Travel Route Planner using GA and solving TSP

Group 12

2019145081 Taejun Kang

2020147049 Hyundong Kim

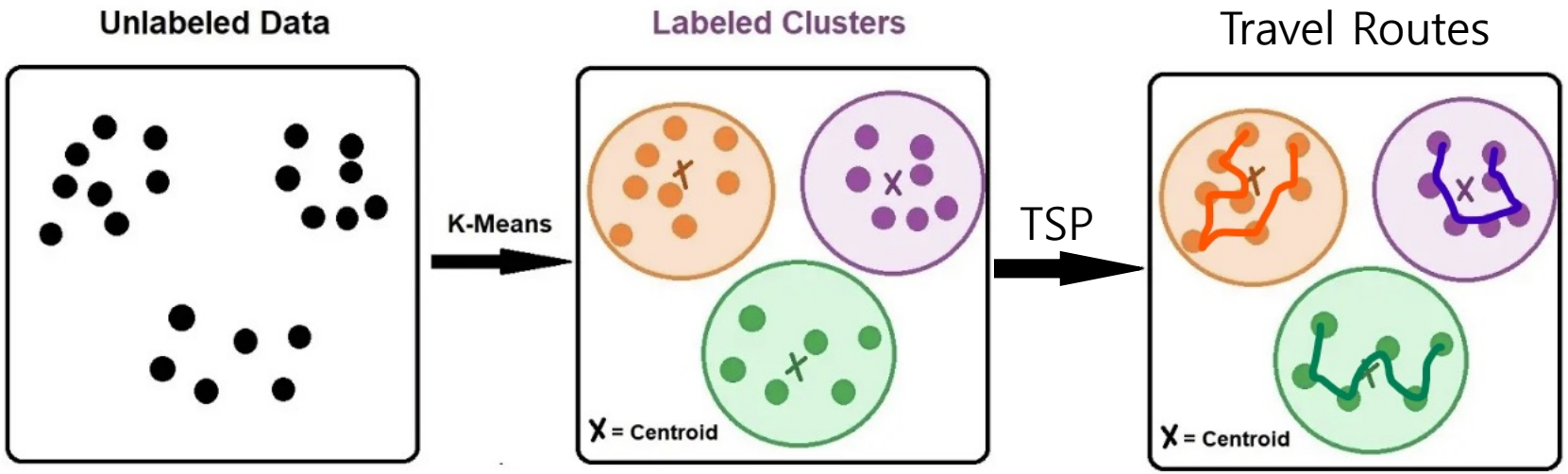
Contents

- Contribution of the Source Paper
- Contribution of the Mid-Term PT
- Limitation of the Mid-Term PT
- Extension
- Data
- Result Comparison
- Conclusion
- Limitations and Future Work
- Reference
- Appendix(Code)

Contribution of the Source Paper

A Development of Travel Itinerary Planning Application using Traveling Salesman Problem and K-Means Clustering Approach. - Rani, S., Kholidah, K. N., & Huda, S. N.

- Algorithm for making travel itinerary using TSP and k-means clustering technique is proposed.

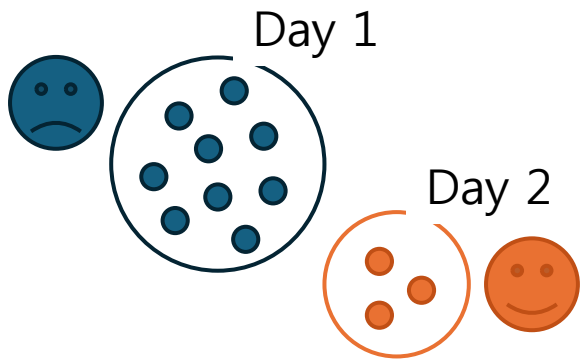


https://www.ejable.com/tech-corner/ai-machine-learning-and-deep-learning/k-means-clustering/

Contribution of the Mid-Term PT

- Clusters are formed based only on the points' distance.
- Without considering the size of clusters

➢Some cluster may contain too many points for daily travel.



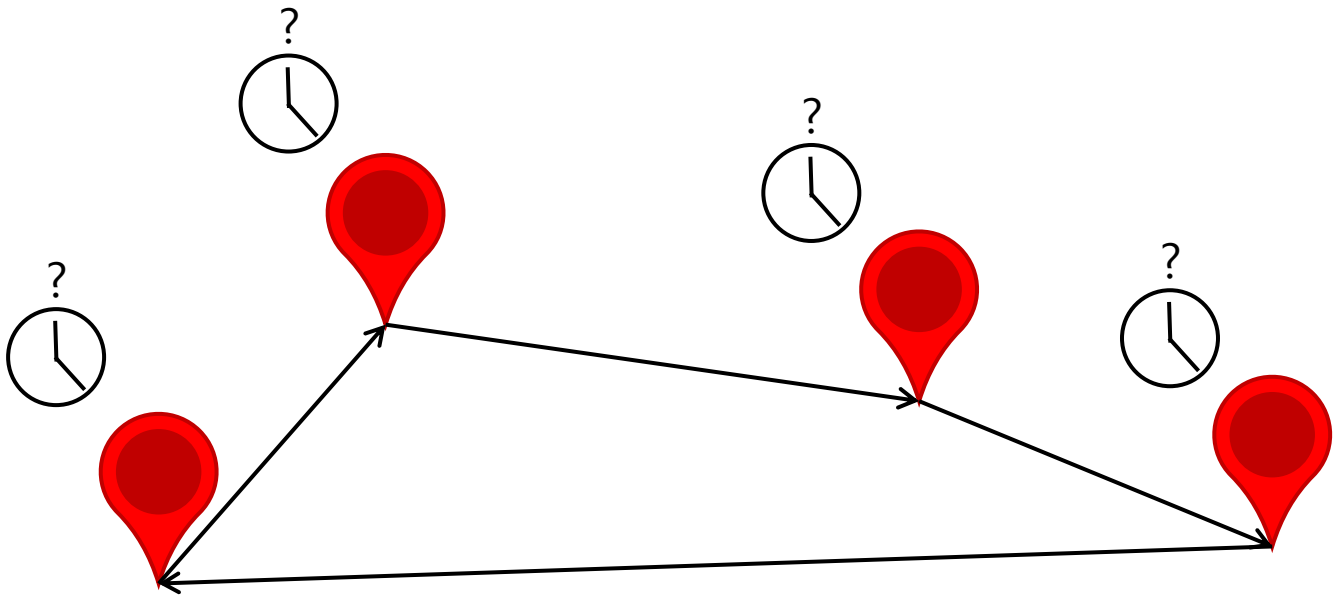
- To overcome the limitation of the original paper,
  - We changed the clustering method from K-means clustering to GA

- Fitness Function of GA

→ $\alpha\times$ Total Time of Travel
$+ (1-\alpha) \times$ Deviation between Travel Time Each Day

Limitation of the Mid-Term PT

- We didn’t consider the stay time at the tourist attraction and only considered travel time, which is far from tour planning in reality.



https://www.ejable.com/tech-corner/ai-machine-learning-and-deep-learning/k-means-clustering/

Extension

- We tried to find out the optimal value of α.
- To overcome the limitation of the mid-term presentation,
  - We added the stay time at each tourist attractions when we calculate the total time of travel of a day.

