

# **Reinforcement Learning for Combinatorial Optimization : A Survey**

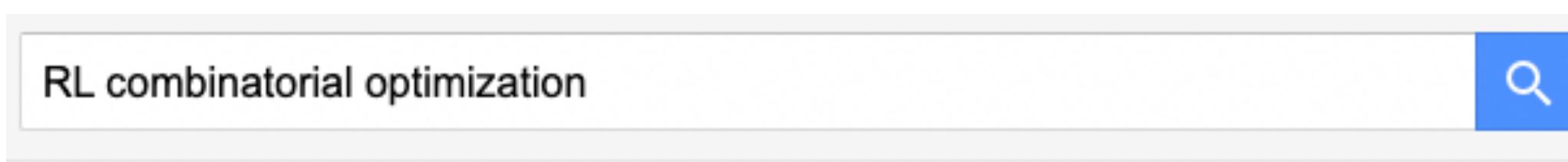
## **Paper review**

**2022.09.08 정규현**

# Paper Review

## 세미나의 목적

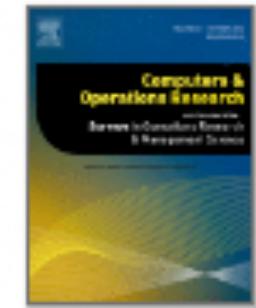
- 강화 학습을 공부했는데
- 산업AI에 어떻게 적용하지?
  - 조합최적화
- survey논문을 읽어서 동향을 파악해보자



검색결과 약 191,000개 (0.10초)



Computers & Operations Research  
Volume 134, October 2021, 105400



Reinforcement learning for combinatorial optimization: A survey ☆

Nina Mazyavkina <sup>a</sup>✉, Sergey Sviridov <sup>b</sup>, Sergei Ivanov <sup>c</sup>, Evgeny Burnaev <sup>a</sup>

[HTML] Reinforcement learning for combinatorial optimization: A survey

N Mazyavkina, S Sviridov, S Ivanov... - Computers & Operations ..., 2021 - Elsevier

... of the **RL** field to solve CO problems. Although many practical **combinatorial optimization** ...  
operations research community, we will focus on **RL** approaches for CO problems. This survey ...

☆ 저장 99 인용 159회 인용 관련 학술자료 전체 6개의 버전

# **Combinatorial Optimization ?**

# RL for CO

## Combinatorial Optimization ?

- **Combinatorial Optimization(CO) problem** : 최적화의 한 분야
  - 탐색공간에서 최적의 해를 찾는 문제
    - discrete space의 최적화 문제
    - continuous space의 최적화 문제와는 다른 유형의 솔루션을 가짐
  - 다양한 산업분야에서 다루고 있는 문제
    - 기존 최적화 알고리즘에는 문제마다 다른 휴리스틱을 사용하는 것이 포함
      - 휴리스틱 : 도메인 전문가에 의해 설계됨
    - RL은 agent를 훈련하여 이러한 휴리스틱 방법들을 자동화 가능

# RL for CO

## Combinatorial Optimization ?

- 대표적으로 많이 사용되는 예제들이 존재
- Canonical example of CO
  - Traveling Salesman Problem
  - Maximum Cut Problem
  - Bin Packing Problem
  - Minimum Vertex Cover
  - Maximum Independent Set

# RL for CO

## Canonical example of CO

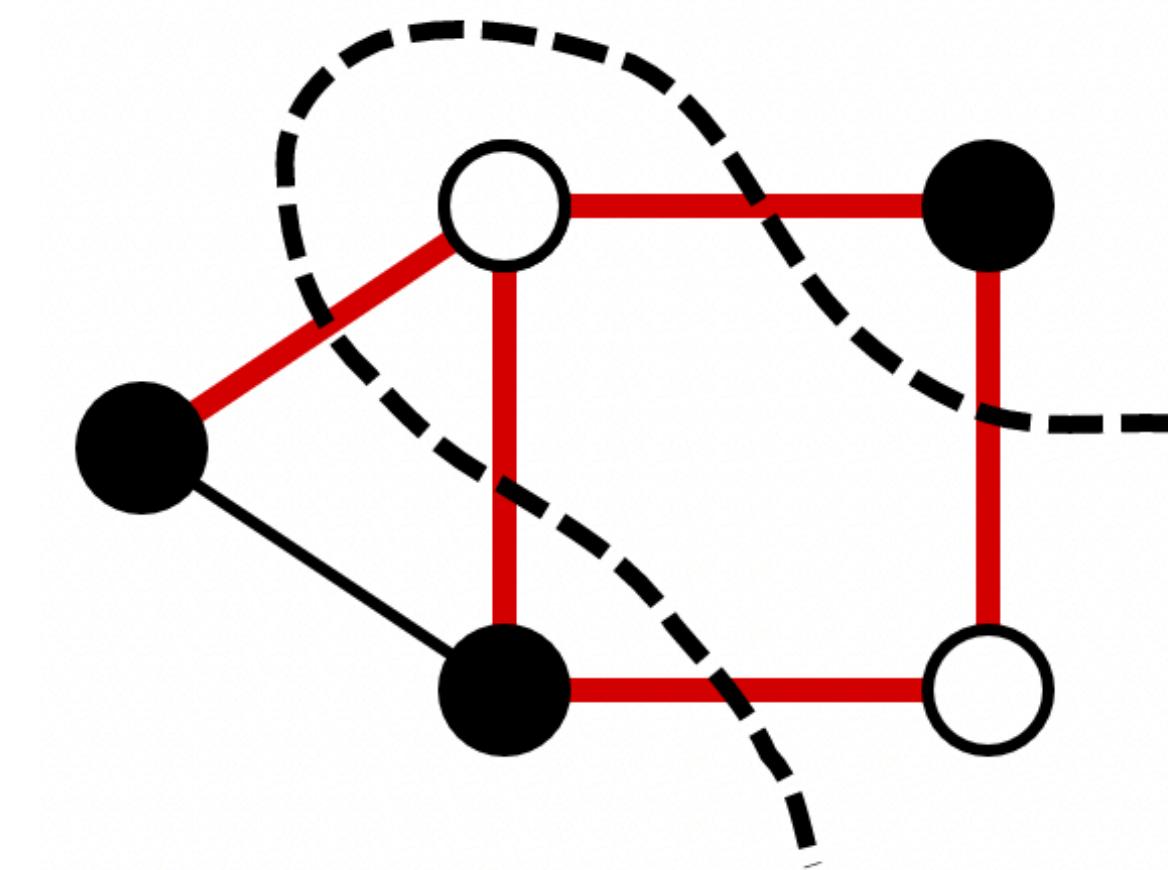
- **Traveling Salesman Problem (TSP)**
- CO문제 중 표준 예제로 유명
- 어떤 외판원이  $n$ 개의 도시를 방문할 계획을 세울 때
  - 각 도시는 다른 도시와 연결되어 있음
  - 출장비용을 최소로 줄이기 위한 최적 경로는?
- 활용 분야
  - Routing 문제 등..



# RL for CO

## Canonical example of CO

- **Maximum Cut Problem (MCP)**
- Cut : 그래프의 node들을 2개의 서로 다른 set으로 나누는 것
- Maximum Cut Problem
  - Cut을 통해 2개의 서로 다른 set으로 나눌 때
  - 잘리는 edge의 수를 최대화 하는 문제



# RL for CO

# Canonical example of CO

- # • Bin Packing Problem(BP)

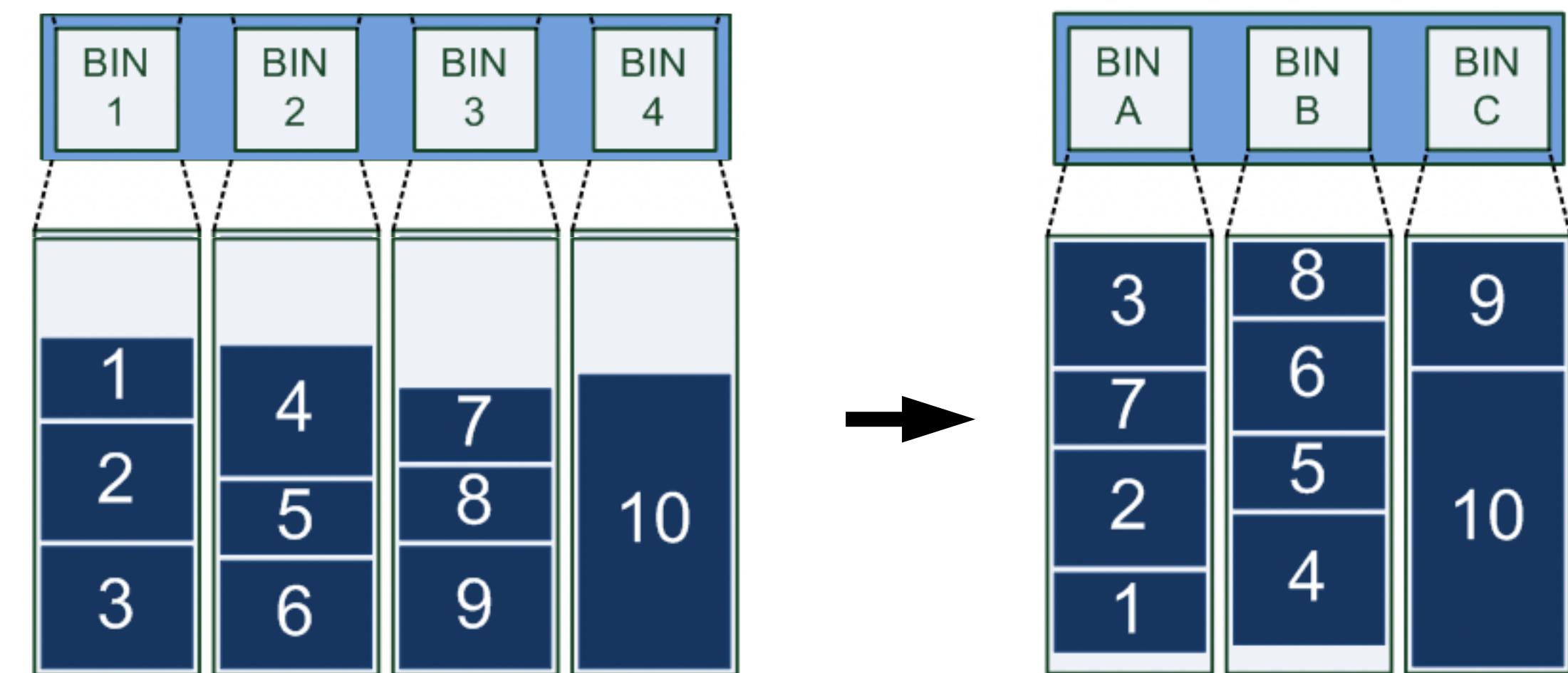
- 상자 채우기 문제
  - 최소한의 공간에 최대한의 아이템을 채워넣는 문제

