Assignment5 Report

2014147550
컴퓨터과학과 강효림

1. 알고리즘 개관

가. repeatedRandom

1) 알고리즘 설명:

define된 randomWalk 방식에 따라, 초기 해에서 iteration만큼의 step을 이동한다. hillClimbing과는 달리, 옮겨간 자리가 옮겨가기 전의 자리보다 해의 quality가 좋지 않아도 무조건 이동하고, 다음 randomWalk는 옮겨간 그 자리에서 시작된다. 이때 옮겨간 자리(해)가 지금까지 탐험한 해 공간 중에서 가장 좋다면 그 값을 저장한다. iteration이 끝난 후, 탐험한 해 공간 중 가장 좋은 질을 가졌던 해를 return한다.

2) 관련된 code:

repeatedRandom()이 주 로직이며, 인접 해로 옮겨가는 데에는 randomWalk()함수가 쓰인다.

나. hillClimbing

1) 알고리즘 설명:

초기해에서 neighboring 해 공간으로 이동하였을 때 해의 quality가 improved된다면 그 이동을 커밋하고, 아니라면 롤백한다. 즉 해의 quality가 증가하는 방향으로 greedy하게 움직인다. iteration 끝날 때 위치하고 있는 해 공간의 해를 return 한다.

2) 관련된 code:

hillClimbing()이 주 로직이며, 인접 해로 옮겨가는 데에는 goToNeighbor()함수가 이용된다.

다. simulatedAnnealing

1) 알고리즘 설명:

hillClimbing과 비슷하나, 더 탐험의 여지를 둔다. 즉 greedy하게 움직이지 않고, 일정 확률로 해의 quality가 imporve되지 않음에도 인접 해로 이동한다. 그 확률은 iteration이 증가할수록 줄어들게 된다.

2) 관련 code:

simulatedAnnealing()이 주 로직이며, hillClimbing과 마찬가지로 goToNeighbor()함수를 사용한다. 다만 탐험 확률을 계산하기 위하여 T()라는 보조 함수를 사용하며, hillClimbing과 같이 iteration의 가장 마지막에 위치한 해를 return하는게 아닌, 탐험한 해 공간 중 가장 좋은 해를 return하는 로직이기 때문에 가장 좋은 해를 저장하기 위한 별도의 공간이 필요하다.(code에선 b1, b2)

라. genetic

1) 알고리즘 설명:

한 generation은 2048개의 gene으로 구성된다. 모든 구성원이 참여하는 토너먼트를 통하여 교배할 한 쌍을 선택하는데, 이 토너먼트에서 좋은 해가 나쁜 해를 이길 확률은 80%이다. 선택된 두 해는 2점 교차와(드문 확률로 1점 교차) 일정 확률로 일어나는 mutation process를 거쳐 교배되고, 그 결과로써 child gene을 생성한다. 이러한

토너먼트-교배는 1024번 일어나고 이 1024개의 새로운 child gene들은 해의 질을 기준으로 전(前)generation의 하위 50%에 속하는 gene들을 대체한다(대체율=0.5). 이렇게 다음 generation이 형성된다. 가장 좋은 유전자형이 전체 generation에서 70% 이상을 차지할 때, 알고리즘이 수렴한 것으로 간주하고 가장 좋은 gene을 해 형식으로 변환하여 return한다. 해의 gene 표현형은 binary string으로 하였다.

