Ch07_진화 연산

이 장에서 다룰 내용

- 01_진화는 지능적인가
- 02_자연의 진화 흉내 내기
- 03_유전 알고리즘
- 04_유전 알고리즘의 작동 원리
- 05_사례 연구: 유전 알고리즘으로 푸는 정비 스케줄링
- 06_진화 전략
- 07_유전 프로로그래밍
- 08_요약

01_진화는 지능적인가

❖ 지능과 진화

- 시스템의 지능
  - 끊임없이 변하는 환경에 시스템이 적응하는 능력이라 정의 할 수 있다.
  - 시스템의 외양이나 형태는 지능과 무관하다(앨린 튜링, 1950).

- 인간의 지능
  - 일상에서 인간의 지능적인 행동을 쉽게 관찰할 수 있다.
  - 인간은 진화의 산물이다.

- 진화연산
  - 진화 연산은 컴퓨터에서 진화를 흉내 내는 것이다.
  - 연살 결과는 대게 단순한 규칙에 바탕을 둔 최적화 알고리즘이다.

- 기계 학습의 진화
  - 기계 학습에 대한 진화론적 방법론은 자연 선택과 유전학 계산 모델에 근거한다.
  - 유전 알고리즘, 진화 전략, 유전 프로그래밍을 포함한다.
  - 선택, 변이, 재생산을 이용하여 진화를 흉내 낸다.

02_자연의 진화 흉내 내기

❖ 컴퓨터의 진화

- 자연의 진화
  - 찰스 다윈의 진화론에 의거한다.
  - 신 다윈주의는 재생산, 진화 경쟁, 선택 과정에 바탕을 둔다.
  - '재생산' 능력은 생명의 본질적인 특성이다.
  - '변이' 능력은 어떤 생명체든 끊임없이 변화하는 환경 속에서 자기 자신을 재생산할 수 있게 한다.
  - '경쟁'과 '선택'은 여러 생물 종의 개체군 확장을 제한하는 자연계에서 일상적으로 일어나는 일이다.

- 진화 적합성(evolutionary fitness)
  - 진화는 특정 환경에서 집단이 생존하고 재생산하는 능력을 유지·향상시키는 과정이다.
  - 생존하고 재생산하는 능력을 진화 적합성이라 한다.
  - 적합성을 직접 측정할 수는 없지만, 생태학이나 유기체에 관한 기능 형태학을 바탕으로 추정할 수 있다(Hoffman).

- 적응형 위상(adaptive topoloty)
  - 적합도를 나타내고자 적응형 위상 개념을 사용한다.
  - 주어진 환경을 적합도 지형으로 나타낼 수 있다.

02_자연의 진화 흉내 내기

- 적응형 위상(adaptive topoloty)
  - 적응형 위상은 연속 함수로 환경, 즉 자연의 위상이 정적이지 않다는 사실을 흉내 낸다.
  - 위상의 형태는 시간에 따라 변하고, 모든 종은 끊임없이 선택 받는다.
  - 진화의 목표는 적합도가 증가하는 개체 집단을 생성하는 것이다.

- 진화의 예: 빠른 토기
  - 적합도가 증가하는 개체 집단
  - 마이클비치의 설명 예
  - 다른 토끼보다 '발이 빠른 토끼'는 여우를 피하여 살아남아 번식할 확률이 높으므로 적합도가 높다. 물론 느린 토끼도 몇 마리 살아남을 것이다. 그 결과, 몇몇 느린 토끼는 빠른 토끼와 새끼를 낳고, 몇몇 빠른 토끼는 다른 빠른 토끼와 새끼를 낳고, 몇몇 느린 토끼는 다른 느린 토끼와 새끼를 낳는다. 즉 번식을 통해 토끼의 유전자가 섞인다. 두 부모의 적합도가 모두 높으면 적합도가 더 높은 자식을 낳을 가능성이 높다. 시간이 지나면서, 전체 토끼 집단은 여우와 우연히 마주치는 환경에 대처하기 위해 점점 더 빨라진다.
  - 환경조건은 변할 수 있다. 예를 들어, 뚱뚱하지만 영리한 토끼에 유리하게 변할 수 있다. 생존을 위해 최적화하려면 토끼 집단의 유전 구조도 이에 따라 변할 것이다. 마찬가지로 더 빠르고 영리하게 태어난 토끼는 더 빠르고 영리한 여우를 태어나게 한다. 자연의 진화과정은 연속적이며, 결코 끝나지 않는다.