- Fitness Function of GA

→ $\alpha\times$ Total Time of Travel + $(1-\alpha)$ × Deviation between Travel Time Each Day

→ Total Time of Travel = Travel Time between Tourist Attractions + Stay Time at Tourist Attractions

Methodology

1. Make the initial solution for GA by assigning travel sites to random clusters(days).
2. Set the fitness function as below:

   $\alpha\times$ Total Time of Travel + $(1-\alpha)$ $\times$ Deviation between Travel Time Each Day

   (Time to travel each cluster is calculated with OR-Tools by solving the TSP)
3. With the fitness function, run GA to find the best cluster combination (which day to visit each sites).
4. Print the result with the travel time of each days.

Data

- Longitude and Latitude of 34 Tourist Attractions (via Google Map)
- Time Duration by Public Transportation between 34 Tourist Attractions (via Google Map)
- List of Attractions

- 'Bukchon Hanok Village', 'Namsan Seoul Tower', 'Gyeongbokgung Palace', '63 Square', 'Changdeokgung Palace', 'Changgyeonggung Palace', 'National Museum of Korea', 'Culture Station Seoul 284', 'War Memorial of Korea', 'Yongsan Family Park', 'Seoul Plaza', 'Seodaemun Prison History Museum', 'Gwanghwamun Square', 'Bosingak Belfry', 'Heunginjimun Gate', 'Deoksugung Palace', 'Samcheong-dong Alley', 'Bangi-dong Baekje Tombs', 'Myeongdong Cathedral', 'Hwangudan Altar', 'The Blue House', 'National Assembly Building', 'National Library of Korea', 'Bukak Skyway Octagonal Pavilion', 'Banpo Bridge Night View', 'Lotte World Tower', 'Sungnyemun Gate', 'Seoullo 7017', 'Deoksugung Stonewall Walkway', 'Dongdaemun Design Plaza (DDP)', 'Starfield Library', 'Namdaemun Market Tourist Information Center', 'Ikseon-dong Hanok Street'

	A	B	C
1	Name	Latitude	Longitude
2	Namsan S	37.55117	126.9882
3	Gyeongbo	37.57962	126.977
4	63 Square	37.51983	126.9401
5	Bukchon H	37.57903	126.9864
6	Changdeo	37.57943	126.991
7	Changgye	37.58058	126.995
8	National M	37.52385	126.9805
9	Culture St	37.55596	126.9716
10	War Mem	37.53661	126.9771
11	Yongsan F	37.5283	126.9829
12	Seoul Plaz	37.5677	126.9773
13	Seodaemu	37.57529	126.955
14	Gwanghwa	37.57169	126.9776
15	Bosingak E	37.56976	126.9837
16	Heunginjir	37.57114	127.0095
17	Deoksugu	37.5674	126.9755
18	Samcheon	37.58201	126.9865

	A	B	C	D	E	F	G	H
1		Bukchon H	Namsan S	Gyeongbo	63 Square	Changdeo	Changgye	National M
2	Bukchon H	0	1706	736	2692	815	670	2062
3	Namsan S	1706	0	2105	2970	2151	1884	2204
4	Gyeongbo	736	2105	0	3185	1019	1236	2390
5	63 Square	2692	2970	3185	0	2823	2884	2272
6	Changdeo	815	2151	1019	2823	0	801	2511
7	Changgye	670	1884	1236	2884	801	0	2244
8	National M	2062	2204	2390	2272	2511	2244	0
9	Culture St	1334	1387	1357	1217	1419	1587	994
10	War Mem	1964	1962	2128	1785	2095	2212	1102
11	Yongsan F	2721	1879	2463	2498	2852	2969	1821
12	Seoul Plaz	1245	1882	1150	1447	1254	1313	1396
13	Seodaemu	582	2208	670	2512	865	1187	1903
14	Gwanghwa	714	1729	589	1455	798	1230	2058
15	Bosingak E	887	1387	1108	1573	972	1136	1522
16	Heunginjir	904	1507	1232	1939	1034	832	1573
17	Deoksugu	1136	1687	1285	1536	1165	1384	1485
18	Samcheon	1233	2037	1293	2999	1490	992	2321

Newly Added Data

- SK Open API Geovision Puzzle Data
  - Stay time at every tourist attractions
- Administrative Standard Code Management System
  - Statutory Neighborhood Code

```
Name,District_Code,Avg_Stay_Duration
Bukchon Hanok Village,1117010200,5899
Namsan Seoul Tower,1111011900,5301
Gyeongbokgung Palace,1156011000,13844
63 Square,1111014800,5420
Changdeokgung Palace,1111013000,5569
Changgyeonggung Palace,1111013000,5569
National Museum of Korea,1117013500,7711
Culture Station Seoul 284,1114012000,4288
War Memorial of Korea,1117010600,6121
Yongsan Family Park,1117013500,5796
Seoul Plaza,1114010300,4495
Seodaemun Prison History Museum,1141010900,5090
Gwanghwamun Square,1111011900,4368
Bosingak Belfry,1111013500,4191
Heunginjimun Gate,1111016400,6396
Deoksugung Palace,1111011900,4368
Samcheong-dong Alley,1111014600,5707
Bangi-dong Baekje Tombs,1171011100,7799
Myeongdong Cathedral,1114012700,4126
Hwangudan Altar,1114011100,5460
The Blue House,1111011900,4368
National Assembly Building,1156011000,11199
National Library of Korea,1165010700,7683
Bukak Skyway Octagonal Pavilion,1111018300,7363
Banpo Bridge Night View,1165010700,7683
Lotte World Tower,1171010200,6868
```

