14주차 3차시 특수한 알고리즘

【학습목표】

1. 확률적 알고리즘을 설명할 수 있다.
2. 유전적 알고리즘을 설명할 수 있다.

학습내용1 : 확률적 알고리즘

1. 기본 개념

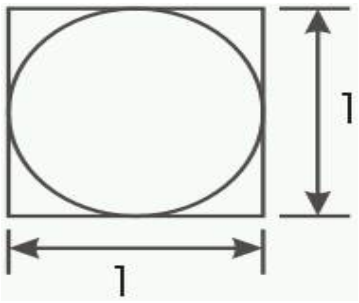
알고리즘의 근본적인 부분에서 확률이 관계하고 있는 것을 확률적 알고리즘(Probabilistic Algorithm)이라 한다. 그러나 주어진 문제를 해결하는 알고리즘은 일반적으로 결정론적 알고리즘으로 기술한다. 결정론적 알고리즘(deterministic Algorithm)은 같은 입력 자료에 대해서 항상 같은 결과를 출력하고 처리하기 때문이다. 그러나 확률적 알고리즘은 항상 같은 결과를 출력하지 않는다. 경우에 따라서는 틀린 결과를 출력 하거나 최적해가 아닌 근사해를 출력하기도 한다. 만약 결정론적 알고리즘에서 같은 입력 자료에 대해 처리할 때 마다 다른 결과를 출력한다면 이 알고리즘으로는 주어진 문제를 처리할 수 없게 된다. 확률적 알고리즘은 원하는 결과를 얻지 못할 확률이 매우 낮다는 조건하에 주어진 문제를 해결하는데 활용 된다. 대부분의 경우는 정확한 답을 반환 하지만 어떤 일정 확률로 틀린 답을 내는 계산법이 있다면 이것은 항상 바른 답을 반환 한다는 알고리즘의 정의로부터 벗어나게 된다. 하지만 이와 같은 것들도 알고리즘의 일종으로 인정하자고 하는 것이 확률적 알고리즘 이다.
일반적으로 확률적 알고리즘을 기술하자면 어떤 확률적 분포를 가진 난수를 필요로 하게 되며 컴퓨터 알고리즘을 기술할 때 난수는 어떤 분포를 가진 의사(Pseudo) 난수를 발생 하는 함수를 사용하게 된다. 예를 들어, P가 어떤 은행에 전송하는 온라인 계좌이체 명령을 암호화해서 전송할 때, 악의적 사용자가 P의 명령을 가로채서 재전송하는 공격을 한다고 가정해 보자 이와 같은 재전송 공격에서 안전하게 전송하려면 P와 해당은행이 프로토콜을 시행할 때 임의의 난수로 매번 다른 과정의 난수를 사용하여 전송해야 악의적인 공격을 피할 수 있을 것 이다. 그러므로 암호학 분야에서 확률적 알고리즘은 매우 중요하게 사용된다.
확률 알고리즘에는 여러 가지가 종류가 있는데 크게 나누어 몬테칼로(Monte Carlo) 알고리즘과 라스베가스(Las Vegas) 알고리즘으로 구분한다.

2. 몬테카를로(Monte Carlo) 알고리즘

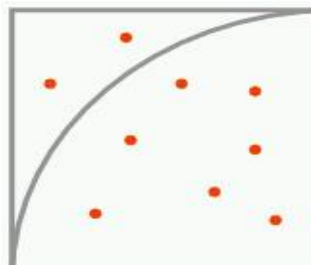
몬테카를로 알고리즘은, 물리적, 수학적 시스템의 행동을 시뮬레이션하기 위한 계산 알고리즘이다. 다른 알고리즘과는 달리 통계학적이고, 일반적으로 무작위의 숫자를 사용한 비결정적인 방법이다. 수학자이자 물리학자인 울람스타니스와프 울람이 모나코의 유명한 도박의 도시 몬테카를로의 이름을 본따 명명하였다. 1930년 엔리코 페르미가 중성자의 특성을 연구하기 위해 이 방법을 사용한 것으로 유명하며 맨해튼 계획의 시뮬레이션이나 수소폭탄의 개발에서도 핵심적인 역할을 담당하였다.
몬테카를로 알고리즘은 주어진 문제를 처리할 때 근사해를 출력하거나 간혹 틀린해를 출력하기도 하며 일정 이상의 정확한 해를 출력하기도 한다. 따라서 몬테카를로 알고리즘은 근사해로 만족할 수 있거나 간혹 틀린 해도 허용할 수 있어야 사용할 수 있다. 아직까지 결정론적인 알고리즘을 찾지 못한 경우 몬테카를로 알고리즘이 효과적이다.
몬테카를로 알고리즘은 적분 등으로 계산하기 어려운 특정한 면적을 구할 때에도 사용되며, 프로젝트 관리에도 사용된다.

예를 들어 몬테카를로 알고리즘을 이용한 파이 계산법을 보면 다음과 같다. 아래 [그림]과 같이 정사각형 안에 꼭지점을 중심으로 사분원을 그린다. 이때 정사각형의 전체 넓이를 1이라고 하면 원의 넓이는 π/4가 된다. 컴퓨터로 난수를 무작위로 정사각형 내부에 점을 찍는다. 그리고 정사각형의 꼭지점과의 거리를 계산하여 점이 사분원의 내부에 있는지 외부에 있는지를 판단한다. 예를 들어 전체 100만개의 점을 찍었다고 할 때 이중 n개가 사분원의 내부에 있었다면 두 숫자의 비율 즉 n/100만의 값은 넓이의 비인 π/4에 근접한다고 예측할 수 있다. 이 값들은 더 많은 점을 찍어 실험할수록 정밀하게 되며 이와 같이 많은 수의 실험을 바탕으로 통계 자료를 얻어 그 자료로부터 역산하여 어떤 특정한 수치나 확률분포를 구하는 방법을 말한다.

