


BUS LINE OPTIMIZATION

성유지 송예은 조규선 황연재





Limits of Geographic Data Analysis

And How This Group will Break that Limit



Explanatory Data Analysis

화성시 버스 노선 최적화

화성시의 교통, 인구 데이터를 분석

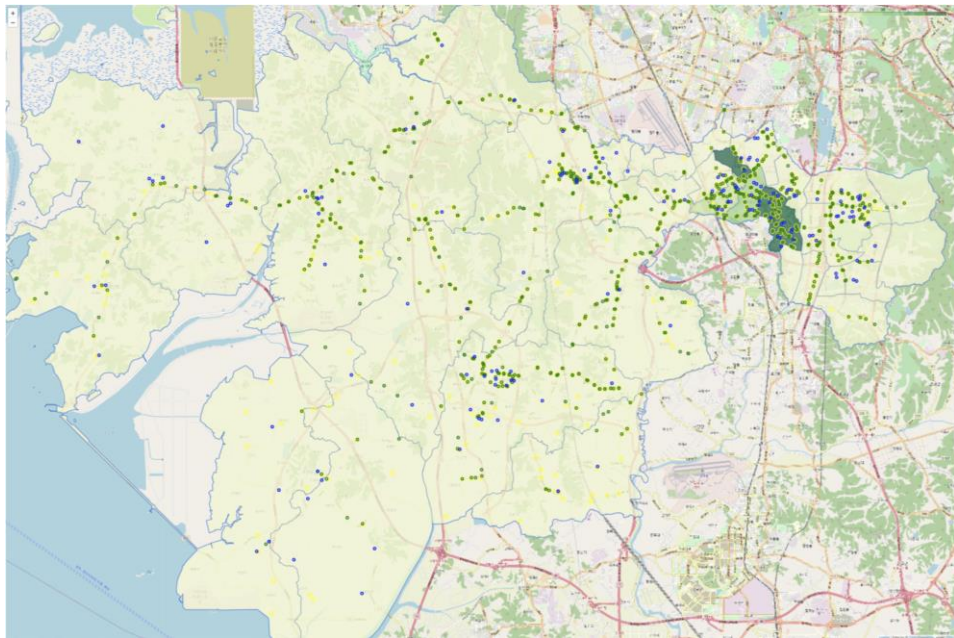


노선 최적화를 위해 화성시 내의 주요 위치(정류장)를 파악



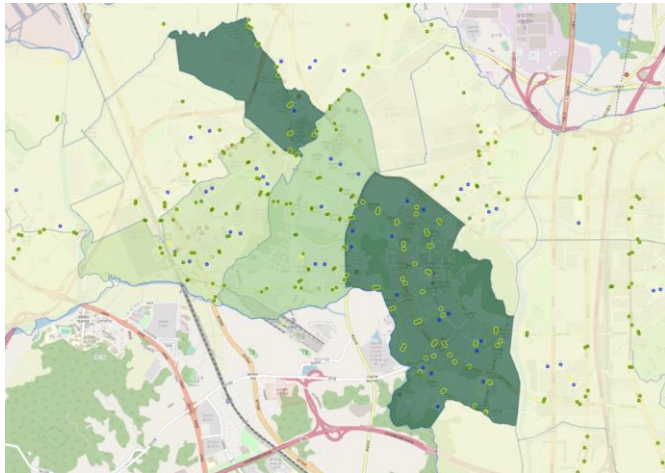
그래프 알고리즘을 통해 정류장들을 잇는 최적의 경로 탐색

EDA



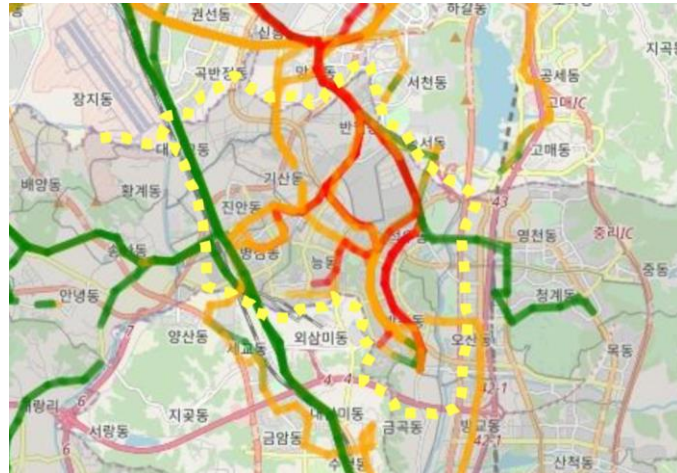
- 출근시간 대 (07시~09시) **유동인구** + 승차(노랑)/하차(초록) **버스정류장** 시각화
- 동탄1동, 동탄2동, 능동, 기산동, 병점동에 몰려있음을 알 수 있음
- 퇴근시간 대 역시 마찬가지

EDA



유동인구가 많은 지역의 경우
버스 정류장 개수는 충분

→



따라서, 수요에 따라
혼잡도가 높은 구간의 노선의 증설이 필요

- 수요 추정을 위해 'TripChain.csv', 'routstationinfo.csv', 'stations_table.csv' 데이터를 매핑해
출퇴근 시간대에 승하차가 가장 빈번하게 이루어지는 정류장을 분석

```
df_79_on_count = Counter(df_79_on['정류장이름'])
df_79_on_count.most_common(20)
```

[('신영통현대타운.두산위브', 1564),
('반월리큰고개', 996),
('신창미션힐.송화초교', 857),
('다은마을(중)', 813),
('와우2리', 792),
('능동마을입구', 706),
('메타폴리스(중)', 694),
('신일해피트리차', 558),
('반석초등학교', 529),
('수영초등학교', 525),
('대진화학', 524),
('수원대학교', 506),
('광도와이드빌.우남그린빌', 490),
('동탄1동행정복지센터', 489),
('홈플러스.별말초교', 476),
('동탄2동행정복지센터', 450),
('한일타운아파트', 413),
('주공5단지', 393),
('송산동입구', 388),
('한빛마을(중)', 381)]

```
df_79_off_count = Counter(df_79_off['정류장이름'])
df_79_off_count.most_common(20)
```

