

# AI 알고리즘

## 동적계획법

## 학습내용

- 동적계획법의 개요
- 동적계획법의 형태

## 학습목표

- 동적계획법의 개요를 이해하고 설명할 수 있다.
- 동적계획법의 형태를 설명할 수 있다.

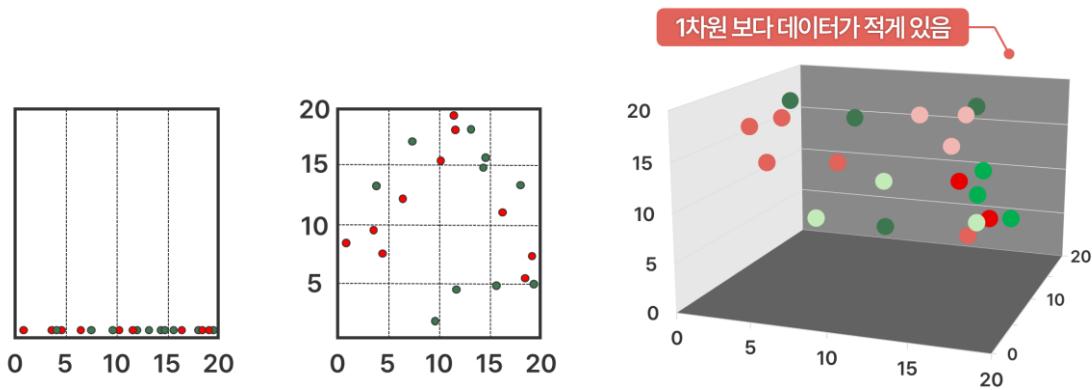
# 동적계획법의 개요

동적계획법의 개요

## 동적계획법의 개요

### ❖ 동적계획법(Dynamic Programming)

- 1957년, 미국 수학자 리처드 어니스트 벨먼(1920-1984) 고안
  - 기존에 개발돼 있던 알고리즘을 문제 상황에 맞춰서 선택
  - 시가변적
  - 다단계적 특성
- 벨먼 방정식
- 벨먼-포드 알고리즘
- 해밀턴-야코비-벨먼 방정식
- 차원의 저주
  - 차원이 증가하면서 학습 데이터 수가 차원 수보다 적어져서 성능이 저하되는 현상
    - 공간 차원(변수 개수)이 늘어나면서 문제 계산법이 지수적으로 커지는 상황
  - 관측치보다 변수 수가 많아지는 경우에 차원의 저주 문제가 발생



## 동적계획법의개요

### ❖ 동적계획법(Dynamic Programming)의 정의

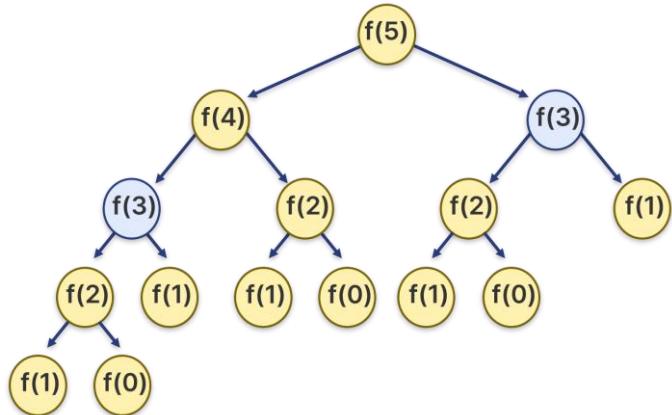
- 특정한 알고리즘이 아닌 하나의 문제 해결 패러다임
- 상호 연관된 의사 결정 단계로 다음 결정에 영향을 미침
- 관심 대상의 전반적인 효과를 극대화하는 의사 결정 조합 문제 해결 위한 절차 제공
- 전체 문제를 작은 문제로 단순화한 다음 점화식으로 만들어 재귀적인 구조를 활용해서 전체 문제를 해결하는 방식

### ❖ 피보나치 수열

- 1, 1, 2, 3, 5, 8, 13, 21, 34, …
  - $f(n) = f(n-1) + f(n-2)$
  - 재귀적 방법

```
def fib(n):
    if n==1 or n==2:
        return 1
    else:
        return fib(n-1) + fib(n-2)

fib(5)
```



- 메모이제이션(Memoization)
  - 함수의 값을 계산한 뒤 계산된 값을 배열에 저장하는 방식
- 동적계획법
  - 연산이 반복되는 결점을 보완
  - 한번 계산된 결과 저장, 활용

```
def fib(n):
    fibList=[1, 1]
    if n==1 or n==2:
        return 1
    for i in range(2,n):
        fibList.append( fibList[i-1] + fibList[i-2] )
    return fibList[n-1]

fib(5)
```