03_유전 알고리즘

❖ 컴퓨터의 진화 과정

- 컴퓨터의 진화 방법
  - 개체 집단을 만들고, 그 집단의 적합도를 평가하며, 유전 연산자를 써서 새로운 집단을 만든다. 그리고 이 과정을 여러 번 반복함으로써 자연의 진화를 흉내 낸다.
  - 컴퓨터의 진화 과정 예: 유전 알고리즘

❖ 유전 알고리즘

- 유전 알고리즘의 탄생
  - 존 홀랜드가 유전 알고리즘 개념을 창안했다.
  - 홀랜드의 유전 알고리즘은 인공적인 '염색체(chromosome)'로 이루어진 한 해집단에서 다른 해집단으로 넘어가는 일련의 절차적인 단계로 나타낼 수 있다.

- 유전 알고리즘의 구성
  - 유전 알고리즘은 '자연' 선택과 유전학에서 영감을 얻은 교차(crossover)와 변이(mutation) 연산을 사용한다. 염색체는 '유전자(gene)'여러 개로 이루어진다.
  - 유전자는 [그림7-1]과 같이 0이나 1로 나타낸다.

1	0	1	1	0	1	0	0	0	0	0	1	0	1	0	1

[그림 7-1] 16비트 이진 문자열로 나타낸 인공적인 염색체

03_유전 알고리즘

- 유전 알고리즘의 구성
  - 유전 알고리즘은 무엇을 할지 가르쳐 주지 않아도 적응하고 학습하는 능력으로 두 가지 메커니즘(인코딩, 평가)이 유전 알고리즘을 풀려는 문제와 연결한다.
  - 인코딩: 염색체를 1과 0으로 된 문자열로 나타낸다.
  - 평가 함수: 풀려는 문제에 대한 염색체의 성능, 즉 적합도를 재는데 사용한다.

