backtracking 알고리즘
- Einstein's riddle

제출일자: 2022. 05. 29

소프트웨어학부
2020203081
심유미

1. 알고리즘 아이디어

15가지의 조건을 보고, 5개의 집이 각각 어떤 특성을 갖고 있는지 분류를 해주는 문제이다.

효율적인 알고리즘을 만들기 위해, 첫째로 불필요한 연산을 줄였다. 각 노드는 5개의 집으로 이루어져 있고, 하나의 집은 5가지 특징(색상, 국적, 음료, 시가, 동물)으로 구성돼있다. 이때 각 특징은 5개의 집 사이 겹치는 게 있어선 안 된다. 조건(Hints)을 따지지 않고 모든 노드를 구하면 600개(5 * 5! = 600)가 구해진다. 즉, 5개의 특징을 단번에 부여할 경우 600개의 노드를 모두 확인해야 한다. 600개의 노드 중에는 조건 한두 개만으로도 Non-Promising 함이 드러나는 노드가 존재하므로, 특징을 하나씩 부여해가며 조건을 즉시 대입하여 불필요한 노드는 일찍이 연산을 중단하도록 했다.

탐색하는 노드의 수를 줄이기 위해, 둘째로 트리의 상단부에서 Non-Promising 한 노드를 최대한 제거했다. 아무런 특징도 부여되지 않은 빈 root 노드부터 시작하여, depth가 1씩 깊어질 때마다 부여받은 특징과 관련된 조건을 확인한다. 조건을 만족한다면 또 하나의 특징을 부여하여 자식 노드를 만들고, 조건을 만족하지 않는다면 연산을 중단하고 이전 노드로 돌아간다.

15가지의 조건을 미리 분석하여 ① 어느 특징의 위치가 확정되는 조건, ② 두 가지 특징을 구해 한 번의 비교를 하는 조건, ③ 두 가지 특징을 구해 여러 번의 비교를 할 수도 있는 조건으로 분류했다. 적은 연산으로 Non-Promising 한 노드를 찾을 수 있도록, 조건을 적용하는 순서를 정하고, 그에 따라 특징들의 할당 순서도 정했다.

2. 알고리즘 흐름

알고리즘의 메인 함수와 다름없는 checknode 함수는 노드에 5가지 특징을 하나씩 부여하고, 조건을 만족하면 다음 자식 노드로 넘어가는 흐름을 잡아준다. 특징을 부여할 때는 allocate_node 함수를 사용하여 별도의 공간에서 변수에 값을 부여하는 것으로 구현했고, 자식 노드로 넘어가는 것은 변수 depth를 증가시키는 것으로 표현했다.

가. allocate_node 함수

allocate_node 함수는 노드와 깊이, 그리고 0부터 4까지 나열된 숫자를 전달받는다. 5개의 숫자는 색상, 국적, 음료, 시가, 동물이 가지고 있는 5가지의 종류들을 의미한다.

```
enum bese_col { blue = 0, green, red, white, yellow } bese_col;
enum bese_nat { Brit = 0, Dane, German, Norwegian, Swede } bese_nat;
enum bese_bev { beer = 0, coffee, milk, tea, water } bese_bev;
enum bese_cig { Blue_Master = 0, Dunhill, Pall_Mall, Prince, Blend } bese_cig;
enum bese_pet { cat = 0, bird, dog, fish, horse } bese_pet;
```

색상, 국적, 음료, 시가, 동물이 가지고 있는 5가지의 종류

allocate_node 함수의 상단에서는 전달받은 5개의 숫자에 중복이 있는지 확인한다. 각 집이 가지고 있는 특징들은 다른 집과 겹쳐서는 안 되기 때문이다. 숫자에 중복이 없다면 다음으로 depth를 확인하여 어떤 특징을 부여할 차례인지를 확인한다. depth(1~5)에 따라 국적, 색상, 음료, 시가, 동물 순서대로 자식 노드에 특징을 부여해준다.

나. 변수 depth와 포인터 변수 pcount

depth는 트리의 깊이를 의미한다. Promising 함을 확인한 노드가 자식 노드를 만들면서 트리의 깊이는 1 증가한다. 이를 변수 depth를 1 증가시키는 것으로 나타냈다.

checknode 함수를 호출했다는 것은 노드를 탐색한다는 의미와도 같다. 노드를 탐색하는 횟수를 구하기 위해 checknode 함수가 호출될 때마다 count라는 변수를 1씩 증가하기로 했고, 모든 checknode 함수 호출에 걸쳐 이를 확인해야 하므로 변수 count를 가리키는 포인터 변수 pcount를 사용하기로 했다.