가로 1, 세로 1의 영역을 생각해 본다. 우선 이러한 사각형과 그 안에 들어가는 지름 1의 원을 고려해보면



임의의 난수(Random Number)를 발생시키되, 사각형 안의 영역에서 발생 시키게 된다.



이 때 발생시킨 난수가 원의 방정식 내부에 들어가면 결과에 1(참)을 더한다.

$X^2 + Y^2 <= 1$ 이라면 1 추가

그 다음에 전체 테스트의 수 n으로 나누어 나온 값이 바로 원의 넓이(정확하게 원의 넓이의 확률)가 되게 된다.

즉, '임의로 발생한 숫자가 원의 내부나 경계에 존재한다면 참이다.' 라는 조건이
전체 테스트 횟수 대비 얼마만큼이 되는가 하는 것이 우리가 구하고자 하는 영역의 면적이 되는 것임을 알 수 있다. 물론 테스트의 수 n이 많으면 많을수록 보다 정확한 값을 얻게 된다.

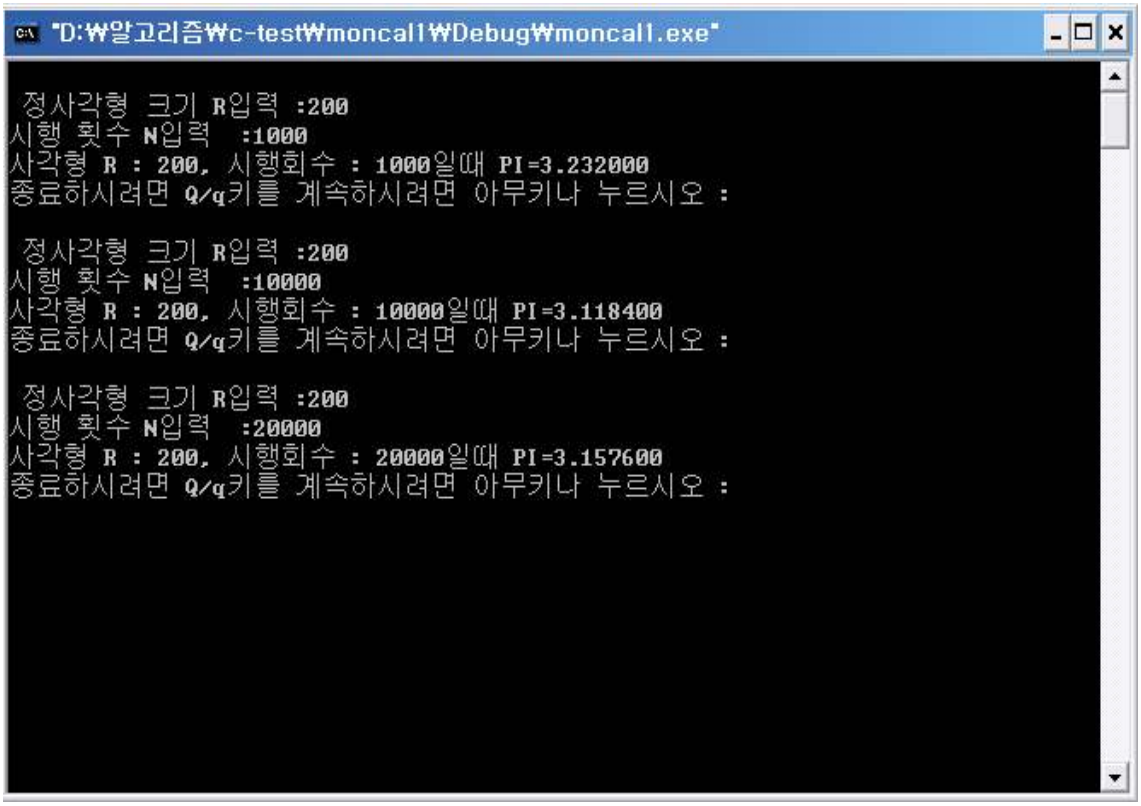
- 몬테카를로 알고리즘을 이용한 파이 계산법

```
#include "stdio.h"
#include "time.h"
#include "stdlib.h"

int main(void)
{
 int c1, c2, x, y;
 double PI;
 int p;
 int n;
 int i;
 int quit;

// 난수 초기화
srand((unsigned int)time(NULL));
while(1)
{
// 자료 입력
   n=0;
   printf("₩n 정사각형 크기 입력 :");
   scanf("%d", &c1);
   c2= c1 * c1;
   printf("시행 횟수 입력  :");
   scanf("%d", &p);
// 계산 과정
// 난수 발생
for (i=0; i< p; i++)
{
x=rand()%(c1+1);
y=rand()%(c1+1);
// 좌표가 사분원 안에 존재하는지 검사
if(x*x+y*y <=c2)
n++;
}
//출력( PI값을 계산)
PI=4. *n/p;
printf("사각형 R : %d, 시행회수 : %d일때 PI=%lf₩n", c1, p, PI);
printf("종료하려면 Q/q키를 계속하려면 아무키나 누르시오 :");
fflush(stdin);
if((quit = getchar())==(int)'Q' || quit == (int)'q')
break;
}
return 0;
}
```

* 몬테카를로 알고리즘을 이용한 파이 계산



* 몬테카를로 알고리즘을 이용한 파이 계산 실행 결과
- 위의 결과에서 시행회수를 크게 하면 3.14에 에 가까운 값이 출력됨을 볼 수 있다.

