

# AI 알고리즘

네트워크 모형(MST, 그리디 알고리즘 등)

## 학습내용

- 그래프의 개념
- 그래프의 표현과 탐색

## 학습목표

- 그래프의 개념과 표현에 대해 이해하고 설명할 수 있다.
- 그래프를 탐색하고 응용할 수 있다.

# 그래프 개요

그래프의 개념

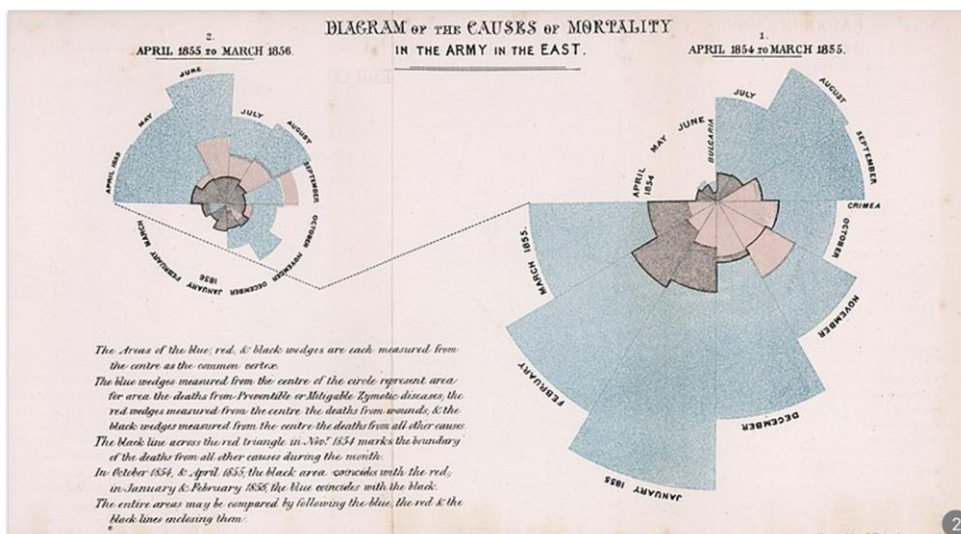
## 그래프의 개념

### ❖ 그래프

- 직선, 곡선, 도형 등 그래픽의 요소에 의해 시각화 된 차트
- 넓은 뜻으로는 다이어그램도 포함
- 함수의 그래프는 주어진 함수가 나타내는 직선 혹은 곡선
- 수학적 측면
  - 꼭짓점의 집합과 변의 집합을 이루는 순서쌍
- 자료 구조적 측면
  - Vertex(정점)와 Edge(정점과 정점을 연결하는 간선)로 구성된 한정된 자료구조 의미
  - 추상적 자료구조
    - 리스트, 스택, 큐, 트리, 그래프

### ❖ 그래프(도표)

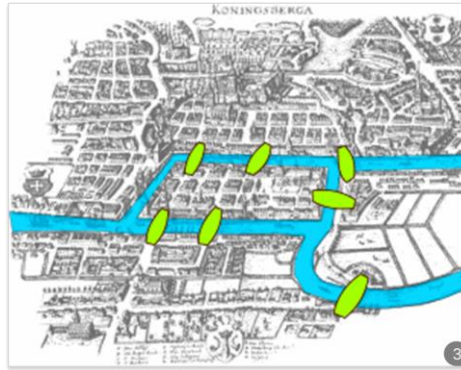
- 플로렌스 나이팅게일
  - 의료계 뿐 아니라 통계학적 학문에서도 유명한 학자
  - 3년 동안 벌어진 크림전쟁에 참전
- 장미 다이어그램
  - 전쟁으로 사망한 사람, 부상당한 사람, 질병으로 죽는 사람들을 조사
    - 의료시스템 개선을 통해 사망률을 42%에서 2%로 낮춤



