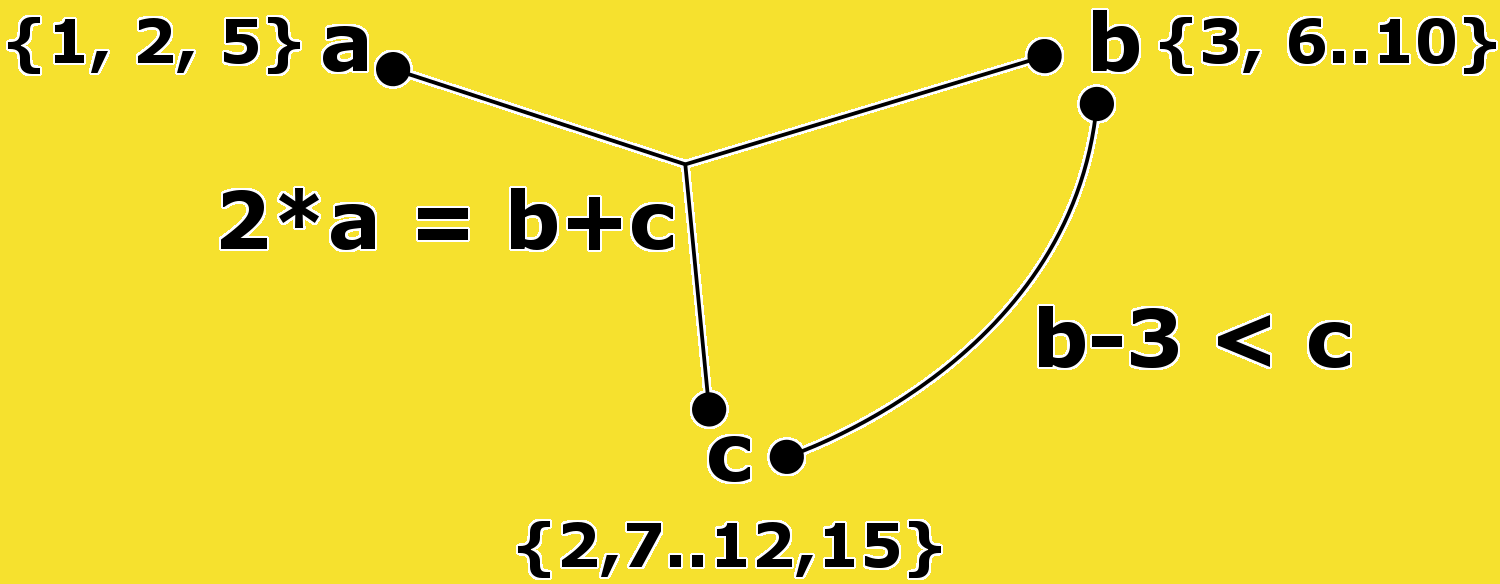
{1, 2, 5} a

b {3, 6..10}

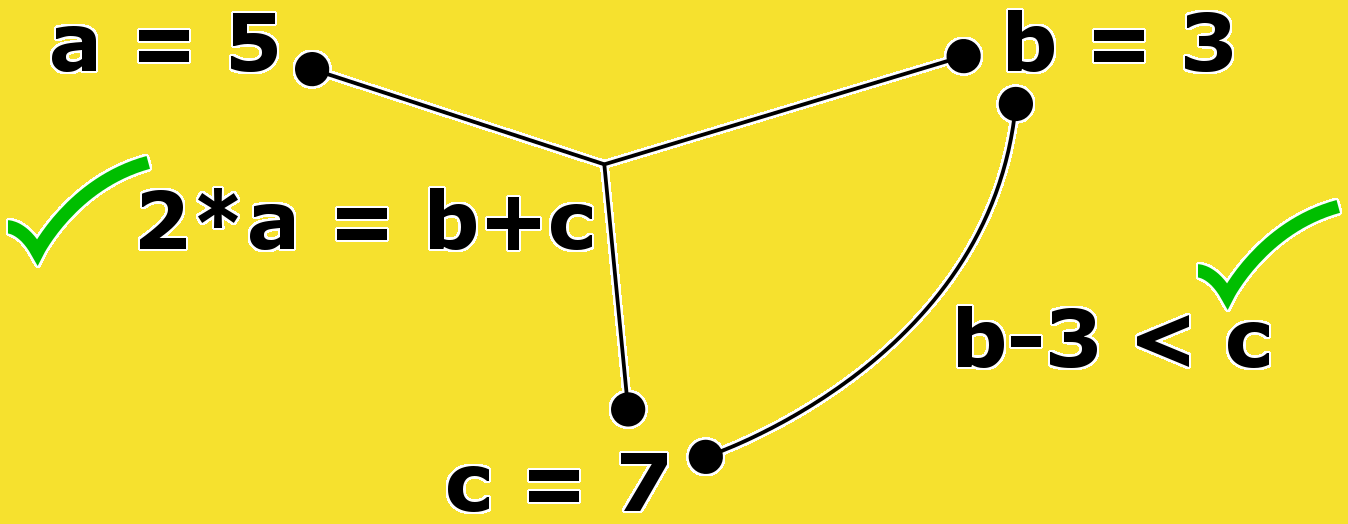
c

{2,7..12,15}

Constraints

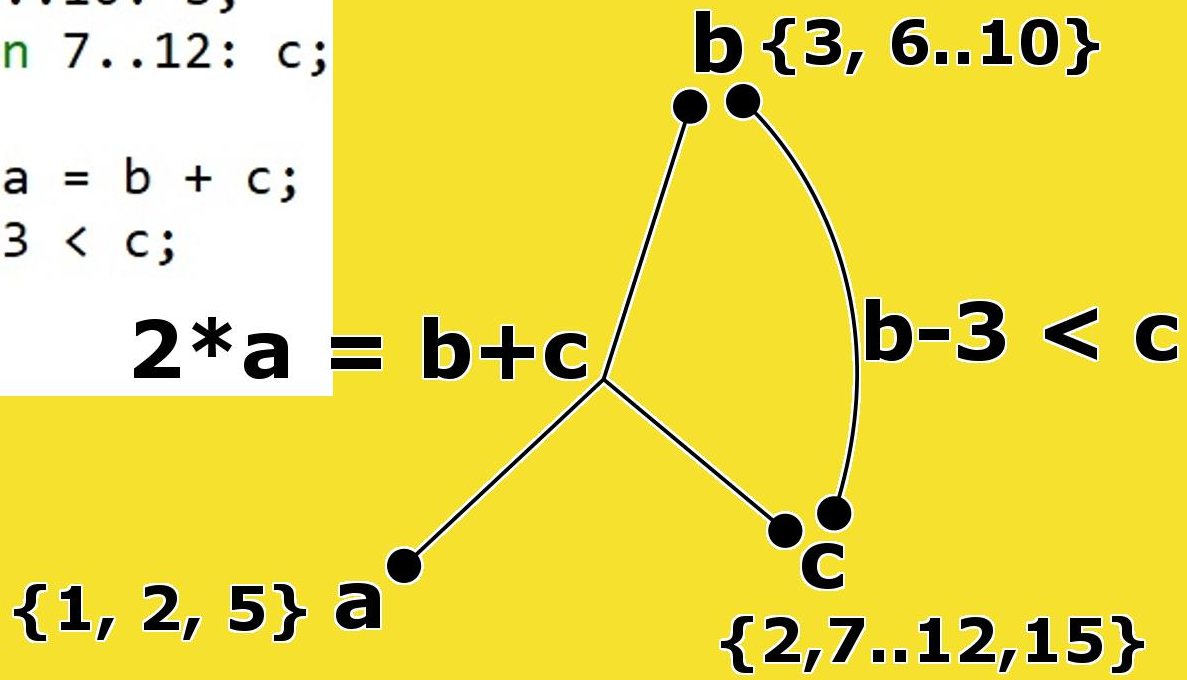


Valid Solution



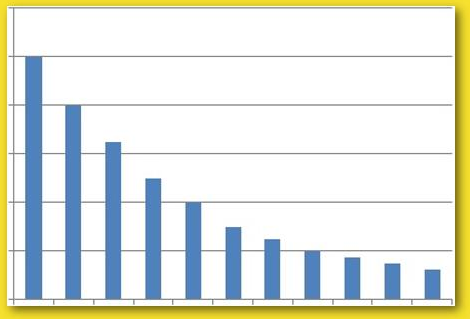
MiniZinc - Source Code

```
1 var {1,2,5}: a;
2 var {3} union 6..10: b;
3 var {2,15} union 7..12: c;
4
5 constraint 2 * a = b + c;
6 constraint b - 3 < c;
7
8 solve satisfy;
```