## 동적계획법의 개요

### ❖ 동적계획법(Dynamic Programming)

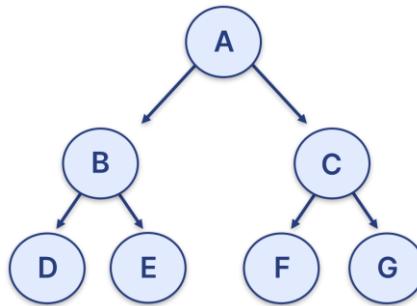
- 2가지조건
  - 겹치는 부분 문제(Overlapping Subproblems)
    - 기본적으로 문제를 나누고 그 문제의 결과값을 재활용해서 전체답을 구함
    - 동일한 작은 문제들이 반복하여 나타나는 경우에 사용 가능
  - 최적 부분 구조(Optimal Substructure)
    - 부분 문제의 최적 결과값을 사용해 전체 문제의 최적 결과를 낼 수 있는 경우 의미



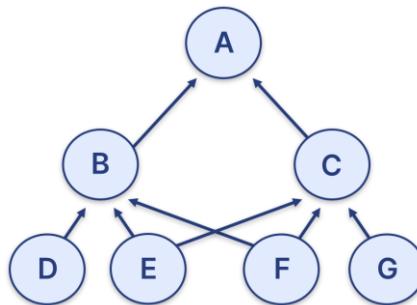
## 동적계획법의개요

### ❖ 동적계획법 vs 분할 정복 알고리즘

- 분할정복알고리즘



- 하단으로내려가며하나씩탐색
- 동적 계획알고리즘



- 하단의 작은 부분 문제들이상단으로작용

## 동적계획법의 개요

### ❖ DP vs 분할 정복 알고리즘

- 분할정복알고리즘
  - A는 B와 C로 분할되고, B는 D와 E로 분할되는데, D와 E의 해를 취합하여 B의 해를 구함
  - 단, D, E, F, G는 각각 더 이상 분할할 수 없는 (또는 가장 작은 크기의) 부분문제
  - 마찬가지로 F와 G의 해를 취합하여 C의 해를 구하고, 마지막으로 B와 C의 해를 취합하여 A의 해를 구함
- 동적 계획알고리즘
  - 최소 단위의 부분문제 D, E, F, G의 해를 각각 구함
  - 그다음 D, E, F의 해를 이용하여 B의 해를 구함
  - E, F, G의 해를 이용하여 C의 해를 구함
  - B와 C의 해를 계산하는데 E와 F의 해 모두 이용

### ❖ 동적계획법을 이용한 알고리즘

- 디익스트라알고리즘
- 플로이드-워셜알고리즘
- 배낭문제

## 동적계획법의 개요

### ❖ 경영과학 기법으로서의 동적계획법

- 작은측면이아닌큰측면에대해집중
- 단계적이거나단계적의사결정을지원하여문제의최적해를구하는수학적기법
  - Dynamic Programming
- 월별생산을결정해야하는생산계획문제
- 한단계내에서만의최적추구하지않고,연쇄적인단계에서의최적추구
- 다른수리계획기법과는달리표준화된알고리즘은없음
  - 심플렉스법과같이정형화된해법은존재하지않음
  - 강력하지만작은문제를해결하는데도계산이복잡하여시간이많이소요

# 동적계획법의 개요

동적계획법의 형태

## 동적계획법의 형태

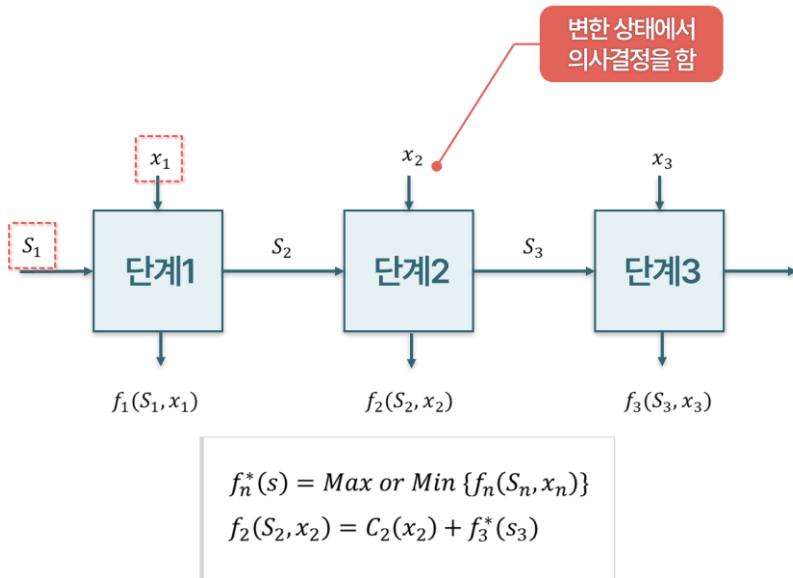
### ❖ 동적계획법의 형태(1)