- 할양부야

- 1차원 : 메모리 할당

- 2차원 : 그래픽 리소스 최적화

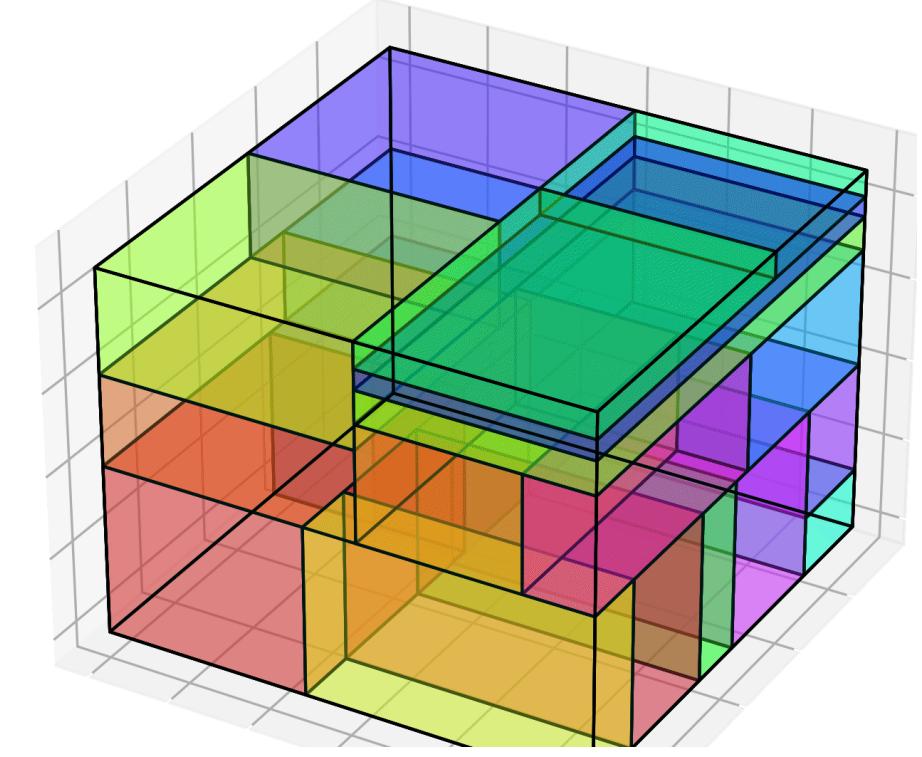
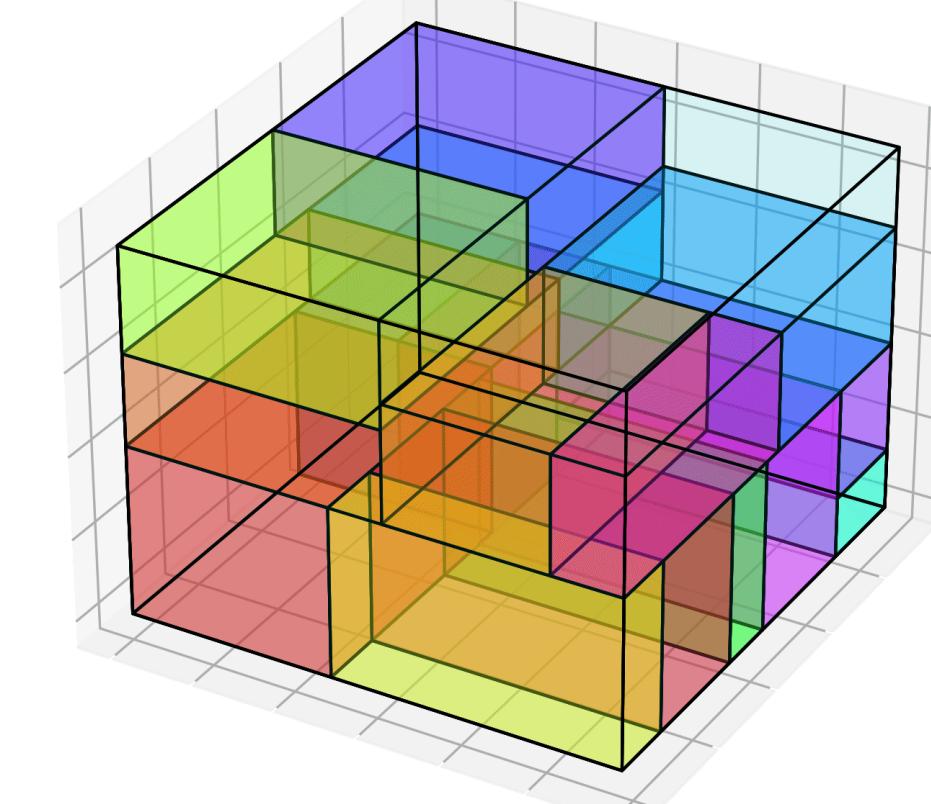
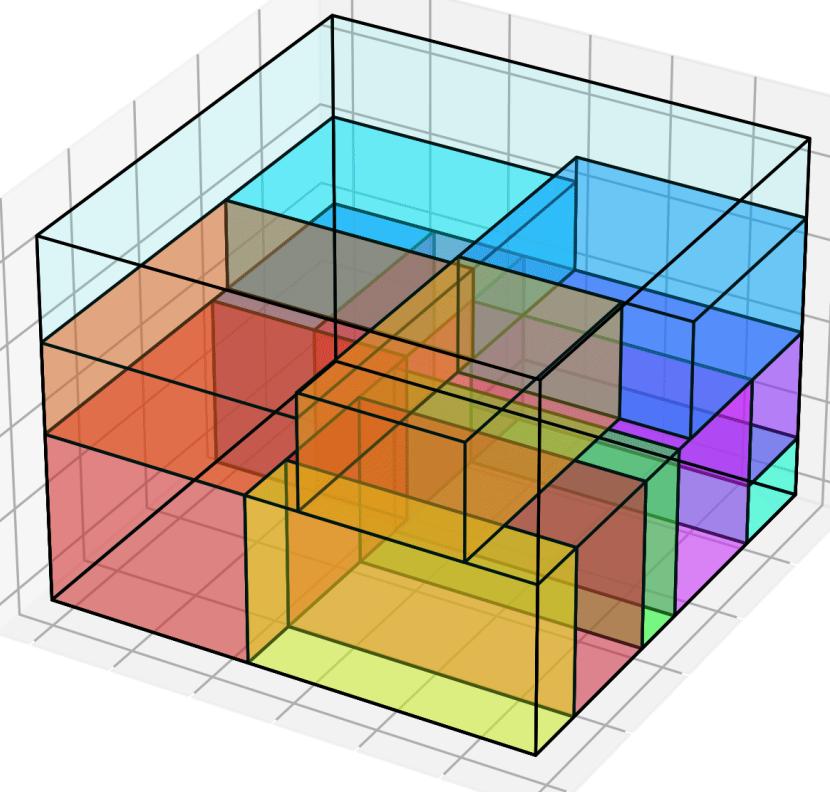
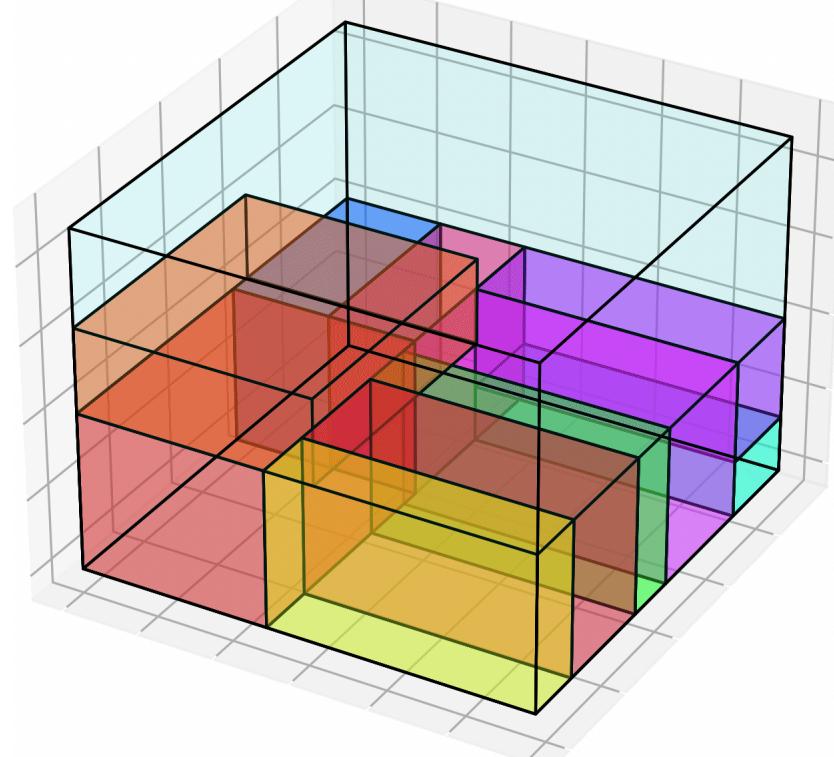
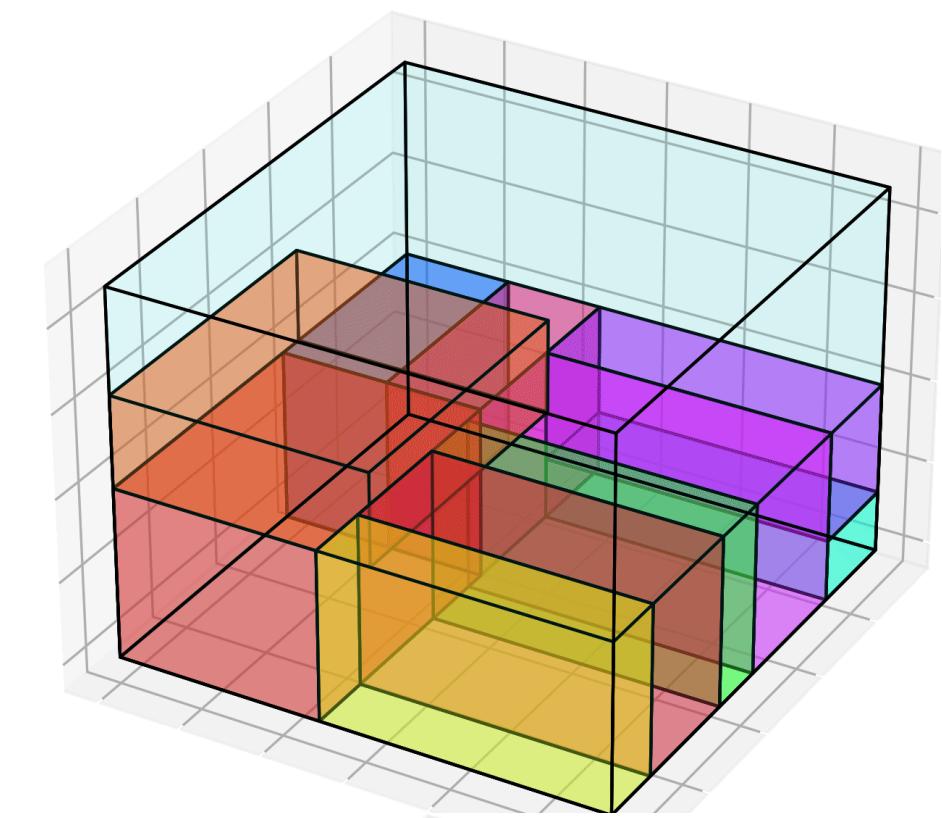
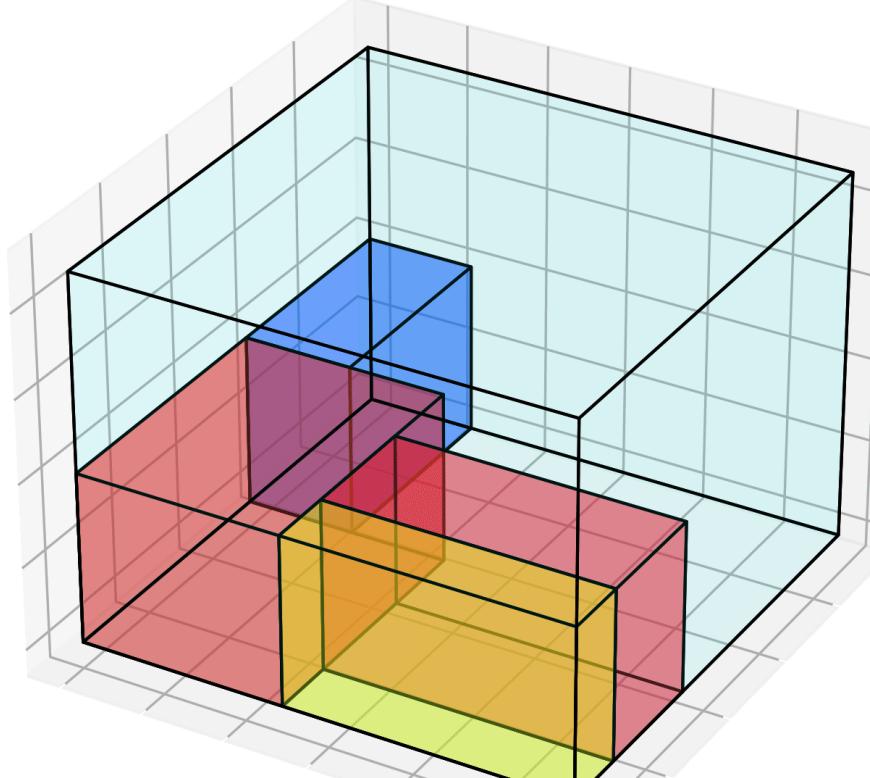
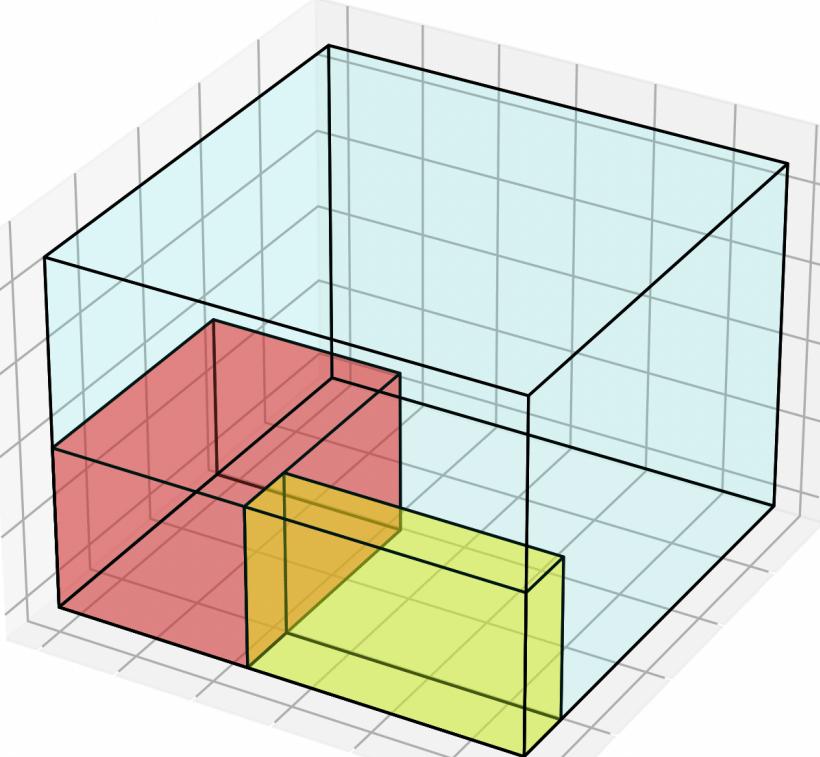
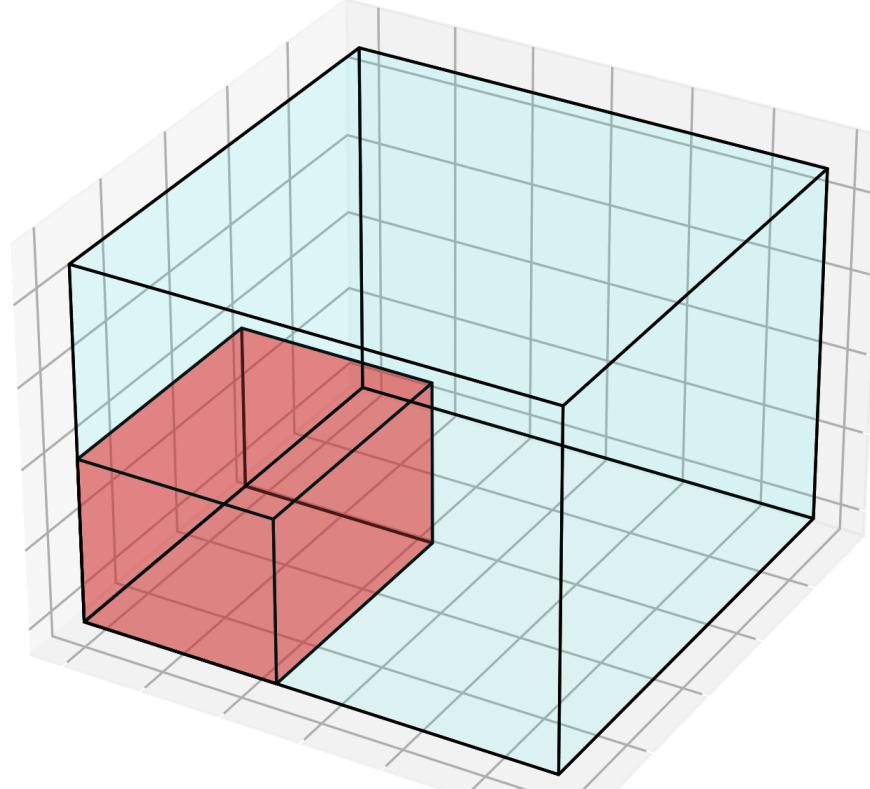
- 3차원 : 화물 적재



# RL for CO

## Canonical example of CO

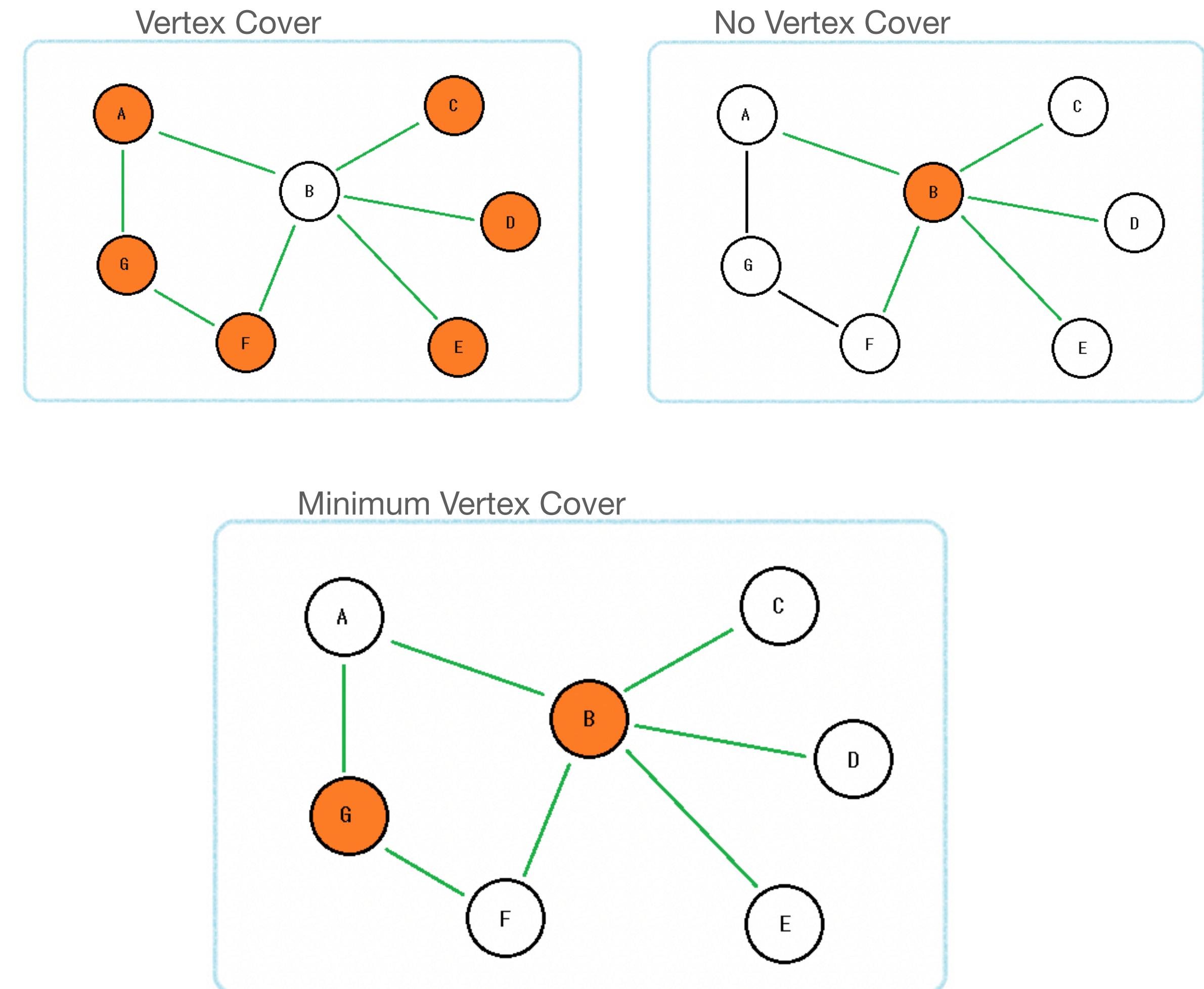
- **Bin Packing Problem(BPP)**
- **3D binpacking example**