[('IT단지(중)', 2387),
('삼성반도체후문', 1618),
('한림대병원(중)', 1375),
('수원대학교', 964),
('메타폴리스(중)', 637),
('예당마을.롯데캐슬(중)', 632),
('반월1통.빅마켓앞', 625),
('신창미션힐.송화초교', 596),
('반월리큰고개', 560),
('삼성전자', 524),
('동탄1동행정복지센터', 517),
('다은마을(중)', 508),
('빅마켓앞', 496),
('롯데캐슬.복합문화센터', 474),
('동탄이마트.한림대병원', 455),
('병점사거리', 431),
('한빛마을.석우중학교(중)', 421),
('상록.경남아파트.이주택지', 412),
('신영통현대타운.두산위브', 399),
('한미약품.푸르지오', 392)]

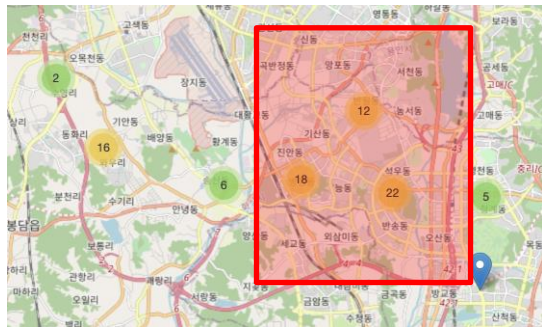
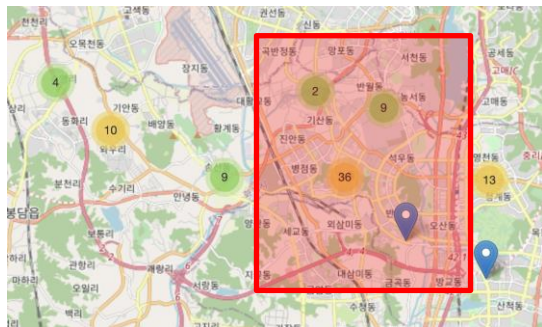
```
df_1820_on_count = Counter(df_1820_on['정류장이름'])
df_1820_on_count.most_common(20)
```

[('삼성반도체후문', 1604),
('IT단지(중)', 1503),
('한림대병원(중)', 1215),
('수원대학교', 1031),
('메타폴리스(중)', 880),
('신영통현대타운.두산위브', 725),
('동탄1동행정복지센터', 679),
('예당마을.롯데캐슬(중)', 588),
('복합문화센터', 573),
('반월리큰고개', 565),
('신창미션힐.송화초교', 555),
('홈플러스.별말초교', 495),
('삼성반도체', 492),
('한미약품.푸르지오', 462),
('다은마을(중)', 450),
('동부출장소.병점초등학교', 447),
('동탄역(동측)', 439),
('동탄이마트.한림대병원', 426),
('이주택지.상록.경남아파트', 414),
('병점역사거리', 392)]

```
df_1820_off_count = Counter(df_1820_off['정류장이름'])
df_1820_off_count.most_common(20)
```

[('신영통현대타운.두산위브', 1276),
('메타폴리스(중)', 795),
('동탄1동행정복지센터', 772),
('반월리큰고개', 763),
('능동마을입구', 671),
('다은마을(중)', 623),
('와우2리', 612),
('롯데캐슬.복합문화센터', 611),
('수원대학교', 569),
('신창미션힐.송화초교', 495),
('홈플러스.별말초교', 472),
('진안동', 452),
('병점사거리', 423),
('한림대병원(중)', 422),
('상록.경남아파트.이주택지', 415),
('한빛마을.석우중학교(중)', 415),
('반도.모아아파트', 404),
('신일우남아파트', 386),
('삼성전자', 379),
('수영초등학교', 376)]

중요한 노드와 그 기준

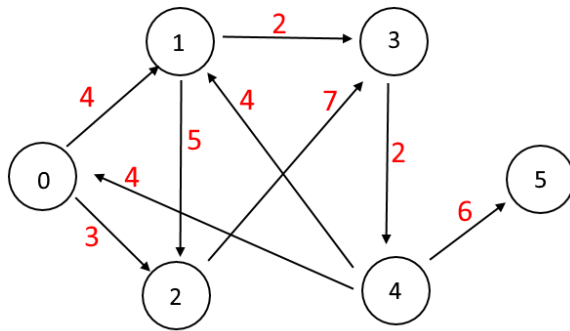
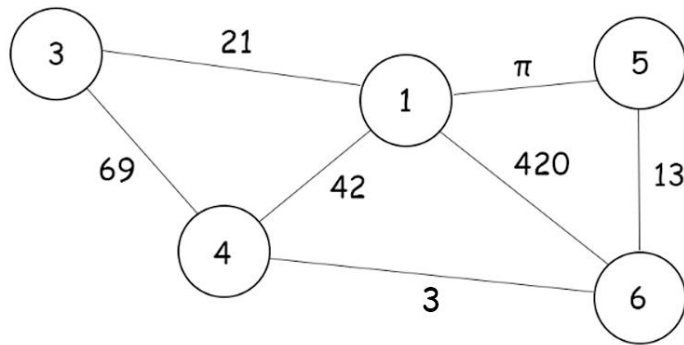


1. 유동인구가 가장 많고, 혼잡도가 높은 구간
 2. 출퇴근 시간대의 승하차가 가장 빈번하게 이루어지는 정류장 50개를 도출
 3. 위 두 기준을 모두 만족하는 정류장을 선별
- 프로젝트를 통해 이 정류장들을 잇는 최적의 노선 도출