## 그래프의 개념

### ❖ 그래프(수학)

#### ■ 쾨니히스베르크의 다리



#### ■ 1736년, 오일러

- 그래프 안에서 경로나 순환을 찾는 방법 연구
- 오일러 경로
  - $G(V, E)$ 의 모든 변을 꼭 한 번씩 지나는 경로
- 오일러 회로
  - 꼭짓점과 변으로 이루어진 그래프가 있을 때, 모든 변을 단 한번씩만 통과하는 회로
- 오일러 그래프
  - 순환에 관련된 오일러 회로를 포함하는 오일러 그래프

오일러 경로

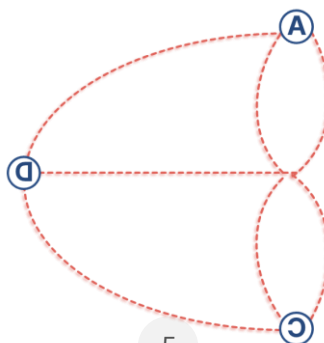
홀수 차수 0개 또는 2개

오일러 그래프

모든 꼭짓점 차수는 짝수

- 홀수 차수는 돌아올 수 없다고 함

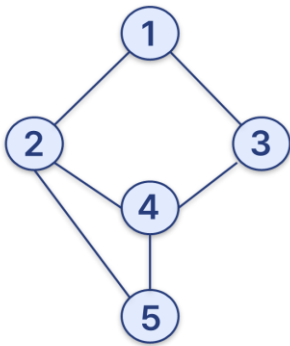
#### ■ 쾨니히스베르크의 다리



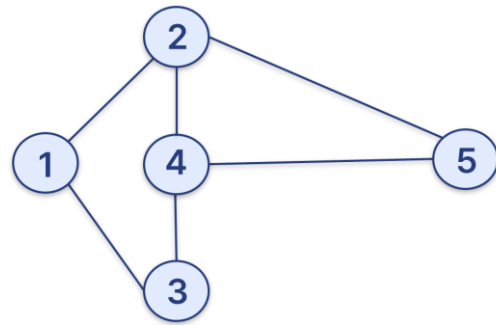
## 그래프의 개념

### ❖ 그래프의 수학적 정의

- 그래프:  $G(V, E)$ 
  - $V(G)$ 
    - 정점(Vertex) 또는 노드(Node)의 집합
  - $E(G)$ 
    - 간선(Edge) 또는 링크(Link)의 집합(정점들 간의 관계 의미)