# RL for CO

## Canonical example of CO

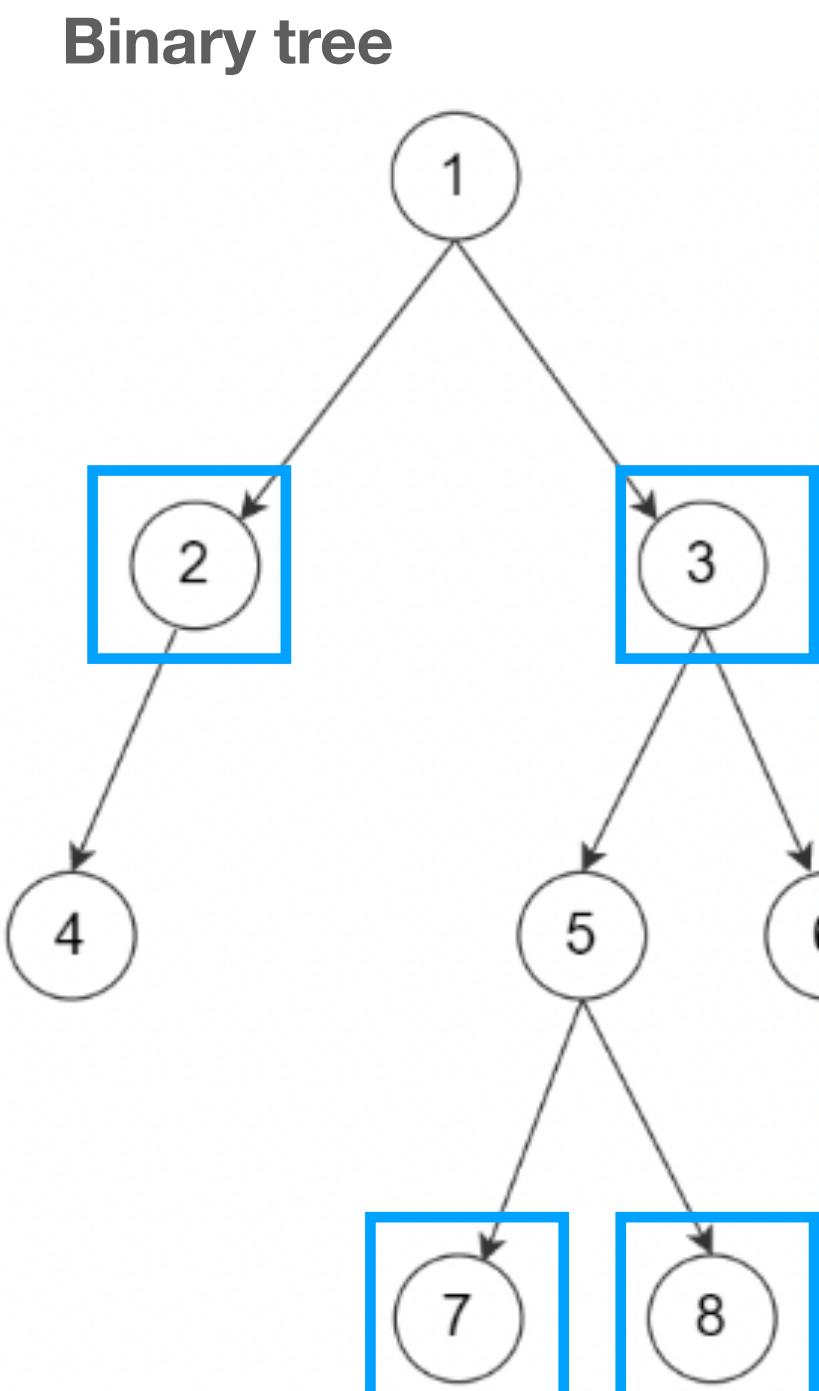
- **Minimum Vertex Cover Problem(MVCP)**
- Vertex Cover
  - node(vertex)의 집합 S가 있을 때, 모든 간선은 양 끝 node 중 하나가 S에 포함되어야 함
- Minimum Vertex Cover
  - node의 집합 S가 있을 때, 모든 간선은 양 끝 node 중 적어도 하나가 S에 포함되어야 함
  - 모든 간선이 node에 적어도 하나가 연결되어야 하는지 확인



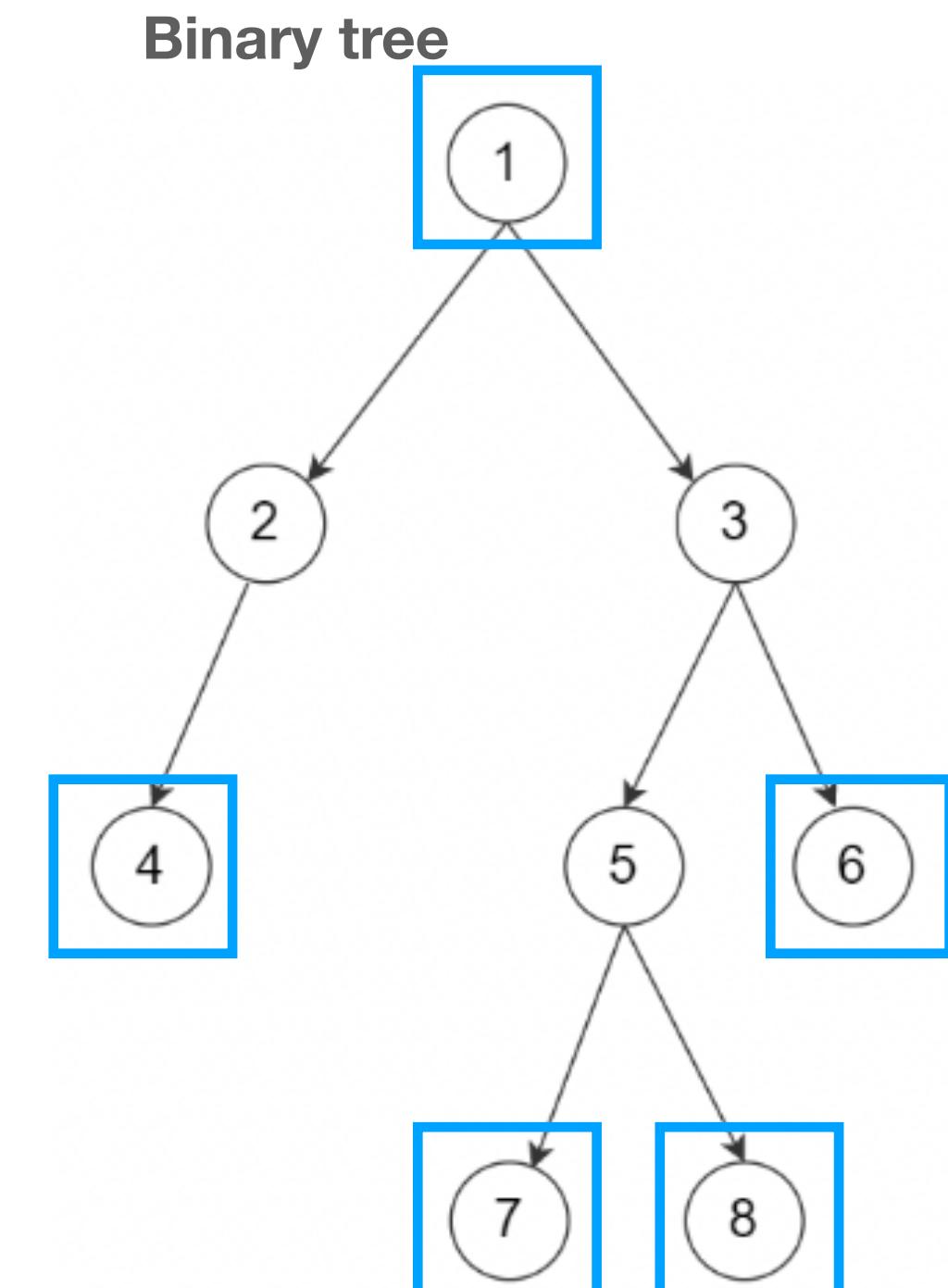
# RL for CO

## Canonical example of CO

- **Maximum Independent Set Problem(MISP)**
- **Independent Set**
  - Binary tree의 node 집합, 두 node가 인접하지 않음
  - 주어진 Binary tree에 대해 가능한 가장 큰 크기의 Independent Set을 찾는 문제



Independent set : 2,3,7,8



Maximum Independent set :  
1,4,6,7,8

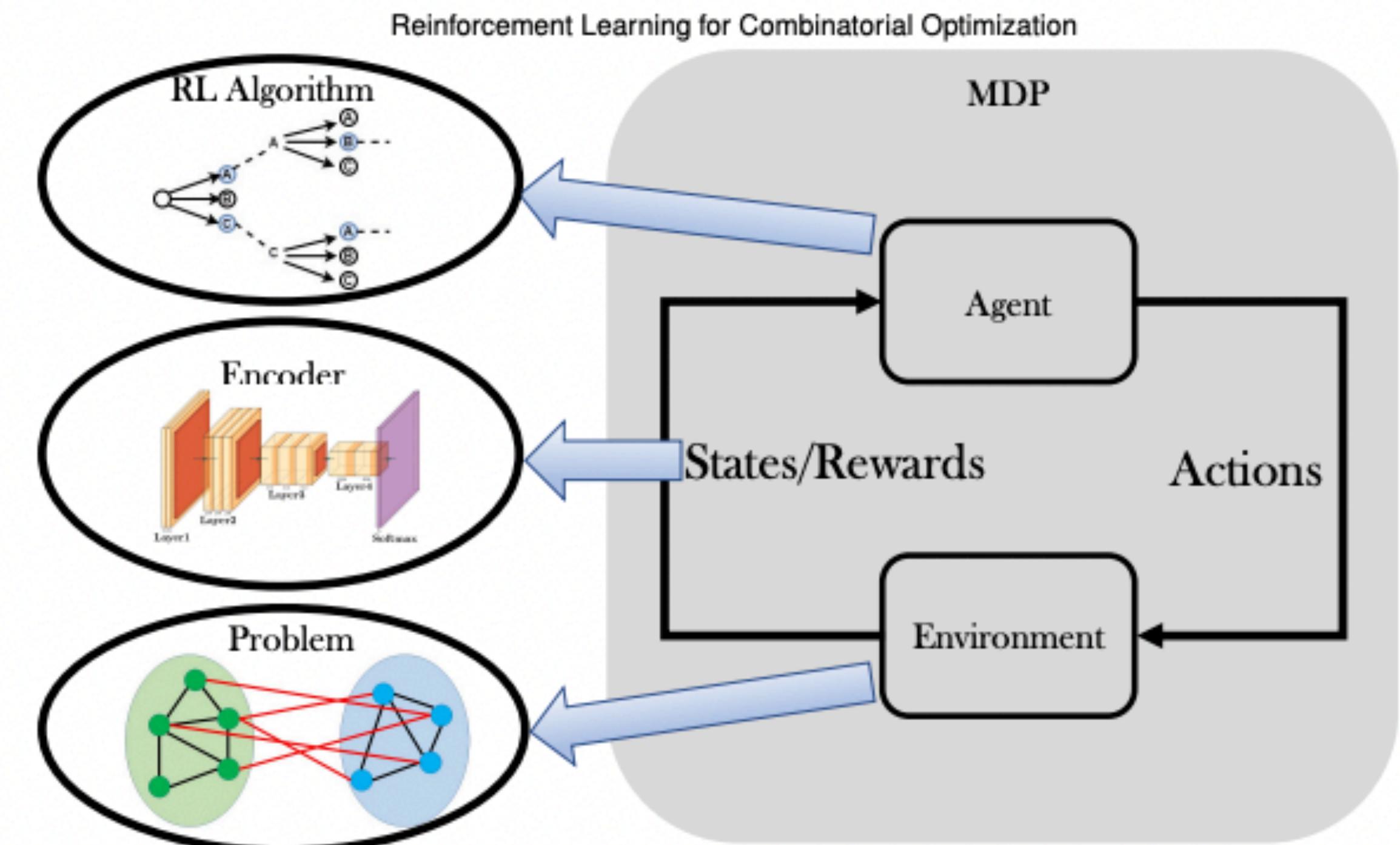
# **CO문제를 풀 때 강화학습을 적용하는 방법**

# RL for CO

## Pipeline for solving CO with RL

### 우리가 풀어야 할 문제

- **Environment : MDP**
  - State / Reward : Environment에 Action을 주고 얻음
- **RL algorithm**
  - State, Reward 등을 통해 Action, state에 대한 가치를 구함
  - optimal policy를 선택함 - 어떤 Action을 할지
- **Encoder**
  - 고차원문제의 input space를 encoding하는데 활용



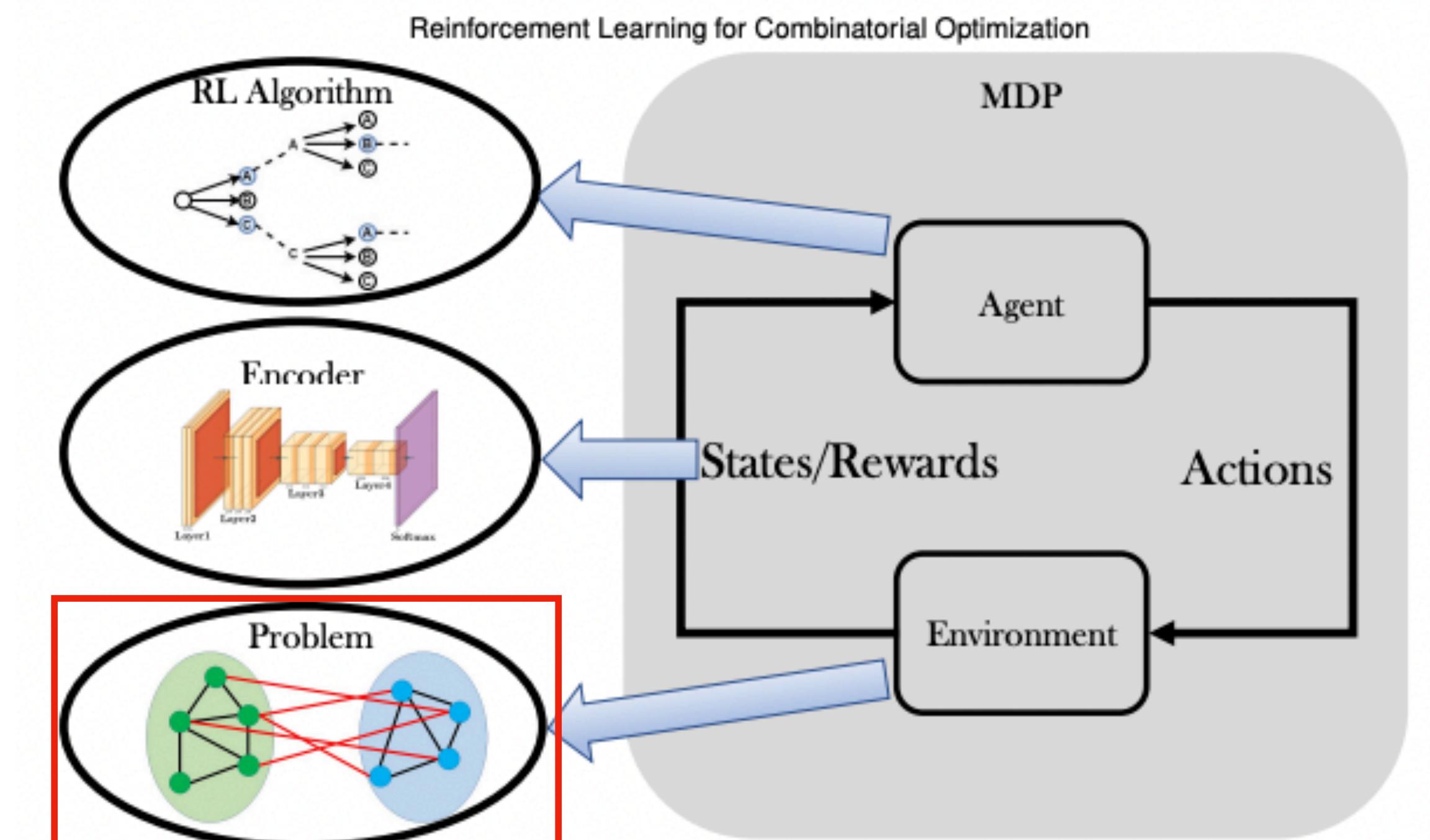
**RL 보충 잠깐**

# RL for CO

## Environment : MDP

### 강화학습 문제

- 주어진 상황이 있을 때, 가장 최적의 행동은?
- **Agent**가 환경과의 상호작용(시행착오)을 통해 목표를 달성하는 방법을 배우는 문제
- 순차적 의사결정 문제(**Sequential decision making**)
  - 행동을 하면, 그로 인해 상황이 바뀌고, 다시 어떤 행동을 하고..
  - 연속적인 행동을 잘 선택해야 하는 문제
  - **Agent**의 **action**에 따라 뒤에 받게될 데이터가 달라짐
- CO문제도 마찬가지!



