**유전 알고리즘 보고서**

김 대 연

1. **동기**

어떠한 문제를 해결한다고 했을 때 그 문제를 단순하게 접근하면 너무 많은 시간이 걸려서 해결하지 못할 때가 있다. 예를 들어서 많은 점들을 이어서 시작한 점으로 돌아온다고 했을 때의 최단경로를 찾는다면 단순하게 생각했을 때 경우의 수를 일일이 찾는 방법을 떠올리게 된다. 하지만 이러한 방법으로 코드를 만들게 되면 엄청난 시간 복잡도를 가지게 된다. 이런 부분들을 해결하고자 여러 방법을 찾던 도중 유전 알고리즘을 이용하면 완벽하진 않지만 정답에 근접할 수 있다는 것을 알게 되어 그것에 대하여 탐구해 보고자 한다.

1. **선행 연구**

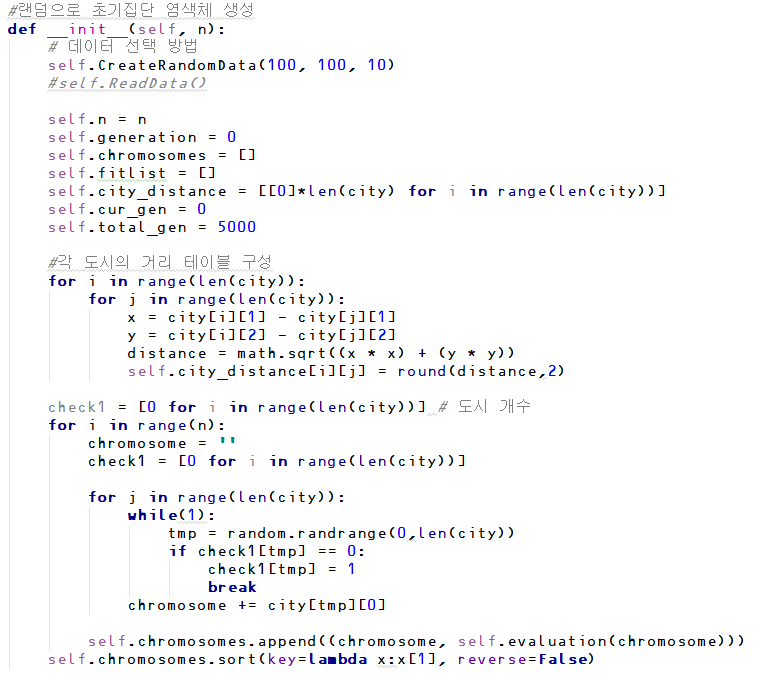
유전 알고리즘은 존 홀랜드(John Holland)가 1975년에 처음 소개한 최적화 기법이며 실제 생물 진화를 모방해서 문제를 해결하는 진화 연산의 대표적인 방법이다. 이는 자연계의 유전학에 바탕을 두며, 특히 다윈의 적자생존 이론을 기본 개념으로 한다. 유전 알고리즘을 정의하기 위해 아래와 같은 개념들을 정의한다.

* **염색체(chromosome) :** 생물학적으로는 유전 물질을 담고 있는 하나의 집합을 의미하며, 유전 알고리즘에는 하나의 해(solution)를 표현한다.
* **유전자(gene) :** 염색체를 구성하는 하나의 요소로서, 하나의 유전 정보를 나타낸다.
* **자손(offspring) :** 특정 세대에 존재했던 염색체들로부터 생성된 염색체를 자손이라고 한다. 자손은 이전 세대와 비슷한 유전 정보를 갖는다.
* **적합도(fitness) :** 어떠한 염색체가 갖고 있는 고유값으로서, 해당 문제에 대해 염색체가 표현하는 해가 얼마나 적합한지를 나타낸다.

1. **일정,계획**

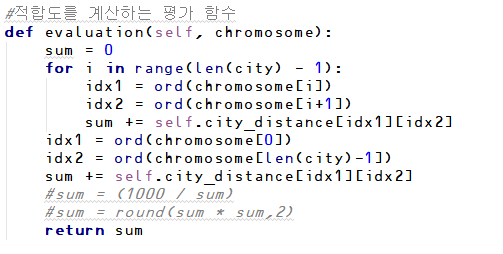
* 7/2 ~ 7/9 : 프로젝트 시작, 자료 조사 및 선행 연구 시작
* 7/11 ~ 7/25 : 핵심 코드 구현 및 인위적 데이터 결과 산출
* 7/30 ~ 8/9 : 시각화 코드 구현 , 랜덤 및 실제 데이터 결과 산출
* 8/13 ~ 8/20 : 보고서 작성, 프로젝트 마무리