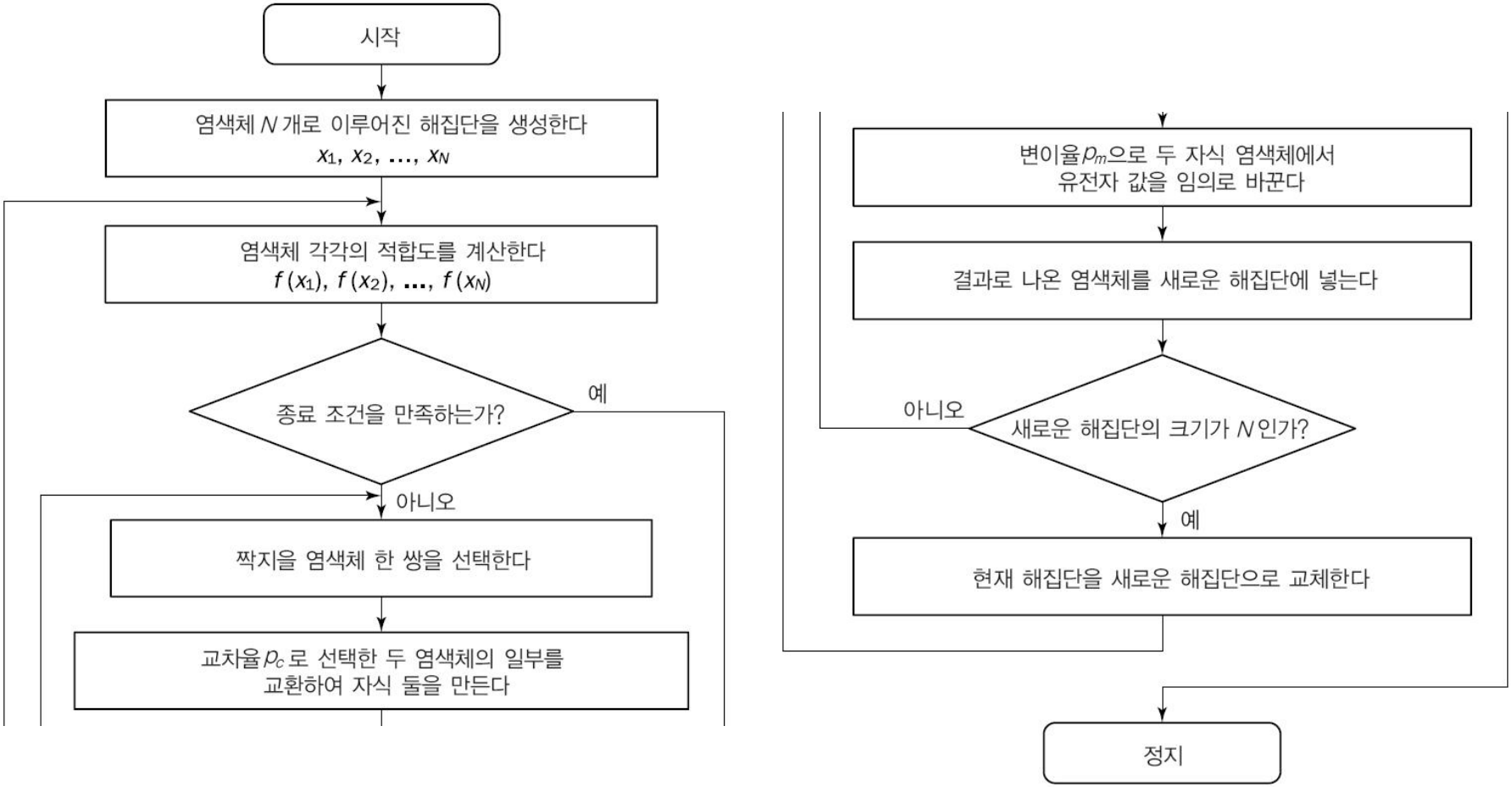
- 평가 함수
  - 유전 알고리즘은 재생산을 할 때 측정한 개별 염색체의 적합도 값을 사용한다.
  - 재생산이 일어나면 교차 연산자는 두 염색체의 일부를 교환하고, 변이 연산자는 염색체에서 임의로 선택한 몇몇 자리에 있는 유전자의 값을 바꾼다.
  - 그 결과, 재생산이 여러 번 연속해서 일어난 후에는 적합도가 낮은 염색체는 소멸되고, 살아남은 염색체가 점차 해집단을 지배한다.

❖ 유전 알고리즘

- 유전 알고리즘은 생물학적 진화에 바탕을 둔 통계적 탐색 알고리즘 집합.
  - 풀어야 할 문제가 명확하게 정의되고, 후보 해를 나타낸 이진 문자열이 주어지면, 기본적인 유전 알고리즘은 [그림 7-2]와 같이 나타낼 수 있다.

03_유전 알고리즘

- **샘플유전 알고리즘은 생물학적 진화에 바탕을 둔 통계적 탐색 알고리즘 집합.**
  - **기본적인 유전 알고리즘** : [그림 7-2]



[그림 7-2] 기본적인 유전 알고리즘

03_유전 알고리즘

❖ 유전 알고리즘의 주요 단계

- 총 10단계로 구성.
- 1단계
  - 문제 변수 영역을 고정된 길이의 염색체로 나타내고, 해집단 크기 N, 교차율(crossover probability) p_c, 변이율(mutation probability) p_m을 정한다.
- 2단계
  - 문제 영역에서 개별 염색체의 성능, 즉 적합도를 재는 적합도 함수를 정의한다.
  - 적합도 함수는 재생산 과정에서 짝지어지는 염색체를 선택하는 근거다.
- 3단계
  - 염색체 N개로 이루어진 초기 해집단을 임의로 생성한다.
  - 다음과 같이 나타낼 수 있다.

$$x_1, x_2, \ldots, x_N$$

- 4단계
  - 염색체 각각의 적합도를 계산한다.