Graph Theory

Graph Algorithm

- $G = (V, E)$
- Vertex (정점)
- Edge (간선)
- Weight (가중치)
- Direction (방향)
- Path (경로)
- Cycle (순환)



Graph Algorithm Representation

- Adjacency – List

(인접리스트)

- Adjacency – Matrix

(인접행렬)

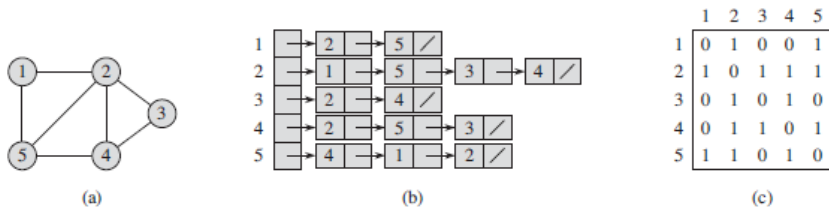


Figure 22.1 Two representations of an undirected graph. (a) An undirected graph G with 5 vertices and 7 edges. (b) An adjacency-list representation of G . (c) The adjacency-matrix representation of G .

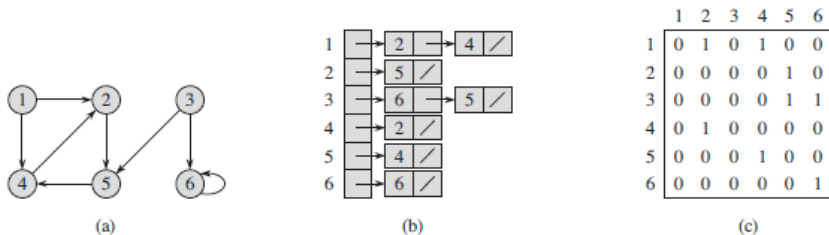


Figure 22.2 Two representations of a directed graph. (a) A directed graph G with 6 vertices and 8 edges. (b) An adjacency-list representation of G . (c) The adjacency-matrix representation of G .

Dijkstra Algorithm

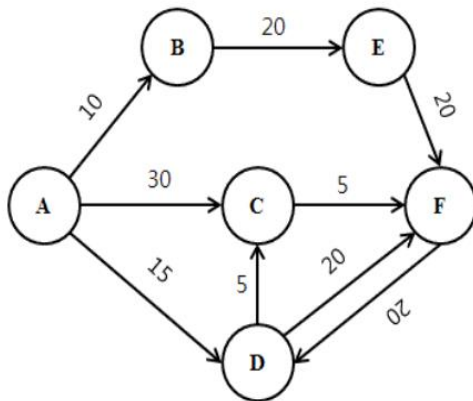
하나의 정점에서 모든 정점들의 최단 경로 구하기

→ 두 노드 간의 가장 짧은 경로를 찾는 알고리즘으로 응용하여 사용

한 정류장에서, 다른 정류장으로 가장 짧은 경로를 찾는 알고리즘으로 응용!

Dijkstra Algorithm - 초기화

- A에서 F로 가는 최단거리
- 시작 노드인 A의 거리 값은 0으로 초기화
- 다른 노드들은 거리 값을 무한으로 초기화
- S 집합: 이미 지난 노드
- Q 집합: 아직 지나지 않은 노드