1. **핵심엔진 및 구성별 세부 내용**
   1. **초기집단 생성 및 도시간 거리 저장**

****

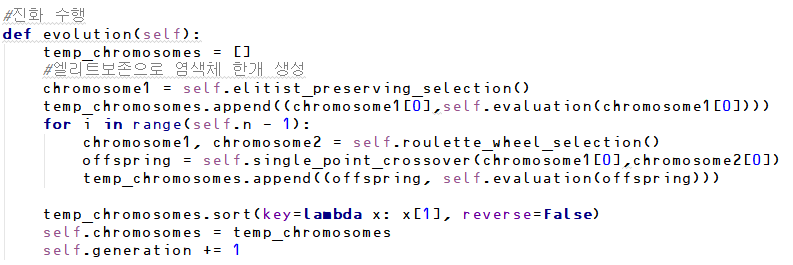
먼저 초기 세대의 염색체는 이전 세대의 염색체가 존재하지 않기 때문에 자손을 생성할 수가 없다. 따라서, 초기 염색체를 생성하는 연산을 별도로 정의했다. 유전 알고리즘에서 가장 많이 이용되는 방법은 어떠한 규칙도 없이 단순히 임의의 값으로 염색체를 생성하는 것이다. 또한 적합도 평가를 효율적으로 하기 위해 각 도시의 거리를 미리 city\_distance 배열에 저장해 두었다.

* 1. **적합도 평가**

****

미리 구해놓은 거리정보 배열(city\_distance)을 통해 각 유전자의 거리를 누적시킨 값을 반환한다.

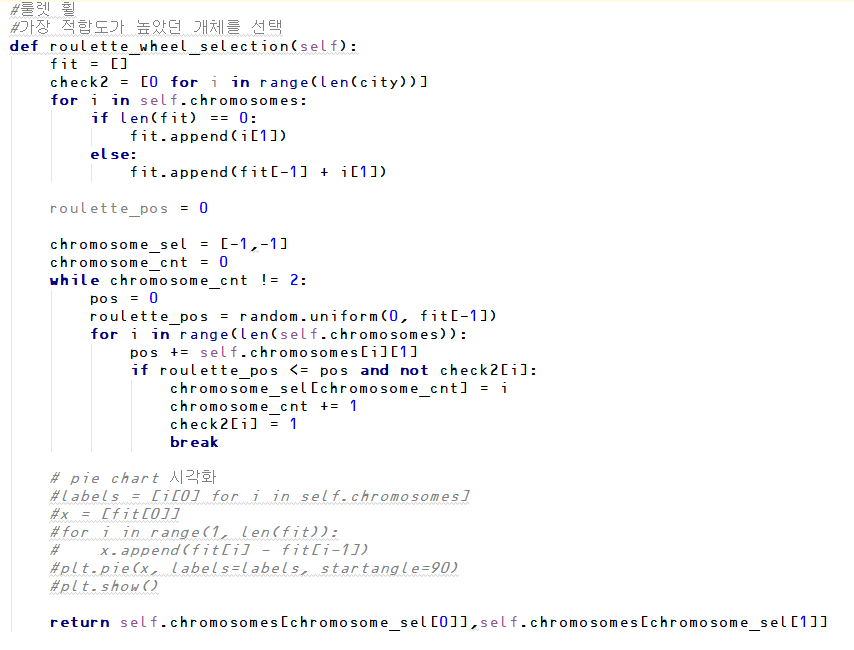
* 1. **진화 수행**



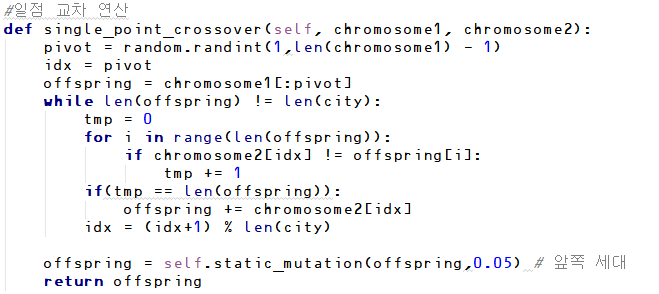
총 N 개의 해집단 중 1개를 엘리트 보존 선택 방식으로 선택하고, 나머지 N-1개의 해집단은 룰렛 휠을 통해 선택했다. 그리고 룰렛 휠을 통해 선택된 2개의 부모 염색체들을 일점 교차 연산을 통해 교배시키고, 5%의 확률로 변이를 일으켰다. 엘리트 보존 연산은 정렬되어있는 염색체들 중 가장 적합도가 낮은 염색체를 반환시켰으며, 룰렛 휠 연산은 적합도가 가장 좋은 염색체가 선택될 확률을 가장 높게 설정하고 확률적으로 염색체를 선택하였다. 현재 염색체에서 모든 적합도를 누적하고, roullete\_pos 변수에 0과 누적된 적합도 사이의 값을 랜덤으로 받아서 2개의 부모 염색체를 chromosome\_sel 배열에 저장할 수 있었다.