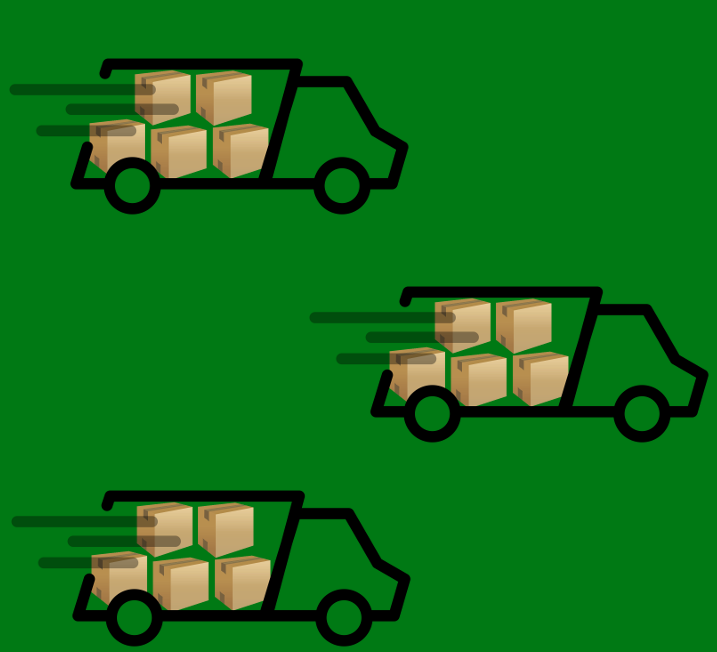
Optimisation

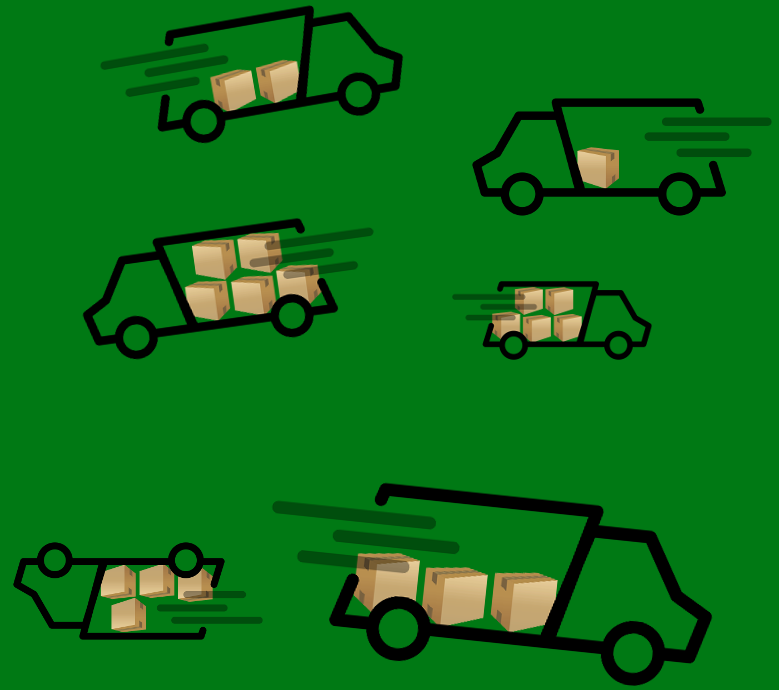
```
1 solve minimize total_distance;
```