$S = \{\}$

$d[A] = 0$

$d[B] = \text{infinity}$

$d[C] = \text{infinity}$

$d[D] = \text{infinity}$

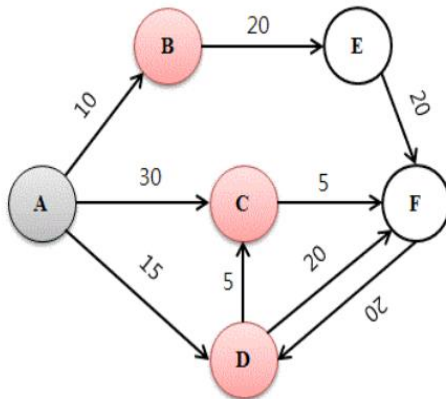
$d[E] = \text{infinity}$

$d[F] = \text{infinity}$

$Q = \{A, B, C, D, E, F\}$

Dijkstra Algorithm – 모든 노드까지의 최소값

- 문제 해결이 될 때까지 루프를 돌며 최소값을 찾는다



$S = \{A\}$

$d[A] = 0$

$d[B] = 10$

$d[C] = 30$

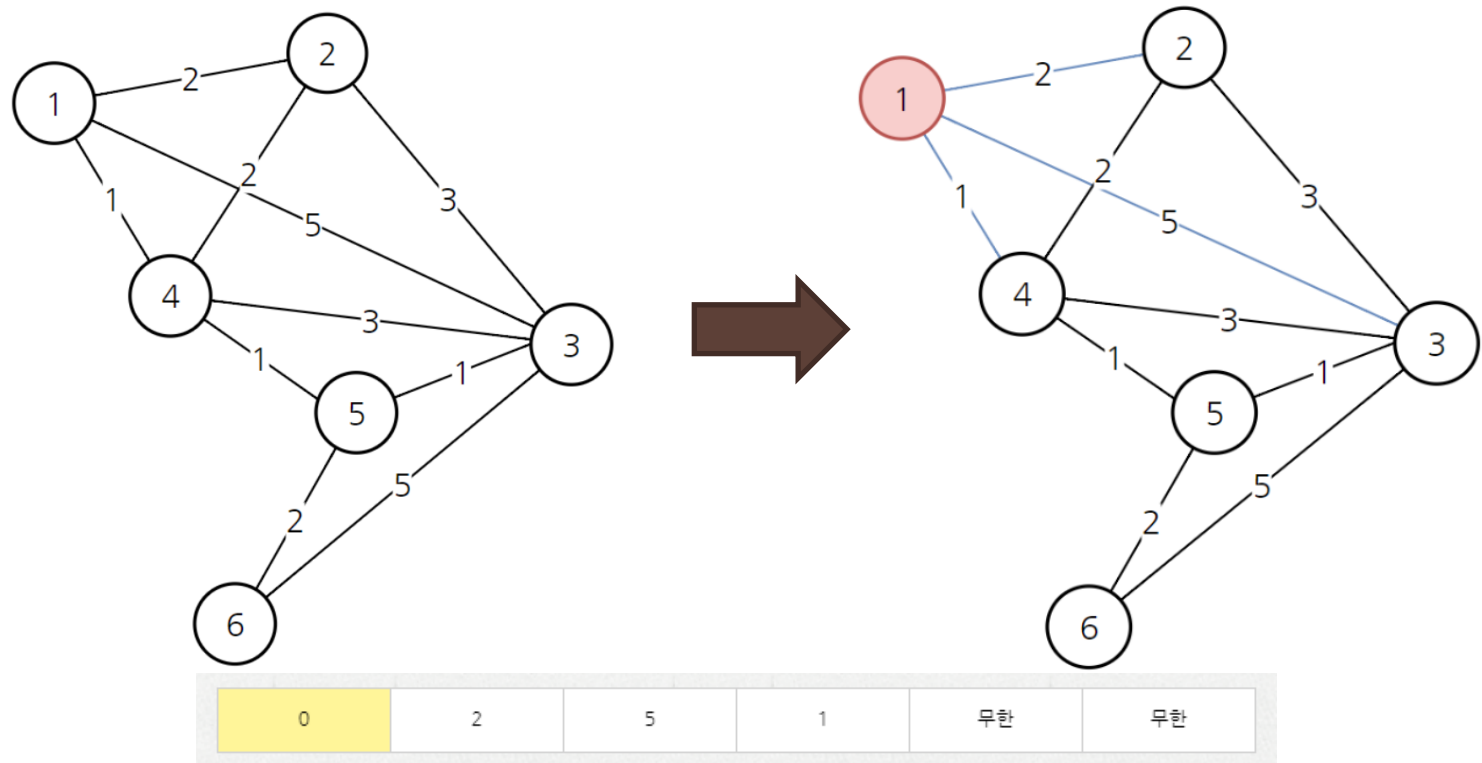
$d[D] = 15$

$d[E] = \text{infinity}$

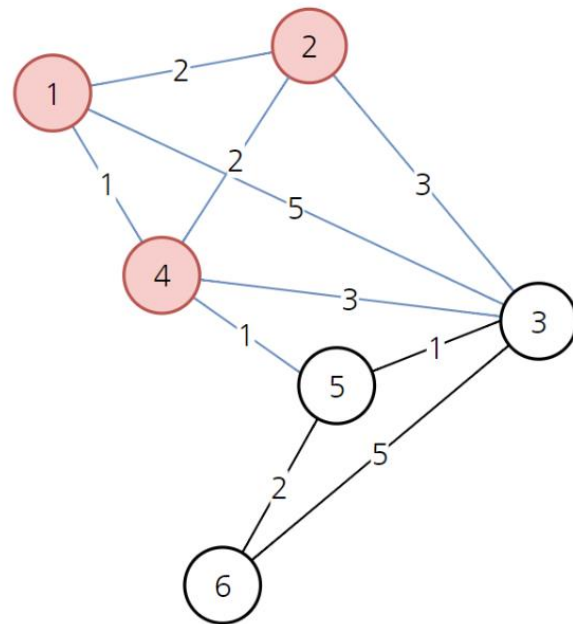
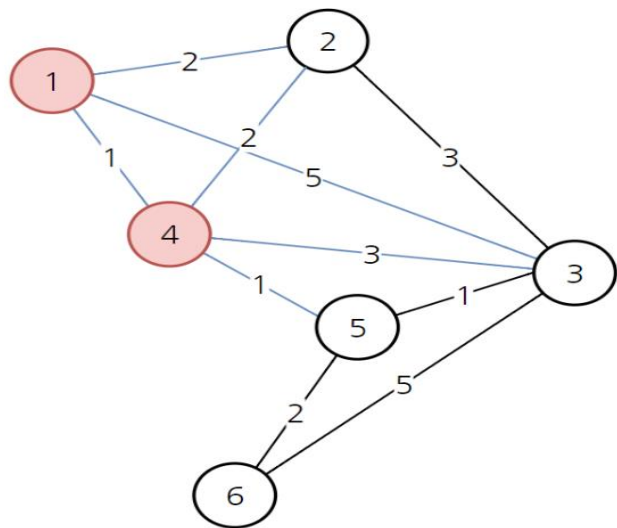
$d[F] = \text{infinity}$

$Q = \{B, C, D, E, F\}$

Dijkstra Algorithm



Dijkstra Algorithm



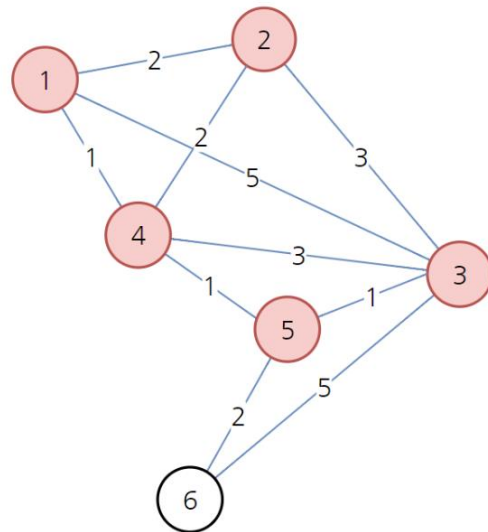
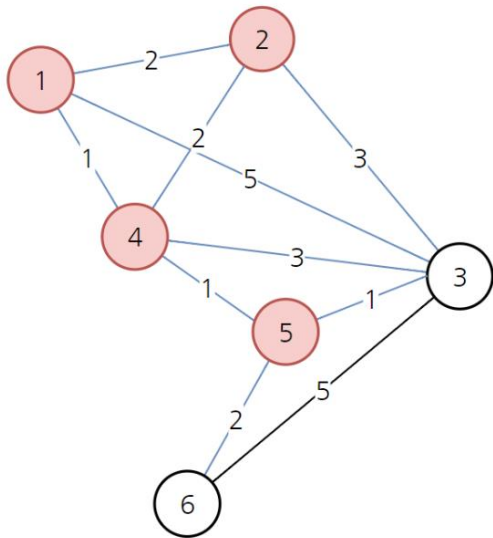
0	2	4	1	2	무한
---	---	---	---	---	----