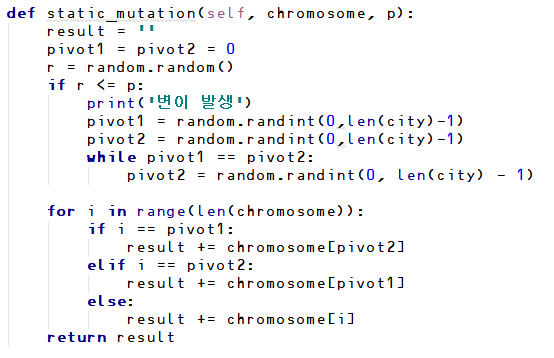
▲ 엘리트 보존 연산



▲ 룰렛 휠 연산

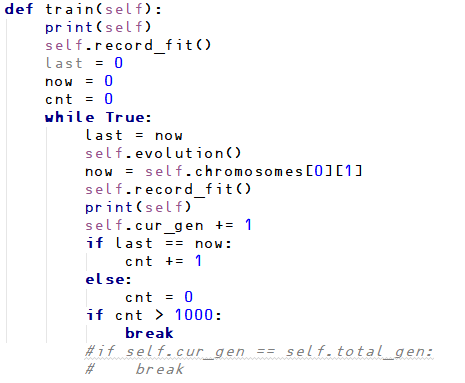


선택된 2개의 부모 염색체를 교배시켜 자식 염색체를 생성한다. 염색체의 길이가 N 일때 1 과 N-1 사이의 값을 지정하고, 지정된 위치를 기준으로 두개의 부모 염색체에서 가져와 자손 염색체를 생성할 수 있었다. TSP 에서는 도시를 한 번만 경유해야 하기 때문에 중복되는 도시가 존재할 경우, 중복되지 않는 도시들을 순서대로 가져와야 했다. 예를 들어 선택된 두 염색체가 [B C E D] 와 [D E C B] 이고 지정된 위치가 2일 경우, 자손은 [B C C B]가 아니라 [B C D E]가 된다.



