Introduction

Objective function

$$Min\ WRW = \sum_{j=1}^{m}\sum_{i=1}^{n+1}\left[\left(\sum_{k=1}^{n+1} w_{k_j} - \sum_{k=1}^{i-1} w_{k_j}\right) \times Dist(Loc_i, Loc_{i-1})\right]$$

m : the number of trips

n : the number of gifts for each trip j

$\sum_{k=1}^{n+1} w_{k_j} - \sum_{k=1}^{i-1} w_{k_j}$: the remaining weight of gifts and sleigh after delivering a gift

Loc_i: location of gift i

$Dist(Loc_i, Loc_{i-1})$: Haversine Distance between two locations Loc_i and Loc_{i-1}

10 < total weight capacity per trip < 1010

Dataset (problem size):

 $\times 10$

 $\times 100$

 $\times 1000$

Algorithm1: Random Search

- **Number of iterations:** 5000 iterations for datasets N=10 and N=100; 1000 iterations for dataset N=1000
- **Times run:** the algorithm was run 30 times for each dataset
- **Calculation of WRW:** random combination of trips and gifts

- **The Result of Algorithm1:**

	Itr = 5000		Itr = 1000
WRW	N =10	N =100	N =1000
Minimum	1410694.79	23517666.19	293013207.5
Maximum	3841349.84	32933004.2	336138949.5
Mean	2365071.49	28261115.95	310811367
Standard deviation	616983.68	2496912.05	10003972
Coefficient of Variation (CV)	0.260873163	0.088351502	0.032186635
Total time spend(sec)	912.03	8717.56	17613.92

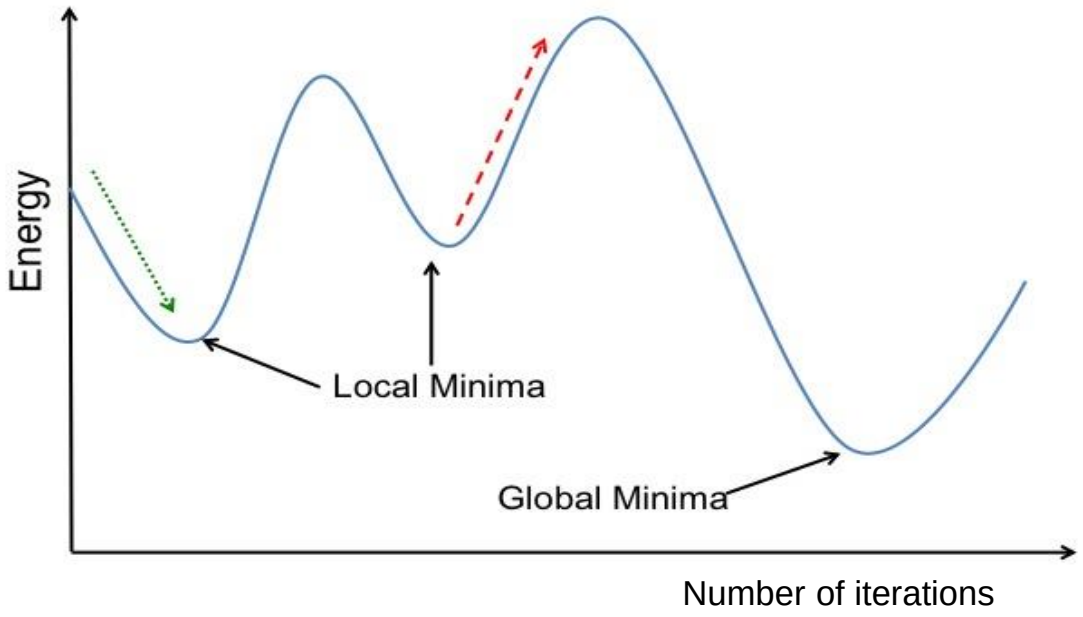
- **Observations:**

✓ Other algorithms may be better since standard deviation and mean of WRW is quite large in all datasets

✓ WRW will increase if we use a big dataset.