ppt 제목

5 / n

0	2	4	1	2	무한
---	---	---	---	---	----

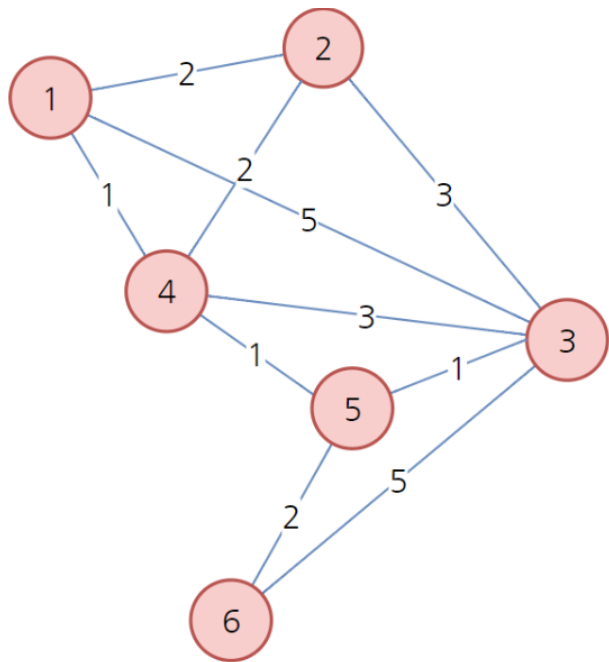
Dijkstra Algorithm



0	2	3	1	2	4
---	---	---	---	---	---

0	2	3	1	2	4
---	---	---	---	---	---

Dijkstra Algorithm



- 노드 1로부터 다른 노드까지의 최단경로길이
- 1-2: 2
- 1-3: 3
- 1-4: 1
- 1-5: 2
- 1-6: 4

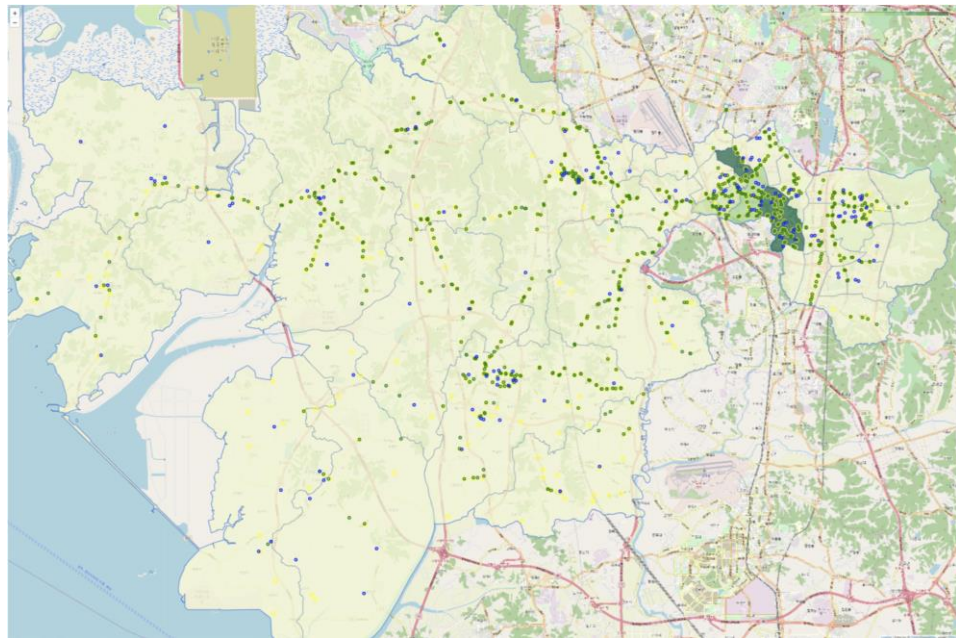
0	2	3	1	2	4
---	---	---	---	---	---

화성시 급행버스

- 큰 대로나 직선구간 위주로만 다닌다
- 여러 정류장에 정차하지 않음
- 일반 버스보다 빠르게 다니는 버스
- 디익스트라 알고리즘을 활용해서 최단 경로를 구하자!



화성시 급행버스



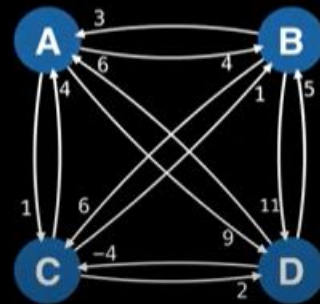
- 동탄1동, 동탄2동, 능동, 기산동, 병점동에 몰려있는 유동인구
- 동탄동의 수요 증가
- 이를 관통하는 급행버스 노선 제안

Travelling Salesman Problem

- 모든 도시들을 단 한 번만 방문 후, 원래 시작점으로 돌아오는 최소 비용의 이동 순서를 구하는 알고리즘

In other words, the problem is: given a **complete graph** with weighted edges (as an adjacency matrix) what is the **Hamiltonian cycle** (path that visits every node once) of minimum cost?