$$f(x_1), f(x_2), \ldots, f(x_N)$$

03_유전 알고리즘

❖ 유전 알고리즘의 주요 단계

- 5단계
  - 현재 해집단에서 짝지을 염색체 한 쌍을 선택한다. 적합도에 따라 확률적으로 부모 염색체(parent chromosomes)를 선택한다.
  - 적합도가 높은 염색체는 적합도가 낮은 염색체보다 선택될 확률이 높다.
- 6단계
  - 유전 연산자인 교차와 변이를 적용하여 자식 염색체 한 쌍을 만든다.
- 7단계
  - 만들어진 자식 염색체를 새로운 해집단에 넣는다.
- 8단계
  - 새로운 해집단의 크기가 초기 해집단 크기인N이 될 때까지 5단계를 반복한다.
- 9단계
  - 초기(부모) 해집단을 새로운(자식) 해집단으로 교체한다.
- 10단계
  - 4단계로 가서 종료 조건을 만족할 때까지 이 과정을 반복한다.

03_유전 알고리즘

❖ 유전 알고리즘의 종료 조건

- 전형적인 종료 조건
  - 만족할 만한 해를 찾을 때까지 계속 해를 구한다.
  - 유전 알고리즘은 통계적 탐색 기법을 쓰기 때문에 더 좋은 염색체가 나타나기 전까지 몇 세대 동안은 해집단의 적합도가 정체될 수 있다.

- 적응적 종료 조건
  - 보통 전해진 세대수가 되면 유전 알고리즘을 종료하고 해집단에서 가장 좋은 염색체를 찾는다.
  - 만족스러운 해를 발견하지 못하면 유전 알고리즘을 다시 시작한다.

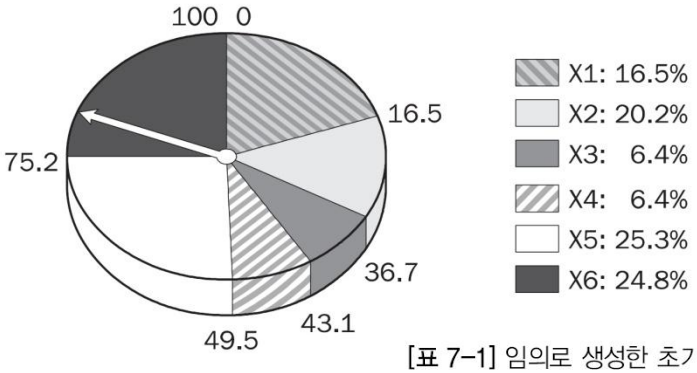
❖ 평균 적합도를 높이는 방법

- 룰렛 휠 선택
  - 해집단의 크기를 일정하게 유지하면서 평균 적합도를 높이는 방법이다.
  - 가장 흔히 쓰이는 염색체 선택 기법이다.

- 룰렛 휠 선택 동작
  - 염색체 각각은 원형 룰렛 휠 조각을 하나씩 할당 받는다.

03_유전 알고리즘

- 룰렛 휠 선택 동작
  - 휠에서 조각의 넓이는 염색체의 적합도 비율과 같다. : [그림 7-4], [표 7-1]
  - 짝을 지을 염색체를 선택하려면 [0, 100]에서 난수를 만들고 난수가 걸쳐있는 구간에 있는 염색체를 선택한다.
  - 이는 모든 염색체가 적합도에 비례하여 공간을 차지하고 있는 룰렛 휠을 돌리는 것과 같다. 룰렛 휠을 돌린 후 화살표가 멈춘 구획의 염색체가 선택된다.



[그림 7-4] 룰렛 휠 선택

[표 7-1] 임의로 생성한 초기 해집단

염색체 이름	염색체 문자열	디코딩된 정수	염색체 적합도	적합도 비율(%)
X1	1100	12	36	16.5
X2	0100	4	44	20.2
X3	0001	1	14	6.4
X4	1110	14	14	6.4
X5	0111	7	56	25.7
X6	1001	9	54	24.8