2) 관련 code:

genetics()가 주 로직이며, child는 f()라는 함수로 만들어진다. f() 내에서 토너먼트 시행을 위한 doTournament()함수, 선택된 두 gene의 교배를 위한 sex()함수가 사용된다.

```
public static Gene f(List<Gene> pop) {
    List<Gene> l1, l2, tmp;
    l1 = new ArrayList<>(pop.size());
    l2 = new ArrayList<>(pop.size());
    for (Gene g : pop) {
        l1.add(g);
    }
    while(true) {
        while(!l1.isEmpty()) {
            Gene g1 = l1.remove(0);
            Gene g2 = l1.remove(0);
            l2.add(doTournament(g1, g2, 0.8));
        }
        tmp = l1;
        l1 = l2;
        l2 = tmp;
        if (l1.size() == 2) {
            break;
        }
    }
    return sex(l1.remove(0), l1.remove(0));
}
```

2. 결과 분석

가. 실행 환경 및 가정:

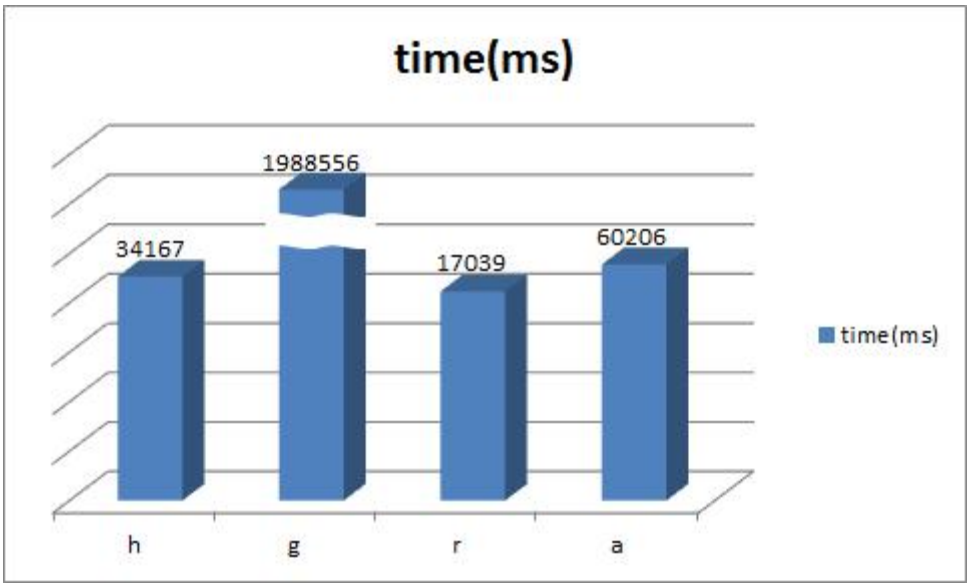
iteration은 250,000번으로 하였고, 횟수는 100으로 하였다. 해의 quality는 residue가 작으면 좋은 것으로 보며, 그래프는 해의 quality가 가장 안 좋은 hillClimbing 알고리즘을 1로 두고, 나머지 알고리즘이 몇 배 더 좋은지를 나타낸 것이다. (모든 iteration에서 각 알고리즘이 도출한 residue를 합하여 측정) 또한 실행시간은 milisecond단위로 측정하였다.

나. 100번의 시행에서 각 알고리즘이 최적해를 도출한 횟수

종류	GA	RR	HC	SA
횟수	4	28	3	65

다. 해의 quality

라. 실행 시간



3. 결과 분석

해의 quality는 넓은 해 공간을 탐색하면서도 어느 정도 guided도 되는 Simulated Annealing이 가장 좋았고, 전혀 guided되지는 않지만 넓은 해 공간을 탐색하는 Repeated Random 알고리즘이 두 번째로 좋은 성능을 나타냈다. Genetic 알고리즘은 위 두 알고리즘보다는 확연히 떨어지지만, 나름대로 가장 나쁜 Hill Climbing알고리즘에 비해 2배 정도 좋은 성능을 나타내어 Hill Climbing보다 유의미하게 좋은 성능을 나타냈고, 극히 적은 공간만을 보게 되는 Hill Climbing이 가장 나쁜 성능을 보였다.

수행 시간은 많은 연산을 동반하는 Genetic 알고리즘이 압도적으로 많았고, 확률 계산을 위해 복잡한 수식을 계산해야 하는 Simulated Annealing이 나머지 두 알고리즘에 비해 유의미하게 오래 걸렸다.