2025 겨울방학 알고리즘 스터디

트리 dp

컴퓨터 공학과 20230546 서보경

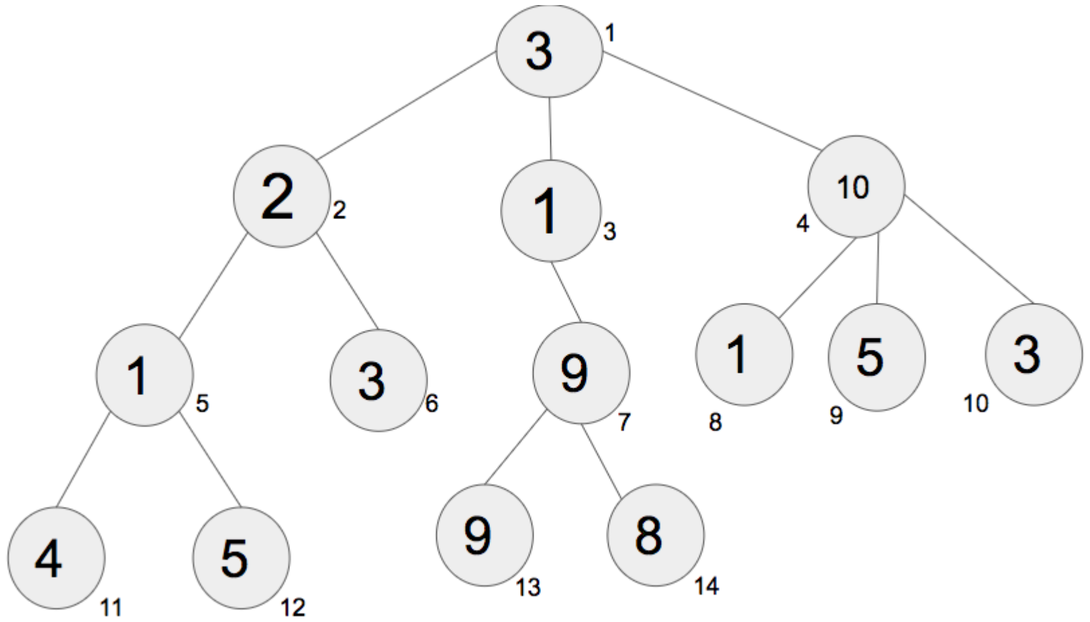
목차

트리 dp란?

트리 DP는 트리 자료구조에 동적 계획법(DP)을 적용해 문제를 해결하는 기법이다. 트리는 사이클이 없는 연결 그래프이기 때문에, DFS(깊이 우선 탐색)를 통해 트리 구조를 탐색하며 하위 노드의 결과를 상위 노드로 병합하는 방식으로 문제를 해결한다.

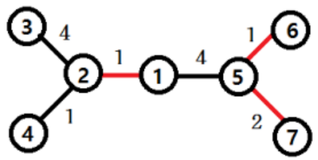
트리 DP의 핵심 아이디어

서브트리 문제 해결 → 부모 노드로 결과 반환 및 병합.



루트 기준 트리 dp

인하니카 공화국은 1번~ *N*번까지 *N*개의 섬으로 이루어진 나라이다. 이 나라에서는 섬 간의 왕래가 매우 어려웠지만, 위대한 다리 설계자 '진'이 두 섬을 연결하는 다리를 최소한의 개수로 만들어 모든 섬 간의 왕래가 가능하도록 만들었다. 1번섬에서 살고 있는 진은 어느 날 위험한 소문을 듣게 되었다. 1번섬을 제외한 다리가 하나밖에 없는 어느 섬에서 유명한 연쇄 살인마 괴도'루팡'이 자신의 목숨을 노리고 있다는 소문이었다. 너무 불안한 나머지 진은 몇 개의 다리를 폭파하여, 루팡이 있을 가능성이 있는 모든 섬에서 자신의 섬으로의 모든 경로를 차단하려고 한다. 하지만 각 다리를 폭파하려면 다리의 크기에 따라 다이너마이트의 개수가 다르다. 다이너마이트는 매우 비싸기 때문에 진은 사용하는 다이너마이트의 개수를 최소화하고 싶어 한다. 각 섬을 연결하는 다리를 폭파하기 위한 다이너마이트의 개수가 주어졌을 때, 진을 도와 필요한 최소 다이너마이트의 개수를 구하라.



예를 들어, 위의 그림과 같이 섬과 다리를 폭파하기 위한 다이너마이트의 수가 주어졌을 때, 빨간색 다리를 폭파하면 다이너마이트의 개수를 최소화하면서 루팡으로부터 안전할 수 있다.

입력

입력의 첫 줄에는 테스트 케이스의 개수 $T(1 \le T \le 100)$가 주어진다.

각 테스트 케이스의 첫 번째 줄에는 섬의 수 $N(1 \le N \le 1,000)$과 다리의 수 M이 주어진다.

다음으로 M개의 줄에는 각 다리를 통해 이어진 두 섬의 번호와 다리를 파괴하기 위한 다이너마이트의 수 $D(1 \le D \le 20)$가 주어진다.

출력

각 테스트 케이스마다 필요한 최소 다이너마이트의 개수를 출력한다.

사실상 거의 모든 트리 dp를 푸는 방법인 루트 기반 트리 dp이다.