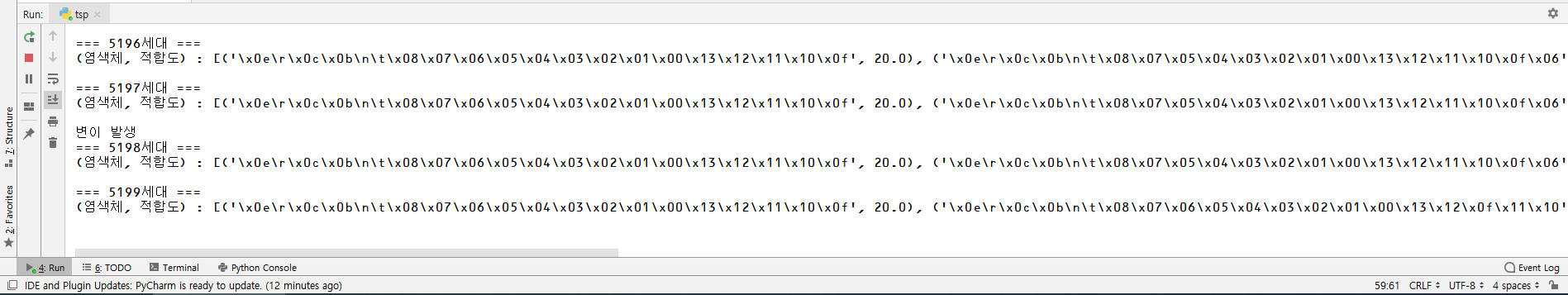
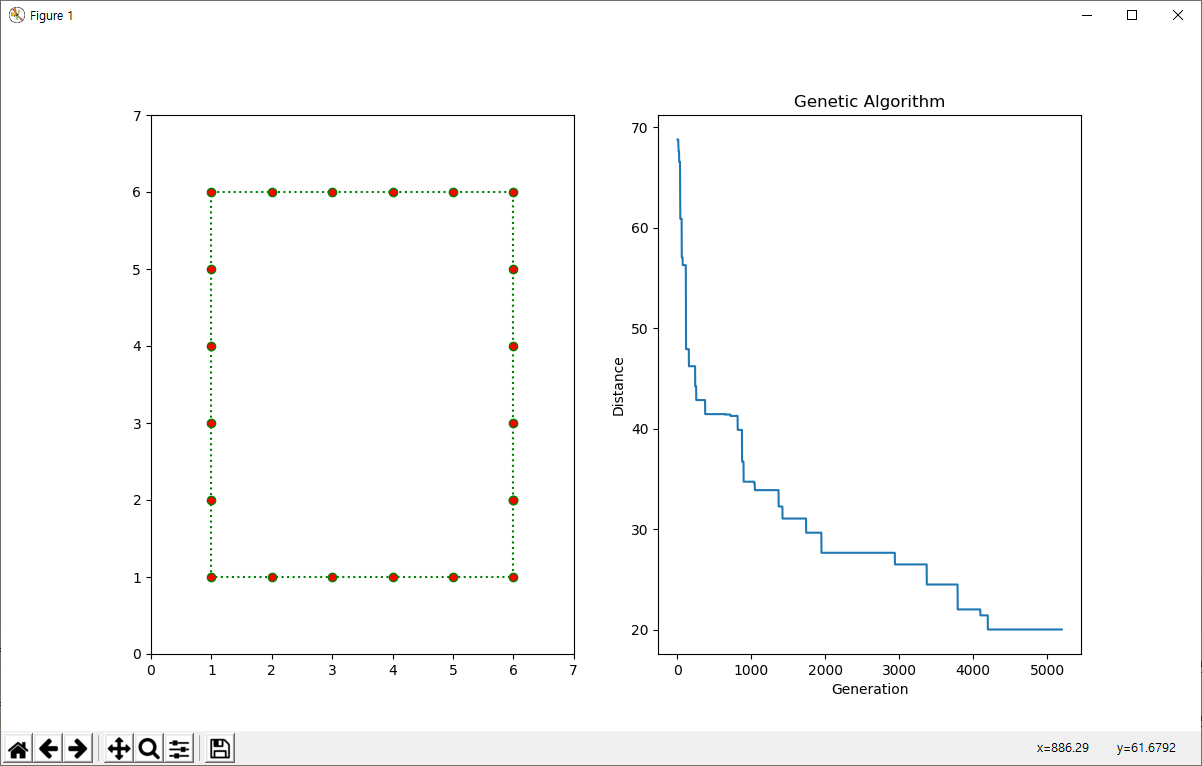
돌연변이를 일으키는 과정은 두 유전자를 임의로 선택하여 교환하는 방식으로 하였다. 그리고 돌연변이가 일어났을 때 별도의 출력을 하게 했다. 여기서는 돌연변이를 일으킬 확률을 5%로 지정했다. 처음 변이확률을 0.1%로 지정했을 때와는 확연한 차이를 느낄 수 있었다.