Result Comparison with Mid-Term PT

Mid-Term PT (solved without stay time)

```
7510
Day 1: Bukchon Hanok Village -> Gwanghwamun Square -> National Assembly Building -> 63 Square -> Culture Station Seoul 284 -> Samcheon

6887
Day 2: Bukchon Hanok Village -> Myeongdong Cathedral -> National Museum of Korea -> Banpo Bridge Night View -> War Memorial of Korea -

6733
Day 3: Bukchon Hanok Village -> Seodaemun Prison History Museum -> Seoul Plaza -> Bukak Skyway Octagonal Pavilion -> Changgyeonggung P

7033
Day 4: Bukchon Hanok Village -> Ikseon-dong Hanok Street -> Bosingak Belfry -> Heunginjimun Gate -> Dongdaemun Design Plaza (DDP) -> N

6968
Day 5: Bukchon Hanok Village -> Changdeokgung Palace -> The Blue House -> Deoksugung Stonewall Walkway -> Hwangudan Altar -> Namsan Se

7396
Day 6: Bukchon Hanok Village -> National Library of Korea -> Starfield Library -> Lotte World Tower -> Bukchon Hanok Village

6774
Day 7: Bukchon Hanok Village -> Bangi-dong Baekje Tombs -> Bukchon Hanok Village
```

Low fluctuation between days.
Min. 1hour 52min
Max. 2hour 5min for travel time
→ Enough time for sight-seeing (Estimation)

Suggestion (with stay time)

```
Best Individual =  [4, 6, 1, 2, 2, 5, 2, 5, 3, 2, 1, 6, 3, 2, 5, 6, 0, 3, 4, 0, 4, 1, 3, 5, 1, 6, 4, 3, 2, 0, 0, 5]
Fitness =  (50776.605430608855,)
Day 1 Tour Order(with 11 hours 58 minutes):
Bukchon Hanok Village -> The Blue House -> Namdaemun Market Tourist Information Center -> Starfield Library -> Bangi-dong Baekje Tombs -> Bukchon Hanok Village

Day 2 Tour Order(with 11 hours 11 minutes):
Bukchon Hanok Village -> Seodaemun Prison History Museum -> 63 Square -> National Library of Korea -> Lotte World Tower -> Bukchon Hanok Village

Day 3 Tour Order(with 11 hours 28 minutes):
Bukchon Hanok Village -> Changdeokgung Palace -> Seoul Plaza -> Culture Station Seoul 284 -> Dongdaemun Design Plaza (DDP) -> Heunginjimun Gate -> Changgyeonggung Pa
lace -> Bukchon Hanok Village

Day 4 Tour Order(with 11 hours 34 minutes):
Bukchon Hanok Village -> Myeongdong Cathedral -> Yongsan Family Park -> Bosingak Belfry -> Deoksugung Stonewall Walkway -> Bukak Skyway Octagonal Pavilion -> Bukchon
 Hanok Village

Day 5 Tour Order(with 11 hours 20 minutes):
Bukchon Hanok Village -> Namsan Seoul Tower -> National Assembly Building -> Seoullo 7017 -> Hwangudan Altar -> Bukchon Hanok Village

Day 6 Tour Order(with 11 hours 52 minutes):
Bukchon Hanok Village -> Ikseon-dong Hanok Street -> Deoksugung Palace -> War Memorial of Korea -> Banpo Bridge Night View -> National Museum of Korea -> Bukchon Han
ok Village

Day 7 Tour Order(with 11 hours 38 minutes):
Bukchon Hanok Village -> Samcheong-dong Alley -> Sungnyemun Gate -> Gwanghwamun Square -> Gyeongbokgung Palace -> Bukchon Hanok Village
```

Low fluctuation between days.
Min. 11hour 11min
Max. 11hour 58min for travel time and sightseeing
→ We can find out that there actually is enough time for both travelling between attractions and sightseeing

Result Comparison with Mid-Term PT

Mid-Term PT (after adding stay time)

```
Best Individual =  [0, 3, 1, 0, 3, 6, 6, 6, 0, 3, 3, 2, 3, 2, 3, 3, 4, 5, 6, 2, 5, 5, 2, 6, 4, 6, 3, 3, 2, 1, 3, 5]
Fitness =  (7653.014455170805,)
Day 1 Tour Order(with 8 hours 16 minutes):
Bukchon Hanok Village -> Changdeokgung Palace -> Yongsan Family Park -> Namsan Seoul Tower -> Bukchon Hanok Village

Day 2 Tour Order(with 7 hours 58 minutes):
Bukchon Hanok Village -> 63 Square -> Starfield Library -> Bukchon Hanok Village

Day 3 Tour Order(with 11 hours 2 minutes):
Bukchon Hanok Village -> Heunginjimun Gate -> Dongdaemun Design Plaza (DDP) -> Gwanghwamun Square -> Bukak Skyway Octagonal Pavilion -> The B
lue House -> Bukchon Hanok Village

Day 4 Tour Order(with 20 hours 0 minutes):
Bukchon Hanok Village -> Changgyeonggung Palace -> Samcheong-dong Alley -> Deoksugung Palace -> Bosingak Belfry -> Seoul Plaza -> Namdaemun M
arket Tourist Information Center -> Seoullo 7017 -> Deoksugung Stonewall Walkway -> Seodaemun Prison History Museum -> Gyeongbokgung Palace -
> Bukchon Hanok Village

Day 5 Tour Order(with 7 hours 43 minutes):
Bukchon Hanok Village -> Lotte World Tower -> Bangi-dong Baekje Tombs -> Bukchon Hanok Village

Day 6 Tour Order(with 11 hours 11 minutes):
Bukchon Hanok Village -> Ikseon-dong Hanok Street -> National Assembly Building -> National Library of Korea -> Myeongdong Cathedral -> Bukch
on Hanok Village

Day 7 Tour Order(with 14 hours 13 minutes):
Bukchon Hanok Village -> Hwangudan Altar -> Culture Station Seoul 284 -> National Museum of Korea -> Banpo Bridge Night View -> War Memorial
of Korea -> Sungnyemun Gate -> Bukchon Hanok Village
```

Increased fluctuation between days.
Min. 7hour 43min
Max. 20hour for travel time and sightseeing
→ 20 hours for travel time and sightseeing for a day might be inappropriate.