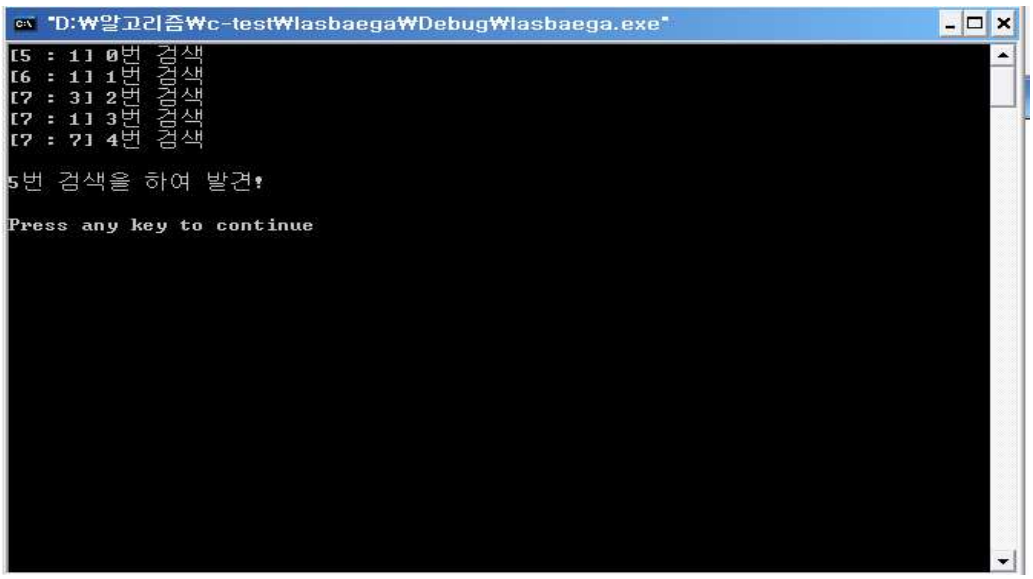
3. 라스베가스 알고리즘

라스베가스 알고리즘은 언제나 정확한 결과를 출력하지만 난수 발생기에 의해 확률적으로 이루어지기 때문에 실행시간이 오래 걸릴 확률이 조금이나마 존재한다. 어떤 경우에는 오류가 발생할 확률이 어느 정도 존재하더라도 알고리즘이 빠르게 실행되는 것을 원할 경우가 있는데 이를 몬테카를로 알고리즘이라 하며, 라스베가스 알고리즘은 올바른 해를 보다 빨리 출력하기 위한 확률적 선택을 하게 한다. 몬테칼로 알고리즘은 근사해를 얻게 하거나 낮은 확률로 오답을 낼 수 있지만 라스베거스 알고리즘은 항상 올바른 해를 출력하는 특징이 있다. 결정론적 알고리즘으로 빨리 수행되는 입력에 대해서 라스베가스 알고리즘은 수행시간이 길어질 수 있다. 하지만 결정론적 알고리즘으로 해결할 때 장시간이 걸리는 경우에 라스베가스 알고리즘은 빠르게 해를 구할 수 있다.