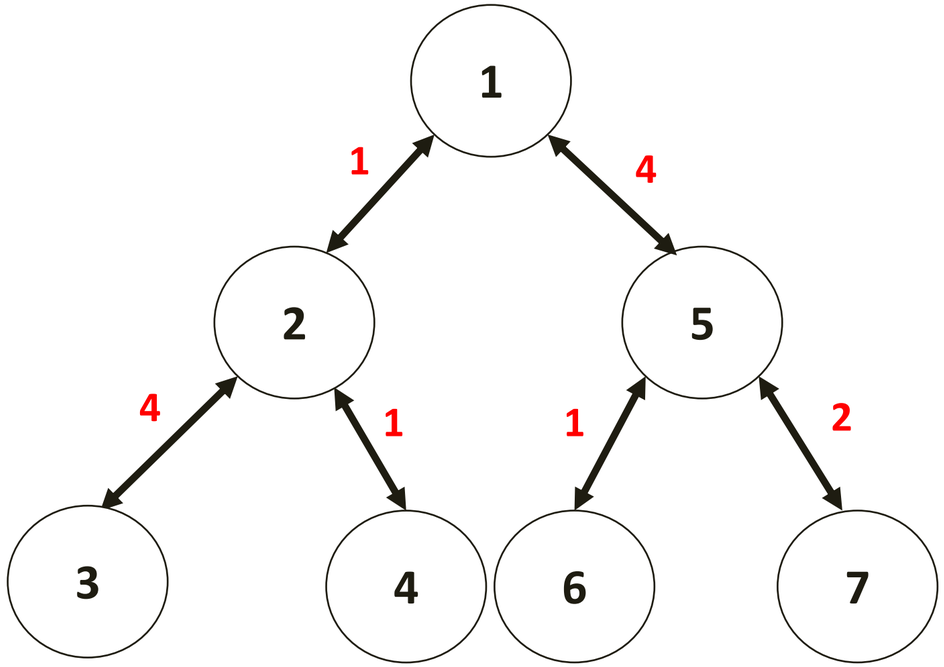
이 기법은 주로 서브트리의 크기, 서브트리 내 최대 최소 값, 경로 합을 구할 때 유용하게 사용이 가능하다.

즉 하위 노드의 결과를 병합해 상위 노드의 결과를 도출 해낼 수 있다는 것이다.

깊이 우선 탐색과 다이나믹 프로그래밍의 메모이제이션을 결합하여 문제를 효율적이고 직관적으로 해결할 수 있다!

이 문제에서
다리가 하나밖에 없다 == 차수가 1이다 == 리프 노드
이라는 것을 직관적으로 알 수 있으므로 우리가 구해야 하는 값은 리프에서 루트로 못가게 하기 위한 최소 간선 가중치의 합이다.

루트 기준 트리 dp



예제 입력 1 복사

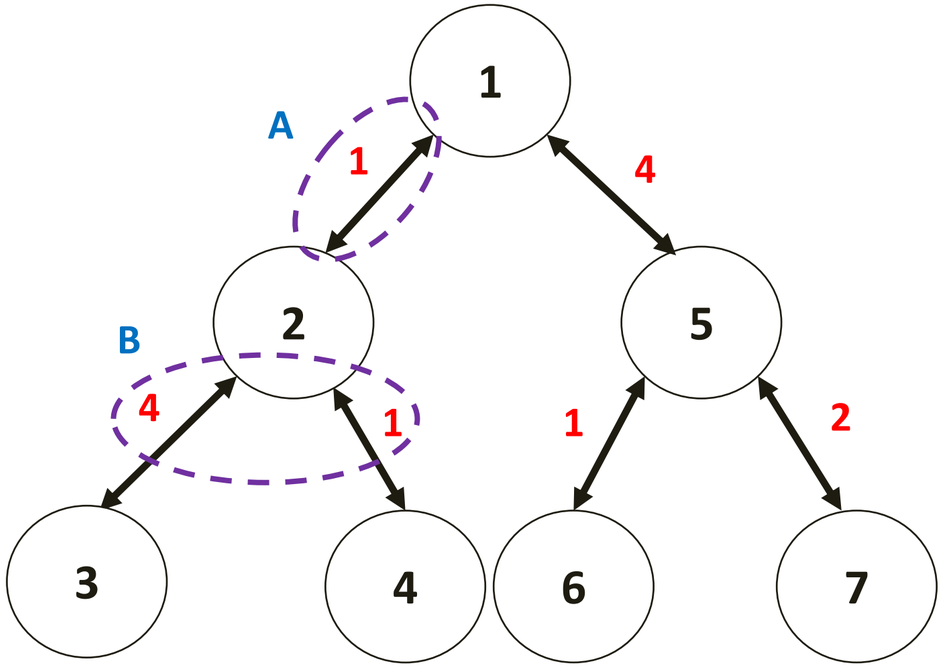
```
1
7 6
1 2 1
2 3 4
2 4 1
5 1 4
6 5 1
7 5 2
```

아까 언급 하였다싶이, 루트를 기준으로 하는 트리 dp가 사실상 절대적인 유형이다.

어느 정점을 루트로 잡아도 상관이 없지만 보통 1번 정점을 루트로 잡는다.

1번 정점이 루트로 가능하게 만들기 위해 그래프도 양방향 그래프로 만들어 준다.

루트 기준 트리 dp

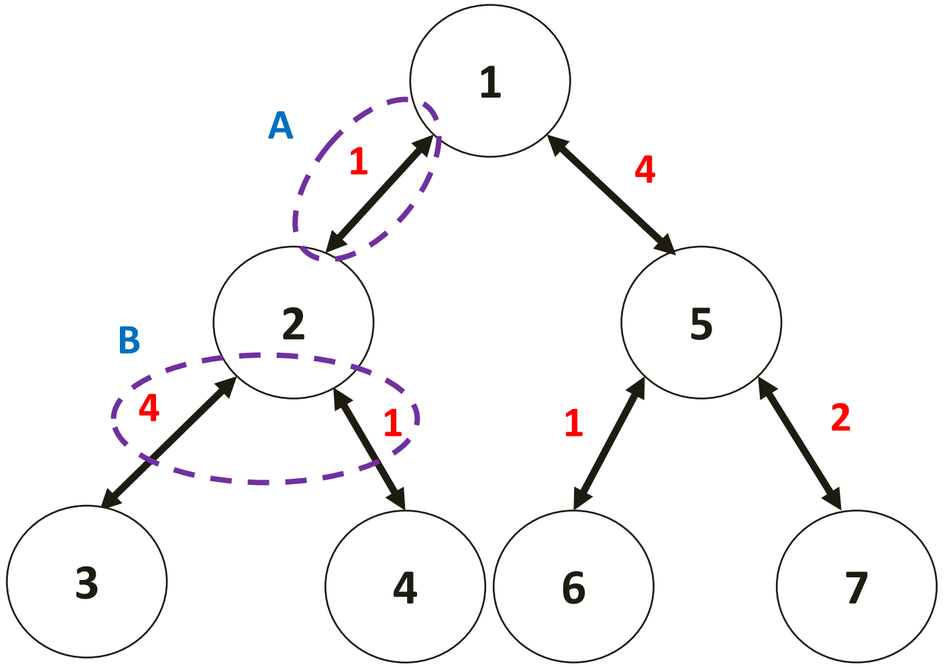


사실 모두 어떻게 해야 할지 대강 감이 왔을 것이다.