# RL for CO

## Environment : MDP

- 강화학습은 순차적 의사결정 문제를 푸는 것 !

- 순차적 의사결정 문제를 수학적 모델 **MDP**로 표현

- **MP**(Markov process/Markov Chain)

- 미래의 state는 바로 직전 state에만 영향을 받음

- **MRP**(Markov Reward Process) : MP에 reward 개념이 추가

- **MDP** : MRP에 **Action** 추가

- 미래의 state는 바로 직전 state + action에만 영향을 받음

- state가 있다면 history는 필요 없다

- state는 미래에 대한 충분한 표현형이다

MP



MRP



MDP



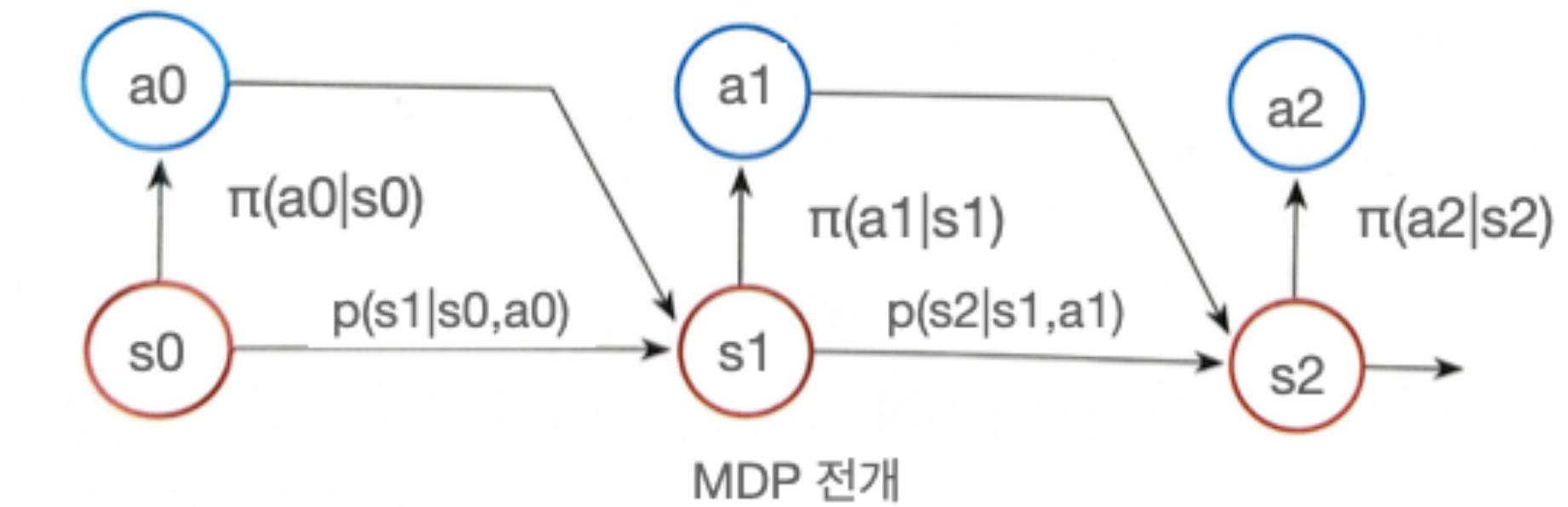
# Reinforcement Learning

## MDP(Markov Decision Process)

- 강화학습은 **MDP** 문제를 푸는것
  - MDP에서 Reward를 Maximize 하는 것
  - **policy** (정책)
    - $\pi(a_t | s_t) = P(a_t | s_t)$
  - **State transition probability**(상태 전이 확률) - transition
    - 이전 state, action을 취할때 다음 state로 이동할 확률
    - $P(s_t | s_{t-1}, a_{t-1})$
- MDP의 수학적 특징

$$1. P(a_1 | \cancel{s_0, a_0}, s_1) = P(a_1 | s_1)$$

$$2. P(s_2 | \cancel{s_0, a_0}, s_1, a_1) = P(s_2 | s_1, a_1)$$



$s_0, a_0, s_1, a_1, s_2, a_2, \dots$

**MDP로 구성했기에 Value function, 강화학습 알고리즘을 사용 가능!**

# Reinforcement Learning

## MDP(Markov Decision Process)

- Return = Expected Reward

- Value function in MDP

- State-Value function : 현재 state의 가치

- 현재 state부터 기대되는 Return

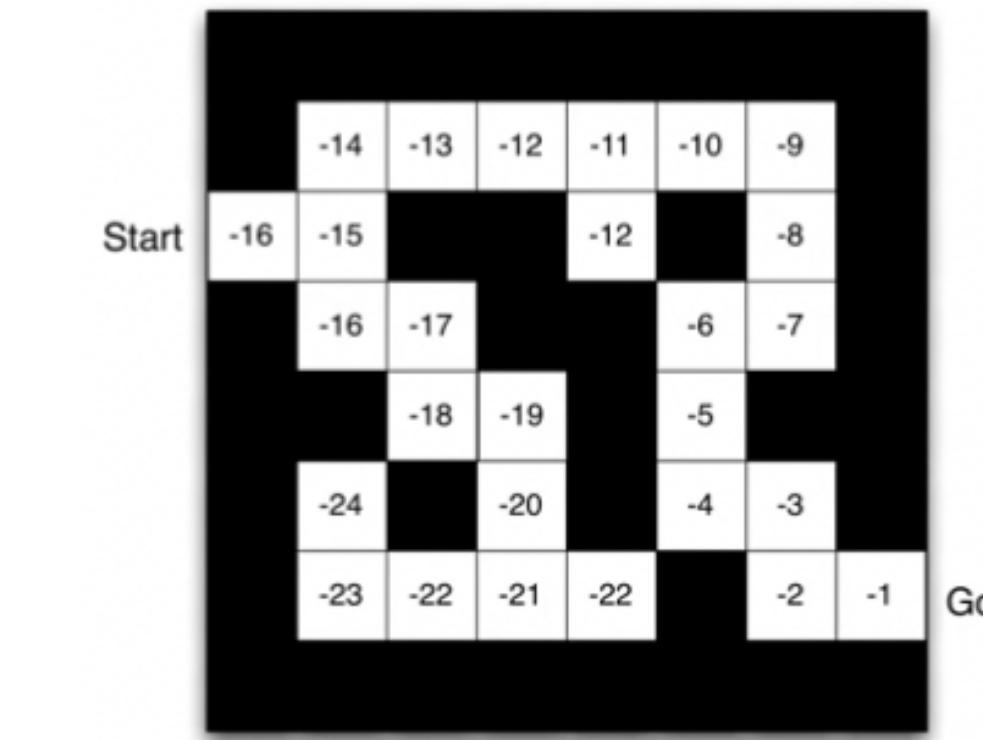
- Action-Value function : 현재 action의 가치

- 현재 Action으로 부터 기대되는 Return

- Optimal Policy

- state-value function을 최대로 하는 policy

- 현재 state부터 기대되는 Return을 최대로 하는 policy



~~-11  
-11 -14  
-12~~

Discount factor

Return  $G_t = R_t + \boxed{\gamma} R_{t+1} + \boxed{\gamma^2} R_{t+2} + \dots$

$$E[G(t)] = \int G(t)p(t)dx$$

State-value function : 현재 state  $S_t$ 부터 모든 action을 다했을 때 기대되는 Return

$$V(S_t) = \int_{a_t:a_\infty} G_t p(a_t, S_{t+1}, a_{t+1}, \dots | S_t) da_t : a_\infty$$

action-value function :  $S_t$ 에서 action  $a_t$ 를 했을 때부터 기대되는 Return

$$Q(S_t, a_t) = \int_{S_{t+1}:a_\infty} G_t p(S_{t+1}, a_{t+1}, S_{t+2}, a_{t+2}, \dots | S_t, a_t) ds_{t+1} : a_\infty$$

**Optimal Policy**

$V(S_t)$ 을 maximize 하는 Policy  $P(a_t|S_t), P(a_{t+1}|S_{t+1}), P(a_\infty|S_\infty)$

# Reinforcement Learning

## Bellman equation

- 벨만 방정식

- 강화학습에서 자주 등장하는 방정식
- V와 Q사이의 관계를 방정식으로 표현
  - V를 Q로 표현

- V를 next V로 표현
- Q를 next V로 표현
- Q를 next Q로 표현
- Q-Learning, SARSA의 motive

Bayesian rule  $p(x, y) = p(x|y)p(y)$      $p(x, y|z) = p(x|y, z)p(y|z)$

$$G_t = R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \dots = R_t + \gamma G_{t+1}$$

action-value function : St에서 action at를 했을 때부터 기대되는 Return

$$Q(S_t, a_t) = \int_{S_{t+1}:a_\infty} G_t p(S_{t+1}, a_{t+1}, S_{t+2}, a_{t+2}, \dots | S_t, a_t) dS_{t+1} : a_\infty$$

State-value function : 현재 state St부터 모든 action을 다했을 때 기대되는 Return

$$V(S_t) = \int_{a_t:a_\infty} G_t p(a_t, S_{t+1}, a_{t+1}, \dots | S_t) d_{a_t:a_\infty}$$

$$p(S_{t+1}, a_{t+1}, \dots | S_t, a_t) p(a_t | S_t)$$

$$p(a_{t+1}, \dots | S_t, a_t, S_{t+1}) p(a_t, S_{t+1} | S_t)$$

$$V(S_t) = \int_{a_t} \int_{S_{t+1}:a_\infty} G_t p(S_{t+1}, a_{t+1}, \dots | S_t, a_t) d_{S_{t+1}:a_\infty} p(a_t | S_t) da_t$$

V를 Q로 표현  $V(S_t) = \int_{a_t} Q(S_t, a_t) p(a_t | S_t) da_t$  State-value function  
= action-value function의 기댓값

$$V(S_t) = \int_{a_t, S_{t+1}} \int_{a_{t+1}:a_\infty} (R_t + \gamma G_{t+1}) p(a_{t+1}, \dots | S_{t+1}) d_{a_{t+1}:a_\infty} p(a_t, S_{t+1} | S_t) d_{a_t, S_{t+1}}$$

$$V(S_t) = \int_{a_t, S_{t+1}} (R_t + \gamma V(S_{t+1})) p(a_t, S_{t+1} | S_t) d_{a_t, S_{t+1}}$$

$$p(S_{t+1} | S_t, a_t) p(a_t | S_t)$$

Transition :  
environment에  
서 주어짐  
Policy :  
전체를 maximize하는  
policy를 찾아야 함

# Reinforcement Learning

## Bellman equation

- 벨만 방정식

- 강화학습에서 자주 등장하는 방정식
- V와 Q사이의 관계를 방정식으로 표현
  - V를 Q로 표현

- V를 next V로 표현
- Q를 next V로 표현
- Q를 next Q로 표현
- Q-Learning, SARSA의 motive

$$\text{Bayesian rule } p(x, y) = p(x|y)p(y) \quad p(x, y|z) = p(x|y, z)p(y|z)$$

$$G_t = R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \dots = R_t + \gamma G_{t+1}$$

State-value function : 현재 state  $S_t$ 부터 모든 action을 다했을 때 기대되는 Return

$$V(S_t) = \int_{a_t:a_\infty} G_t p(a_t, S_{t+1}, a_{t+1}, \dots | S_t) d_{a_t:a_\infty}$$

$$p(S_{t+1}, a_{t+1}, \dots | S_t, a_t) p(a_t | S_t)$$

$$p(a_{t+1}, \dots | S_t, a_t, S_{t+1}) p(a_t, S_{t+1} | S_t)$$

action-value function :  $S_t$ 에서 action  $a_t$ 를 했을 때부터 기대되는 Return

$$Q(S_t, a_t) = \int_{S_{t+1}:a_\infty} G_t p(S_{t+1}, a_{t+1}, S_{t+2}, a_{t+2}, \dots | S_t, a_t) d_{S_{t+1}:a_\infty} : a_\infty$$

$$p(a_{t+1}, S_{t+2}, a_{t+2}, \dots | S_t, a_t, S_{t+1}) p(S_t | S_{t+1}, a_t)$$

$$Q(S_t, a_t) = \int_{S_{t+1}} \left[ \int_{a_{t+1}:a_\infty} (R_t + \gamma G_{t+1}) p(a_{t+1} : a_\infty | S_{t+1}) d_{a_{t+1}:a_\infty} \right] p(S_{t+1} | S_t, a_t) d_{S_{t+1}}$$

$$Q(S_t, a_t) = \int_{S_{t+1}} (R_t + \gamma V(S_{t+1})) p(S_{t+1} | S_t, a_t) d_{S_{t+1}} \quad Q \text{를 next V로 표현}$$

$$Q(S_t, a_t) = \int_{S_{t+1}:a_\infty} G_t p(S_{t+1}, a_{t+1}, S_{t+2}, a_{t+2}, \dots | S_t, a_t) d_{S_{t+1}:a_\infty} : a_\infty$$

$$p(S_{t+2} : a_\infty | S_t, a_t, S_{t+1}, a_{t+1}) p(S_{t+1}, a_{t+1} | S_t, a_t)$$

$$Q(S_t, a_t) = \int_{S_{t+1}, a_{t+1}} \left[ \int_{S_{t+2}:a_\infty} (R_t + \gamma G_{t+1}) p(S_{t+2} : a_\infty | S_{t+1}, a_{t+1}) d_{S_{t+2}:a_\infty} \right] p(S_{t+1}, a_{t+1} | S_t, a_t) d_{S_{t+1}, a_{t+1}}$$

$$Q(S_t, a_t) = \int_{S_{t+1}, a_{t+1}} (R_t + \gamma Q(S_{t+1}, a_{t+1})) p(S_{t+1}, a_{t+1} | S_t, a_t) d_{S_{t+1}, a_{t+1}} \quad Q \text{를 next Q로 표현}$$

$$Q(S_t, a_t) = \int_{S_{t+1}, a_{t+1}} (R_t + \gamma Q(S_{t+1}, a_{t+1})) p(a_{t+1} | S_t, a_t, S_{t+1}) p(S_{t+1} | S_t, a_t) d_{S_{t+1}, a_{t+1}}$$