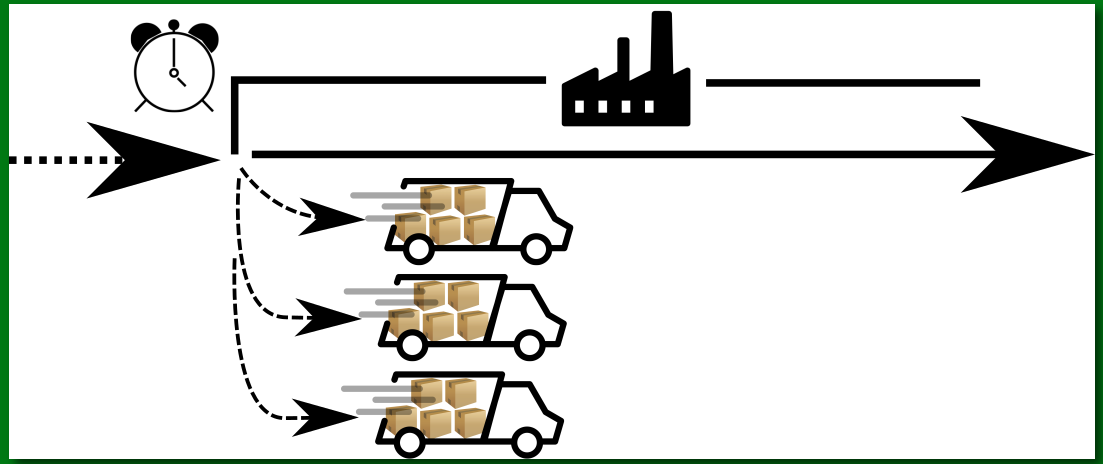
MODEL 2

Homogeneous Fleet VS Heterogeneous Fleet



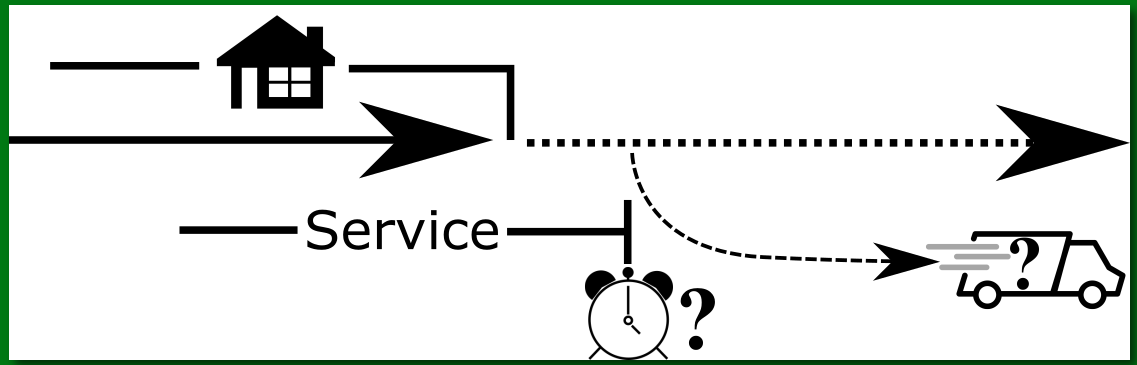


All used vehicles leave the depot



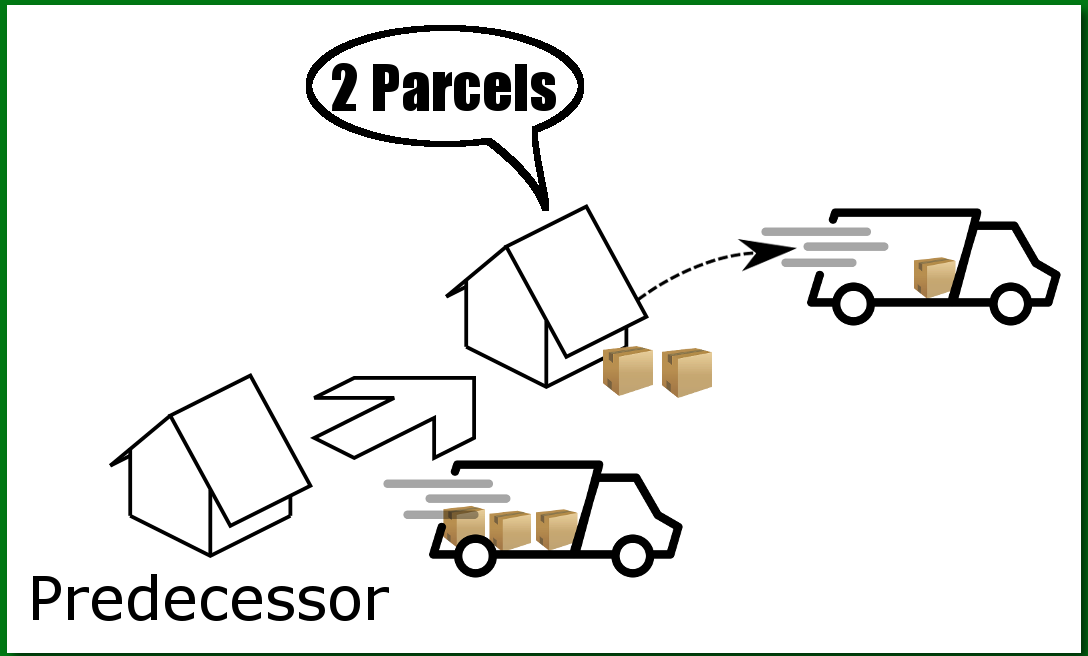
- as early as possible
- fully loaded

