<알고리즘>

5장 요약문

# backtracking

백트래킹은 단계적으로 선택을 하면서 진행하는 것이다. 조건이 만족되지 않으면 이전 상태로 복귀하여 남은 것 중에 선택을 하는 것이다. 단계적으로 선택을 하기 때문에 Greedy 알고리즘으로도 구현할 수 있고, 동적 프로그래밍 혹은 분할정복 모두 구현할 수 있다.

백트래킹의 원리가 원하는 답을 얻을 때까지 반복을 하는 것이고, 탐색이 잘못되었다는 것을 알고나서 다시 원래 자리로 돌아와 아직 탐색을 시도하지 않은 곳으로 진행하는 것이기 때문에 worst case time complexity는 나쁘다. 정말로 운이 나쁘거나, 제대로 된 사전정보가 없을 경우 탐색을 할 가장 마지막에 원하는 해답이 있을 수 있기 때문이다. 하지만 average case time complexity는 다른 알고리즘 접근에 비해 꽤 효율적인데, 그 이유는 분할정복처럼 중복계산도 없고 제대로 된 사전정보나 가능성에 기반을 둔 탐색 순서를 정한다면 해답까지 탐색하여 도달하는 시간이 유의미하게 줄어들기 때문이다.

백트래킹의 일반적인 접근법은 가능한 모든 결과를 생성해둔 다음에 decision function에 의해 탐색해야 할 상태공간 트리를 가지치기하는 것이다. 앞서 언급한 것처럼 이와 같은 방식 때문에 worst case time complexity는 여전히 지수함수의 시간복잡도를 가지지만, instance가 큰 경우에 매우 효율적인 알고리즘이다.

코딩을 어떠한 문제를 풀 때, 8 Queen 같이 특정 상태를 얻어내고자 하는 문제라면 백트래킹은 정말로 감사한 문제이다. 처음 이 문제를 풀어보았을 때 백트래킹 없이 순수하게 반복문과 재귀호출로만 만들어보려고 했으나, 도저히 감당할 수 없을 정도로 많은 경우의 수를 처리할 수 없었기에 헛된 시간을 보냈었다. 나중에 tictactoe 같은 문제에서도 유용하게 쓰이는 알고리즘이니까 이번 참에 어떤 방식으로 동작하는지 제대로 알아둘 필요가 있다.