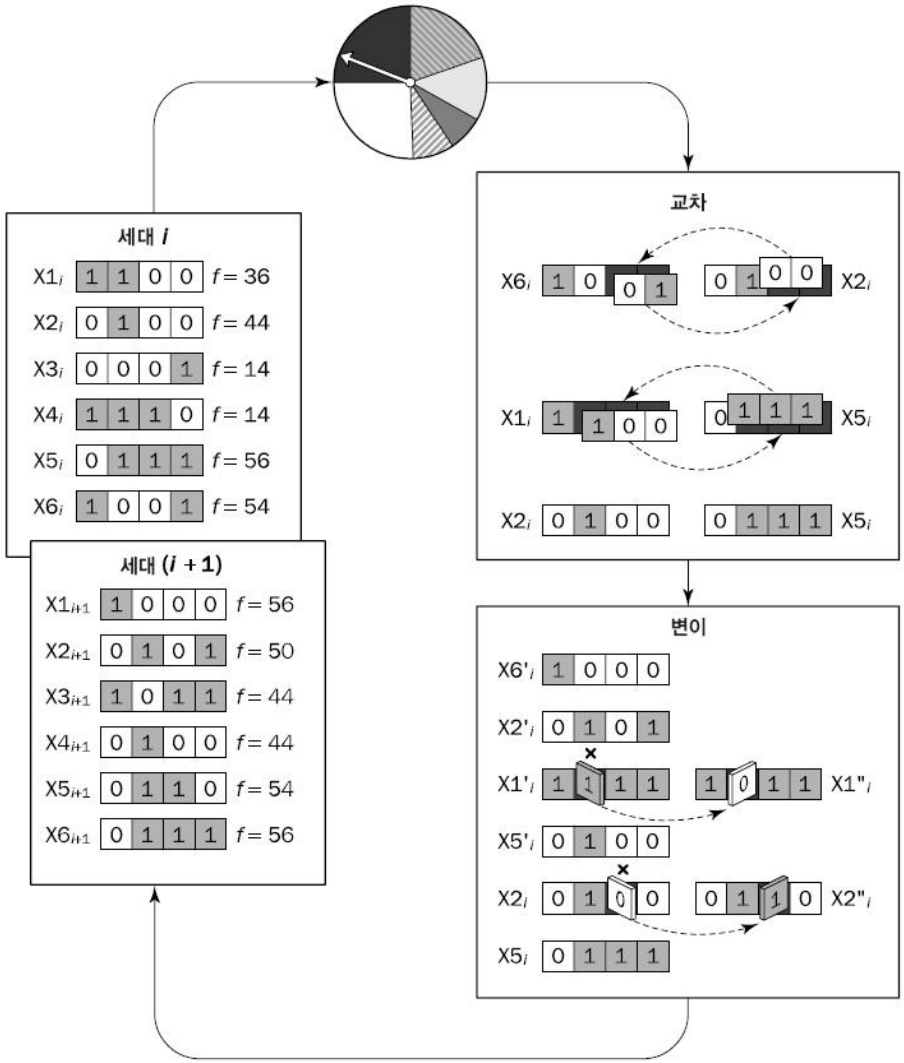
03_유전 알고리즘

- 룰렛 휠 선택 동작
  - 초기 해집단이 염색체 여섯 개로 이루어 졌다. 다음 세대에서도 같은 규모를 유지하려면 룰렛 휠을 여섯 번 돌려야 한다.

- 교차 연산자
  - 교차 연산자는 부모 염색체 둘이 '끊어지는' 지점(교차점)을 임의로 선택하고 염색체에서 이 지점보다 뒤쪽에 있는 부분을 교환한다. 이 결과 새로운 자식 염색체 두 개가 만들어진다.
  - 염색체 쌍이 교차되지 않으면 염색체 복제가 이루어져 부모를 똑같이 복사해서 자식을 만든다. [그림 7-5]
  - 일반적으로 교차율 값이 0.7일 대 좋은 결과를 낸다.
  - 선택과 교차를 수행한 후 해집단의 평균 적합도가 36에서 42로 올라갔다.

03_유전 알고리즘

- **교차 연산자** : [그림 7-5]



[그림 7-5] 유전 알고리즘의 한 주기

03_유전 알고리즘

- 변이
  - 변이는 유전자가 바뀌는 것을 나타내며, 자연계에서는 드물게 일어난다.
  - 변이는 적합도를 상당히 바람직하지 않은 결과를 만드는 경우가 많다.

- 변이의 사용 이유
  - 변이는 탐색 알고리즘이 지역 최적점에 갇히지 않도록 보장하는 역할을 한다.
  - 선택과 교차 연산만 적용하다 보면 동질적인 해집단에서 정체될 수 있다. 그런 경우에는 모든 염색체가 동일하기 때문에 해집단의 평균 적합도가 향상되지 않는다.
  - 변이는 임의 탐색(random search)과 동등하며 유전적 다양성을 잃지 않도록 도와준다.