Policy :  
전체를 maximize하는  
policy를 찾아야 함

Transition :  
environment에  
서 주어짐

# RL for CO

## Environment : MDP

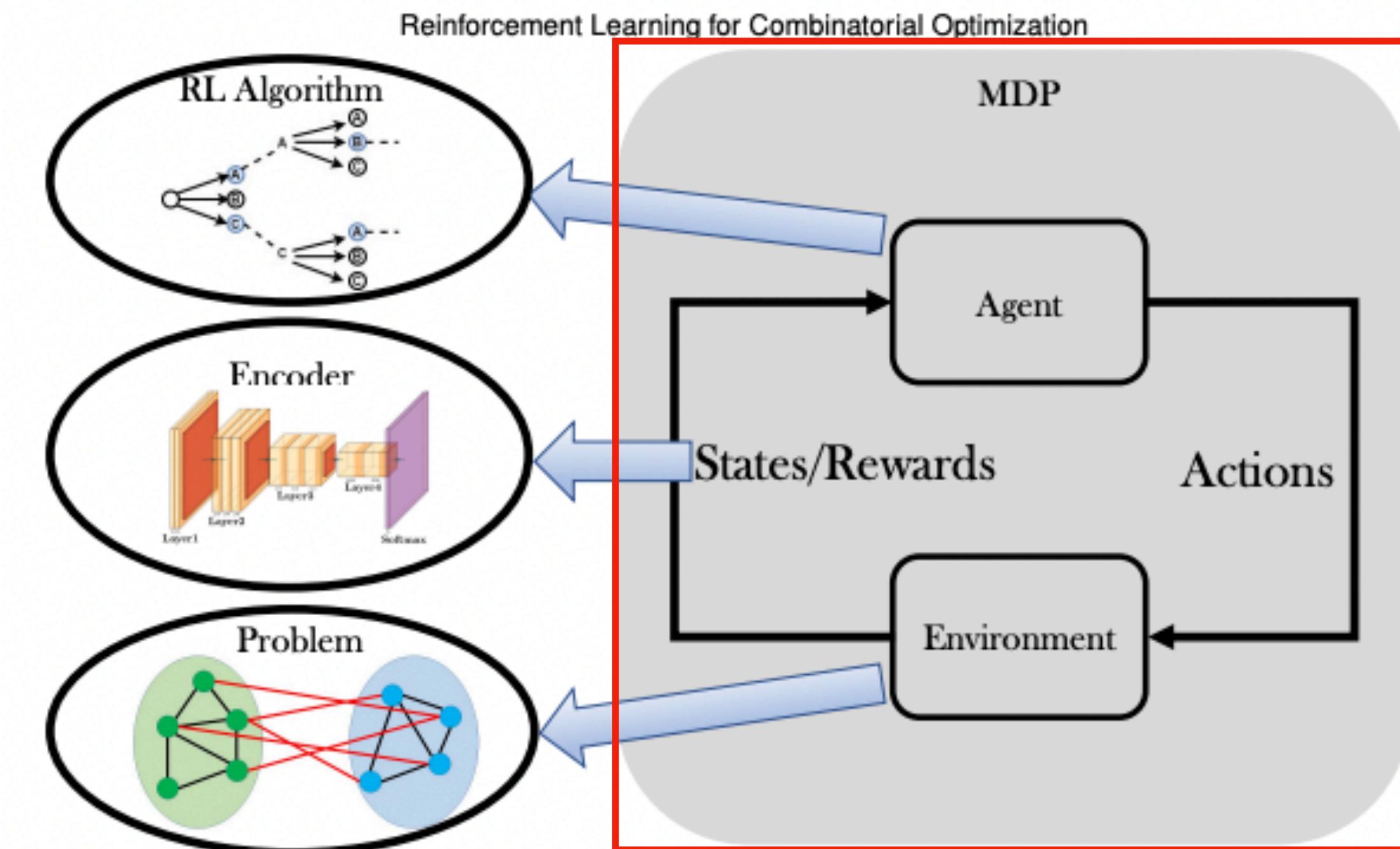
주어진 현실에서 해결해야할 문제에 맞게 MDP 환경을 구성



최적화 관점에서 최적화 surface를 바꿔주는 것과 유사한 관점

MDP는 tuple  $M = \langle S, A, R, T, \gamma, H \rangle$  으로 정의됨

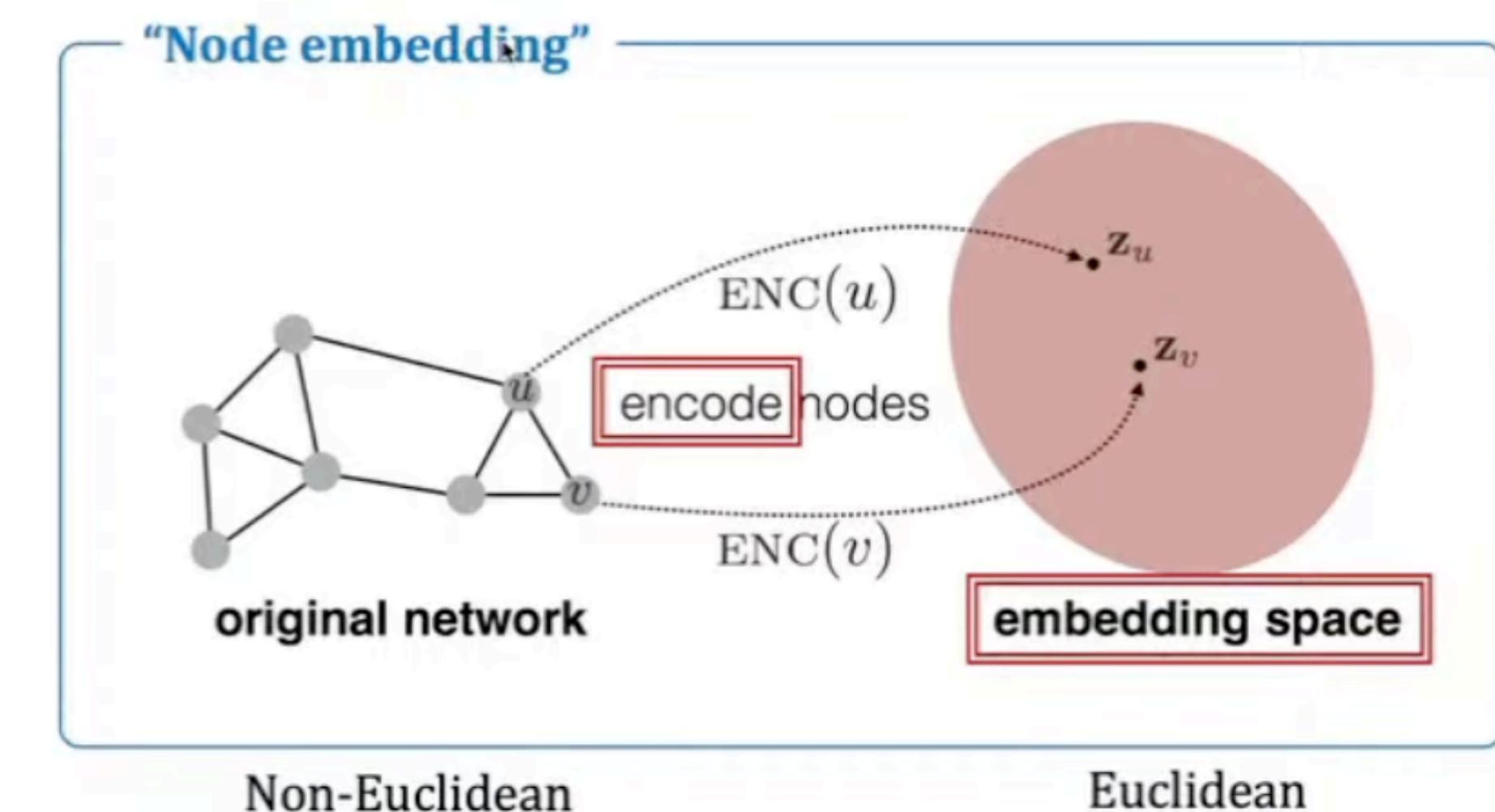
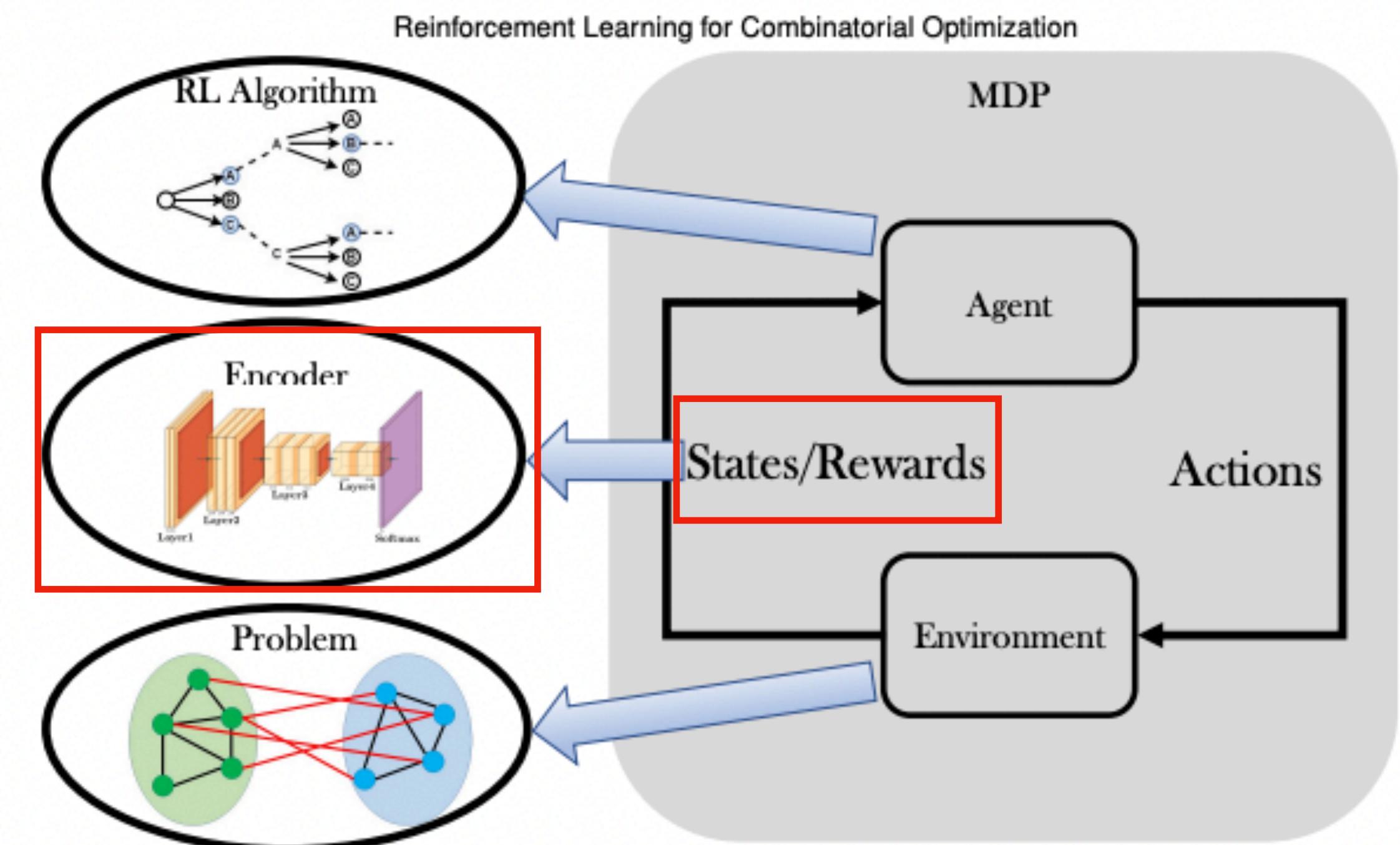
- $S$  : state space
- $A$  : Action space
- $R$  : reward function
- $T$  : Transition function
- $\gamma$  : scalar discount factor ( $0 < \gamma < 1$ )
- $H$  : horizon ( length of the episode )



# RL for CO Encoder

Encoder는 고차원 input space를 encoding 함

- Input state : high-dimensional vector(ex. Graph...)
- 원래 input space를 다른 차원의 space로 mapping
  - Mapping encoder라고 불림
  - 주로 CO 문제는 Graph 형태
- 기존에 연구되던 encoder - Deep learning model
  - **RNN (recurrent neural network)**
  - **Attention**
  - **GNN (graph neural network)**

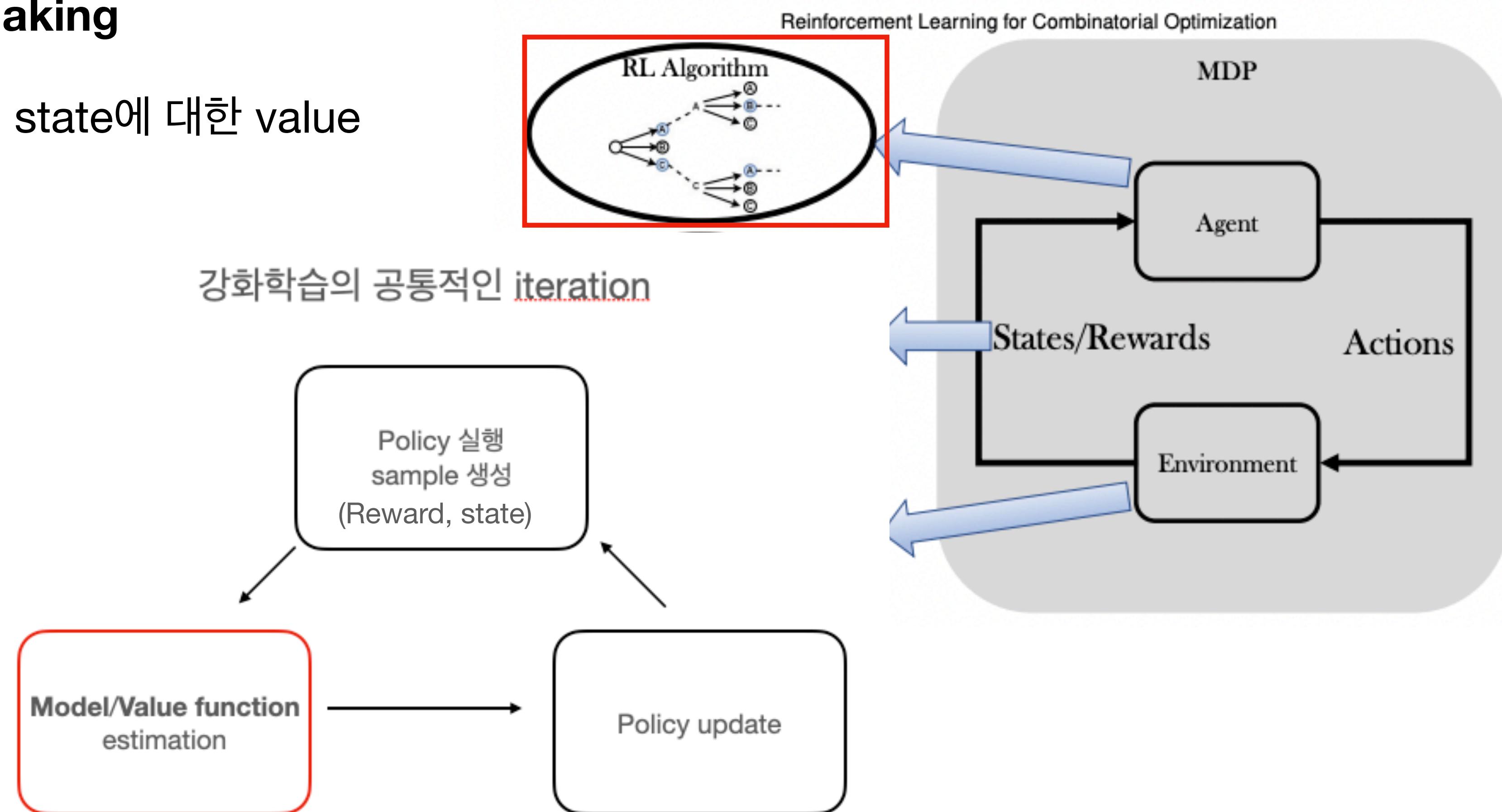


# RL for CO

## RL algorithms

**RL algorithm** : agent가 encoder의 parameter를 학습하고 주어진 MDP에 대해 decision making

- State, Reward 등을 통해 Action, state에 대한 value function을 구함
- **Value function**
  - Action value function
  - State value function
- optimal policy를 찾음
  - 어떤 Action을 할지