* 라스베가스 알고리즘

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
void main()
{
 int n[10] = {3,2,1,5,6,7,7,7,7,7}; //임의의 수
 int r1=1, r2=0;
 int i=0;
 int count=0; //검색하는 횟수
 srand((unsigned int)time(NULL)); //난수 초기화
 while(n[r1] != n[r2])
 {
    r1 = rand()%10; //10 보다 적은 난수 생성
    for( i =0; i<1 ; i++){
    r2= rand()%10;
    if(r1 == r2)
     i--;
  }
   printf("[%d : %d] %d번 검색 ₩n", n[r1], n[r2], count);
   count++;
 }
 printf("₩n");
 printf("%d번 검색을 하여 발견!₩n₩n", count);
}
```

* 라스베가스 알고리즘 실행화면



라스베가스 알고리즘은 n개의 숫자들 가운데서 임의로 두 개를 선택 한다. 그리고 이 두 개의 선택된 숫자들이 같은지를 확인한다. 만약 두 개의 선택된 숫자들이 같으면 바로 그 숫자가 정답이다. 만약 두 개의 숫자들이 다르다면 다시 앞에서 설명한 과정을 계속해서 반복한다. 이 알고리즘은 반드시 정답을 도출하지만 언제 연산이 끝날지는 정확히 보장할 수가 없다. 다만 평균적으로 O(log n)정도의 시간 내에 정답을 찾을 수 있다는 것을 증명할 수 있다.

학습내용2 : 유전적 알고리즘

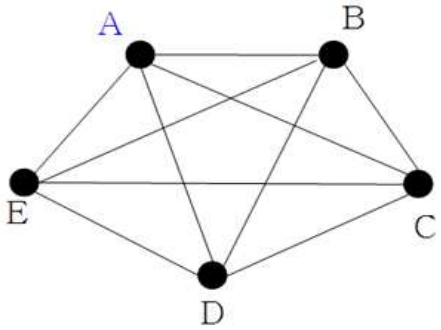
1. 유전자 알고리즘 (Genetic Algorithm, GA)

- 유전자 알고리즘 (Genetic Algorithm, GA)은 다윈의 진화론으로부터 창안된 해 탐색 알고리즘이다.
- 즉, '적자생존'의 개념을 최적화 문제를 해결하는데 적용한 것이다.
- 유전자 알고리즘은 다음과 같은 형태를 가진다.

GeneticAlgorithm
1. 초기 후보해 집합G_0을 생성한다.
2. G_0의 각 후보해를 평가한다.
3. t ←0
4. repeat
5. G_t로부터 G_{t+1}을 생성한다.
6. G_{t+1}의 각 후보해를 평가한다.
7. t ←t + 1
8. until (종료 조건이 만족될 때까지)
9. return G_t의 후보해 중에서 가장 우수한해

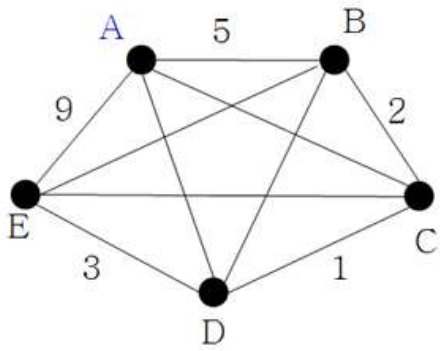
- 유전자 알고리즘은 여러 개의 해를 임의로 생성하여 이들을 초기 세대 (generation) G_0로 놓고, repeat-루프에서 현재 세대의 해로부터 다음 세대의 해를 생성해가며, 루프가 끝났을 때의 마지막 세대에서 가장 우수한 해를 리턴 한다.
- 이 해들은 repeat-루프의 반복적인 수행을 통해서 최적해 또는 최적해에 근접한 해가 될 수 있으므로 후보해 (candidate solution)라고 한다.

• 후보해에 대한이해: 5개의 도시 (A, B, C, D, E)에 대한 여행자 문제 (Traveling Salesman Problem)를 예로 들어보자.