Suggestion (with stay time)

```
Best Individual =  [4, 6, 1, 2, 2, 5, 2, 5, 3, 2, 1, 6, 3, 2, 5, 6, 0, 3, 4, 0, 4, 1, 3, 5, 1, 6, 4, 3, 2, 0, 0, 5]
Fitness =  (50776.605430608855,)
Day 1 Tour Order(with 11 hours 58 minutes):
Bukchon Hanok Village -> The Blue House -> Namdaemun Market Tourist Information Center -> Starfield Library -> Bangi-dong Baekje Tombs -> Bukchon Hanok Village

Day 2 Tour Order(with 11 hours 11 minutes):
Bukchon Hanok Village -> Seodaemun Prison History Museum -> 63 Square -> National Library of Korea -> Lotte World Tower -> Bukchon Hanok Village

Day 3 Tour Order(with 11 hours 28 minutes):
Bukchon Hanok Village -> Changdeokgung Palace -> Seoul Plaza -> Culture Station Seoul 284 -> Dongdaemun Design Plaza (DDP) -> Heunginjimun Gate -> Changgyeonggung Pa
lace -> Bukchon Hanok Village

Day 4 Tour Order(with 11 hours 34 minutes):
Bukchon Hanok Village -> Myeongdong Cathedral -> Yongsan Family Park -> Bosingak Belfry -> Deoksugung Stonewall Walkway -> Bukak Skyway Octagonal Pavilion -> Bukchon
 Hanok Village

Day 5 Tour Order(with 11 hours 20 minutes):
Bukchon Hanok Village -> Namsan Seoul Tower -> National Assembly Building -> Seoullo 7017 -> Hwangudan Altar -> Bukchon Hanok Village

Day 6 Tour Order(with 11 hours 52 minutes):
Bukchon Hanok Village -> Ikseon-dong Hanok Street -> Deoksugung Palace -> War Memorial of Korea -> Banpo Bridge Night View -> National Museum of Korea -> Bukchon Han
ok Village

Day 7 Tour Order(with 11 hours 38 minutes):
Bukchon Hanok Village -> Samcheong-dong Alley -> Sungnyemun Gate -> Gwanghwamun Square -> Gyeongbokgung Palace -> Bukchon Hanok Village
```

Low fluctuation between days.
Min. 11hour 11min
Max. 11hour 58min for travel time and sightseeing
→ We can find out that there actually is enough time for both travelling between attractions and sightseeing

Result Comparison with Paper

Paper (K-means Clustering)

```
✔ python original.py
Day 1 Tour Order(with 29 hours 31 minutes):
Bukchon Hanok Village -> Changdeokgung Palace -> Myeongdong Cathedral -> Samcheong-dong Alley -> Ikseon-dong Hanok Street -> Gwanghwamun Square -> Deoksugung Palace
-> Seoul Plaza -> Bosingak Belfry -> Hwangudan Altar -> Deoksugung Stonewall Walkway -> Bukak Skyway Octagonal Pavilion -> The Blue House -> Gyeongbokgung Palace ->
Seodaemun Prison History Museum -> Bukchon Hanok Village -> Bukchon Hanok Village

Day 2 Tour Order(with 8 hours 49 minutes):
Bukchon Hanok Village -> Bangi-dong Baekje Tombs -> Lotte World Tower -> Bukchon Hanok Village

Day 3 Tour Order(with 8 hours 53 minutes):
Bukchon Hanok Village -> 63 Square -> National Assembly Building -> Bukchon Hanok Village

Day 4 Tour Order(with 12 hours 12 minutes):
Bukchon Hanok Village -> National Museum of Korea -> Yongsan Family Park -> Banpo Bridge Night View -> National Library of Korea -> Bukchon Hanok Village

Day 5 Tour Order(with 6 hours 49 minutes):
Bukchon Hanok Village -> Starfield Library -> Bukchon Hanok Village

Day 6 Tour Order(with 14 hours 54 minutes):
Bukchon Hanok Village -> Namsan Seoul Tower -> Namdaemun Market Tourist Information Center -> Culture Station Seoul 284 -> War Memorial of Korea -> Seoullo 7017 -> S
ungnyemun Gate -> Bukchon Hanok Village

Day 7 Tour Order(with 9 hours 8 minutes):
Bukchon Hanok Village -> Changgyeonggung Palace -> Dongdaemun Design Plaza (DDP) -> Heunginjimun Gate -> Bukchon Hanok Village
```

High fluctuation between days and infeasible solution.
Min. 6hour 49min
Max. 29hour 31min for travel time and sightseeing
→ In case of maximum time, it exceeds 24hours which means it is not appropriate for one-day plan.