위에 보이는 보라색 영역의 간선에 주목 해보자.

편의상 위에 있는 영역을 A, 아래에 있는 두 영역을 B라고 하겠다.

루트 기준 트리 dp

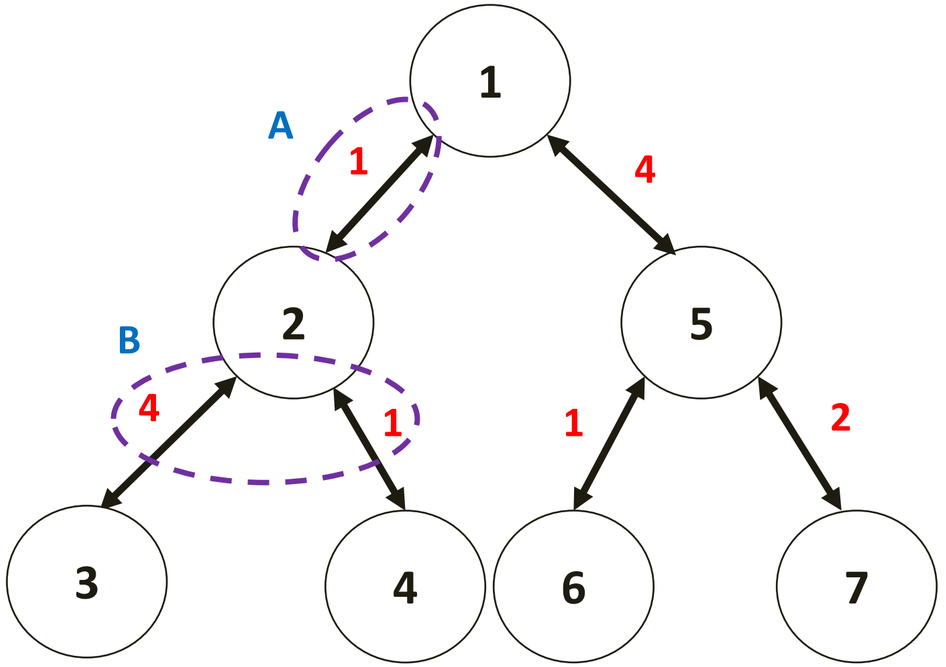


우리의 목표는 리프 노드가 루트 노드로 가는 경로가 없게끔 간선을 지우는 것이다.

이때 지우는 간선의 가중치의 합이 최소가 되어야 한다.

제일 쉬운 방법은 루트에 직접 연결된 간선을 모두 지우는 것이다.

루트 기준 트리 dp

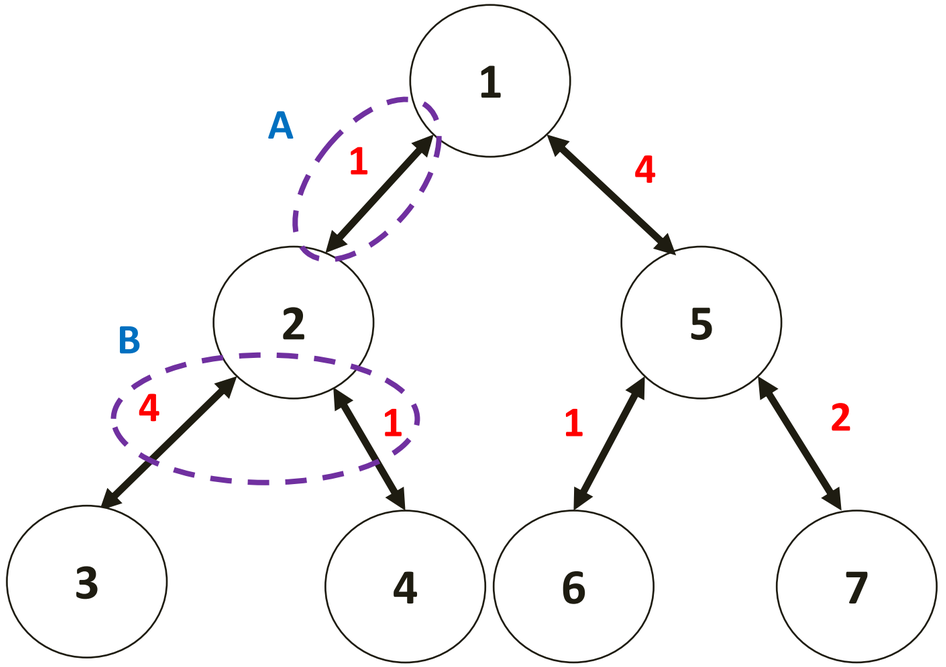


하지만 그 값이 항상 최소라고 보장 될 수가 없다.

현재 선택지는 2가지 있다.

1) 1번 ~ 2 번 간선을 끊어서 3번과 4번 정점을 무력화 시키기.
2) 2번 ~ 3번 간선 + 2번 ~ 4번 간선을 끊어서 3번과 4번 정점을 무력화 시키기.