	A	B	C	D
A	0	4	1	9
B	3	0	6	11
C	4	1	0	2
D	6	5	-4	0



TSP Solution1 – Naïve Solution

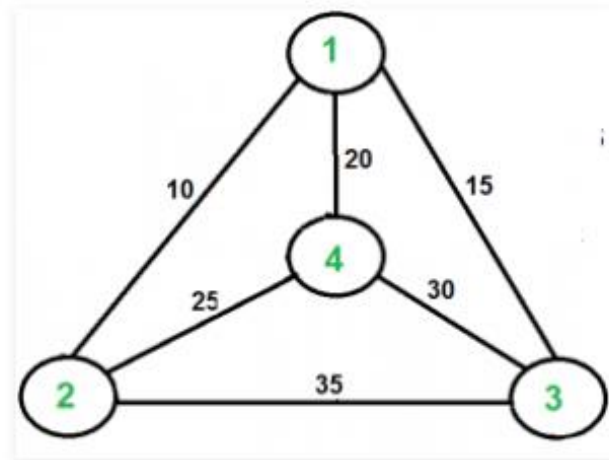
- Naive Solution:

- 1) Consider city 1 as the starting and ending point.
- 2) Generate all $(n-1)!$ Permutations of cities.
- 3) Calculate cost of every permutation and keep track of minimum cost permutation.
- 4) Return the permutation with minimum cost.

- Time Complexity: $\Theta(n!)$

- 10개 탐색 $\rightarrow 3,628,800$

- 20개 탐색 $\rightarrow 2,432,902,176,008,640,000$



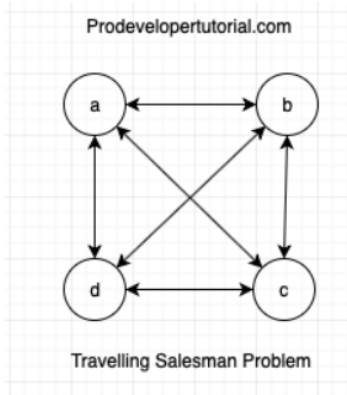
TSP Solution2 – Dynamic Programming

If size of S is 2, then S must be $\{1, i\}$,

$$C(S, i) = \text{dist}(1, i)$$

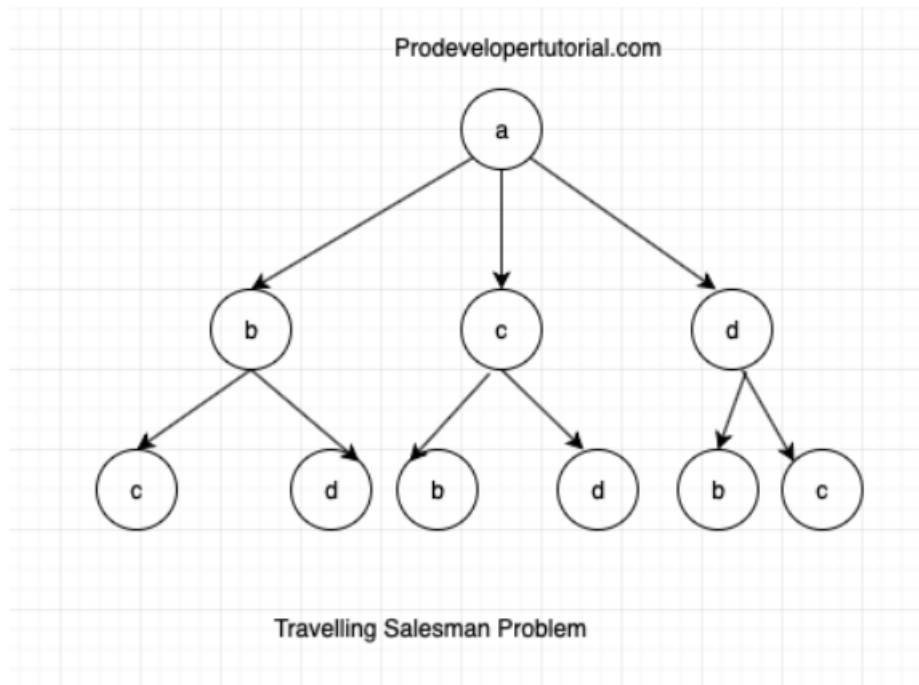
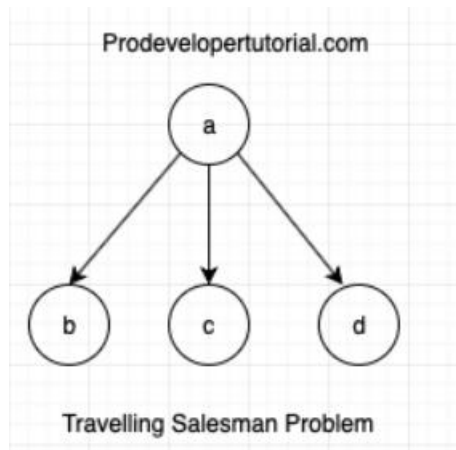
Else if size of S is greater than 2.

$$C(S, i) = \min \{ C(S - \{i\}, j) + \text{dis}(j, i) \} \text{ where } j \text{ belongs to } S, j \neq i \text{ and } j \neq 1.$$