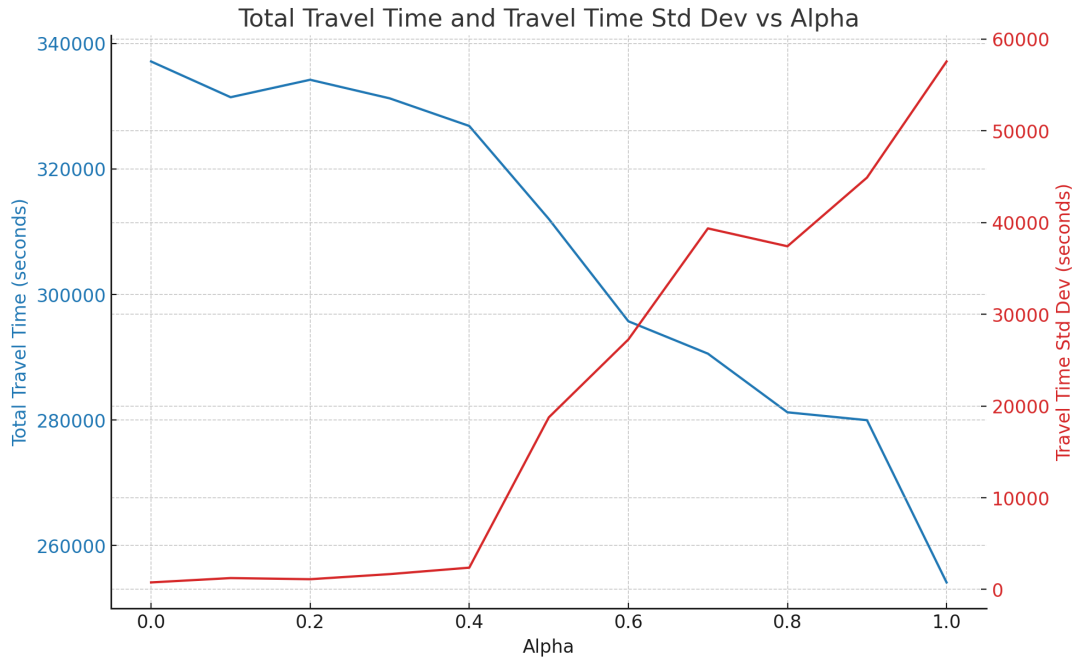
Suggestion (GA)

```
Best Individual =  [4, 6, 1, 2, 2, 5, 2, 5, 3, 2, 1, 6, 3, 2, 5, 6, 0, 3, 4, 0, 4, 1, 3, 5, 1, 6, 4, 3, 2, 0, 0, 5]
Fitness =  (50776.605430608855,)
Day 1 Tour Order(with 11 hours 58 minutes):
Bukchon Hanok Village -> The Blue House -> Namdaemun Market Tourist Information Center -> Starfield Library -> Bangi-dong Baekje Tombs -> Bukchon Hanok Village

Day 2 Tour Order(with 11 hours 11 minutes):
Bukchon Hanok Village -> Seodaemun Prison History Museum -> 63 Square -> National Library of Korea -> Lotte World Tower -> Bukchon Hanok Village

Day 3 Tour Order(with 11 hours 28 minutes):
Bukchon Hanok Village -> Changdeokgung Palace -> Seoul Plaza -> Culture Station Seoul 284 -> Dongdaemun Design Plaza (DDP) -> Heunginjimun Gate -> Changgyeonggung Pa
lace -> Bukchon Hanok Village

Day 4 Tour Order(with 11 hours 34 minutes):
Bukchon Hanok Village -> Myeongdong Cathedral -> Yongsan Family Park -> Bosingak Belfry -> Deoksugung Stonewall Walkway -> Bukak Skyway Octagonal Pavilion -> Bukchon
 Hanok Village

Day 5 Tour Order(with 11 hours 20 minutes):
Bukchon Hanok Village -> Namsan Seoul Tower -> National Assembly Building -> Seoullo 7017 -> Hwangudan Altar -> Bukchon Hanok Village

Day 6 Tour Order(with 11 hours 52 minutes):
Bukchon Hanok Village -> Ikseon-dong Hanok Street -> Deoksugung Palace -> War Memorial of Korea -> Banpo Bridge Night View -> National Museum of Korea -> Bukchon Han
ok Village

Day 7 Tour Order(with 11 hours 38 minutes):
Bukchon Hanok Village -> Samcheong-dong Alley -> Sungnyemun Gate -> Gwanghwamun Square -> Gyeongbokgung Palace -> Bukchon Hanok Village
```

Low fluctuation between days and feasible solutions.
Min. 11hour 11min
Max. 11hour 58min for travel time and sightseeing
→ We can find out that there is enough time for both travelling between attractions and sightseeing

Hyperparameter Tuning

Fitness Function: $\alpha \times$ Total Time of Travel + $(1-\alpha) \times$ Deviation between Travel Time Each Day



	Alpha	Best Individual	Fitness	Total Travel Time	Travel Time Std Dev
0	0.0	[4, 3, 2, 0, 6, 3, 3, 1, 6, 4, 5, 5, 4, 0, 5, ...	(769.0780857889636,)	337110	769.078086
1	0.1	[4, 2, 1, 0, 4, 6, 3, 1, 3, 6, 4, 0, 1, 2, 3, ...	(34260.88128318409,)	331415	1243.756981
2	0.2	[0, 0, 3, 5, 0, 3, 1, 3, 3, 6, 2, 2, 0, 2, 4, ...	(67734.26162232278,)	334196	1118.827028
3	0.3	[2, 5, 3, 6, 3, 1, 4, 0, 2, 6, 1, 2, 0, 1, 1, ...	(100542.88511545087,)	331234	1675.264451
4	0.4	[0, 5, 2, 1, 4, 0, 4, 6, 0, 5, 1, 2, 4, 4, 1, ...	(132161.79801581602,)	326837	2378.330026
5	0.5	[0, 2, 4, 5, 6, 0, 5, 1, 0, 6, 2, 2, 1, 5, 4, ...	(165394.54421150935,)	312028	18761.088423
6	0.6	[6, 4, 4, 5, 1, 0, 4, 6, 6, 1, 6, 5, 1, 1, 5, ...	(188341.9486967036,)	295735	27252.371742
7	0.7	[0, 2, 0, 2, 2, 0, 1, 2, 0, 2, 0, 1, 3, 0, 2, ...	(215210.63844726578,)	290575	39360.461491
8	0.8	[2, 0, 5, 2, 3, 0, 5, 5, 5, 3, 2, 5, 3, 0, 5, ...	(232469.4612477507,)	281234	37411.306239
9	0.9	[1, 0, 3, 0, 0, 3, 0, 3, 3, 0, 1, 3, 0, 5, 5, ...	(256472.08048080228,)	279980	44900.804808
10	1.0	[4, 6, 4, 6, 6, 6, 6, 4, 4, 4, 4, 4, 4, 6, 4, ...	(254142.0,)	254142	57557.274677

Candidates of Alpha

	Alpha	Total Travel Time Decrease	Travel Time Std Dev Increase
1	0.1	5695.000000	474.678895
2	0.2	-2781.000000	-124.929953
3	0.3	-2962.000000	556.437423
4	0.4	-4397.000000	703.065575
5	0.5	-14809.000000	16382.758397
6	0.6	-16393.000000	8483.283318
7	0.7	-5160.000000	12108.089749
8	0.8	-9341.000000	-1949.155251
9	0.9	-1254.000000	7490.303569
10	1.0	-25838.000000	12656.469869

- Alpha = 0.3~0.4
  Total Travel Time Std. Dev/Alpha increased rapidly right after this point
- Alpha = 0.6
  As two graphs intersect at alpha = 0.6

Hyperparameter Tuning

Fitness Function: $\alpha\times$ Total Time of Travel + $(1-\alpha)\times$ Deviation between Travel Time Each Day