루트 기준 트리 dp

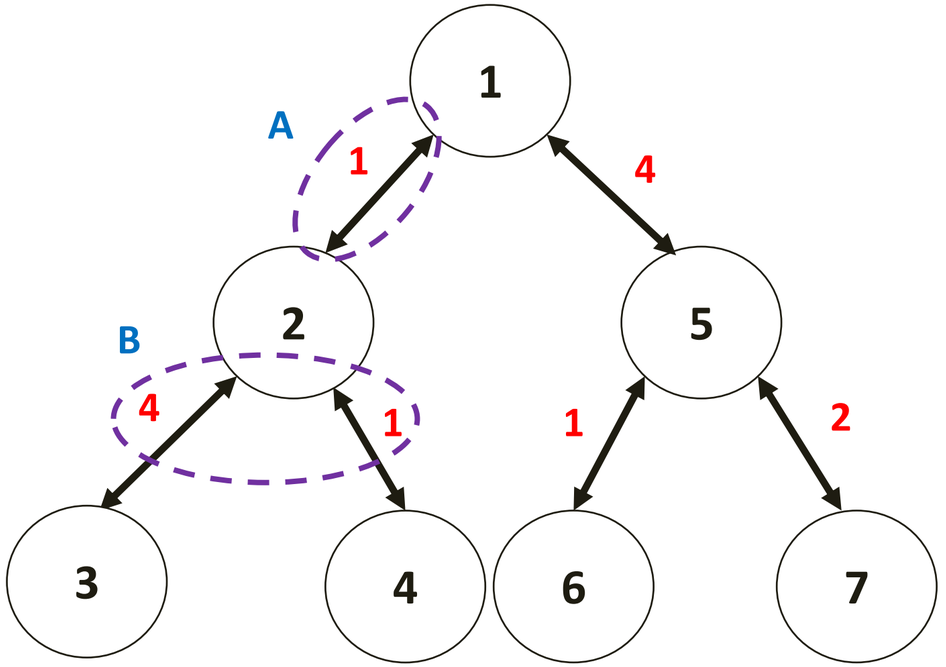


물론 2번 ~ 3번 간선과 1번 ~ 2번 간선을 끊는 방법도 유효하다.

하지만 1번 ~ 2번 정점을 끊어주는 것 만으로도 3번 간선과 4번 간선이 봉쇄 된다.

즉 가중치가 1 이상인 경우에 2번 ~ 3번 간선을 끊는건 필요 이상으로 과다하게 간선을 끊는다는 결론이 나온다.

루트 기준 트리 dp

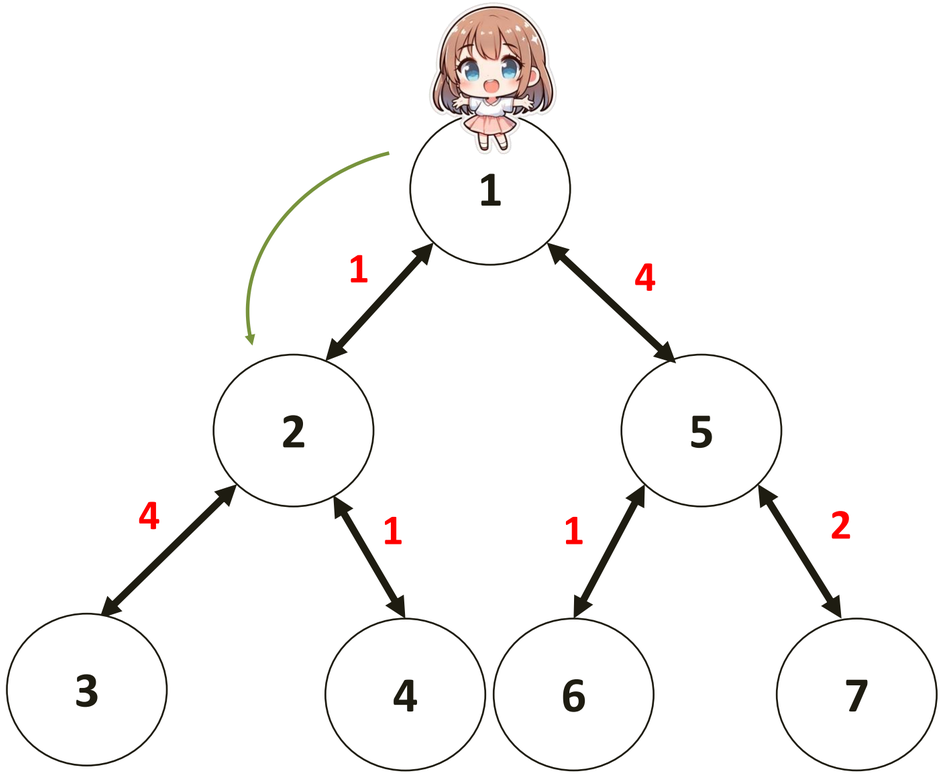


우리가 해결 해야하는 방법을 제시하면 아래와 같다.

현재 정점에서 자식 간선과의 연결을 끊을지, 혹은 하위 서브트리에서 끊을지 이다.

물론 트리가 이렇게 간단한 형태로 제시되지 않기 때문에 서브트리에 대해 재귀적으로 문제를 해결해 나가야 한다.

루트 기준 트리 dp



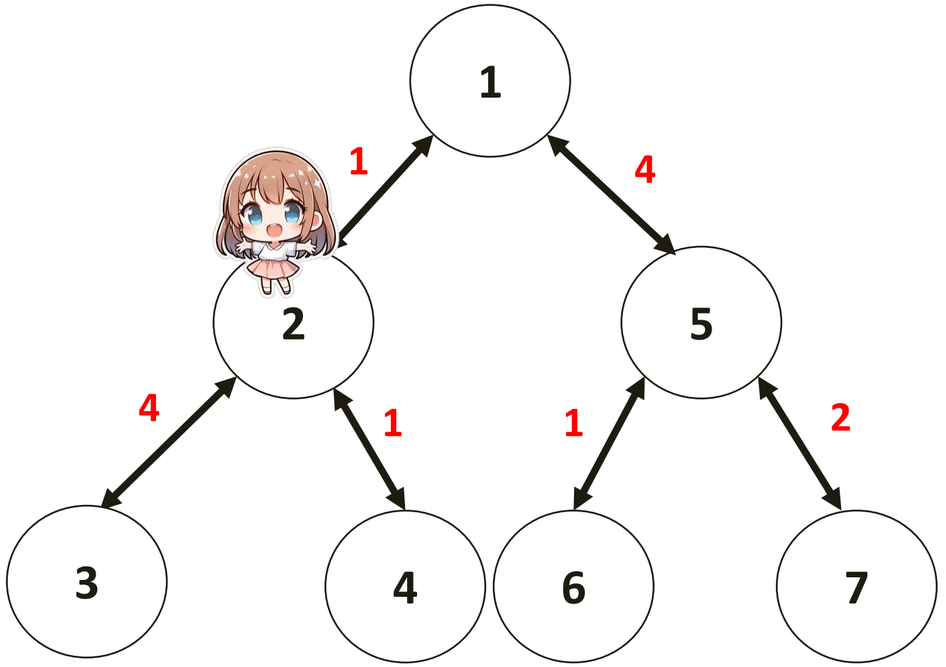
current : 1번

현재 정점은 1번(루트)이다.