What is the vehicle's

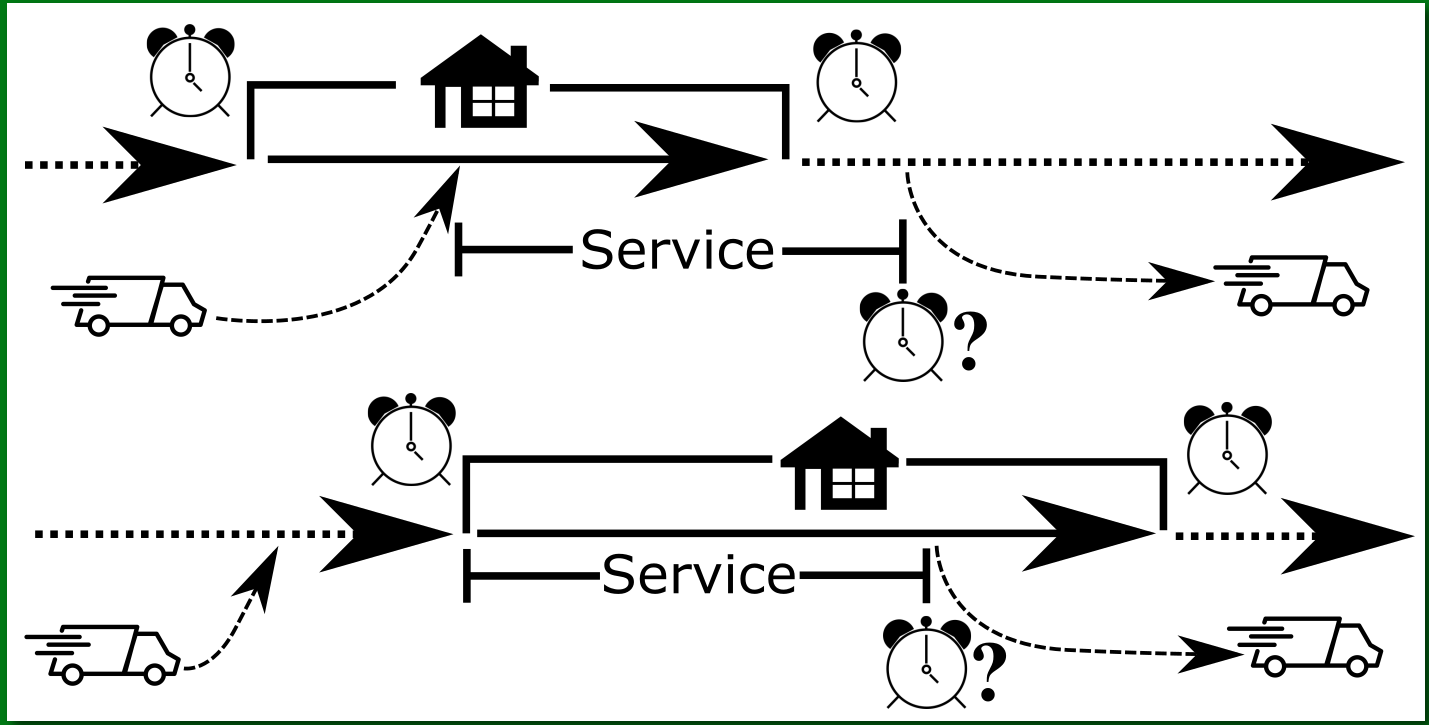


- leaving time
- remaining load

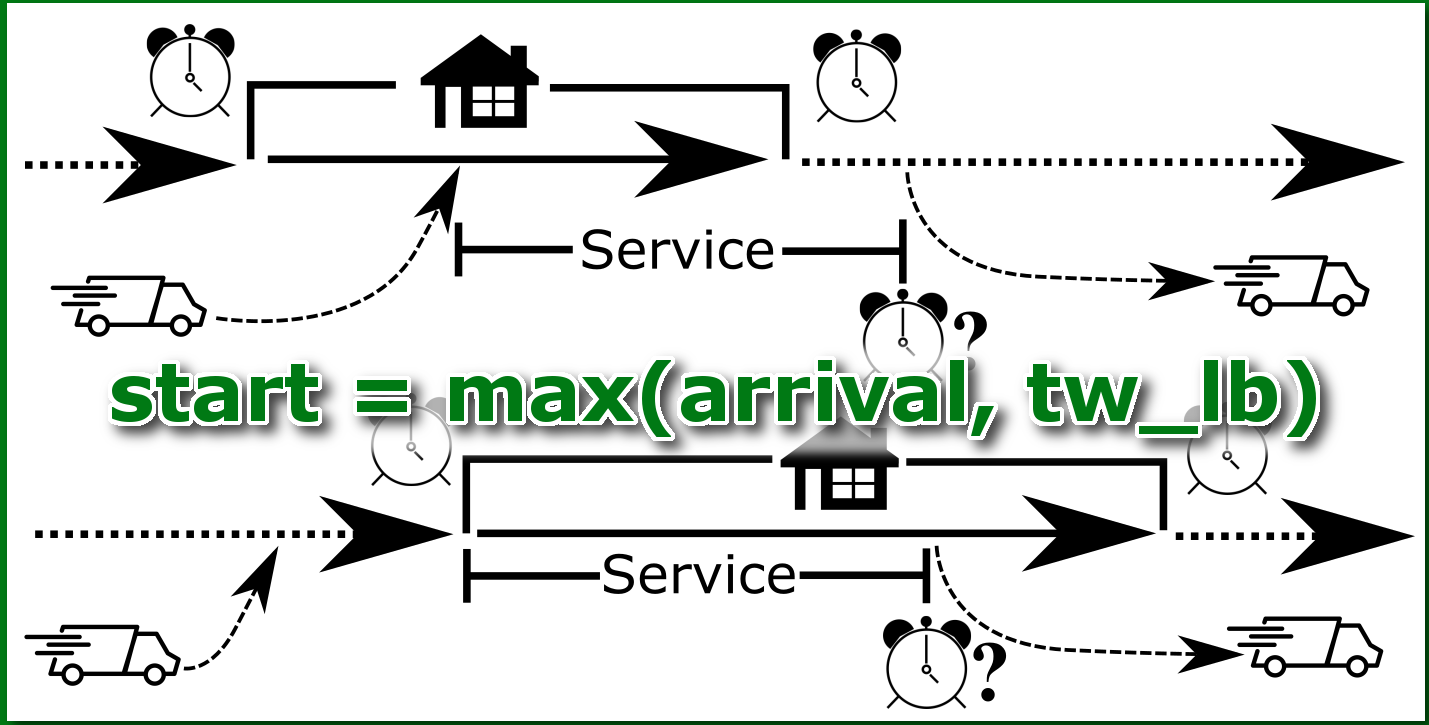
Load Propagation



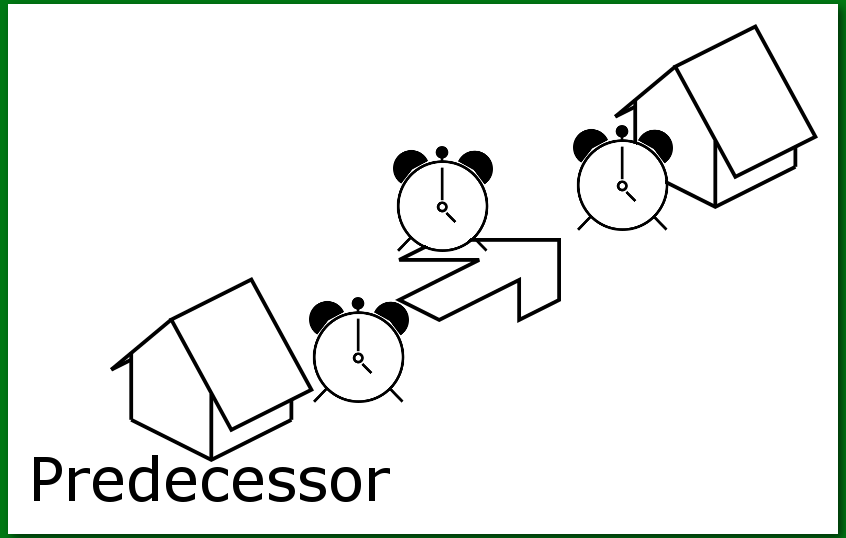
Service Start Time



Service Start Time