• 단, 시작 도시는 A이다. 여행자 문제의 조건은 시작 도시에서 출발하여 모든 다른 도시를 1번씩만 방문하고 시작 도시로 돌아와야 하므로, ABCDEA, ACDEBA, AECDBA 등이 후보해 이다.



• 이 문제의 후보해의 수는 시작도시 를제외한 5개의 도시를 일렬로 나열하는 방법의 수와 같으므로 5! = 120이다.
• 만일 n개의 도시가 있다면, 후보해의 수는(n-1)!이다.
• 후보해의 평가: 후보해를 평가 한다는 것은 후보해의 값을 계산 하는 것 이다.

• 문제의 후보해의 값은 도시간의 거리가 입력으로 주어지므로, 다음과 같이 계산한다.
후보해 ABCDEA의 값 =
(A와 B 사이의 거리)
+ (B와 C 사이의 거리)
+ (C와 D 사이의 거리)
+ (D와 E 사이의 거리)
+ (E와 A 사이의 거리)
= 5 + 2 + 1 + 3 + 9
= 20



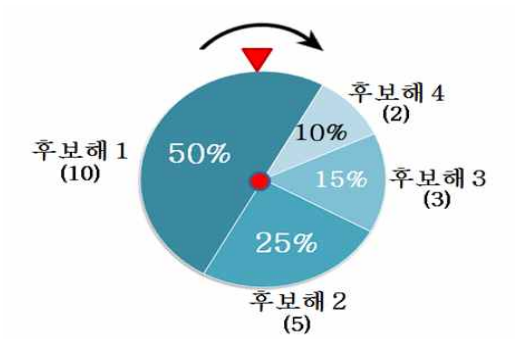
• 후보해의 값을 후보해의 적합도(fitness value)라고 한다.
• 후보해 중에서 최적해의 값에 근접한 적합도를 가진 후보해를 '우수한' 해라고 부른다.
• GeneticAlgorithm에서 가장 핵심적인 부분은 현재 세대의 후보해에 대해서 다음과 같은 3개의 연산을 통해서 다음 세대의 후보해를 생성 하는 것이다.
- 선택 (selection) 연산
- 교차 (crossover) 연산
- 돌연변이 (mutation) 연산

2. 선택 연산

• 선택 연산은 현재 세대의 후보해 중에서 우수한 후보해를 선택하는 연산이다.
• 현재 세대에 n개의 후보해가 있으면, 이들 중에서 우수한 후보해는 중복되어 선택될 수 있고, 적합도가 상대적으로 낮은 후보해 선택 되지 않을 수도 있다.
• 이렇게 선택된 후보해의 수는n개로 유지된다. 이러한 선택은 '적자생존' 개념을 모방한 것이다.

• 선택 연산을 가장 간단히 구현하는 방법은 룰렛 휠 (roulette wheel) 방법이다.
→ 각 후보해의 적합도에 비례하여 원반의 면적을 할당하고, 원반을 회전시켜서
→ 원반이 멈추었을 때 핀이 가리키는 후보해를 선택한다.
→ 면적이 넓은 후보해가 선택될 확률이 높다.

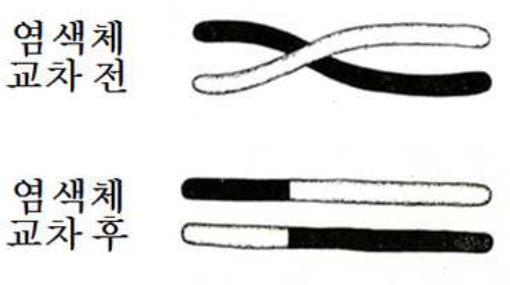
• 후보해 1의 적합도: 10
• 후보해 2의 적합도: 5
• 후보해 3의 적합도: 3
• 후보해 4의 적합도: 2



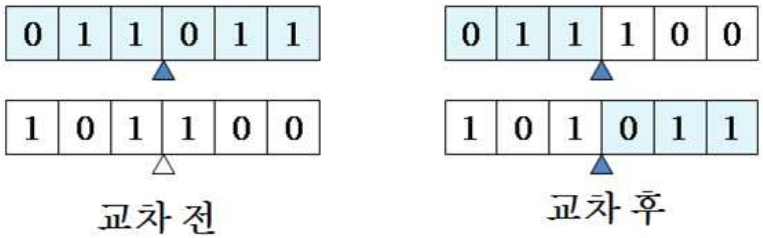
• 각 후보해의 원반 면적은 (후보해의 적합도/ 모든 후보해의 적합도의합)에 비례한다.
• 앞의 예제에서 모든 적합도의 합이 10 + 5 + 3 + 2 = 20이므로,
→ 후보해 1의 면적은 10/20 = 50%
→ 후보해 2의 면적은 5/20 = 25%
→ 후보해 3의 면적은 3/20 = 15%
→ 후보해 4의 면적은 2/20 = 10%
• 현재 4개의 후보해가 있으므로, 4번 원반을 돌리고 회전이 멈추었을 때 핀이 가리키는 후보해를 각각 선택한다.

3. 교차 연산