루트에서 어느 정점을 지울지 선택해야 하므로 깊이 우선 탐색을 실시한다.

먼저 2번 정점으로 탐색한다.

루트 기준 트리 dp



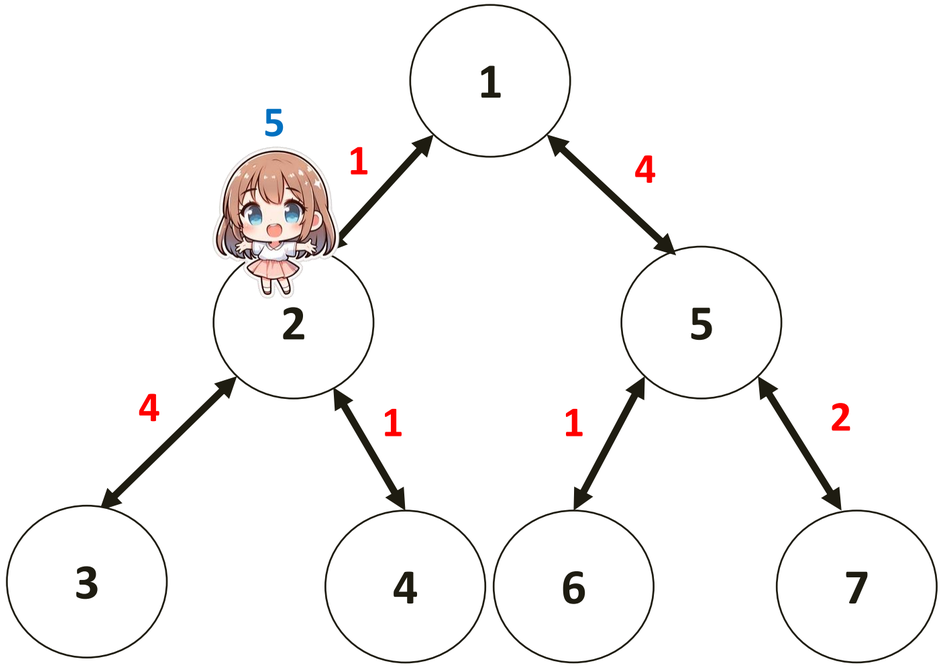
current : 2번

현재 정점은 2번이다.

2번 정점에서 어느 정점을 지울지 선택 해야 한다.

2번 정점은 3번 정점과 4번 정점을 자식으로 가진다.
이때, 3번 정점과 4번 정점은 모두 리프 노드이다.

루트 기준 트리 dp



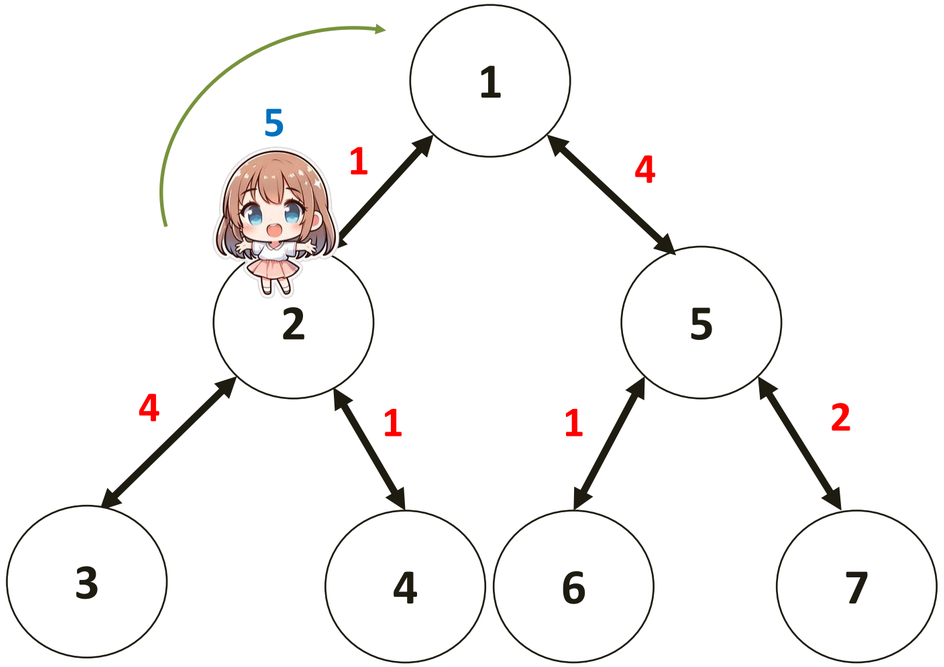
current : 2번

우리의 목적은 리프 노드가 루트로 연결되지 않게 간선을 끊어 주는 것이다.

리프 노드에서는 더 이상 자식이 없기 때문에 탐색을 할 필요가 없다.

즉, 현재 정점에서는 현재 자식들에 대한 간선을 전부 끊어 주는 것이 이득이다.
이러면 현재 정점(2번) 상태 공간에서의 값이 5(4+1) 로 갱신된다.