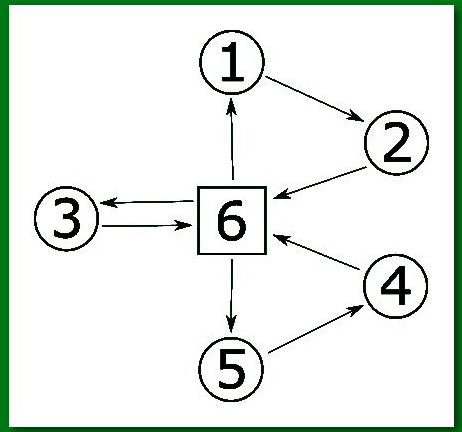


Arrival Time

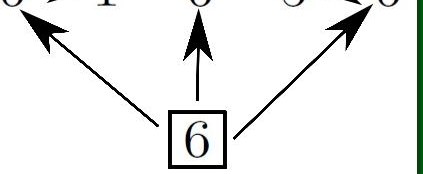


arrival = leave(pred) + distance

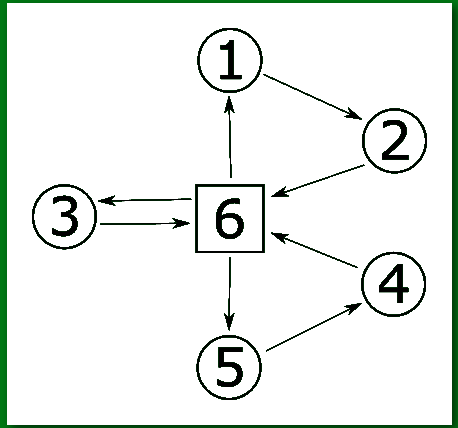
Predecessor Array



Customer	1	2	3	4	5
pred	6	1	6	5	6