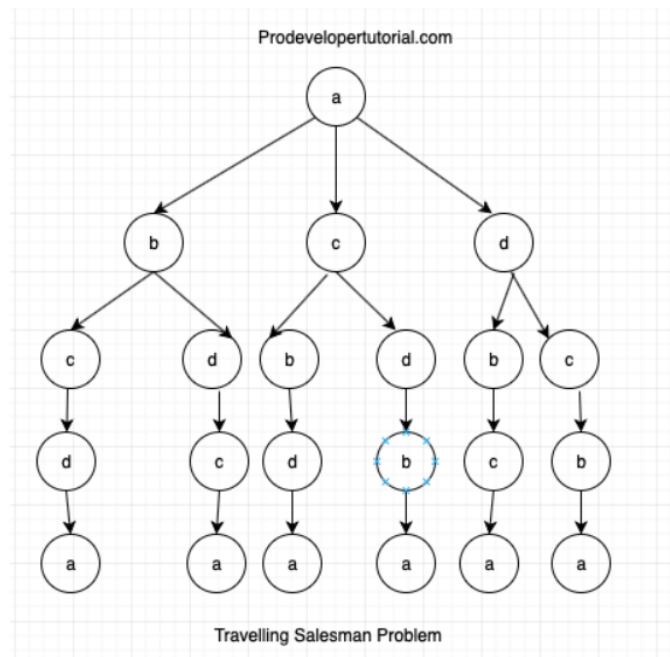
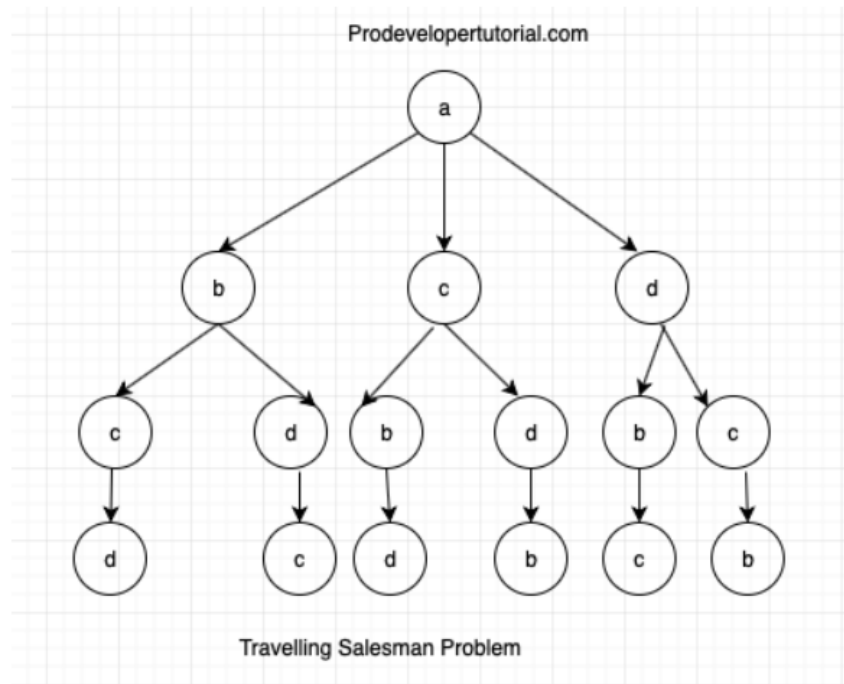


- A에서 시작하여, A로 돌아오는 TSP

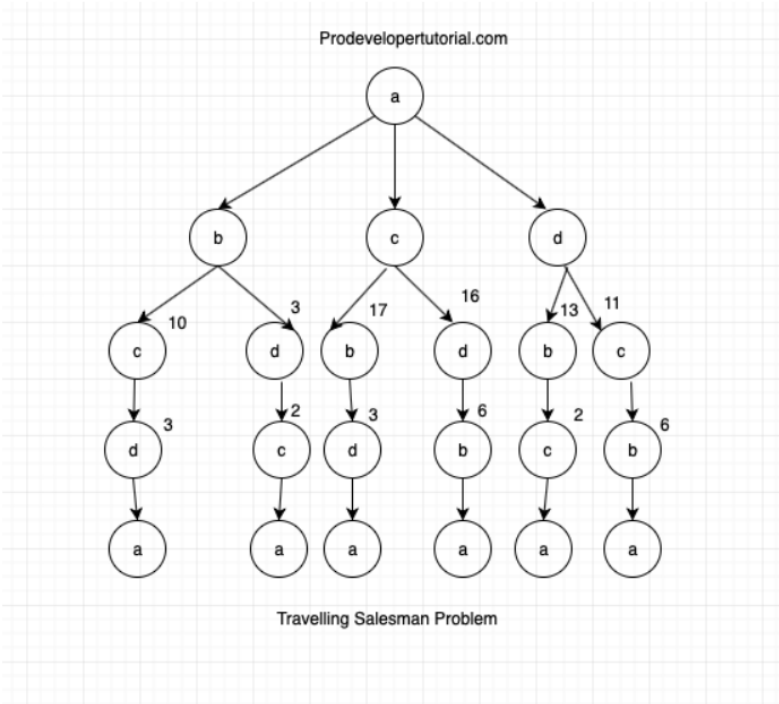
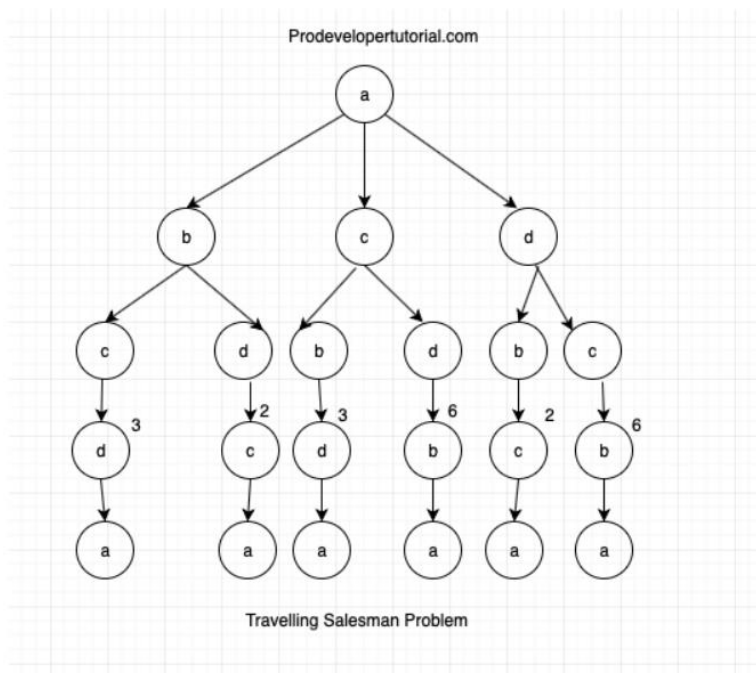
TSP Solution2 – Dynamic Programming