루트 기준 트리 dp



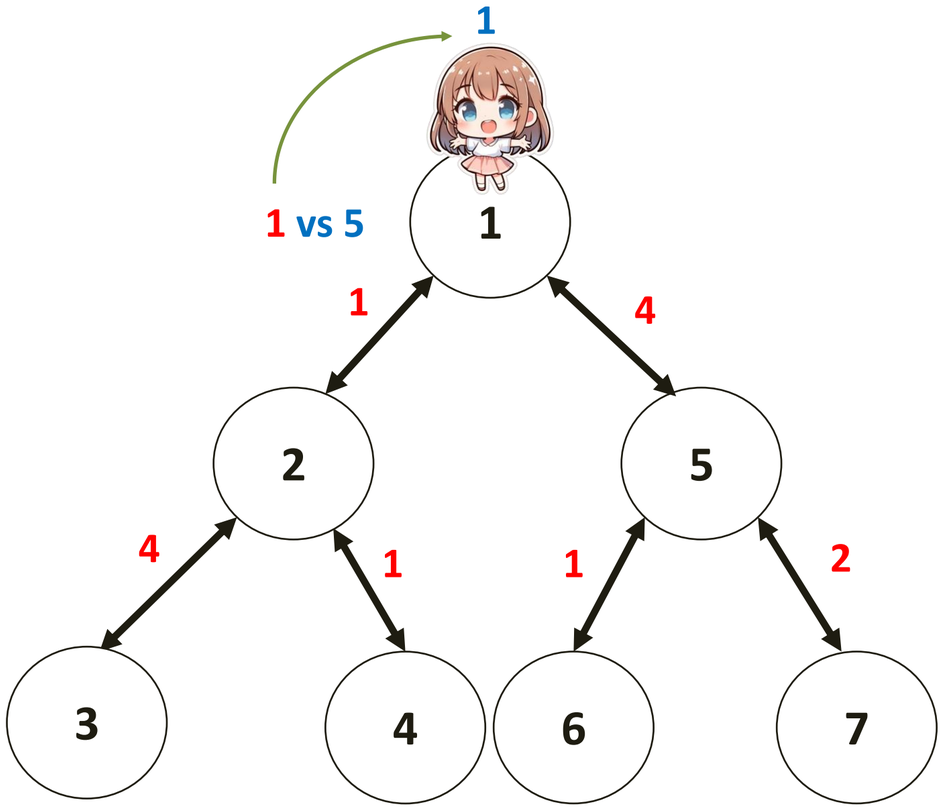
current : 2번

현재 2번 정점에서 탐색할 수 있는 모든 정점을 탐색 하였다.

현재 상태 공간의 최소값은 5였다.

5 값을 들고 1번 정점으로 복귀한다.(return)

루트 기준 트리 dp

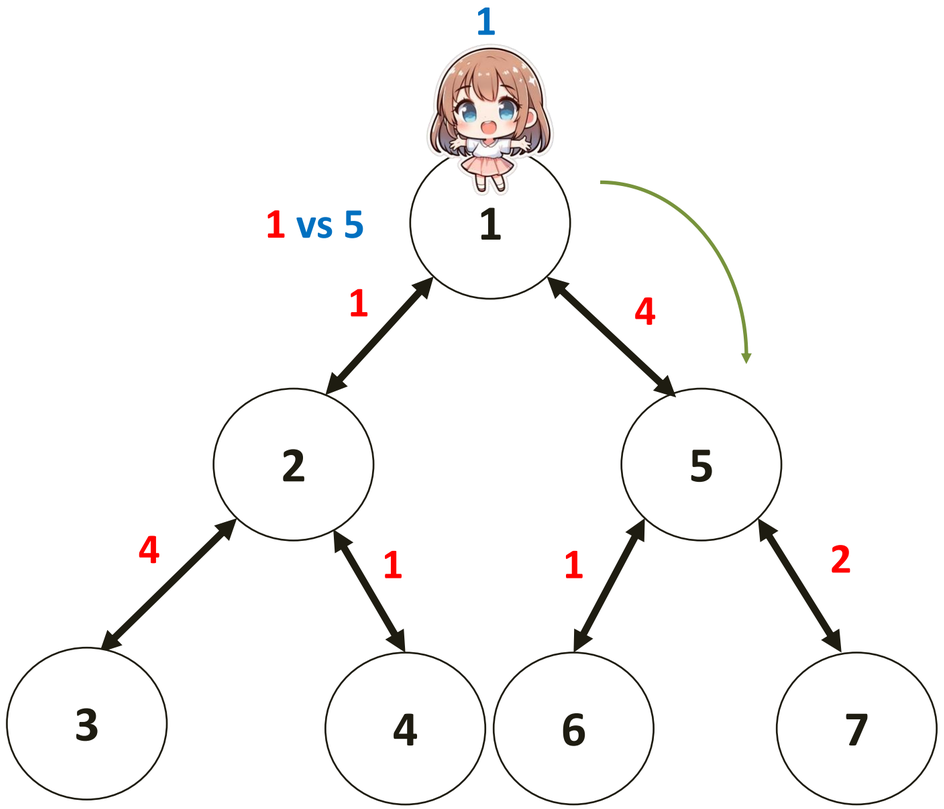


current : 1번

1번 정점으로 복귀 하였다.

2번 정점에서 가져온 값은 5였다. 이는 2번 정점에서 자식 간선을 끊어서 나올 수 있는 값 중에 최소 값이다.
하지만 2에서 5의 가중치로 간선을 끊는 것 보다 1번 정점에서 2번 정점으로 가는 간선을 끊는 것이 더 최소 이므로 1번 ~ 2번 간선을 끊어준다. 현재 1번 정점에서의 값은 1이다.