• 교차 연산은 선택 연산을 수행한 후의 후보해 사이에 수행 되는데, 이는 염색체가 교차하는 것을 모방한 것이다.
• 예를 들어, 2개의 후보해가 각각2진수로 아래와 같이 표현된다면, 교차점 이후의 부분을 서로 교환하여 교차 연산이 수행되며, 그 결과 각각 새로운 후보해가 만들어진다.
• 이와 같은 교차 연산을 1-점 (point) 교차 연산이 라고 한다.



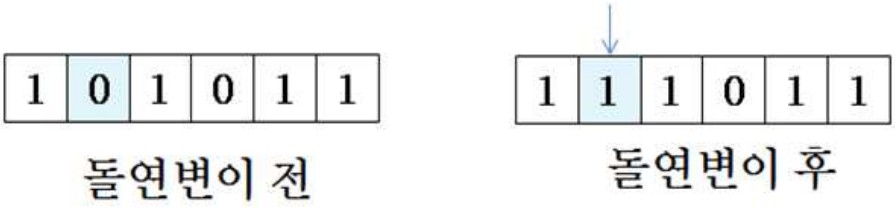
※ 염색체(주어진 당면 문제의 탐색 공간에서 하나의 가능한 해법을 함축하고 있다)



• 후보해가 길게 표현되면, 여러 개의 교차점을 임의로 정하여 교차 연산을 할 수도 있다.
• 교차 연산의 목적: 선택 연산을 통해서 얻은 우수한 후보해보다 우수한 후보해가 생성하는 것
• 문제에 따라서 교차 연산을 수행할 후보해의 수를 조절하는데, 이를 교차율 (crossover rate)이라고 한다. 일반적으로 교차율은 0.2~1.0 범위에서 정한다.

4. 돌연변이 연산

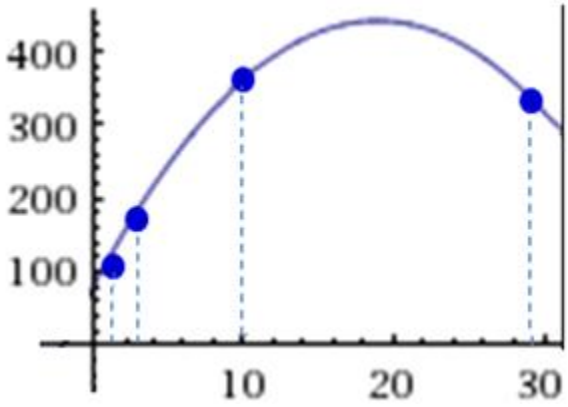
• 교차 연산이 수행된 후에 돌연변이 연산을 수행한다.
• 돌연변이 연산은 아주 작은 확률로 후보해의 일부분을 임의로 변형시키는 것이다.
• 이 확률을 돌연변이율 (Mutation Rate)이라고 하며, 일반적으로 (1/PopSize)~(1/Length)의 범위에서 사용된다.
→ 여기서 PopSize란 모집단 크기(Population Size)로서 한 세대의 후보해의 수이고,
→ Length란 후보해를 이진 표현으로 했을 경우의 bit 수이다.
• 다음의 예는 두 번째 bit가 0에서 1로 돌연변이된 것을 보여주고 있다.



• 돌연변이가 수행된 후에 후보해의 적합도가 오히려 나빠질 수도 있다.
• 돌연변이 연산의 목적은 다음 세대에 돌연변이가 이루어진 후보해와 다른 후보해를 교차 연산함으로써 이후 세대에서 매우 우수한 후보해를 생성하기 위한 것이다.
• GeneticAlgorithm의 종료 조건은 일정하지 않다. 왜냐하면 유전자 알고리즘이 항상 최적해를 찾는다는 보장이 없기 때문이다.
• 따라서 일반적으로 알고리즘을 수행시키면서 더 이상 우수한 해가 출현하지 않으면 알고리즘을 종료시킨다.

• 다음의 2차 함수에 대해 유전자 알고리즘으로 $0 \leq x \leq 31$ 구간에서 최댓값을 찾아보자.
$f(x) = -x^2 + 38x + 80$

• 먼저 한 세대의 후보해 수를 4로 정하고, 0~31에서 랜덤하게 4개의 후보해인 1, 29, 3, 10을 선택하였다고 가정하자.
• 이들이 초기 세대를 구성하는 후보해들이다.
• 각 후보해의 적합도:
→ $f(1) = -(1)^2 + 38(1) + 80 = 117$,
→ $f(29) = 341$,
→ $f(3) = 185$,
→ $f(10) = 360$



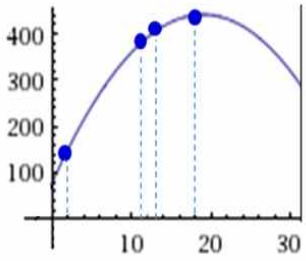
후보해	2진 표현	x	적합도 f(x)	원반 면적 (%)
1	0 0 0 0 1	1	117	12
2	1 1 1 0 1	29	341	34
3	0 0 0 1 1	3	185	18