# **Taxonomy of RL for CO - 분류법**

# Taxonomy of RL for CO

## RL method에 따라 분류

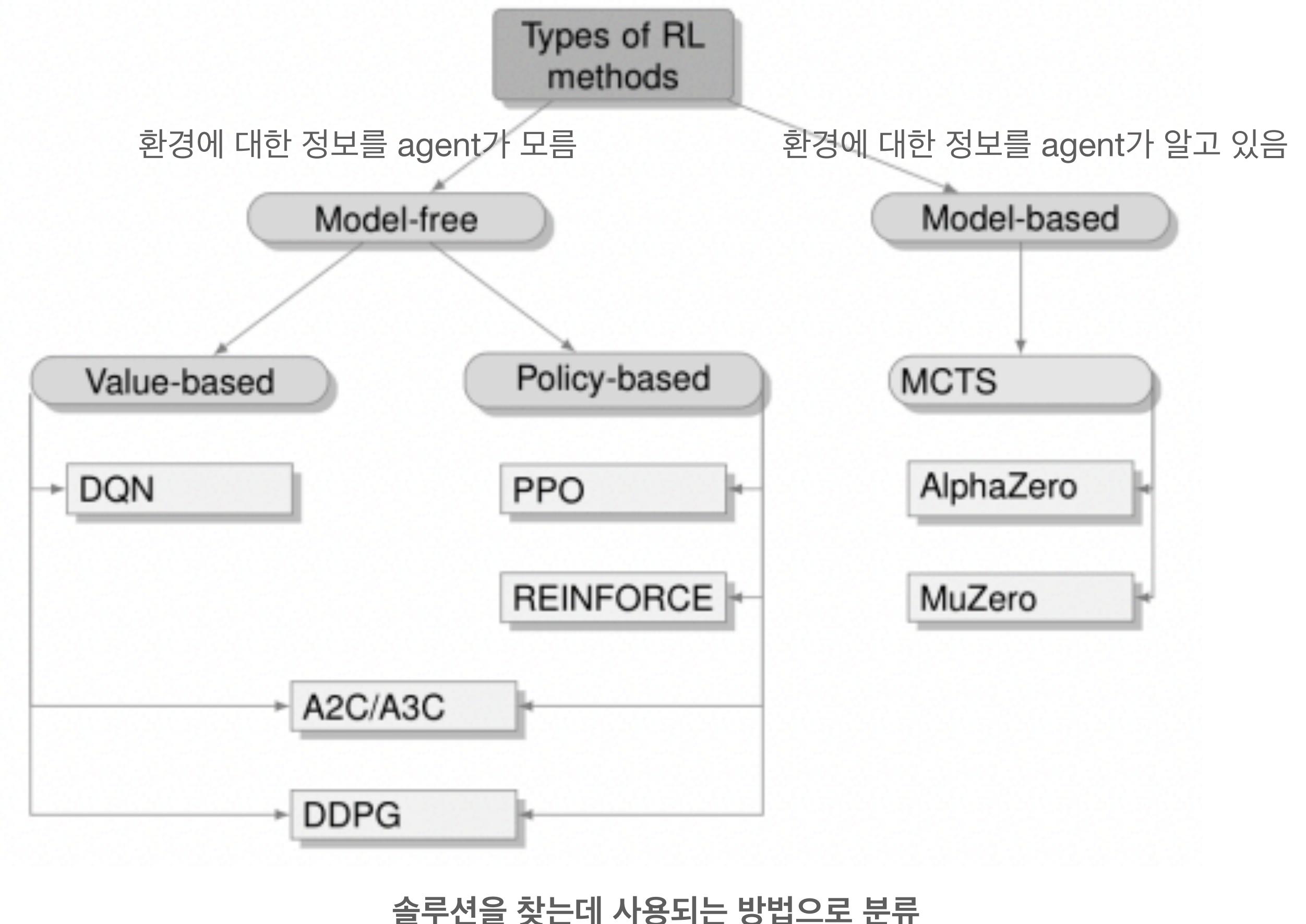
### Model-free method

- **Value-based method**
  - DQN
- **Policy-based method**
  - Policy gradient
  - Actor-critic algorithms
  - PPO / DDPG
  - REINFORCE

### Model-based method

- MCTS(Monte Carlo Tree Search)
- AlphaZero, MuZero

Reinforcement Learning for Combinatorial Optimization

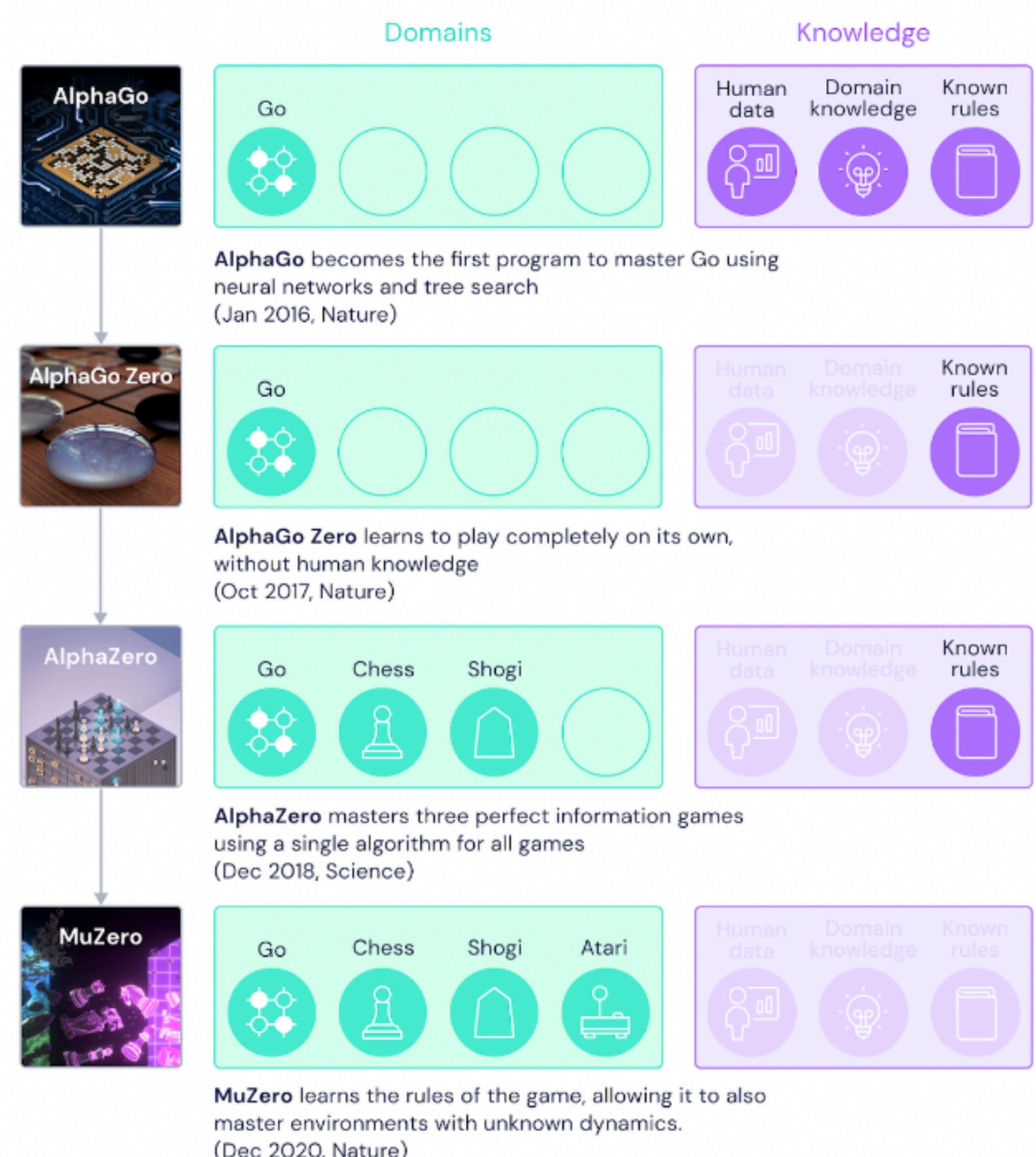


# Taxonomy of RL for CO

## RL method에 따라 분류

### Model-based method

- Environment에 대해 알고 있을 때
  - transition probability와 reward function을 정의 가능
- 주로 게임같은 분야는 환경 자체가 만들어져있기 때문에 모든 rule을 알고 있을 수 있음
- real world에는 매우 드문 경우
- episode 끝까지 갈 필요 없이 계속해서 state update가 가능
- **MCTS(Monte Carlo Tree Search)**
  - AlphaZero, MuZero

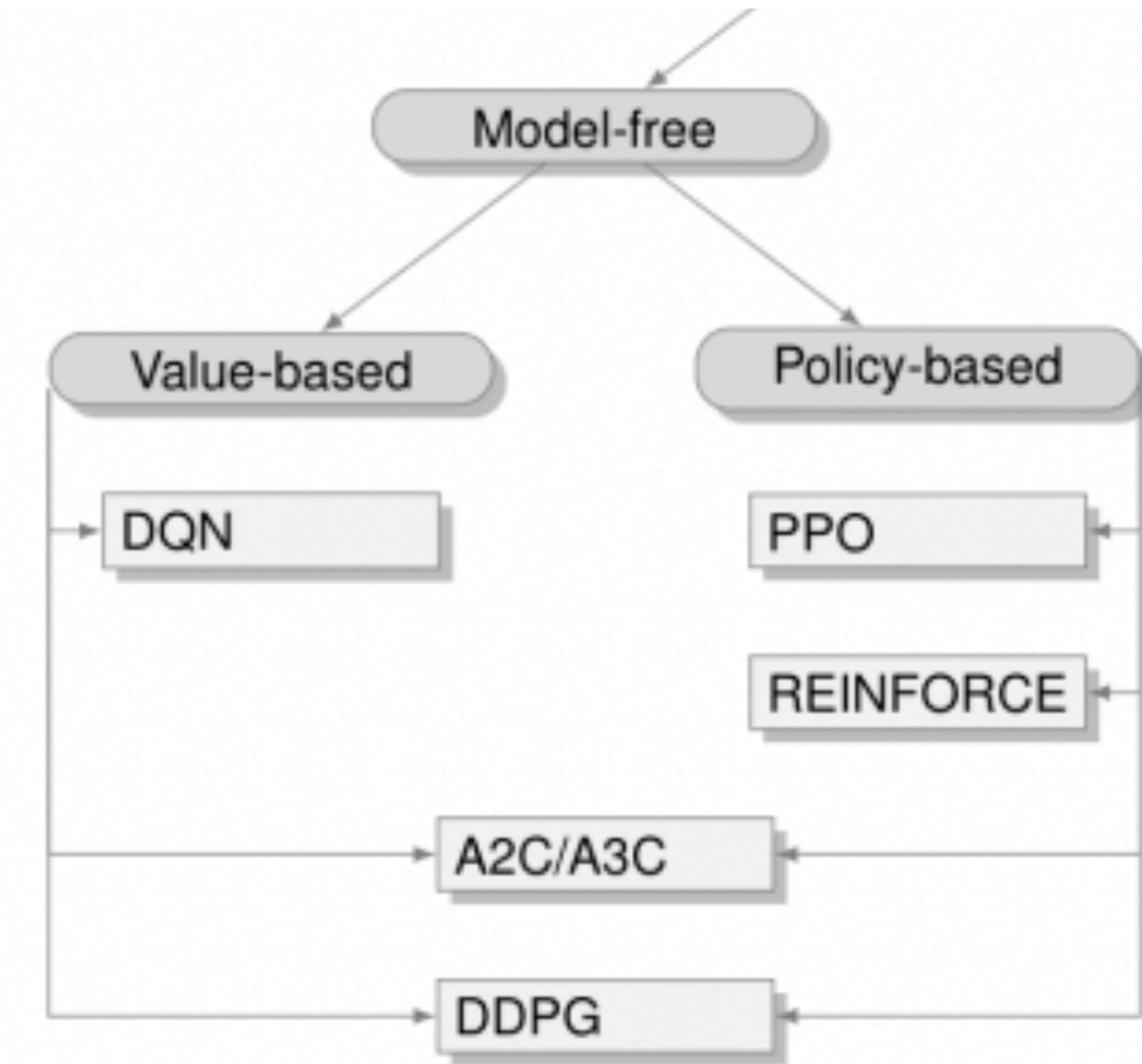


# Taxonomy of RL for CO

## RL method에 따라 분류

### Model-free method

- Environment에 대해 잘 모를 때
  - agent의 experience를 활용
  - transition probability와 reward function을 **Sampling**을 통해 추정
- **Policy-based/ Value-based**로 나뉨
  - Approximate 하는 것에 따라 (뭘 학습할거?)
    - **Policy-based** : Policy
      - PPO / REINFORCE
    - **Value-based** : Value function
      - DQN