Predecessor Array



Customer	1	2	3	4	5
pred	6	1	6	5	6
last	0	1	1	1	0

Objective

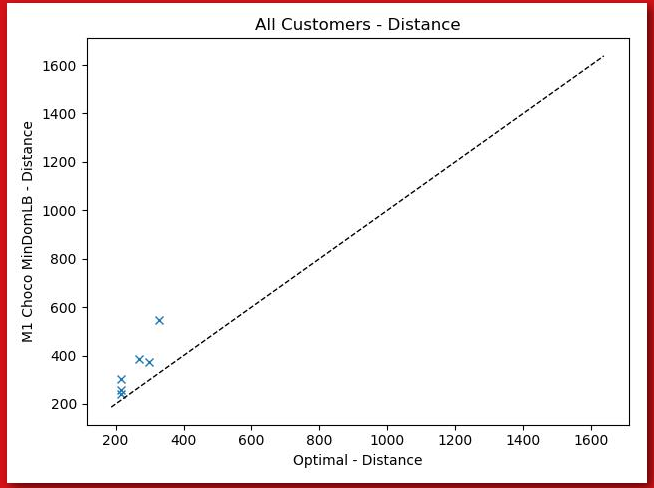
```
solve minimize sum(c in CUSTOMER)
      (
          dist[pred[c],c]
          +
          dist[c, DEPOT] * last[c]
      );
```