루트 기준 트리 dp



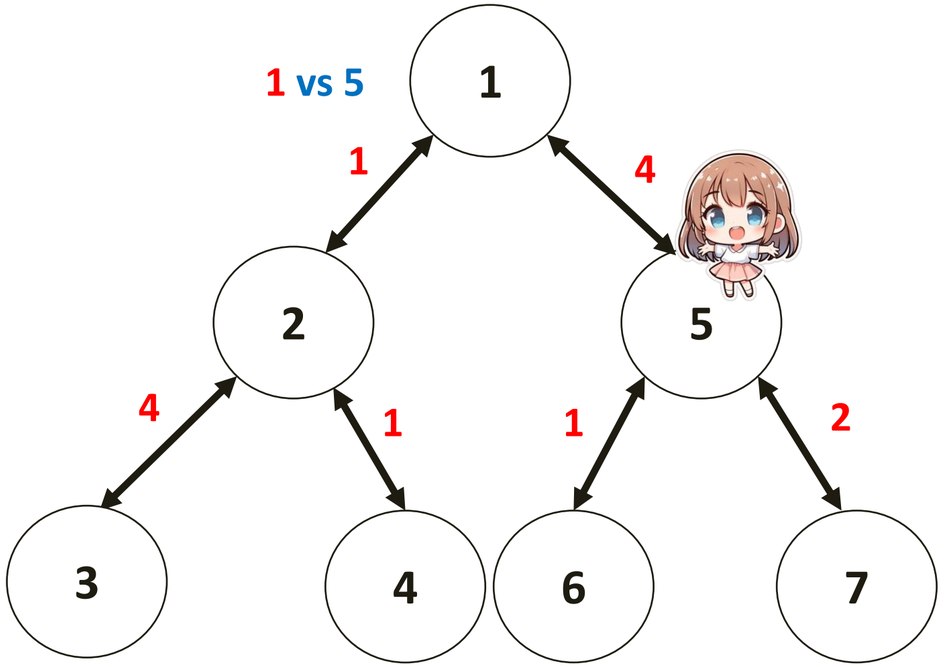
current : 1번

아직 1번 정점에서 갈 수 있는 정점이 남아있다.

현재 값을 변수에 보관한다.

이후, 5번 정점으로 깊이 우선 탐색을 실시한다.

루트 기준 트리 dp



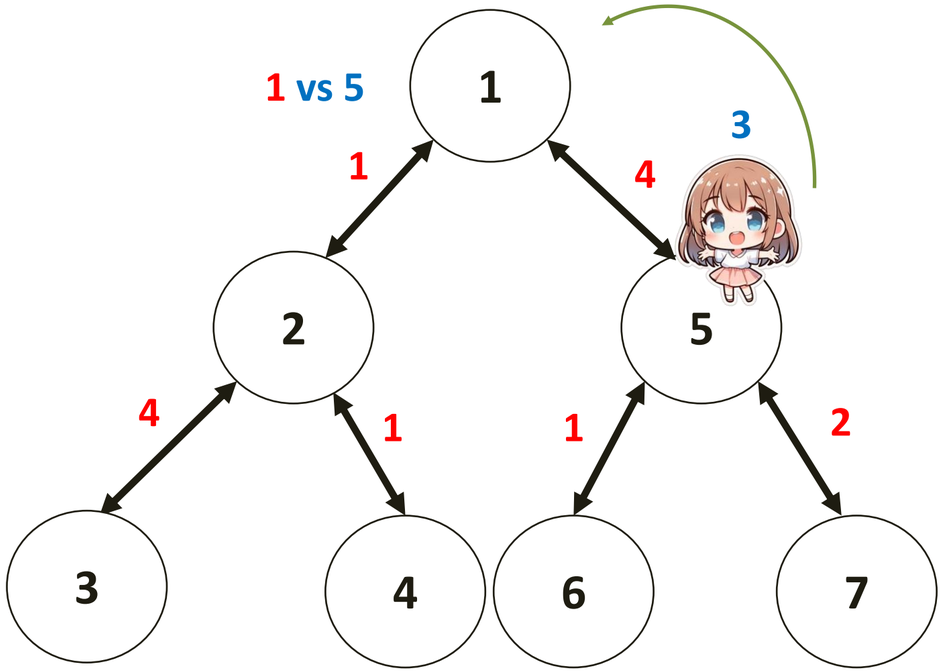
current : 5번

현재 정점은 5번이다.

5번 정점에서 어느 정점을 지울지 선택 해야 한다.

5번 정점은 6번 정점과 7번 정점을 자식으로 가진다.
이때, 6번 정점과 7번 정점은 모두 리프 노드이다.

루트 기준 트리 dp



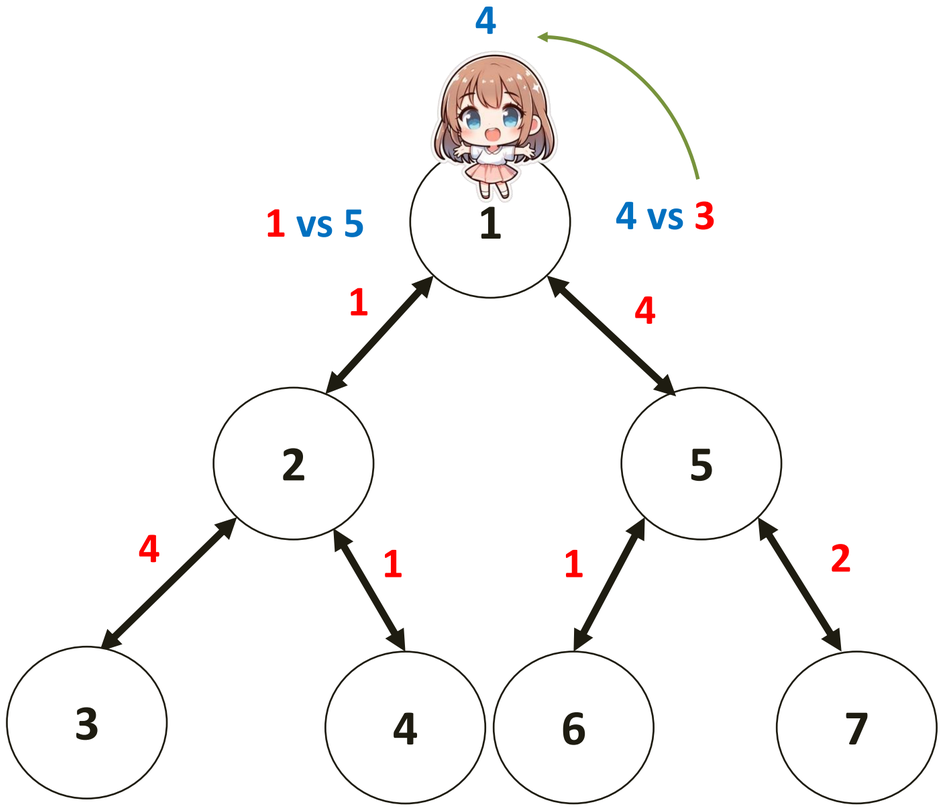
current : 5번

우리의 목적은 리프 노드가 루트로 연결되지 않게 간선을 끊어 주는 것이다.

리프 노드에서는 더 이상 자식이 없기 때문에 탐색을 할 필요가 없다.