$$V(G) = \{1, 2, 3, 4, 5\}$$



$$E(G) = \{(1,2), (1,3), (2,4), (2,5), (3,4), (4,5)\}$$

- 그래프 이론으로 정리하면 둘이 똑같은
- 표현하는 방법을 정립하면 다른 상황을 인지할 수 있음

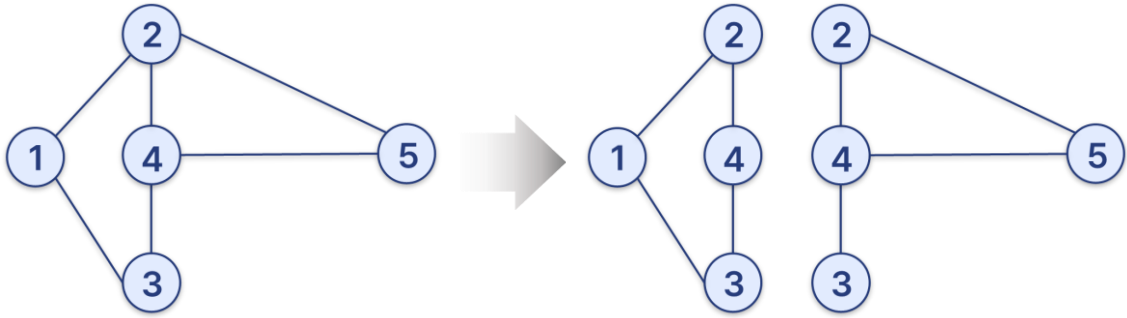
### ❖ 그래프의 종류

- 간선의 종류에 따라 분류한 경우
  - 무방향 그래프(Undirected Graph)
    - $(1, 2) = (2, 1)$
  - 방향 그래프(Directed Graph)
    - $\langle A, B \rangle \neq \langle B, A \rangle$
- 가중치 그래프, 네트워크
  - 간선에 비용이나 가중치가 할당된 그래프
  - 점이 두 개가 있으며 그 둘을 연결하는 선에 수치가 할당된 것을 네트워크라고 함

## 그래프의 개념

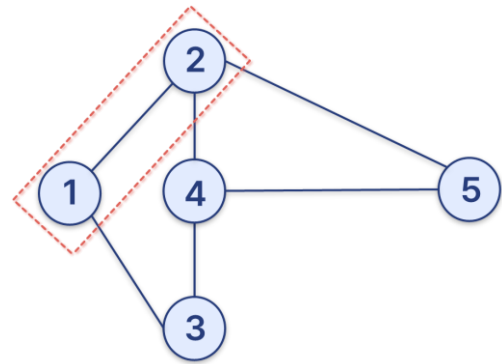
### ❖ 그래프의 수학적 정의

- 부분 그래프
  - 그래프 속에 속하는 부분 그래프

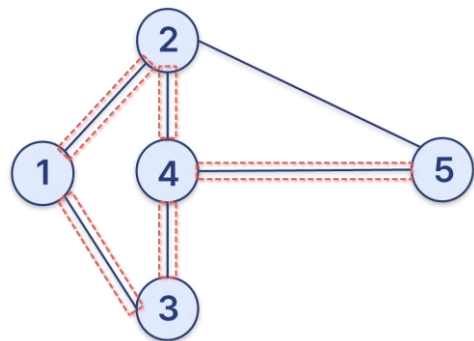


### ❖ 그래프의 용어

- 인접 정점
  - 간선에 의해 직접 연결된 정점
  - 멀리 있어도 어떤 선으로 연결되어 있음



- 차수(Degree)
  - 정점에 연결된 간선의 수
  - 무방향 그래프의 경우
    - 차수의 합은 간선 수의 2배
  - 방향 그래프의 경우
    - 진입 차수, 진출 차수를 모두 계산해야 함
    - 모든 진입(진출) 차수의 합은 간선의 수

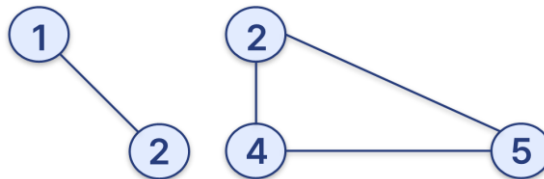


## 그래프의 개념

### ❖ 그래프의 용어

#### ■ 연결 성분

- 그래프에서 정점들이 서로 연결되어 있는 부분



2개 연결 성분  
[1,2], [2,4,5]

#### ■ 경로(Path)

- 무방향 그래프의 정점 1로부터 정점 5까지의 경로

- 정점의 나열 1, 2, 5

- 방향 그래프의 정점 1로부터 정점 5까지의 경로

- 정점의 나열 1, 2, 4, 5

- 반드시 간선  $\langle 1, 2 \rangle$ ,  $\langle 2, 4 \rangle$ ,  $\langle 4, 5 \rangle$  존재해야 함

- 경로의 길이(Length)

- 경로를 구성하는 데 사용된 간선의 수

- 숫자 의미에 따라서 경로 길이 개념이 달라짐

- 단순 경로(Simple Path)

- 경로 중에서 반복되는 정점이 없는 경로(1, 2, 4, 5)

- 사이클(Cycle)

- 시작 정점과 종료 정점이 동일한 경로(1, 2, 4, 3, 1)

- 연결 그래프(Connected Graph)