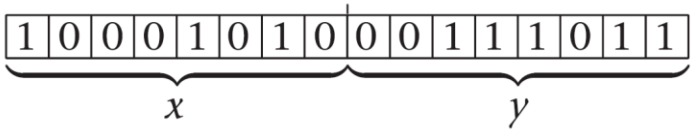
- 변이 연산자의 동작
  - 변이 연산자는 염색체에서 임의로 선택한 유전자를 뒤집는다.

    예) [그림 7-5]에서처럼 염색체 X1'의 두 번째 유전자가, 염색체 X2는 세 번째 유전자가 변이될 수 있다.
  - 변이는 염색체의 모든 유전자에 일정한 확률로 일어날 수 있다.
  - 변이율은 자연계에서도 꽤 낮고 유전 알고리즘에서도 0.001~0.01 사이의 범위로 꽤 낮다.

03_유전 알고리즘

❖ 적합도 계산

- 염색체를 x와 y로 변환하여 디코딩한 다음 디코딩 값을 피크 함수에 대입한다.
- x와 y 값의 범위는 -3 ~3 사이다.
- 다음과 같은 염색체가 가정하자.



- 디코딩
  - 16비트 문자열로 된 염색체를 8비트 문자열 두 개로 분할한다.

  - 이진수 문자열을 십진수로 변환 한다.

$$(10001010)_2 = 1\times 2^7 + 0\times 2^6 + 0\times 2^5 + 0\times 2^4 + 1\times 2^3 + 0\times 2^2 + 1\times 2^1 + 0\times 2^0$$
$$= (138)_{10}$$

$$(00111011)_2 = 0\times 2^7 + 0\times 2^6 + 1\times 2^5 + 1\times 2^4 + 1\times 2^3 + 0\times 2^2 + 1\times 2^1 + 1\times 2^0$$
$$= (59)_{10}$$

  - 8비트로 다룰 수 있는 정수의 범위 0~(2^8 - 1)을 인자 x와 y의 실제 범위 -3~3에 대응시킨다.

$$\frac{6}{256-1} = 0.0235294$$

03_유전 알고리즘

- 디코딩
  - x와 y의 실제 값을 얻으려면 십진 값에 0.0235294를 곱하고, 결과에서 3을 뺀다.

$$x = (138)_{10} \times 0.0235294 - 3 = 0.2470588$$

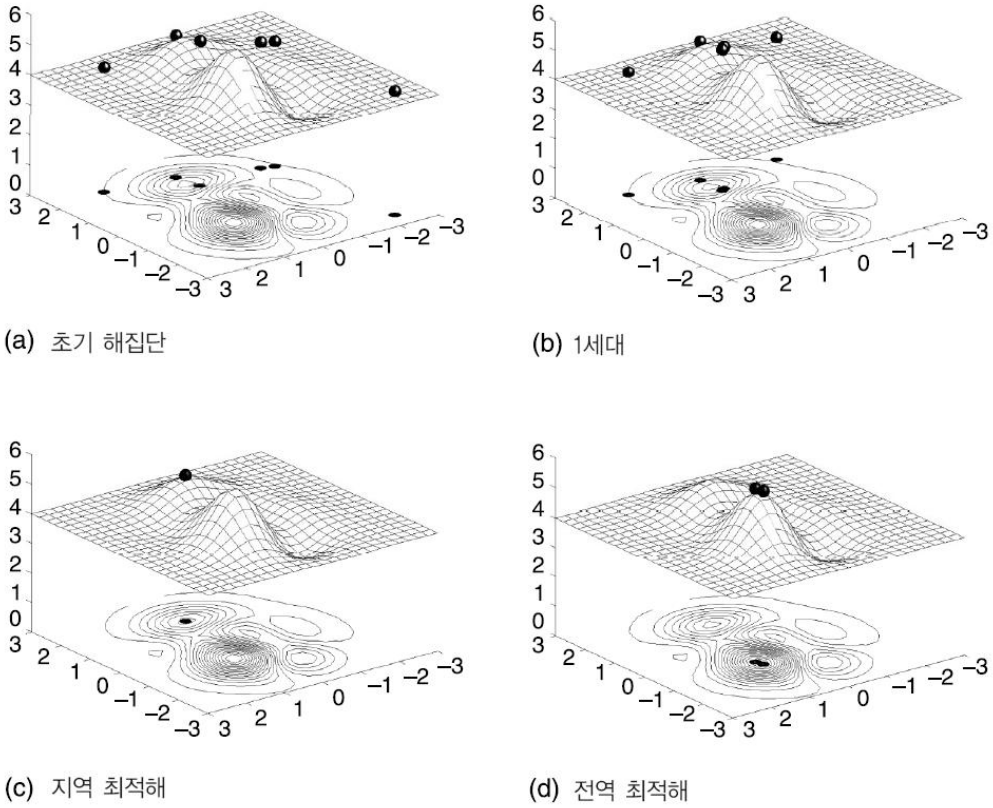
$$y = (59)_{10} \times 0.0235294 - 3 = -1.6117647$$