Introduction of Simulated Annealing(SA)

- Simulated Annealing(SA) is a form of optimization based on the slow cooling of a material
- **Aim:** avoid staying in local minima and try to find global minima by using the function



- In our case, the new solution could be accepted in one of the following cases:

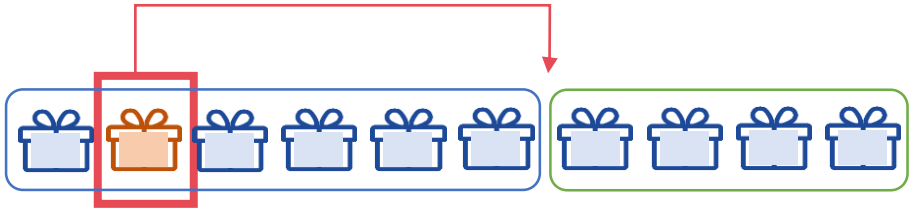
1. The new solution is better than the original one
2. random(0,1) < exp(1/T·($WRW(s)$−$WRW(s')$)/$WRW(s)$)

s = random search solution
s' = new solution
T = temperature
α = temperature decay
(Larger T leads to high probability accepting other solutions)

Algorithm2: Simulated Annealing with neighbourhood move 2

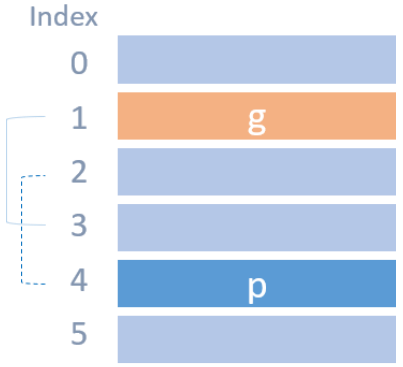
- **Number of iterations:** 5000 iterations for datasets N=10 and N=100; 1000 iterations for dataset N=1000
- **Times run:** the algorithm was run 30 times (different random seed) for each dataset
- **Calculation of WRW:** combining SA with NM2, which applies an arbitrary movement of one gift within a trip

NM2:



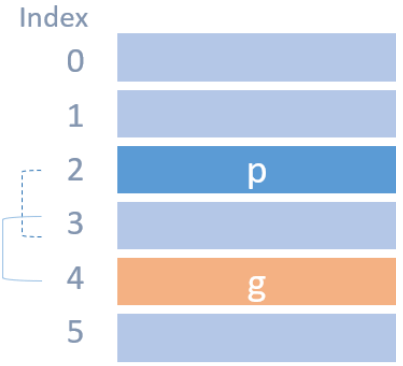
< Nm2 function >

- Within the chosen trip, randomly select a gift (g) and a position (p).
- Change the position of the chosen gift to the newly assigned position.



If g < p :

```
tmpA = df.iloc[ g+1 : p+1 ]
df.iloc[ p ] = df.iloc[ g ]
df.iloc[ g : p ] = tmpA
```



If g > p :

```
temp = df.iloc[ p : g ]
df.iloc[ p ] = df.iloc[ g ]
df.iloc[ p+1 : g+1 ] = tmpB
```

If g = p : sample a random trip again

Algorithm3: Simulated Annealing with neighbourhood move 2 and 6

- **Number of iterations:** 5000 iterations for datasets N=10 and N=100; 1000 iterations for dataset N=1000
- **Times run:** the algorithm was run 30 times (different random seed) for each dataset
- **Calculation of WRW:** combining SA with NM2 and NM6, which apply a three-way suffix to the selected three trips.

NM6:

< Nm6 function >

Randomly select 3 trips and pick 1 gift in each of the 3 selected trips to split each trip into two dataframe for a three-way swap.

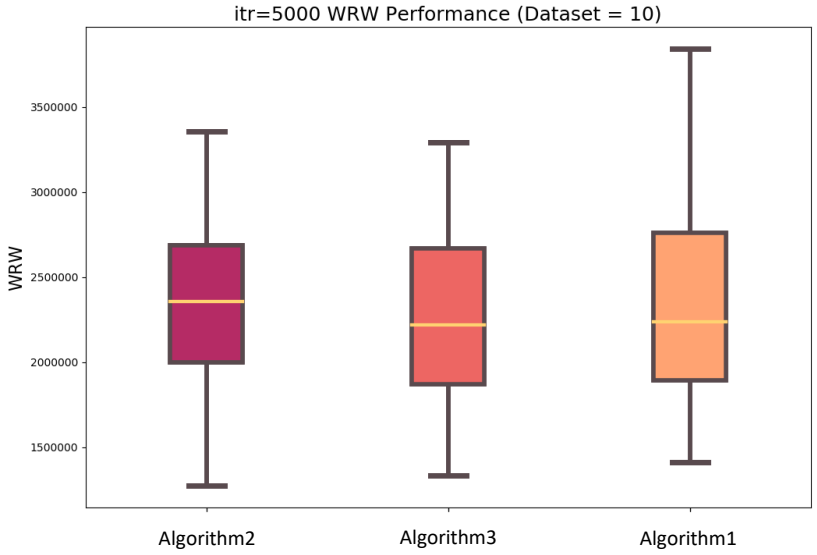
- If the number of the trip less than 3, we return the original dataframe.