- 의사결정문제를 여러 개의 하위문제로 분할
- 각 문제들의 상호연관성을 고려하여 전체문제의 최적해를 찾게됨
- 동적계획법의 해를 구할 수 있는 의사결정의 단계는 후진접근법과 전진접근법 사용
- 최적해는 항상 동일
- 후진접근법이 전진접근법보다는 계산상의 효율성 때문에 많이 사용되고 있음
- 용어정리
  - 단계
    - 시점을 나타낼 수도 있고, 문제를 해결하는 전환점이 될 수도 있음
    - 각 단계에서 의사결정자는 다음 단계에 영향을 줄 정책 결정을 내려야 하고 이 결정은 보통 다음 단계로 들어가는 상태를 바꾸게 함
  - 상태
    - 각 단계에서 시스템은 가능한 여러 가지 상태 중에서 한 가지 상태에 있음
    - 즉 각 단계는 어떤 상태와 관련이 있음
  - 상태변수
    - 시스템의 상태를 규정하는 값을 취하는 변수
  - 보상
    - 한 단계에서 다음 단계로 시스템이 변화됨에 따라 발생하는 이득 또는 손실
  - 정책
    - 각 단계의 가능한 상태에 대한 일련의 결정 대안
  - 최적정책
    - 전 과정의 보상을 최적화하는 정책
    - 최적 정책은 앞 단계의 상태와 결정이 무엇이든지 간에 이후의 결정이 앞 단계의 결정에 의해서 정해지는 상태에 대해서 최적 정책이 되어야 하는 특성을 지님(독립적인 특성)

## 동적계획법의 형태

### ❖ 동적계획법의 형태(1)

- 일반적인구조



- 확정적 동적계획법

- 의사결정문제를 분할한 단계들 사이에서 다음 단계에 발생할 상태는 현재의 상태와 현재 단계의 정책결정(의사결정)에 의해 확정적으로 결정되는 형태

- 확률적 동적계획법

- 다음의 상태가 확률분포에 의해 결정되는 의사결정문제에 적용이 되는 동적계획법
- 많은 현실세계에서의 의사결정 문제는 현재의 단계와 정책결정이 알려져 있다 하더라도 현재 기간의 비용이나 다음 기간의 상태에 대해서는 불확실한 것이 일반적

## 동적계획법의 형태

### ❖ 동적계획법의 특징

- 문제는 몇 개의 단계(Stage)나 부분 문제로 나누어질 수 있음
- 각 단계에서는 정책이나 방침의 결정이 이루어짐
- 상태는 각 단계에서의 시스템이 치할 수 있는 모든 가능한 조건을 나타냄
  - 종류가 한정되어 있는 유한(Finite) 상태와 매우 많은 종류를 갖는 무한(Infinite) 상태로 구분
- 각 단계의 정책은 현재의 상태를 다음 상태와 연결된 상태로 변환시키기 위해 각 가능한 상태에 대한 결정을 확정 짓는 의사결정 규칙이나 일련의 의사결정을 하는 역할
- 최적성의 원리(Principle Of Optimality)을 만족
  - 주어진 현상태에 대해서 나머지 단계들의 최적 정책은 그 이전 상태에서 채택된 정책과는 무관하다는 성질
- 해를 구하는 절차는 마지막 단계의 각 상태에 대한 최적 방침을 찾는 것부터 시작
  - 각 단계의 순서는 시간 흐름과 관련된 문제가 아니라면 각 단계 별 부문제(Subproblem)를 어느 순서에 배정하더라도 상관없음

### ❖ 동적계획법의 형태(2)

표기법	
$S_n$	단계 n에서의 <b>상태</b> 집합
$D_n$	단계 n에서의 <b>결정 변수</b> 집합
$r_{n(S_n, D_n)}$	단계 n에서의 상태 $S_n$ 의 값이 $D_n$ 일 때의 <b>보상</b> (목적식에의 기여)
$f_{n(S_n, D_n)}$	단계 n에서의 상태 $S_n$ 에 대해 $D_n$ 이 선택되었을 때의 <b>목적함수 값</b>
$f_n^*(S_n)$	최적 정책 $D_n$ 을 의사결정한 상황에서의 <b>목적값</b>