4	0 1 0 1 0	10	360	36
계			1,003	100
평균			250.75	

- 위의 표는 각 후보해의 2진 표현, 적합도, 룰렛 휠 선택을 위한 원반 면적을 보이고 있다.
- 또한 초기 세대의 평균 적합도는 250.75이다.

• 선택 연산: 룰렛 휠 선택 방법을 이용하여, 후보해 4는 2번 선택되었고, 후보해 2와 3은 각각 1번 선택되었으며, 후보해 1은 선택 안되었다고 가정하자.

• 교차 연산: 후보해 4가 2개이므로, 후보해 2와 4를 짝짓고, 후보해 3과 4를 짝지어 아래와 같이 교차 연산을 수행한다. 단, 1점-교차 연산을 위해 아래와 같이 임의의 교차점이 선택 되었다고 가정하자.

• 돌연변이 연산: 교차 연산 후에 후보해 1의 왼쪽에서 두 번째 bit가 돌연변이가 되어서 '1'에서 '0'으로 바뀌었다고 가정하자. 다른 후보해는 교차 연산 후와 동일하다.



후보해	2진 표현	x	적합도 f(x)	원반 면적 (%)
1	1 0 0 1 0	18	440	32
2	0 1 1 0 1	13	405	29
3	0 0 0 1 0	2	152	11
4	0 1 0 1 1	11	377	27
계			1,374	100
평균			343.5	

• 두 번째 세대의 평균 적합도가 343.5로 많이 향상되었다.
• repeat-루프를 더 수행하여 후보해의 적합도가 변하지 않으면, 알고리즘을 종료하고, 후보해 중에서 가장 적합도가 높은 후보해를 리턴 한다.

5. 외판원 문제 (Traveling Salesman Problem)

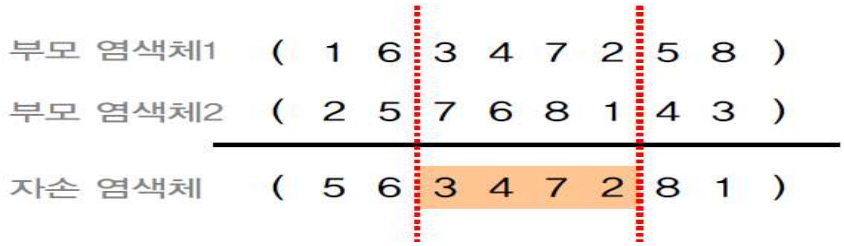
최소한의 비용으로 모든 도시를 한 번씩만 방문하고 처음 도시로 돌아오는 방법을 찾는 문제

⦿ 염색체 인코딩
▪ 순열 인코딩

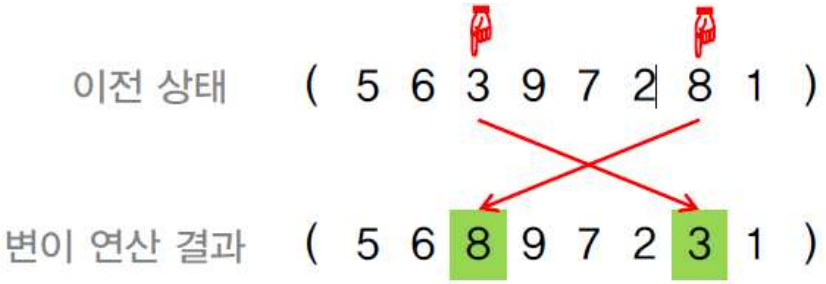
1. 서울	2. 도쿄	3. 싱가포르	4. 베이징
5. 홍콩	6. 뉴델리	7. 삿포르	8. 시드니

(1 2 3 4 5 6 7 8)
⋮
(2 5 7 6 8 1 4 3)

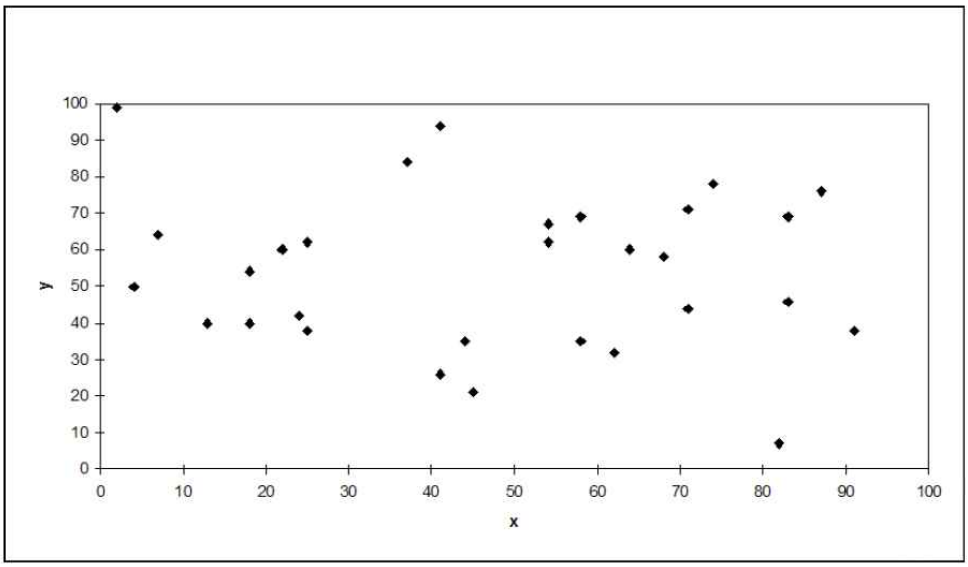
⦿ 교차 연산



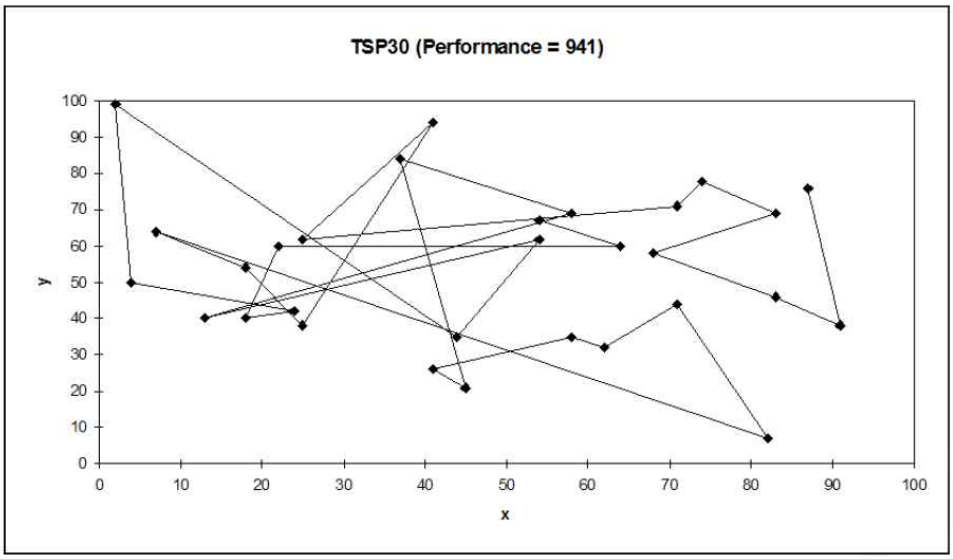
⦿ 변이 연산



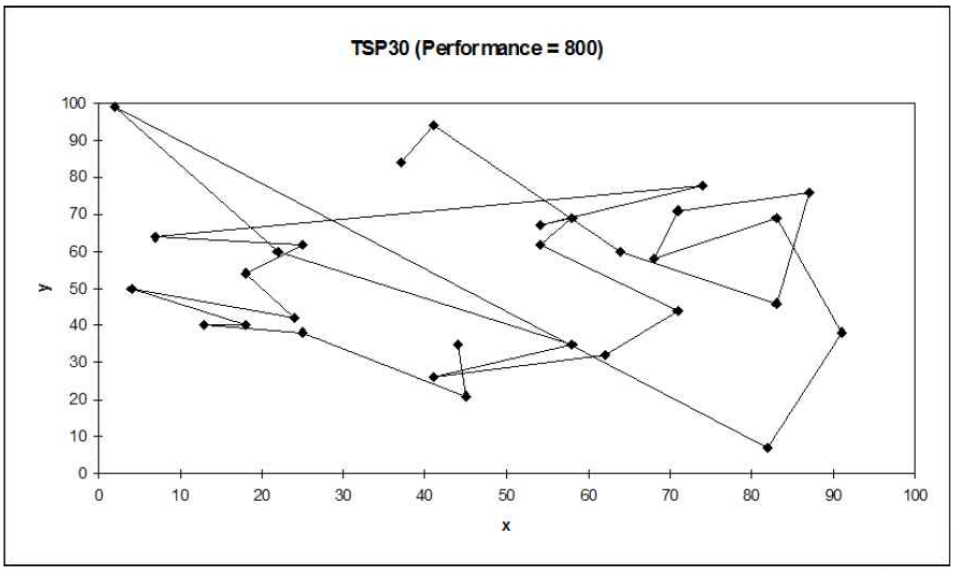
⦿ 30개 도시 (by Wendy Williams, “Genetic Algorithms: A Tutorial”)



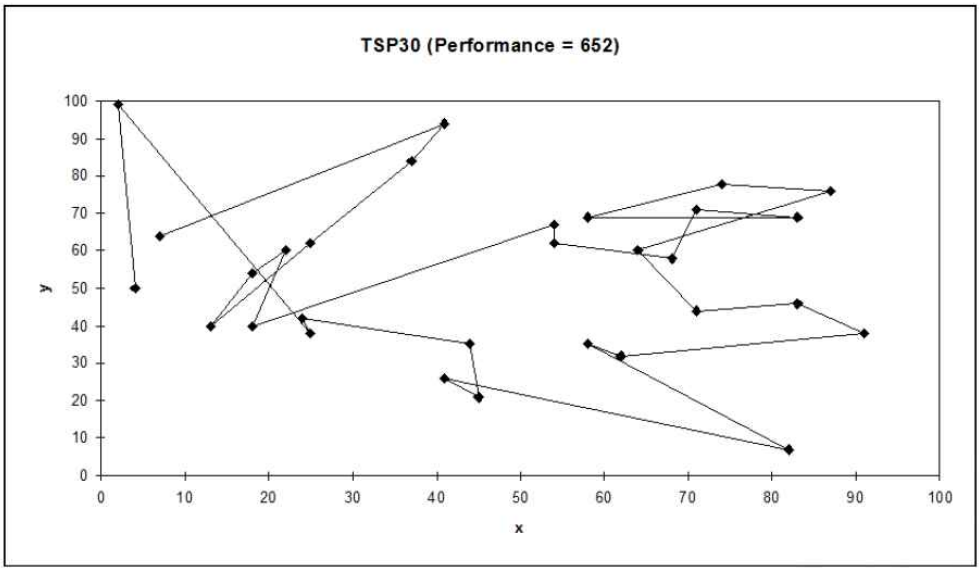
i번째 세대의 해 → 거리 = 941



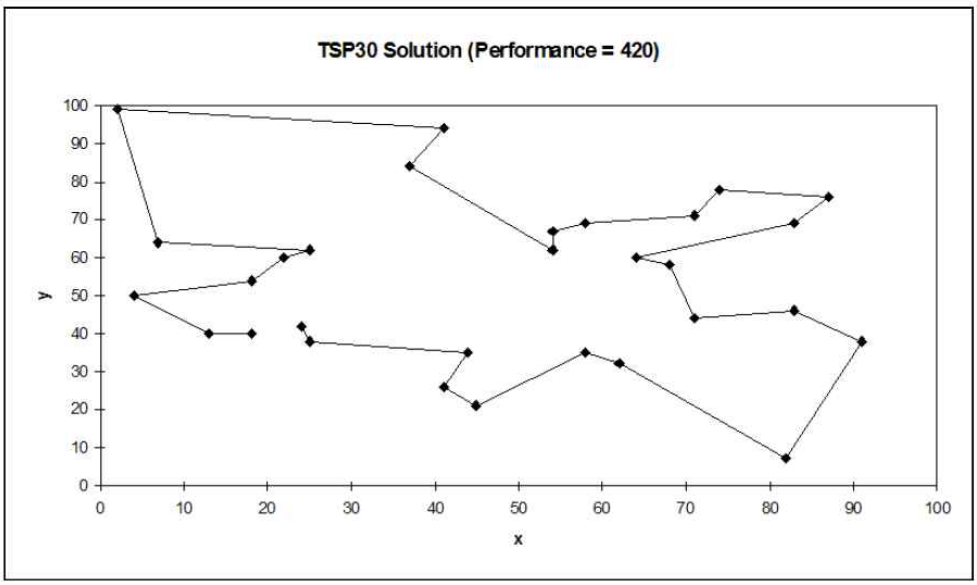
j번째 세대의 해 → 거리 = 800



k번째 세대의 해 → 거리 = 652



Best solution → 거리 = 420



6. 응용

• 유전자 알고리즘은 문제의 최적해를 알 수 없고, 기존의 어느 알고리즘으로도 해결하기 어려운 경우에, 최적해에 가까운 해를 찾는데 매우 적절한 알고리즘이다.
• 유전자 알고리즘이 최적해를 반드시 찾는다는 보장은 없으나 대부분의 경우 매우 우수한 해를 찾는다.