  - 그레이 코드(Gray coding)와 같은 다른 디코딩 기법도 적용할 수 있다.

- 피크 함수
  - x와 y가 디코딩된 값을 피크 함수에 입력하여 염색체 각각의 적합도를 계산한다.
  - 피크 함수의 최대값을 찾기 위해 교차율 0.7과 변이율 0.001을 사용한다.
  - 종료 세대수를 정한다.
    세대수를100으로 정하면 유전 알고리즘은 멈출 때까지 염색체 여섯 개를100세대만큼 생성한다.

- 피크 함수
  - 피크 함수 결과 그래프 : [그림 7-6]
  - 피크 함수의 3차원 표면과 등고선 상에 나타낸 것이다.

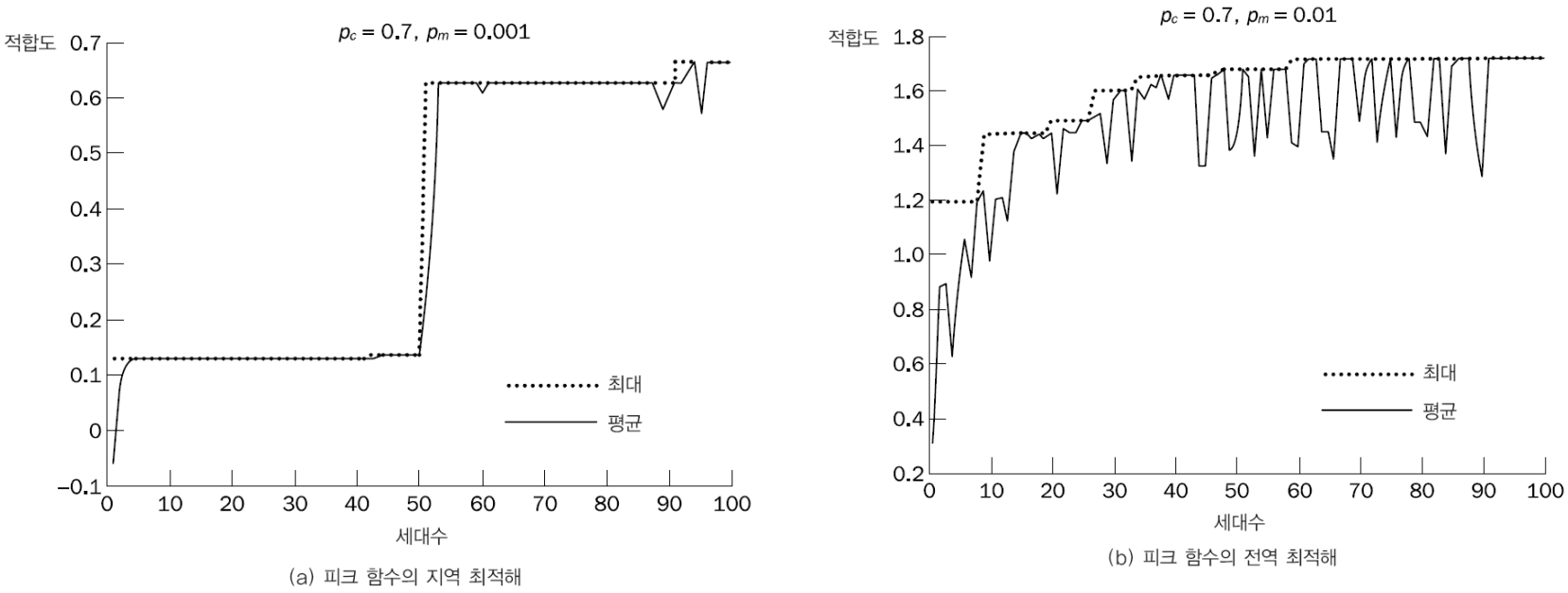
  - (a): 염색체의 초기 위치를 초기 해 집단은 임의로 생성되어 서로 비슷하지 않은 이질적인 개체로 이루어져 있다.
  - (b): 교차는 가장 좋은 염색체의 특성을 재조합하기 시작하고, 해집단이 최고값을 포함한 정점 주위로 모이기 시작한다.
  - (c): 해집단이 피크 함수의 지역 최적해에 해당하는 염색체로 수렴한다.
  - (d): 염색체가 전역 최적해로 수렴한다. 그러나 안정된 결과라는 확신을 얻으려면 해집단의 크기를 늘려야 한다.