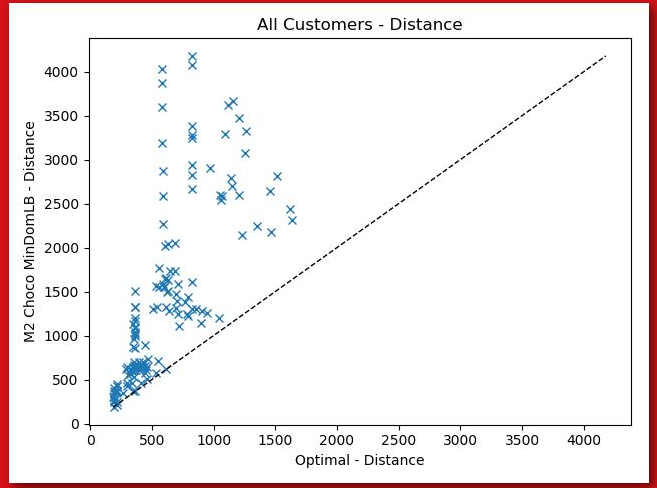
outcome /ˈaʊtkʌm/

The way a thing turns out

Oxford Dictionary

Model 1 VS Model 2

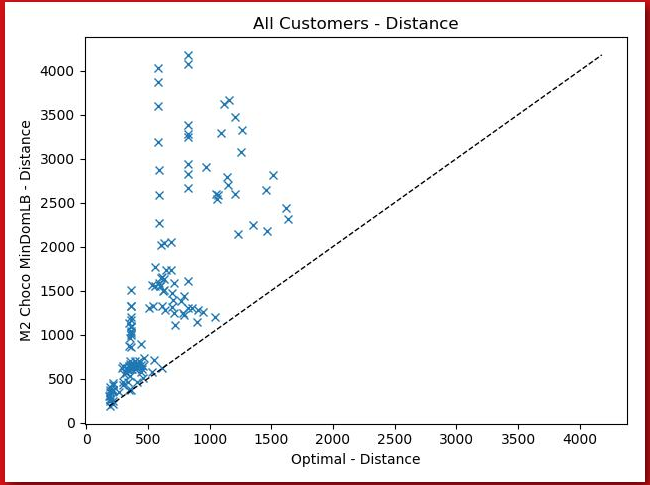


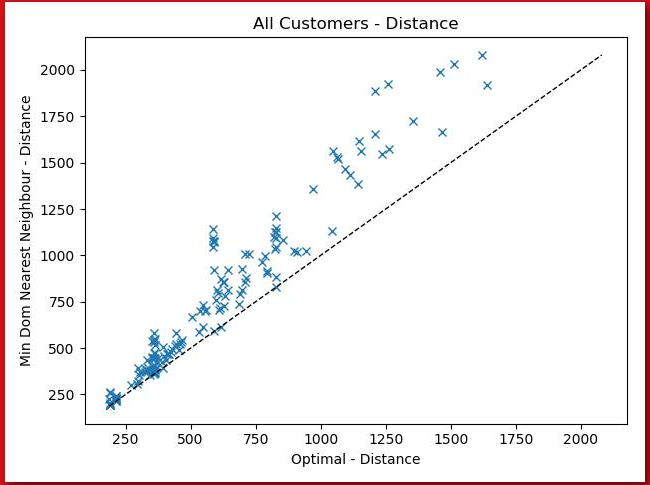


Smallest Value

VS

Nearest Neighbour





Thank you!

Questions?

University of Glasgow | School of Computing Science