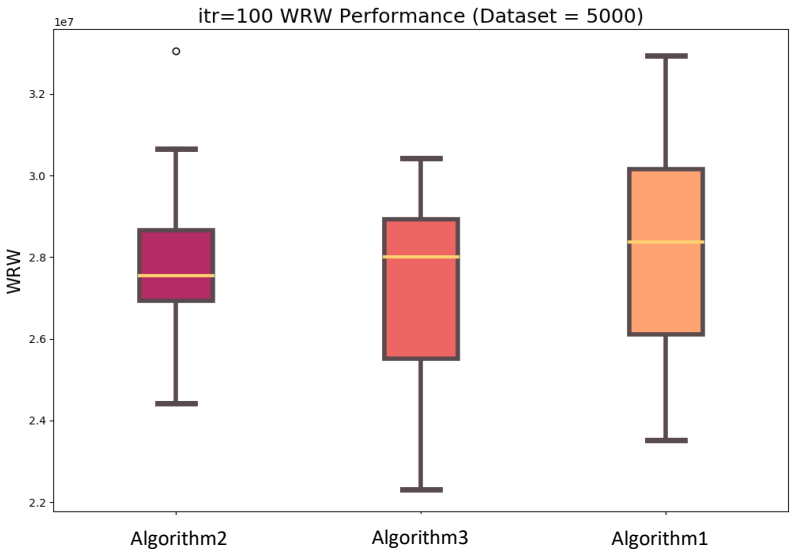
```
if len(df["TripId"].unique()) <= 2:
    return(df)
elif len(df["TripId"].unique()) >= 3:
```

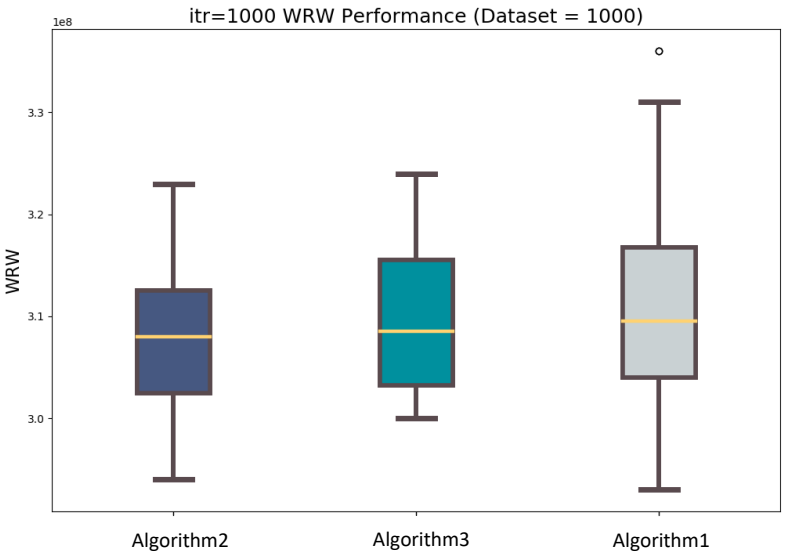
- After selecting three trips, we should also meet the weight limitation; otherwise, three new trips will be selected again.

```
for i in range(3):
      if sum(new[i].Weight) > weight_limit:
            break
      else:
            for i in range(3):
                  new[i].TripId = rd_trip[i]
```

Visualisation of Algorithm1, 2 and 3







- According to these boxplots, **Algorithm2** indicates the lowest standard deviation among all algorithms in three datasets.
- In dataset N=1000, **Algorithm2** shows the lowest minimum WRW and median.

make a better promise in producing lowest WRW value

Santa should use Algorithm2 in the future