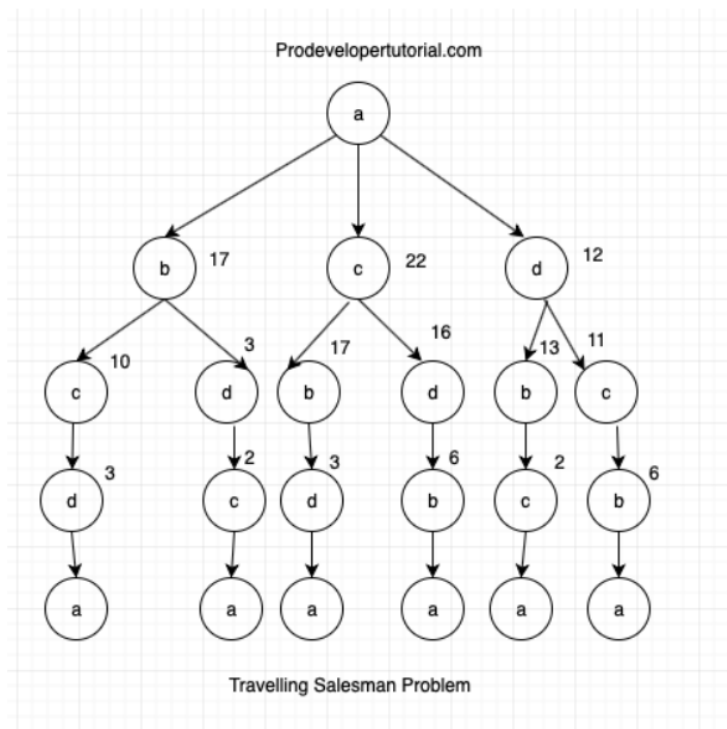
TSP Solution2 – Dynamic Programming



TSP Solution2 – Dynamic Programming



TSP Solution2 – Dynamic Programming



Now we go one last level up and calculate the final path:

$$a \rightarrow b + (b \rightarrow d \rightarrow c \rightarrow a) = 14 + 17 = 31$$

$$a \rightarrow c + (c \rightarrow b \rightarrow d \rightarrow a) = 9 + 22 = 31$$

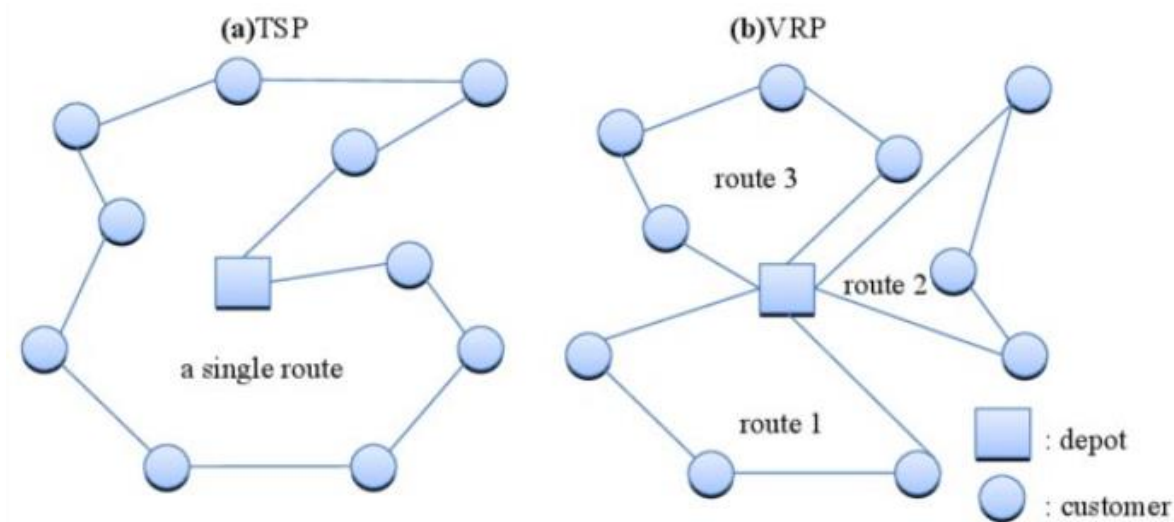
$$a \rightarrow d + (d \rightarrow c \rightarrow b \rightarrow a) = 4 + 12 = 16$$

The min value is 16. Hence our result.

TSP Time Complexity Improvement

n	n!	$n^2 2^n$
1	1	2
2	2	16
3	6	72
4	24	256
5	120	800
6	720	2304
...
15	1307674368000	7372800
16	20922789888000	16777216
17	355687428096000	37879808

Vehicle Route Problem



화성시 다람쥐버스

- 일정 구간을 똑같이 반복하여 도는 버스



350x350

ppt 제목

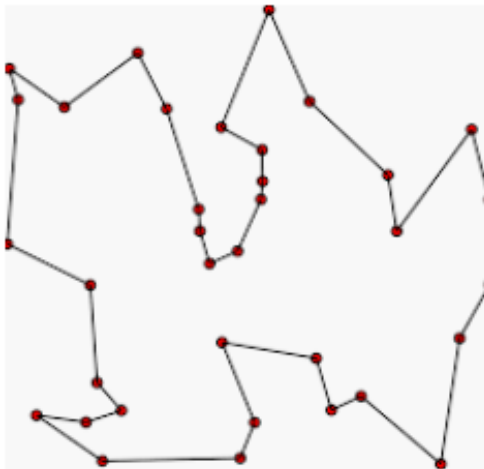
출퇴근 시간의 히어로, 다람쥐버스



19 /
n

화성시 다람쥐버스

- 국토교통부가 주관하는 '2018 지속가능 교통도시평가' 교통정책분야에서 최우수정책으로 선정
- 현재 서울에서 정책 도입 -> 화성시에도 적용해보자!





감사합니다!

BUS LINE
OPTIMIZATION