α=0.3

```
Day 1 Tour Order(with 11 hours 33 minutes):
Bukchon Hanok Village -> Myeongdong Cathedral -> Bosingak Belfry -> Seoul Plaza -> The Blue House -> Gyeongbokgung Palace -> Bukchon Hanok Village

Day 2 Tour Order(with 11 hours 41 minutes):
Bukchon Hanok Village -> Gwanghwamun Square -> Bangi-dong Baekje Tombs -> Lotte World Tower -> Banpo Bridge Night View -> Bukchon Hanok Village

Day 3 Tour Order(with 11 hours 25 minutes):
Bukchon Hanok Village -> Ikseon-dong Hanok Street -> Seoullo 7017 -> National Museum of Korea -> War Memorial of Korea -> Heunginjimun Gate -> Bukchon Hanok Village

Day 4 Tour Order(with 11 hours 12 minutes):
Bukchon Hanok Village -> Namsan Seoul Tower -> Namdaemun Market Tourist Information Center -> Culture Station Seoul 284 -> Deoksugung Stonewall Walkway -> Bukak Skyway Octagonal Pavilion -> Bukchon Hanok Village

Day 5 Tour Order(with 11 hours 31 minutes):
Bukchon Hanok Village -> 63 Square -> National Assembly Building -> Starfield Library -> Bukchon Hanok Village

Day 6 Tour Order(with 11 hours 26 minutes):
Bukchon Hanok Village -> Seodaemun Prison History Museum -> Deoksugung Palace -> Dongdaemun Design Plaza (DDP) -> Yongsan Family Park -> National Library of Korea -> Bukchon Hanok Village

Day 7 Tour Order(with 11 hours 22 minutes):
Bukchon Hanok Village -> Changgyeonggung Palace -> Samcheong-dong Alley -> Hwangudan Altar -> Sungnyemun Gate -> Changdeokgung Palace -> Bukchon Hanok Village
```

Shows feasible solutions in an acceptable Avg. traveling time and its Std. Dev.

α=0.6

```
Fitness =  (188151.20676861584,)
Day 1 Tour Order(with 18 hours 3 minutes):
Bukchon Hanok Village -> National Museum of Korea -> Banpo Bridge Night View -> National Library of Korea -> Starfield Library -> Lotte World Tower -> Bangi-dong Baekje Tombs -> Bukchon Hanok Village

Day 2: No visits planned.
Day 3 Tour Order(with 12 hours 8 minutes):
Bukchon Hanok Village -> Changgyeonggung Palace -> Bukak Skyway Octagonal Pavilion -> The Blue House -> Gyeongbokgung Palace -> Bukchon Hanok Village

Day 4 Tour Order(with 16 hours 30 minutes):
Bukchon Hanok Village -> Ikseon-dong Hanok Street -> Bosingak Belfry -> Seoul Plaza -> Namdaemun Market Tourist Information Center -> Namsan Seoul Tower -> Yongsan Family Park -> War Memorial of Korea -> Dongdaemun Design Plaza (DDP) -> Heunginjimun Gate -> Bukchon Hanok Village

Day 5: No visits planned.
Day 6 Tour Order(with 14 hours 26 minutes):
Bukchon Hanok Village -> Changdeokgung Palace -> Gwanghwamun Square -> National Assembly Building -> 63 Square -> Sungnyemun Gate -> Seodaemun Prison History Museum -> Bukchon Hanok Village

Day 7 Tour Order(with 12 hours 40 minutes):
Bukchon Hanok Village -> Myeongdong Cathedral -> Culture Station Seoul 284 -> Deoksugung Palace -> Seoullo 7017 -> Deoksugung Stonewall Walkway -> Hwangudan Altar -> Samcheong-dong Alley -> Bukchon Hanok Village
```

The mean value reduced a lot, but it leads to large variance, making infeasible solution.

We have to find appropriate value of α. In this case, we can see that focusing on the Std. Dev slightly more than Avg. by decreasing α leads to a better result.

Conclusion

- We found out that clustering using GA gave better tour plan than the source paper.
- We also found out that including stay time during solving is important for getting ideal travel plan.
- We explained the correlation of Average traveling time and Std. Dev of traveling time with the Alpha value.
- We suggested reasonable Alpha value that gives best result considering both Avg, Std. Dev.

Limitation and Future Work

- We might also have to consider the opening time window of tourist attractions.
- As the travel time and cost varies depending on the time window and on the choice of means of transportation.
- We attempted to use Naver Map API to construct data considering real-time and multiple transportations, but it was not included in the current presentation stage.
- Because Naver Map API deleted public transportation data and using other methods might cause some legal problems.
- By getting admission for the data matter, we might try to implement these real-time and multi transportation terms.

Reference

Rani, S., Kholidah, K. N., & Huda, S. N. (2018). A Development of Travel Itinerary Planning Application using Traveling Salesman Problem and K-Means Clustering Approach. In Proceedings of the 2018 7th International Conference on Software and Computer Applications (ICSCA '18) (pp. 327–331). https://doi.org/10.1145/3185089.3185142

https://data.seoul.go.kr/dataList/OA-21050/S/1/datasetView.do

https://www.google.co.kr/maps/?hl=ko

https://openapi.sk.com/products/detail?svcSeq=57&menuSeq=297

https://www.code.go.kr/stdcode/regCodeL.do

Appendix 1 – Original paper code

```
# 필요한 라이브러리 불러오기
import pandas as pd
import numpy as np
from sklearn.cluster import KMeans
from ortools.constraint_solver import routing_enums_pb2
from ortools.constraint_solver import pywrapcp

# 이동 시간 매트릭스 로드
duration_df = pd.read_csv('./tour_distances_matrix.csv', index_col=0)

# 위치 정보 로드
geo_df = pd.read_csv('./geo_info.csv')
geo_df['Cluster'] = KMeans(n_clusters=total_day, random_state=42).fit(geo_df[[
'Latitude', 'Longitude']]).labels_

# 숙소 정보
base_location = 'Bukchon Hanok Village'

# 각 클러스터별 TSP 문제 해결 및 출력
for day in range(total_day):
    # 현재 클러스터의 관광지 선택
    cluster_locations =  base_location] + geo_df[geo_df['Cluster'] == day][
'Name'].tolist()       [
    # 이동 시간 매트릭스 추출
    cluster_matrix = duration_df.loc[cluster_locations, cluster_locations].
to_numpy()

    # TSP 문제 설정 및 해결
    manager = pywrapcp.RoutingIndexManager(len(cluster_matrix), 1, 0)
    routing = pywrapcp.RoutingModel(manager)
    transit_callback_index = routing.RegisterTransitCallback(lambda from_index
, to_index: cluster_matrix[manager.IndexToNode(from_index)][manager.IndexToNode
(to_index)])
    routing.SetArcCostEvaluatorOfAllVehicles(transit_callback_index)
    search_parameters = pywrapcp.DefaultRoutingSearchParameters()
    search_parameters.first_solution_strategy = routing_enums_pb2.
FirstSolutionStrategy.PATH_CHEAPEST_ARC
    solution = routing.SolveWithParameters(search_parameters)
    # 결과 출력
    print(f'Day {day+1} Tour Order(with {solution.ObjectiveValue()}):')
    index = routing.Start(0)
    while not routing.IsEnd(index):
        print(f'{cluster_locations[manager.IndexToNode(index)]} -> ', end='')
        index = solution.Value(routing.NextVar(index))
    print(base_location)
    print()
```