즉, 현재 정점에서는 현재 자식들에 대한 간선을 전부 끊어 주는 것이 이득이다.
이러면 현재 정점(5번) 상태 공간에서의 값이 3(1+2) 로 갱신된다.

루트 기준 트리 dp

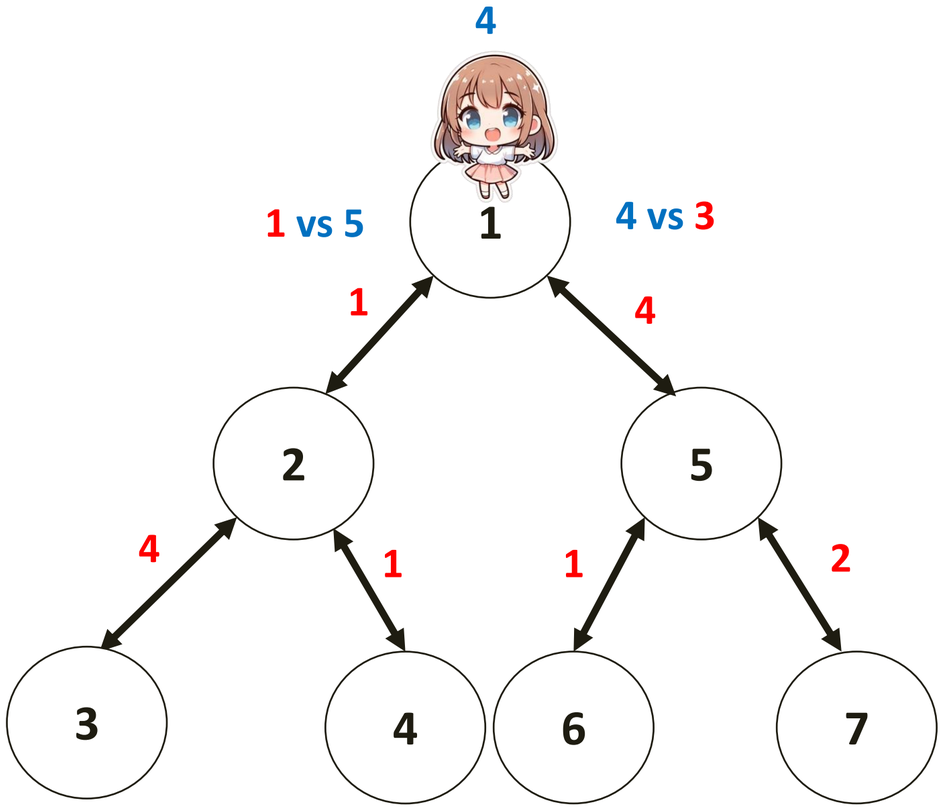


current : 1번

1번 정점으로 복귀 하였다.

5번 정점에서 가져온 값은 3였다. 이는 2번 정점에서 자식 간선을 끊어서 나올 수 있는 값 중에 최소 값이다.
아까와 다르게 5에서 3의 가중치로 간선을 끊는 것이 1번 정점에서 5번 정점으로 가는 간선을 끊는 것 보다 더 최소이다. 고로 반환값을 그대로 더해준다. 현재 1번 정점에서의 값은 4이다.

루트 기준 트리 dp

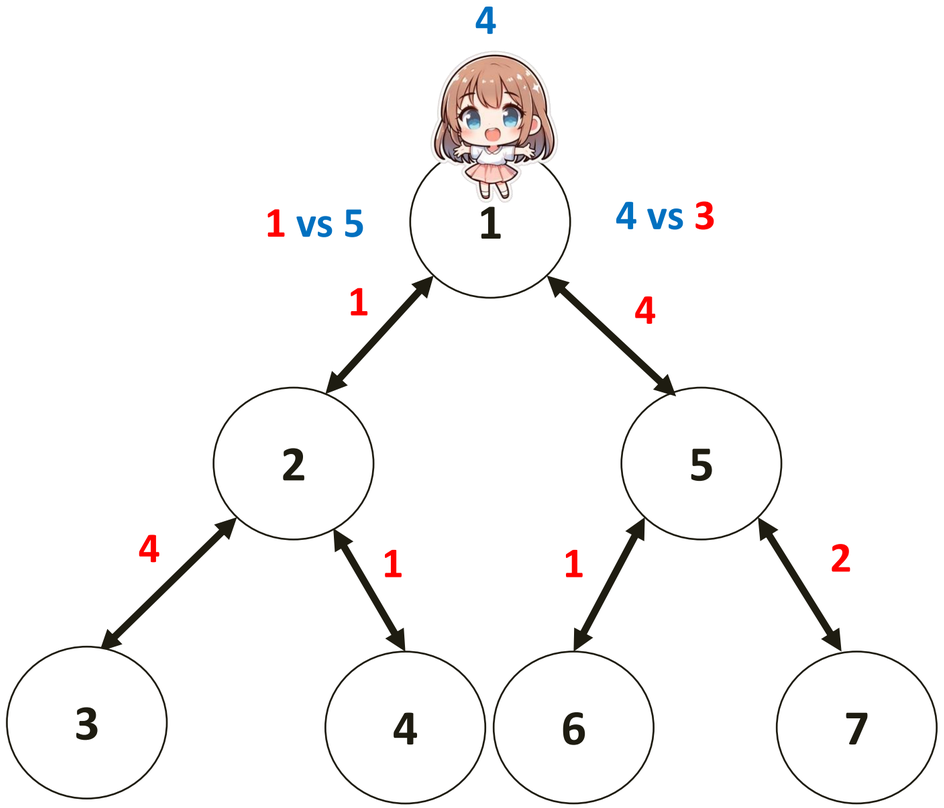


current : 1번

깊이 우선 탐색이 모두 끝났다.

리프 노드와의 연결된 경로가 없게 하기 위해 지울 수 있는 간선 가중치 최소의 합은 4이다!

루트 기준 트리 dp



트리 dp는 사실 dp 유형에서 매우 쉬운 축에 속한다.

분할 정복의 개념과도 매우 유사한 면이 있다.

트리를 그려 본 후, 상태를 어떻게 탐색 해야 할지만 고민하면 문제를 쉽게 풀 수 있다!