• 통 채우기
• 작업 스케줄링
• 차량 경로
• 배낭 문제
• 로봇 공학
• 기계 학습 (Machine Learning)
• 신호 처리 (Signal Processing)
• 반도체 설계
• 항공기 디자인
• 통신 네트워크
• 패턴 인식

• 그 외에도 경제, 경영, 환경, 의학, 음악, 군사 등과 같은 다양한 분야에서 최적화 문제를 해결 하는데 활용된다.

【학습정리】

1. 확률 알고리즘
확률 알고리즘에는 여러 가지가 종류가 있는데 크게 나누어 몬테칼로(Monte Carlo) 알고리즘과 라스베가스(Las Vegas) 알고리즘으로 구분한다.

* 몬테카를로(Monte Carlo) 알고리즘
몬테카를로 알고리즘은, 물리적, 수학적 시스템의 행동을 시뮬레이션하기 위한 계산 알고리즘이다. 다른 알고리즘과는 달리 통계학적이고, 일반적으로 무작위의 숫자를 사용한 비결정적인 방법이다. 수학자이자 물리학자인 울람스타니스와프 울람이 모나코의 유명한 도박의 도시 몬테카를로의 이름을 본따 명명하였다. 1930년 엔리코 페르미가 중성자의 특성을 연구하기 위해 이 방법을 사용한 것으로 유명하며 맨해튼 계획의 시뮬레이션이나 수소폭탄의 개발에서도 핵심적인 역할을 담당하였다.

* 라스베가스 알고리즘
라스베가스 알고리즘은 언제나 정확한 결과를 출력하지만 난수 발생기에 의해 확률적으로 이루어지기 때문에 실행시간이 오래 걸릴 확률이 조금이나마 존재한다. 어떤 경우에는 오류가 발생할 확률이 어느 정도 존재하더라도 알고리즘이 빠르게 실행되는 것을 원할 경우가 있는데 이를 몬테카를로 알고리즘이라 하며, 라스베가스 알고리즘은 올바른 해를 보다 빨리 출력하기 위한 확률적 선택을 하게 한다. 몬테칼로 알고리즘은 근사해를 얻게 하거나 낮은 확률로 오답을 낼 수 있지만 라스베거스 알고리즘은 항상 올바른 해를 출력하는 특징이 있다.

2. 유전자 알고리즘 (Genetic Algorithm, GA)

• 유전자 알고리즘 (Genetic Algorithm, GA)은 다윈의 진화론으로부터 창안된 해 탐색 알고리즘이다.

• 즉, '적자생존'의 개념을 최적화 문제를 해결하는데 적용한 것이다.

• 유전자 알고리즘은 여러 개의 해를 임의로 생성하여 이들을 초기 세대 (generation) G_0로 놓고, repeat-루프에서 현재 세대의 해로부터 다음 세대의 해를 생성해가며, 루프가 끝났을 때의 마지막 세대에서 가장 우수한 해를 리턴 한다.

• 이 해들은 repeat-루프의 반복적인 수행을 통해서 최적해 또는 최적해에 근접한 해가 될 수 있으므로 후보해 (candidate solution)라고 한다.

• 후보해의 값을 후보해의 적합도(fitness value)라고 한다.

• 후보해 중에서 최적해의 값에 근접한 적합도를 가진 후보해를 '우수한' 해라고 부른다.

• GeneticAlgorithm에서 가장 핵심적인 부분은 현재 세대의 후보해에 대해서 다음과 같은 3개의 연산을 통해서 다음 세대의 후보해를 생성 하는 것이다.

① 선택 (selection) 연산

② 교차 (crossover) 연산

③ 돌연변이 (mutation) 연산