# Taxonomy of RL for CO

## RL method에 따라 분류

### Model-free method

- **Actor-Critic methods**

- Policy-based + value-based 혼합

- **Critic model**

- Value function approximating

- State에 가치를 둠

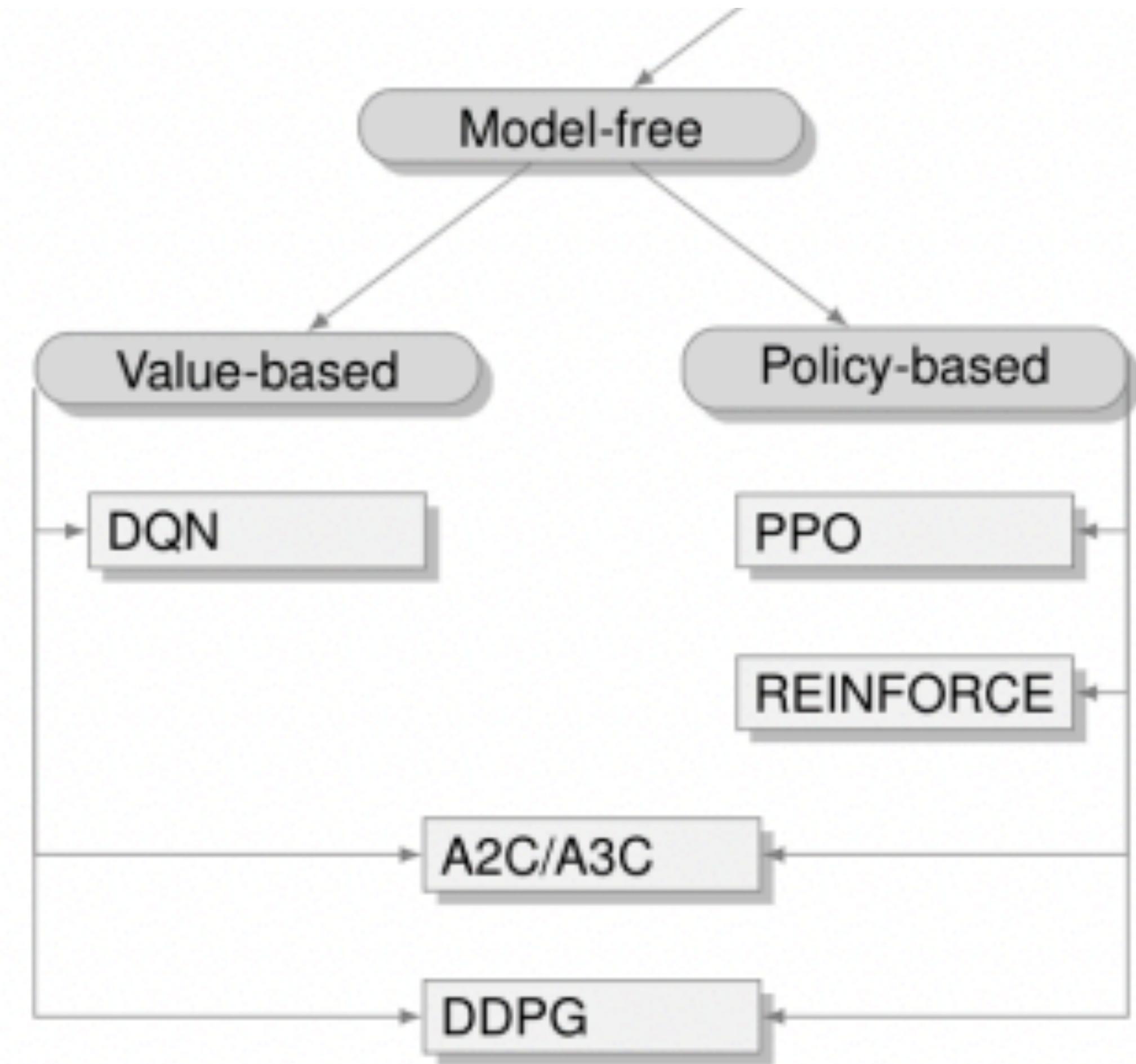
- **Actor model**

- Policy approximating

- Action에 가치를 둠

- A2C/A3C

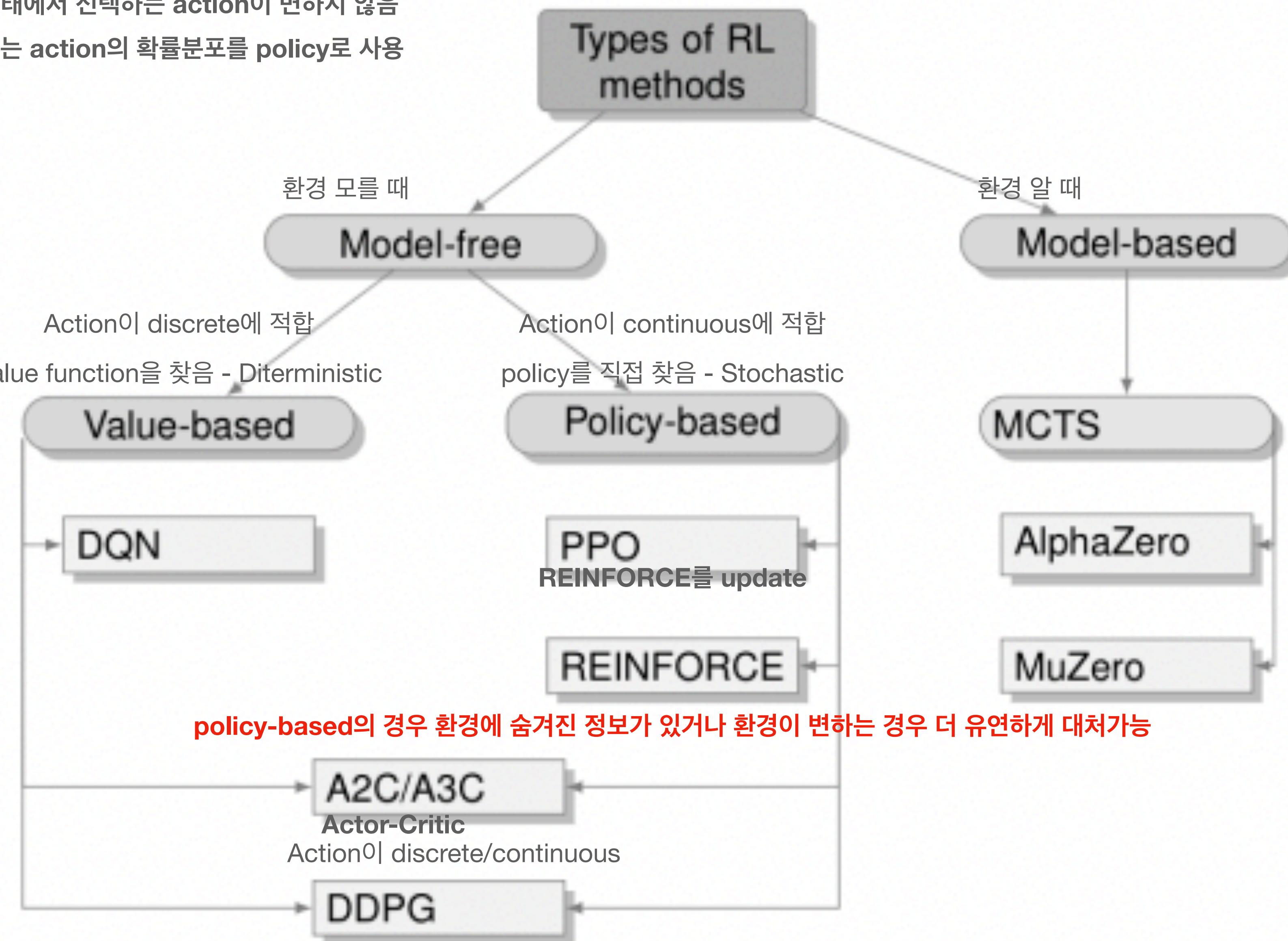
- DDPG



## Reinforcement Learning for Combinatorial Optimization

Deterministic : 모든 상태 s에 대해 각 상태에서 선택하는 action이 변하지 않음

Stochastic : 모든 상태 s에 대해 할 수 있는 action의 확률분포를 policy로 사용



# Taxonomy of RL for CO

## encoder에 따라 분류

- States encoding에 쓰인 encoder의 종류
- Encoder : Graph에 있는 관계를 modeling, representation을 생성

- **RNN**

- **Attention**

Combinatorial Optimization	Combinatorial Optimization (Khalil et al., 2017; Nowak et al., 2018; Li et al., 2018e; Kool et al., 2019; Bello et al., 2017; Vinyals et al., 2015b; Sutton and Barto, 2018; Dai et al., 2016; Gasse et al., 2019; Zheng et al., 2020a; Selsam et al., 2019; Sato et al., 2019)
----------------------------	--

- **GNN**

- Graph convolutional network(GCN)
- Graph attention network(GAT)
- Graph Isomorphism network(GIN)
- Structure to Vecor Network(S2V)



### Graph neural networks: A review of methods and applications

Jie Zhou <sup>a,1</sup>, Ganqu Cui <sup>a,1</sup>, Shengding Hu <sup>a</sup>, Zhengyan Zhang <sup>a</sup>, Cheng Yang <sup>b</sup>, Zhiyuan Liu <sup>a,\*</sup>, Lifeng Wang <sup>c</sup>, Changcheng Li <sup>c</sup>, Maosong Sun <sup>a</sup>

<sup>a</sup> Department of Computer Science and Technology, Tsinghua University, Beijing, China

<sup>b</sup> School of Computer Science, Beijing University of Posts and Telecommunications, China

<sup>c</sup> Tencent Incorporation, Shenzhen, China

해당 논문에서 GNN관점에서 CO문제의 formulation에 대한 내용을 설명했다고 함

계속해서 단점을 보완하는식으로 발전

# Taxonomy of RL for CO

## Searching Solution

Reinforcement Learning for Combinatorial Optimization

Approach	<i>Searching Solution</i>		Encoder	<i>Training</i>	RL
	Joint	Constructive			
[Bello et al., 2017]	No	Yes	Pointer Network	REINFORCE with baseline	
[Khalil et al., 2017]	No	Yes	S2V	DQN	
[Nazari et al., 2018]	No	Yes	Pointer Network with Convolutional Encoder	REINFORCE (TSP) and A3C (VRP)	
[Deudon et al., 2018]	No	Yes	Pointer Network with Attention Encoder	REINFORCE with baseline	
[Kool et al., 2019]	No	Yes	Pointer Network with Attention Encoder	REINFORCE with baseline	
[Emami and Ranka, 2018]	No	No	FF NN with Sinkhorn layer	Sinkhorn Policy Gradient	
[Cappart et al., 2020]	Yes	Yes	GAT/Set Transformer	DQN/PPO	
[Drori et al., 2020]	Yes	Yes	GIN with an Attention Decoder	MCTS	
[Lu et al., 2020]	Yes	No	GAT	REINFORCE	
[Chen and Tian, 2019]	Yes	No	LSTM encoder + classifier	Q-Actor-Critic	

Joint training : 기존 solver와 joint training을 통해 RL agent의 policy 학습 - 기존 solver에 새로운 데이터를 추가해 다시 update

Constructive : 기존에 이미 학습된 heuristic을 통해 solution을 활용하여 policy improvement - 기존 solver의 policy를 활용해서 초기 policy에 활용

**RL for CO - task마다 진행된 연구**

# RL for CO

## TSP

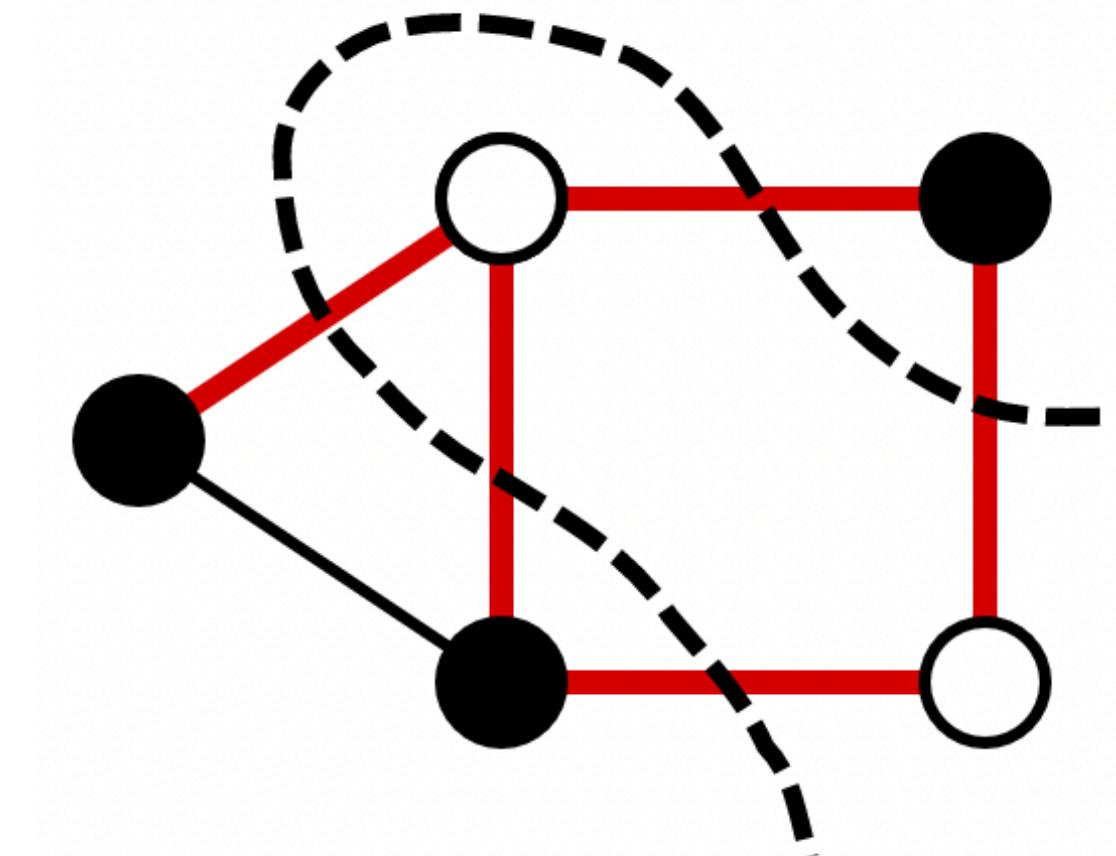


Reinforcement Learning for Combinatorial Optimization

Approach	<i>Searching Solution</i>		Encoder	<i>Training</i>	RL
	Joint	Constructive			
[Bello et al., 2017]	No	Yes	Pointer Network	REINFORCE with baseline	
[Khalil et al., 2017]	No	Yes	S2V	DQN	
[Nazari et al., 2018]	No	Yes	Pointer Network with Convolutional Encoder	REINFORCE (TSP) and A3C (VRP)	
[Deudon et al., 2018]	No	Yes	Pointer Network with Attention Encoder	REINFORCE with baseline	
[Kool et al., 2019]	No	Yes	Pointer Network with Attention Encoder	REINFORCE with baseline	
[Emami and Ranka, 2018]	No	No	FF NN with Sinkhorn layer	Sinkhorn Policy Gradient	
[Cappart et al., 2020]	Yes	Yes	GAT/Set Transformer	DQN/PPO	
[Drori et al., 2020]	Yes	Yes	GIN with an Attention Decoder	MCTS	
[Lu et al., 2020]	Yes	No	GAT	REINFORCE	
[Chen and Tian, 2019]	Yes	No	LSTM encoder + classifier	Q-Actor-Critic	

# RL for CO

## Maximum Cut Problem



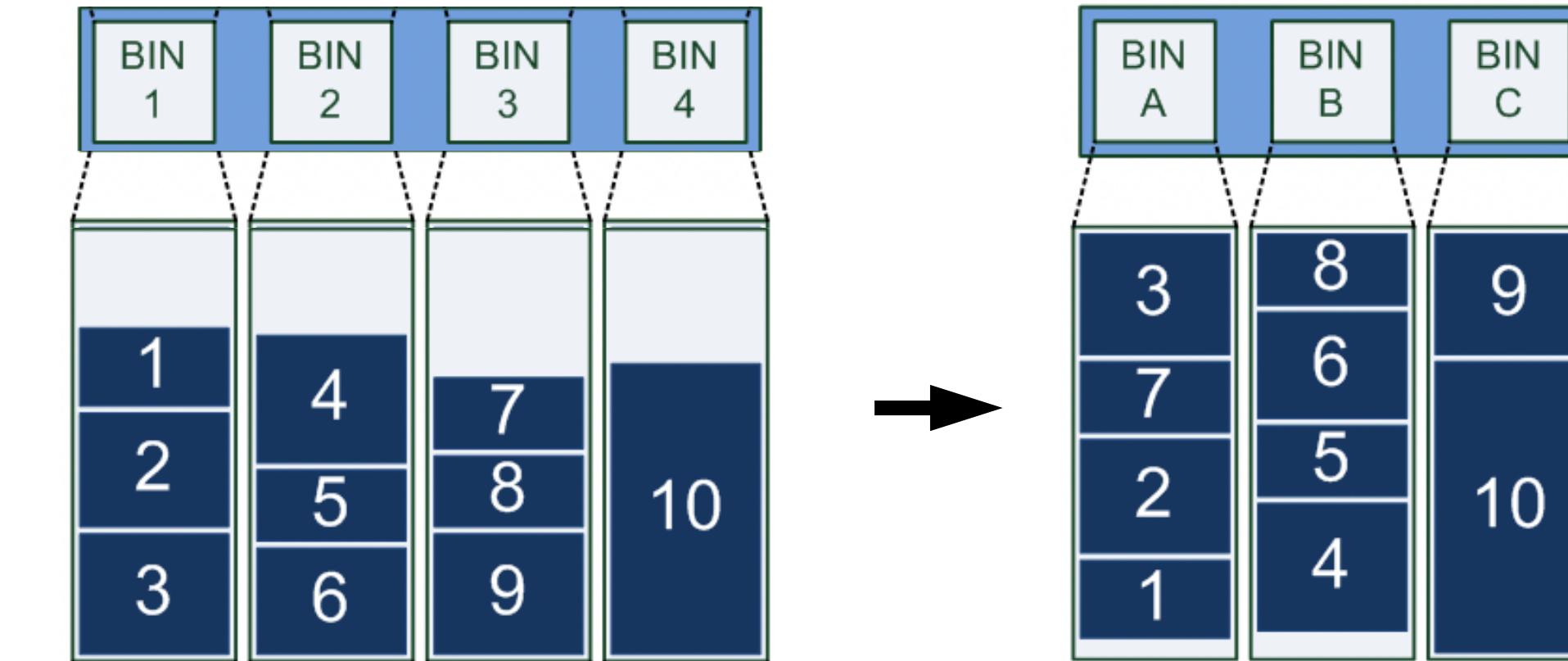
Reinforcement Learning for Combinatorial Optimization

Approach	<b>Searching Solution</b>		<b>Training</b>	
	Joint	Constructive	Encoder	RL
[Khalil et al., 2017]	No	Yes	S2V	DQN
[Barrett et al., 2020]	No	Yes	S2V	DQN
[Cappart et al., 2019]	Yes	Yes	S2V	DQN
[Tang et al., 2020]	Yes	No	LSTM + Attention	Policy Gradient + ES
[Abe et al., 2019]	No	Yes	GNN	Neural MCTS
[Gu and Yang, 2020]	No	Yes	Pointer Network	A3C

**Table 2**  
Summary of approaches for Maximum Cut Problem.

# RL for CO

## Bin Packing



### Reinforcement Learning for Combinatorial Optimization

Approach	<i>Searching Solution</i>		<i>Training</i>	
	Joint	Constructive	Encoder	RL
[Hu et al., 2017]	No	Yes	Pointer Network	REINFORCE with baseline
[Duan et al., 2019]	No	Yes	Pointer Network + Classifier	PPO
[Laterre et al., 2018]	No	Yes	FF NN	Neural MCTS
[Li et al., 2020]	No	No	Attention	Actor-Critic
[Cai et al., 2019]	Yes	No	N/A	PPO

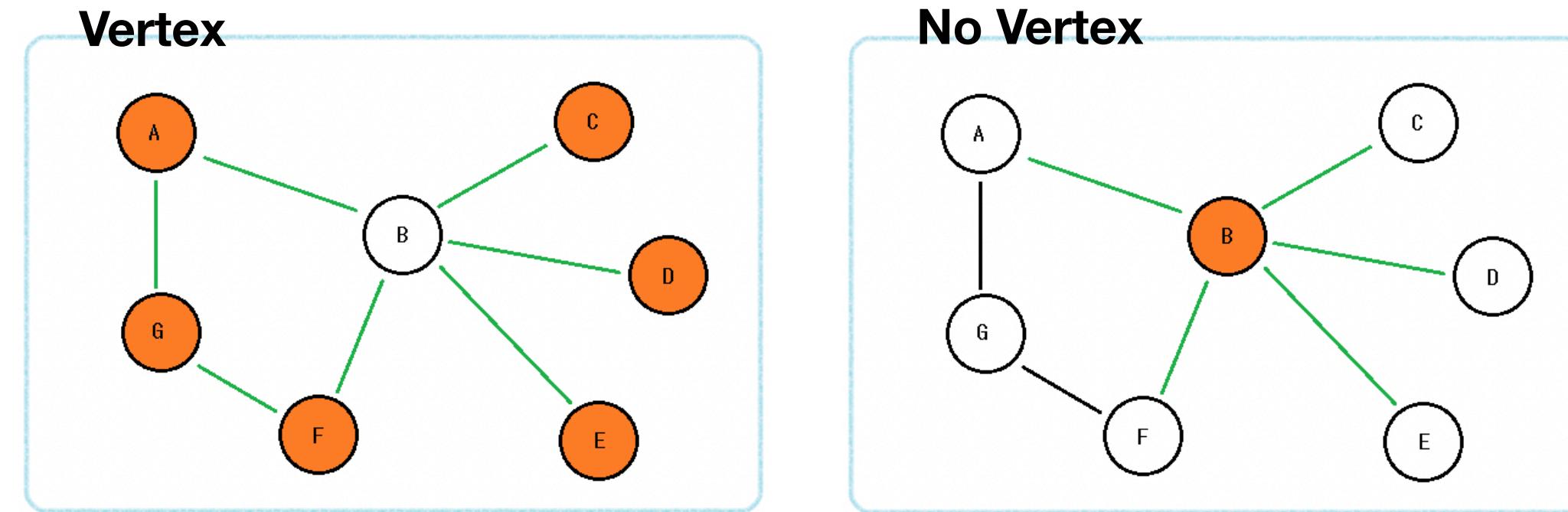
**Table 3**  
Summary of approaches for Bin Packing Problem.