Appendix 2 – Proposed code (GA)

```python
import random
import numpy as np
import pandas as pd
from deap import base, creator, tools, algorithms
from ortools.constraint_solver import pywrapcp, routing_enums_pb2

# CSV 파일 경로 설정
DISTANCE_PATH = "./tour_distances_matrix.csv"
DURATION_PATH = "./residence_time.csv"

# CSV 파일 로드
distance_matrix_df = pd.read_csv(DISTANCE_PATH, index_col=0)
duration_matrix_df = pd.read_csv(DURATION_PATH, index_col=0)


moving_time_matrix = distance_matrix_df.values
duration_time_list = duration_matrix_df["Avg_Stay_Duration"].values
travel_point_name = distance_matrix_df.columns.to_list()


random.seed(64)
algorithms.random.seed(64)

total_day = 7

# 유전 알고리즘 설정
creator.create("FitnessMin", base.Fitness, weights=(-1.0,))
creator.create("Individual", list, fitness=creator.FitnessMin)

toolbox = base.Toolbox()


# 각 관광지를 무작위 군집에 할당하는 함수
def create_individual():
    return
        random.randint(0, total_day - 1) for _ in range(len(moving_time_matrix) - 1)
    ]


toolbox.register("individual", tools.initIterate, creator.Individual, create_individual)
toolbox.register("population", tools.initRepeat, list, toolbox.individual)


# OR-Tools를 사용하여 군집 내 관광지 순회 시간 계산
def calculate_cluster_time(cluster):
    # 숙소(인덱스 0)를 시작점과 종료점으로 추가
    cluster_with_base =  0] + cluster +  0]
                        [              [
    if len(cluster_with_base) <= 2:
        return 0  # 군집 내 관광지가 없으면 이동 시간은 0

    manager = pywrapcp.RoutingIndexManager(len(cluster_with_base), 1, 0)
    routing = pywrapcp.RoutingModel(manager)

    # 거리 콜백 정의
    def distance_callback(from_index, to_index):
        from_node = manager.IndexToNode(from_index)
        to_node = manager.IndexToNode(to_index)
        return
            moving_time_matrix[cluster_with_base[from_node], cluster_with_base[to_node]]
            + duration_time_list[cluster_with_base[to_node]]
        )

    transit_callback_index = routing.RegisterTransitCallback(distance_callback)
    routing.SetArcCostEvaluatorOfAllVehicles(transit_callback_index)

    search_parameters = pywrapcp.DefaultRoutingSearchParameters()
    search_parameters.first_solution_strategy =
        routing_enums_pb2.FirstSolutionStrategy(AUTOMATIC
    )

    solution = routing.SolveWithParameters(search_parameters)
    if solution:
        # 순회 시간 계산
        return solution.ObjectiveValue()
    else:
        return 0


# 평가 함수
def evalTour(individual):
    # 군집별로 관광지 인덱스를 분류
    clusters = [[] for _ in range(total_day)]
    for idx, cluster_id in enumerate(individual):
        clusters[cluster_id].append(idx + 1)  # 관광지 인덱스는 1부터 시작
    # 각 군집의 순회 시간 계산
    times =
        calculate_cluster_time(cluster) for cluster in clusters
    ]  # 숙소 포함하여 계산
    total_travel_time = np.sum(times)
    # 순회 시간의 표준 편차 반환
    if len(times) > 1:
        travel_time_std = np.std(times)
    else:
        travel_time_std = 0  # 단일 군집의 경우 표준 편차는 0

    alpha = 0.15
    fitness = alpha * total_travel_time +  1 - alpha) * travel_time_std
                                          (
    return  fitness,)
          (
```

Appendix 2 – Proposed code (GA)

```
toolbox.register("evaluate", evalTour)
toolbox.register("mate", tools.cxTwoPoint)
toolbox.register("mutate", tools.mutShuffleIndexes, indpb=0.2)
toolbox.register("select", tools.selTournament, tournsize=3)

# 유전 알고리즘 실행
population_size = 100
num_generations = 100

pop = toolbox.population(n=population_size)
hof = tools.HallOfFame(1, similar=np.array_equal)

result = algorithms.eaSimple(
    pop,
    toolbox,
    cxpb=0.7,
    mutpb=0.2,
    ngen=num_generations,
    halloffame=hof,
    verbose=True,
)

# 최적 개체 출력
best_individual = hof.items[0]
print("Best Individual = ", best_individual)
print("Fitness = ", evalTour(best_individual))


def solve_tsp_for_cluster(cluster):
    # 숙소를 포함하여 TSP 문제 설정
    if len(cluster) == 0:
        return  0]  # 군집 내 관광지가 없으면 숙소만 반환
    cluster_with_base =  0] + cluster  # 숙소를 포함
    manager = pywrapcp.RoutingIndexManager(len(cluster_with_base), 1, 0)
    routing = pywrapcp.RoutingModel(manager)

    # 거리 콜백
    def distance_callback(from_index, to_index):
        from_node = manager.IndexToNode(from_index)
        to_node = manager.IndexToNode(to_index)
        return
            moving_time_matrix[cluster_with_base[from_node], cluster_with_base[to_node]]
            + duration_time_list[cluster_with_base[to_node]]
        )

    transit_callback_index = routing.RegisterTransitCallback(distance_callback)
    routing.SetArcCostEvaluatorOfAllVehicles(transit_callback_index)

    # TSP 문제 해결
    search_parameters = pywrapcp.DefaultRoutingSearchParameters()
    search_parameters.first_solution_strategy =
        routing_enums_pb2.FirstSolutionStrategy(AUTOMATIC
    )

    solution = routing.SolveWithParameters(search_parameters)
    time = solution.ObjectiveValue()
    if solution:
        index = routing.Start(0)
        route = []
        while not routing.IsEnd(index):
            route.append(cluster_with_base[manager.IndexToNode(index)])
            index = solution.Value(routing.NextVar(index))
        route.append(manager.IndexToNode(index))
        return route, time  # 숙소에서 시작하고 종료하는 경로 인덱스 반환
    else:
        return  0], time  # 해결할 수 없는 경우 숙소만 반환
              [

# 각 날짜별 군집화된 관광지를 식별하고 TSP 해결
for day in range(total_day):
    # 군집화된 관광지 인덱스 식별
    cluster =  i + 1 for i, cid in enumerate(best_individual) if cid == day]
    if not cluster:
        print(f"Day {day + 1}: No visits planned.")
        continue
    # 해당 군집에 대한 TSP 경로 해결
    tsp_route, total_seconds = solve_tsp_for_cluster(cluster)
    hours = total_seconds // 3600
    minutes =  total_seconds % 3600) // 60
    # 경로 출력(
    print(f"Day {day+1} Tour Order(with {hours} hours {minutes} minutes):")
    route_names =  travel_point_name[idx] for idx in tsp_route]
    print(" -> ".join(route_names))
    print()
```