* 1. **종료 조건**

****

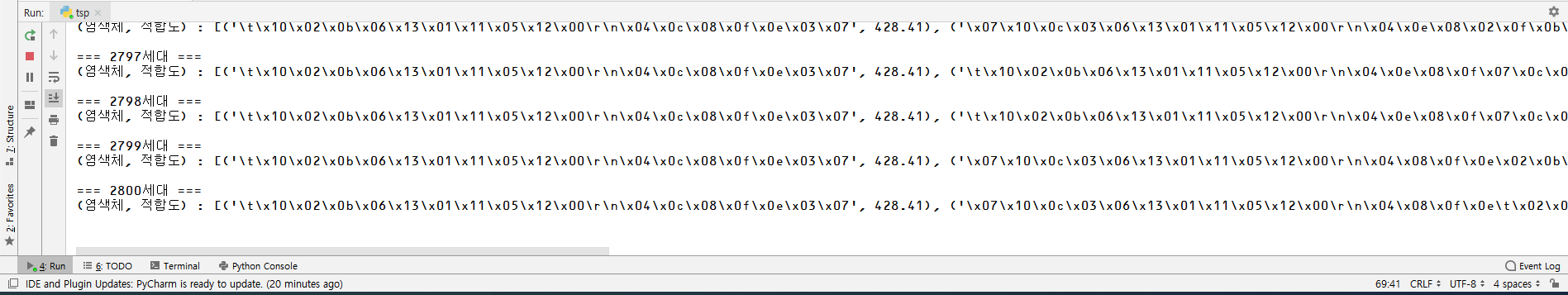
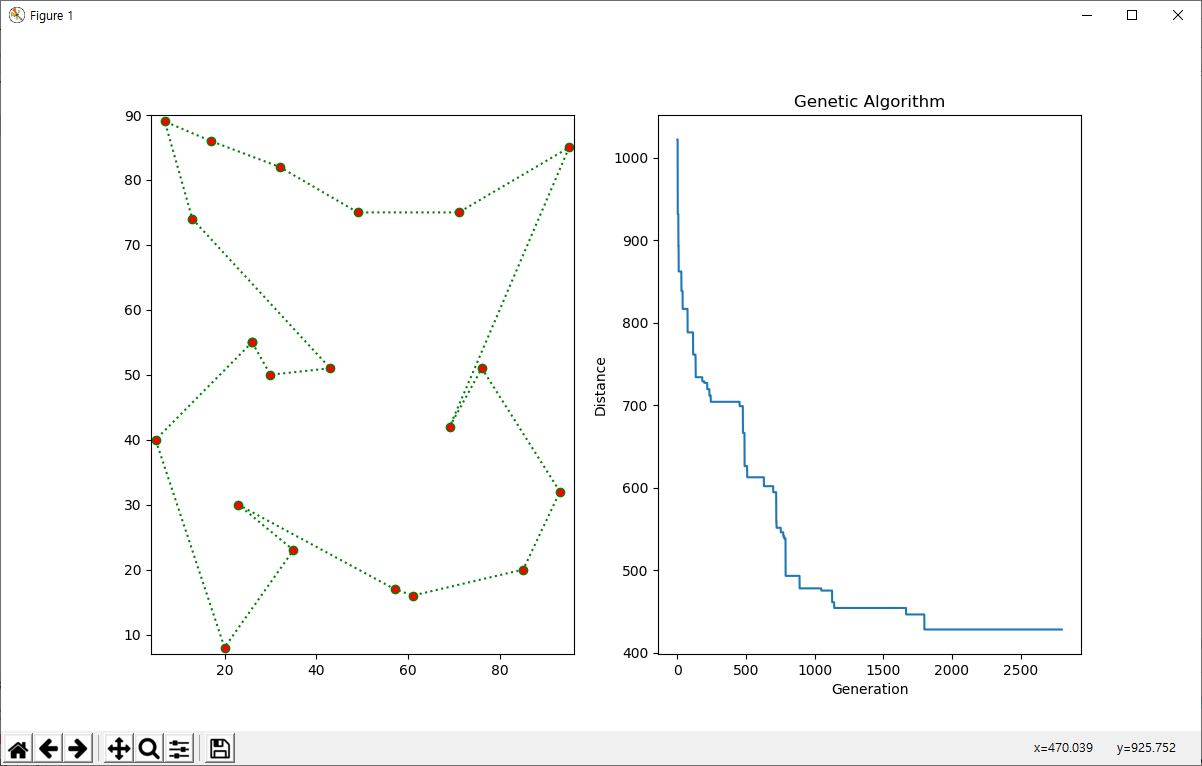
여러 데이터를 바탕으로 직접 종료조건을 정해서 실행해본 결과 똑 같은 적합도에서 1000세대 길게는 3000세대 까지 머무르는 것을 확인할 수 있었다. 따라서 마지막으로 종료 조건을 ‘적합도가 가장 낮은 염색체의 값이 1000세대 동안 바뀌지 않을 때’ 로 설정했다.

1. **실험데이터 산출 및 검증**
   1. **인위적 데이터**



먼저 20개의 도시를 인위적으로 city배열에 넣고 알고리즘을 실행해 보았다. TSP문제의 특성상 최적의 경로를 구하는데 N!의 시간 복잡도가 걸리기 때문에 랜덤으로 설정한 데이터로는 실제의 답과 비교할 수가 없었다. 그래서 도시들을 사각형 모양으로 만들고 알고리즘을 실행한 결과 답을 찾을 수 있었다.

* 1. **랜덤 데이터**



또한 X좌표와 Y좌표를 랜덤으로 설정한 20개의 도시를 데이터로 실행한 결과 다음과 같은 그래프를 얻었다. 이 그래프가 최적의 경로인지는 알 수 없지만, 적합도(경로의 총 거리)를 대략 1000에서 400까지 줄일 수 있었다.

1. **완료 느낀점**
2. **참고문헌**

<https://untitledtblog.tistory.com/110>

<https://twinw.tistory.com/1>

<https://ko.wikipedia.org/wiki/TSP>