# RL for CO

## Minimum Vertex Cover

### Reinforcement Learning for Combinatorial Optimization

Approach	<i>Searching Solution</i>		Encoder	<i>Training</i>	
	Joint	Constructive		RL	
[Khalil et al., 2017]	No	Yes	S2V	DQN	
[Song et al., 2020]	No	Yes	S2V	DQN + Imitation Learning	
[Manchanda et al., 2019]	No	Yes	GNN	DQN	



**Table 4**  
Summary of approaches for Minimum Vertex Cover problem.

# RL for CO

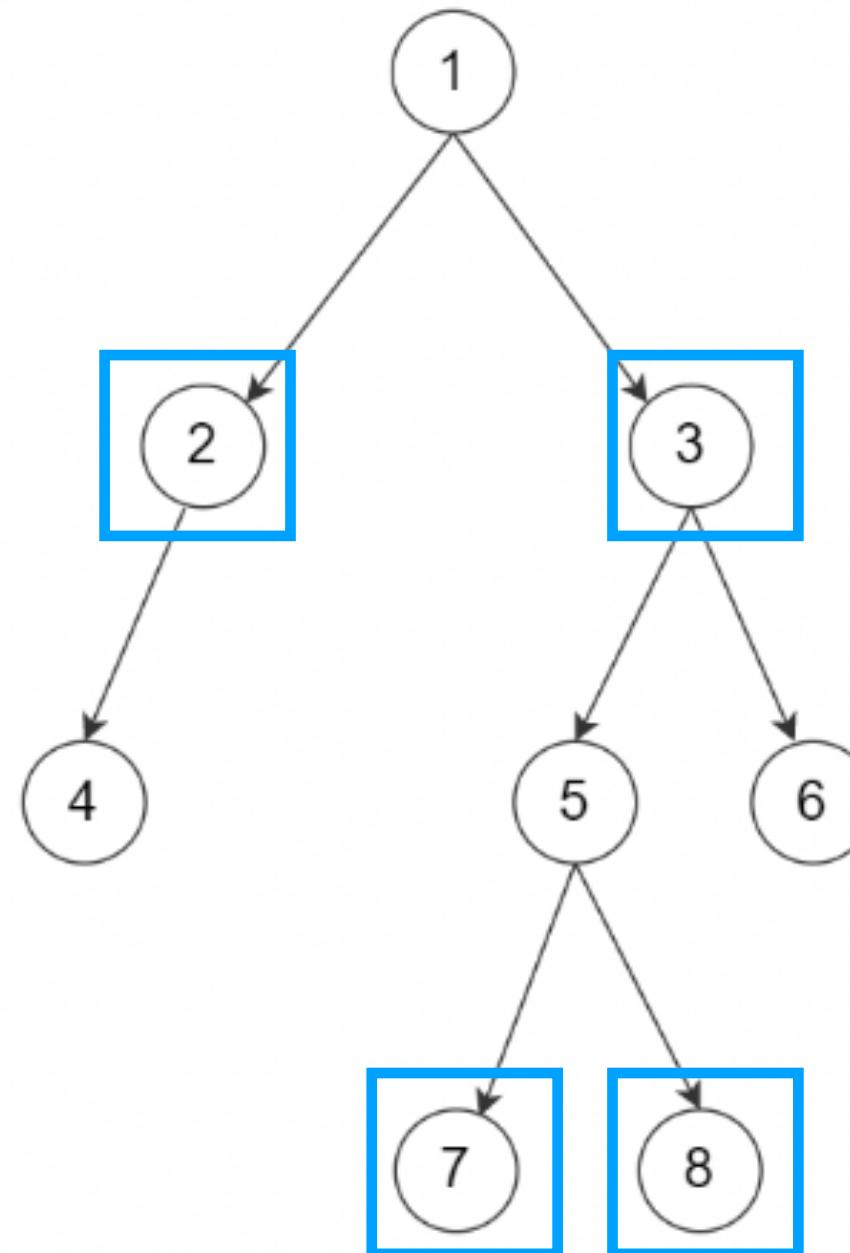
## Maximum Independent Set

Reinforcement Learning for Combinatorial Optimization

Approach	<b>Searching Solution</b>		<b>Training</b>	
	Joint	Constructive	Encoder	RL
[Cappart et al., 2019]	Yes	No	S2V	DQN
[Abe et al., 2019]	No	No	GIN	MCTS
[Ahn et al., 2020]	Yes	Yes	GCN	PPO

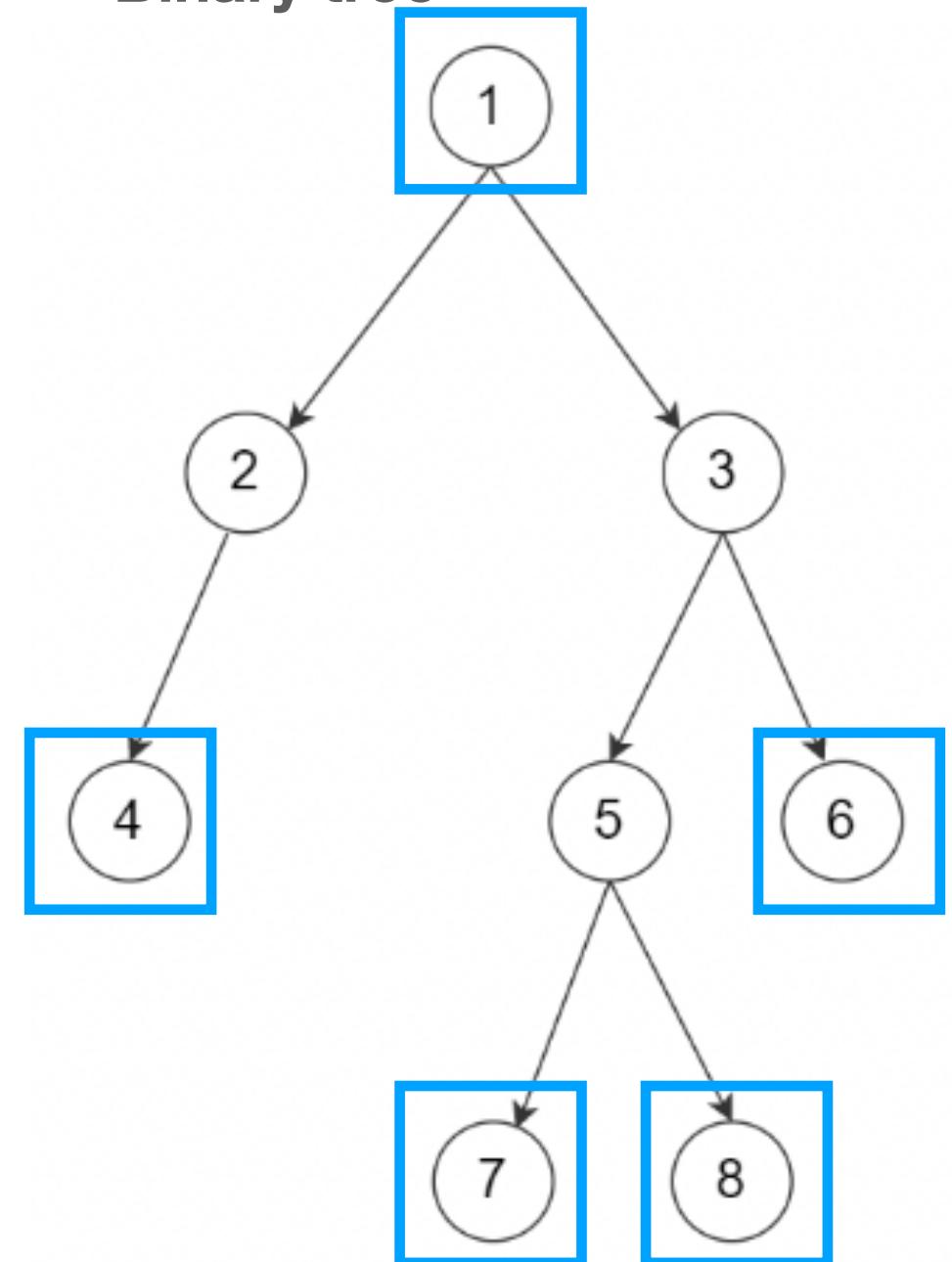
**Table 5**  
Summary of approaches for Maximum Independent Set problem.

Binary tree



Independent set : 2,3,7,8

Binary tree



Maximum Independent set :  
1,4,6,7,8

# **Comparison - 최근 RL연구들과 기존 optimizer를 비교**

# Comparison

## TSP 성능 비교

Reinforcement Learning for Combinatorial Optimization

Algo	Article	Method	Average tour length		
			N=20	N=50	N=100
RL	[Lu et al., 2020]	REINFORCE	4.0	6.0	8.4
	[Kool et al., 2019]		3.8	5.7	7.9
	[Deudon et al., 2018]		3.8	5.8	8.9
	[Deudon et al., 2018]	REINFORCE+2opt	3.8	5.8	8.2
	[Bello et al., 2017]	A3C	3.8	5.7	7.9
	[Emami and Ranka, 2018]	Sinkhorn Policy Gradient	4.6	-	-
	[Helsgaun, 2017]	LKH	3.8	5.7	7.8
	[Perron and Furnon, 2019]	OR-Tools	3.9	5.8	8.0

Table 6

The average tour lengths (the smaller, the better) comparison for TSP for ER graphs with the number of nodes  $N$  equal to 20, 50, 100.

길이가 더 긴 250, 500, 750, 1000인 경우 아직 쉽지 않음

# Comparison TSP

CO문제의 경우 다루는 문제가 일치하지 않아 공정한 비교가 어려운편

Reinforcement Learning for Combinatorial Optimization

Algo	Article	Method	Average tour length						
			N=10, Cap. 10	N=20, Cap. 20	N=20, Cap. 30	N=50, Cap. 30	N=50, Cap. 40	N=100, Cap. 40	N=100, Cap. 50
RL	[Nazari et al., 2018]	REINFORCE	4.7	—	6.4	—	11.15	—	17.0
	[Kool et al., 2019]		—	—	6.3	—	10.6	—	16.2
	best [Lu et al., 2020]		—	6.1	—	10.4	—	15.6	—
	[Chen and Tian, 2019]	A2C	—	—	6.2	—	10.5	—	16.1
	[Helsgaun, 2017]	LKH	—	6.1	6.1	10.4	10.4	15.6	15.6
	[Perron and Furnon, 2019]	OR-Tools	4.7	6.4	6.4	11.3	11.3	17.2	17.2

Table 7

The average tour lengths comparison for Capacitated VRP for ER graphs with the number of nodes  $N$  equal to 10, 20, 50, 100. Cap. represents the capacity of the vehicle for CVRP.

RL이 대표적인 CO문제에서는 기존 방법과 동등한 성능을 내고 있다!

# Comparison

## Running times

- 기존의 meta heuristic 알고리즘 및 solver에서 얻은 것과 비교하여 실행시간이 상당히 단축됨
  - 그러나 다른 연구의 실행시간 결과는 실험에 사용된 하드웨어와 구현에 따라 다를 수 있음
- 그렇기에 survey 논문에서는 논문마다 running time을 비교하는 시도하지 않음
- 하지만 여러 논문에서 일부 task에서 RL방법론이 고전 휴리스틱 알고리즘을 능가한다고 주장
  - [Nazari et al., 2018]
  - [Chen and Tian, 2019]
  - [Lu et al., 2020]
  - [kool et al., 2019]

**RL for CO의 미래 ?**

# RL for CO

## Future direction

- **Generalization to other problem**
  - 이미 특정 문제에서 학습된 policy를 다른 보지 못한 문제에 대해 일반화 해서 활용하는 것
  - 크기나 분포가 다른 instance를 가진 동일한 CO문제를 해결하는 연구
  - 관련 분야로 여러 연구가 활발히 진행중

# RL for CO

## Future direction

- **Improving the solution quality**
  - 이 survey에서 제시된 관련 연구들은 모두 기존 상용 solver에 비해 우수한 성능을 보여줌
  - 그러나 이러한 결과는 덜 복잡한 문제에만 해당된다.
  - 객관적인 quality 측면에서 현재 알고리즘의 추가 개선이 가능
  - **Imitation learning**을 사용하여 RL과 기존 알고리즘을 추가로 통합하는 것 등

# RL for CO

## Future direction

- **Filling the Gaps**

- RL-CO 접근 방식을 분류하는 방법 중 **joint, constructive** 방법으로 그룹화 하는 것
- 각 CO 문제에 대해 탐색 되지 않은 몇가지 접근 방식이 존재
  - Bin Packing문제에 대한 joint, constructive 알고리즘은 모두 공개되지 않음
  - joint-constructive와 joint-nonconstructive type의 methods가 없는 MVCP문제
  - 이러한 알고리즘의 가능성을 탐색하면 여러 접근 방식이 어떤 영향을 미치는지에 대한 인사이트를 얻을 수 있을 것으로 보임

Approach	Searching Solution		Encoder	Training	
	Joint	Constructive		RL	
[Bello et al., 2017]	No	Yes	Pointer Network	REINFORCE with baseline	
[Khalil et al., 2017]	No	Yes	S2V	DQN	
[Nazari et al., 2018]	No	Yes	Pointer Network with Convolutional Encoder	REINFORCE (TSP) and A3C (VRP)	
[Deudon et al., 2018]	No	Yes	Pointer Network with Attention Encoder	REINFORCE with baseline	
[Kool et al., 2019]	No	Yes	Pointer Network with Attention Encoder	REINFORCE with baseline	
[Emami and Ranka, 2018]	No	No	FF NN with Sinkhorn layer	Sinkhorn Policy Gradient	
[Cappart et al., 2020]	Yes	Yes	GAT/Set Transformer	DQN/PPO	
[Drori et al., 2020]	Yes	Yes	GIN with an Attention Decoder	MCTS	
[Lu et al., 2020]	Yes	No	GAT	REINFORCE	
[Chen and Tian, 2019]	Yes	No	LSTM encoder + classifier	Q-Actor-Critic	

**이걸 그래서 어떻게 활용해야 할까?**

# RL for CO

## Conclusion

- 적절한 강화학습 알고리즘 선택
  - 현실 문제를 **MDP**로 만들었을 때, state의 sequence가 명확하게 markovian을 따르는지 안따르는지
  - 전체 Trajectory의 performance를 하나의 **error**가 영향을 많이 끼치는지?
    - CO 문제 : **policy based**가 좀 더 적합
- 선행 연구를 참고 하는 것이 중요
  - 강화학습은 적절한 하이퍼파라미터에 따라 성패여부가 갈림
  - 논문을 꼼꼼히 확인해야 함
  - 관련된 논문에서 **MDP parameter, hyperparameter, trick**들을 참고해야함

**감사합니다.**