Appendix 3 – Proposed code (Alpha fitting)

```
import random
import numpy as np
import pandas as pd
from deap import base, creator, tools, algorithms
from ortools.constraint_solver import pywrapcp, routing_enums_pb2

# CSV 파일 경로 설정
DISTANCE_PATH = "./tour_distances_matrix.csv"
DURATION_PATH = "./residence_time.csv"

# CSV 파일 로드
distance_matrix_df = pd.read_csv(DISTANCE_PATH, index_col=0)
duration_matrix_df = pd.read_csv(DURATION_PATH, index_col=0)

moving_time_matrix = distance_matrix_df.values
duration_time_list = duration_matrix_df["Avg_Stay_Duration"].values
travel_point_name = distance_matrix_df.columns.to_list()

random.seed(64)
algorithms.random.seed(64)

total_day = 7

# 유전 알고리즘 설정
creator.create("FitnessMin", base.Fitness, weights=(-1.0,))
creator.create("Individual", list, fitness=creator.FitnessMin)

toolbox = base.Toolbox()


# 각 관광지를 무작위 군집에 할당하는 함수
def create_individual():
    return
        random.randint(0, total_day - 1) for _ in range(len(moving_time_matrix) - 1)
    ]

toolbox.register("individual", tools.initIterate, creator.Individual, create_individual)
toolbox.register("population", tools.initRepeat, list, toolbox.individual)


# OR-Tools를 사용하여 군집 내 관광지 순회 시간 계산
def calculate_cluster_time(cluster):
    cluster_with_base =  0] + cluster +  0]
    if len(cluster_with_base) <= 2:    [
        return 0

    manager = pywrapcp.RoutingIndexManager(len(cluster_with_base), 1, 0)
    routing = pywrapcp.RoutingModel(manager)

    def distance_callback(from_index, to_index):
        from_node = manager.IndexToNode(from_index)
        to_node = manager.IndexToNode(to_index)
        return
            moving_time_matrix[cluster_with_base[from_node], cluster_with_base[to_node]]
            + duration_time_list[cluster_with_base[to_node]]
        )

    transit_callback_index = routing.RegisterTransitCallback(distance_callback)
    routing.SetArcCostEvaluatorOfAllVehicles(transit_callback_index)

    search_parameters = pywrapcp.DefaultRoutingSearchParameters()
    search_parameters.first_solution_strategy =
        routing_enums_pb2.FirstSolutionStrategy(AUTOMATIC
    )

    solution = routing.SolveWithParameters(search_parameters)
    if solution:
        return solution.ObjectiveValue()
    else:
        return 0
```

Appendix 3 – Proposed code (Alpha fitting)

```python
# 평가 함수
def evalTour(individual, alpha):
    clusters = [[] for _ in range(total_day)]
    for idx, cluster_id in enumerate(individual):
        clusters[cluster_id].append(idx + 1)

    times = [calculate_cluster_time(cluster) for cluster in clusters]
    total_tfavel_time = np.sum(times)
    travel_time_std = np.std(times) if len(times) > 1 else 0

    fitness = alpha * total_travel_time + (1 - alpha) * travel_time_std
    return fitness,

def run_experiment(alpha):
    toolbox.register("evaluate", evalTour, alpha=alpha)
    toolbox.register("mate", tools.cxTwoPoint)
    toolbox.register("mutate", tools.mutShuffleIndexes, indpb=0.2)
    toolbox.register("select", tools.selTournament, tournsize=3)

    population_size = 100
    num_generations = 100

    pop = toolbox.population(n=population_size)
    hof = tools.HallOfFame(1, similar=np.array_equal)

    result = algorithms.eaSimple(
        pop,
        toolbox,
        cxpb=0.7,
        mutpb=0.2,
        ngen=num_generations,
        halloffame=hof,
        verbose=False,
    )

    best_individual = hof.items[0]
    fitness = evalTour(best_individual, alpha)
    return best_individual, fitness

# 알파 값 범위 설정 및 실험 수행
alpha_values = np.arange(0.0, 1.1, 0.1)
results = []

for alpha in alpha_values:
    best_individual, fitness = run_experiment(alpha)
    results.append((alpha, best_individual, fitness))

# 결과 분석
results_df = pd.DataFrame(results, columns=["Alpha", "Best Individual", "Fitness"])
results_df["Total Travel Time"] = results_df["Best Individual"].apply(
    lambda ind: np.sum(
        [
            calculate_cluster_time([i + 1 for i, cid in enumerate(ind) if cid == day])
            for day in range(total_day)
        ]
    )
)
results_df["Travel Time Std Dev"] = results_df["Best Individual"].apply(
    lambda ind: np.std(
        [
            calculate_cluster_time([i + 1 for i, cid in enumerate(ind) if cid == day])
            for day in range(total_day)
        ]
    )
)

# 결과 표시
print(results_df)
```