다. print_node 함수

깊이가 5인 노드는 모든 특징을 할당받은 노드임을 의미한다. 이 노드가 promising 함수를 거쳐 조건 15개를 모두 만족함을 증명하고 나면 노드를 이루고 있는 5개의 집이 각각 어떤 특징을 가지고 있는지 출력한다. 동시에 checknode 함수를 몇 번이나 호출했는지, 즉 노드 탐색이 몇 번 이루어졌는지도 함께 출력한다.

3. promising 함수

depth가 0인 경우를 제외하고, Promising 함수는 총 5단계로 나뉜다. 각각 국적(nationality)이 할당됐을 때, 색상(colors)이 할당됐을 때, 음료(beverage)가 할당됐을 때, 시가(cigar)가 할당됐을 때, 동물(pet)이 할당됐을 때이다.

가. 국적 할당

힌트 9번은 '(①) 가장 앞집에 노르웨이인이 있다.'라는 명확한 조건을 제시하여 Promising 한 노드를 한정할 수 있다. 가장 처음 적용하는 조건으로 힌트 9번을 택함으로써, 노드에 처음 할당하는 특징도 국적으로 정했다.

나. 색상 할당

힌트 14번은 단독으로 볼 경우 '(③) 노르웨이인이 사는 집과 파란색 집은 옆집 사이'임을 확인해야 한다. 그러나 힌트 9번을 만족하는 노드의 경우 노르웨이인이 사는 집의 위치가 고정되었으므로 '(①) 파란색 집의 위치는 두 번째 집으로 고정'된다고 볼 수 있다. 이에 따라 노드에 두 번째 할당하는 특징도 색상으로 정했다. 국적과 색상이 할당되었을 때 추가로 확인할 수 있는 힌트에는 1번이 있다. 힌트 1번을 적용하여 '(②) 영국인이 사는 집과 빨간색 집은 같은 위치'임을 만족하는 노드는 자식 노드를 만들어 다음 특징을 할당받는다.

다. 음료 할당

'(①) 가운데 집 주인이 우유를 마신다.'라는 명확한 조건, 힌트 8번이 있으므로 국적, 색상 다음으로 음료를 할당하기로 했다. 이 세 가지 특징이 할당되었을 때 확인할 수 있는 조건에는 힌트 3번, 4번, 5번이 있다. 힌트 3번, 5번, 4번 순서대로 '(②) 덴마크인이 사는 집과 차를 마시는 집이 같은 위치'인지, '(②) 초록색 집과 커피를 마시는 집이 같은 위치'인지, '(②) 하얀색 집 왼쪽에 초록색 집이 있는지'를 따져본다.

라. 시가 할당

시가와 관련된 조건에는 7번, 12번, 13번, 15번이 있다. 7번, 12번, 13번을 우선 확인하여 '(②) 노란색 집과 Dunhill을 피우는 집이 같은 위치'인지, '(②) Blue Master를 피우는 집과 맥주를 마시는 집이 같은 위치'인지, '(②) 독일인이 사는 집과 Prince를 피우는 집이 같은 위치'인지를 따져본다. 그다음으로 힌트 15번, '(③) Blend를 피우는 집과 물을 마시는 집이 옆집 사이'인지 판별한다. 힌트 15번은 경우에 따라 Blend를 피우는 집이 왼쪽에 위치할 수도, 물을 마시는 집이 왼쪽에 위치할 수도 있기 때문에 특징이 배열된 순서에 따라 비교 횟수가 두 번 이상이 될 수 있다. 그렇기에 나중에 확인하는 것이다.

마. 동물 할당

마지막으로 동물까지 할당하고 나면 노드는 모든 특징을 부여받게 된다. 남은 힌트 중 힌트 2번과 6번을 먼저 적용해 '(②) 스웨덴인이 사는 집과 개를 키우는 집이 같은 위치'이고, '(②) Pall Mall을 피우는 집과 새를 키우는 집이 같은 위치'임을 확인하고, 다음으로 힌트 10번과 11번을 적용해 '(③) Blend를 피우는 집과 고양이를 키우는 집이 옆집 사이'이고, '(③) Dunhill을 피우는 집과 말을 키우는 집이 옆집 사이'임을 확인한다.