- 모든 정점들 사이에 경로가 존재하는 그래프

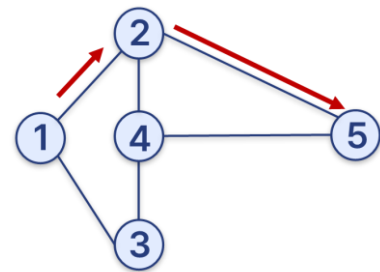
- 트리(Tree)

- 사이클을 가지지 않는 연결 그래프

- 다시 돌아오지 않음

- 완전 그래프(Complete Graph)

- 모든 정점 간에 간선이 존재하는 그래프



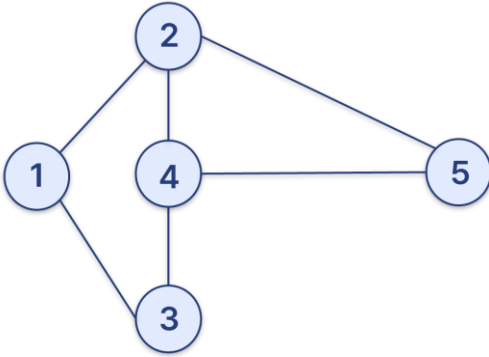


# 그래프 개요

그래프의 표현과 탐색

## 그래프의 표현과 탐색

### ❖ 인접 행렬

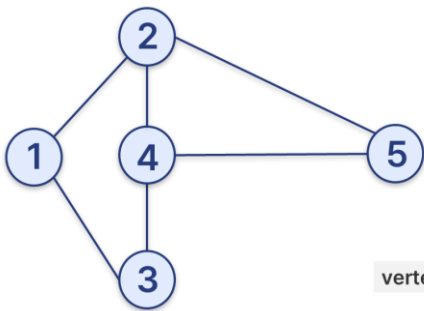


	1	1		
1			1	1
1			1	
	1	1		1
	1		1	

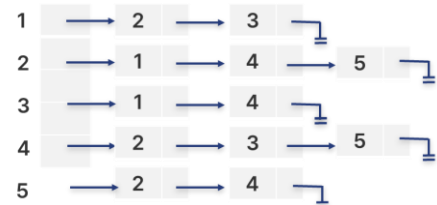
### ❖ 인접 리스트

#### ■ 인접 리스트

- 연결 리스트로 그래프를 표현 가능
- 각 정점에 대하여 정점과 연결된 간선을 한 개의 연결 리스트에 저장
- 전체 연결 리스트는  $n$ 개가 되며, 이 연결 리스트를 배열로 표현



vertex(정점) → link



### ❖ 그 외 그래프의 표현

- 실세계의 그래프는 대부분 정점의 평균 차수가 작은 희소 그래프(Sparse Graph)
- 희소 그래프의 간선 수는 최대 간선 수인  $N(N-1)/2$ 보다 훨씬 작음
- 조밀 그래프(Dense Graph)
  - 간선의 수가 최대 간선 수에 근접한 그래프

## 그래프의 표현과 탐색

### ❖ 그래프 탐색이란

- 그래프 프로그래밍 시에 그래프 자료구조 제작 후 연산 필요
  - 그래프에 관한 연산을 구현 시 사용하는 것이 그래프 탐색
  - 그래프 자료구조
    - 그래프 자료 선언 + 그래프 연산

### ❖ 그래프 ADT(추상 자료형)

- ▶ 데이터란 정점과 간선의 집합

연산	
isEmpty()	그래프가 공백 상태인지 확인
countVertex()	정점의 수
countEdge()	간선의 수
getEdge(u,v)	정점 u에서 정점 v로 연결된 간선
Degree(v)	정점 v의 차수
Adjacent(v)	정점 v에 인접한 모든 정점의 집합을 반환

### ❖ 그래프 탐색의 종류

- 깊이우선탐색, Depth First Search(DFS)
  - 갈 수 있는 데까지 가보는 방문 방법
  - 트리의 전위 탐색 방법을 그래프에 적용한 것(Preorder Tree Traversal)
- 너비우선탐색, Breadth First Search(BFS)
  - 거리가 가까운 곳부터 가보는 방문 방법
  - 트리의 레벨 우선 탐색을 그래프에 적용한 것(Level Order Tree Traversal)

## 그래프의 표현과 탐색

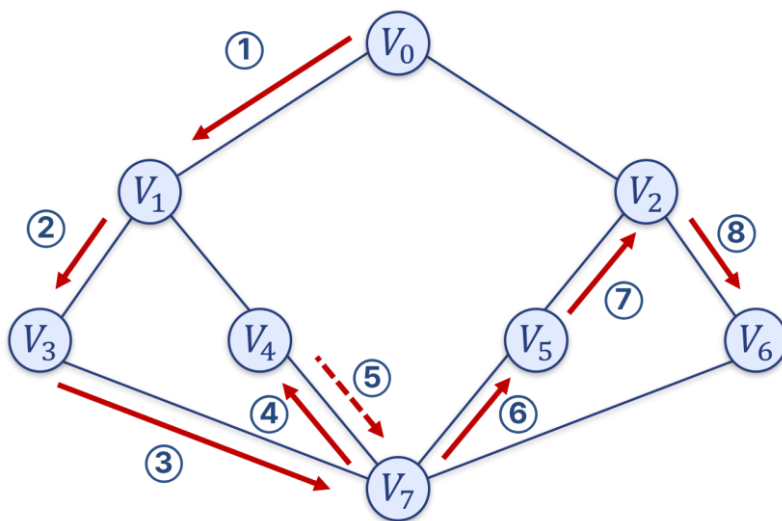
### ❖ 깊이우선탐색

#### ■ 방법

- 더 이상 한 방향으로 진행 불가 시 가장 가까운 갈림길로 돌아와서 다른 방향으로 탐색 진행
  - 탐색중 방문 가능한 곳은 스택자료를 이용하여 저장

#### ■ 절차

- ① 하나의 노드를 택함
- ② 노드를 방문하여 필요한 작업을 한 다음 연결된 다음 노드를 찾고, 현재 방문 노드는 스택에 저장  
2단계를 반복하면서 방문을 계속함
- ③ 더 이상 방문할 노드가 없으면 스택에서 노드를 빼내 다음 방문 노드를 찾아 2번 과정을 다시 반복



### ❖ 너비우선탐색

#### ■ 방법

- 시작 정점으로부터 가까운 정점을 먼저 방문
- 큐 자료구조를 이용하여 다음 방문지를 선정

#### ■ 절차

- ① 하나의 노드를 택함
- ② 노드를 방문하여 필요한 작업을 한 다음 연결된 다음 노드를 찾고, 노드들을 큐에 저장  
더 이상 방문할 곳이 없으면 3단계로 감
- ③ 큐의 맨 앞의 노드를 빼내 2단계를 반복

## 그래프의 표현과 탐색

### ❖ 그래프 탐색의 성능

- 정점의 수  $n$ , 간선의 수  $e$ 라고 할 때

	인접 행렬	인접 리스트
메모리 공간	$n^2$	$n+2e$
<code>getEdge(u, v)</code>	$O(1)$	$O(\text{정점 } u \text{의 차수})$
<code>Degree(u)</code>	$O(n)$	$O(\text{정점 } u \text{의 차수})$
<code>Adjacent(u)</code>	$O(n)$	$O(\text{정점 } u \text{의 차수})$
모든 간선의 수	$O(n^2)$	$O(n+e)$

### ❖ 위상 정렬

- 방향 그래프에 대해 정점들의 선행 순서를 위배하지 않으면서 모든 정점을 나열하는 것
  - 수강신청과 같은 일들을 처리할 때 사용