[그림 7-6] 피크 함수의 3차원 표면과 등고선 상에 나타낸 염색체 위치

03_유전 알고리즘

- 성능 그래프
  - 유전 알고리즘 성능을 나타내기 위한 그래프.
  - [그림 7-6]과 같은 수학 함수의 표현 그래프는 유전 알고리즘의 성능을 보여주기에 유용한 방법이다. 그러나 실제 문제의 적합도 함수는 그래프로 쉽게 나타낼 수 없다.
  - 유전 알고리즘은 확률적이기 때문에 보통 세대마다 성능이 달라진다. 그러므로 해집단의 평균 성능을 나타내는 곡선뿐만 아니라 해집단에서 가장 우수한 개체의 성능을 나타내는 곡선도 일정한 세대수에 걸친 유전 알고리즘 동작을 살피기에 유용하다.
  - [그림 7-7] 100세대에 걸친 적합도 함수의 최대값과 평균값의 그래프.



[그림 7-7] 100세대에 걸친 염색체 6개의 성능 그래프

04_유전 알고리즘의 작동 원리

❖ 스키마 (schema)

- 유전 알고리즘 기법은 탄탄한 이론적 기반, 즉 스키마 정리를 바탕으로 한다.
- 존 홀랜드가 스키마라는 용어를 도입했다.

▪ 스키마 형태

- 스키마는 0, 1, 별표(*)로 이루어진 문자열의 집합이다.
- 별표에는 0과 1 중 아무 값이나 들어가도 된다.
- 스키마에서 1과 0은 고정된 자리를 나타내고, 별표는 와일드 카드를 나타낸다.
- 1로 시작해서 0으로 끝나는 4비트 문자열의 집합으로 나타낸 스키마. 예) | 1 | * | * | 0 |

▪ 스키마와 염색체 사이의 관계

- 단순한 관계다.
- 염색체와 스키마가 부합할 때는 스키마 안에 고정된 자리가 염색체와 대응하는 자리와 일치할 때다.
- 스키마가 | 1 | * | * | 0 | 와 같다고 하자. 이 스키마는 다음 4비트 염색체의 집합에 부합한다.

1	1	1	0
1	1	0	0
1	0	1	0
1	0	0	0

- 1로 시작해서 0으로 끝나는 염색체를 모드 스키마 H의 인스턴스다.
- 스키마에 있는 특정 비트(별표가 아닌 것) 개수를 차주(order)라 한다. 스키마 H에서는 특정 비트가 2개 있으므로 차수가 2다.

04_유전 알고리즘의 작동 원리

❖ 유전 알고리즘에서의 스키마

- 스키마의 조절
  - 유전 알고리즘은 실행할 때 스키마타(schemata)를 조절한다.
  - 스키마타는 스키마의 복수형이다.
  - 유전 알고리즘에서 재생산율이 적합도와 비례한다고 하면, 스키마 정리에 따라 특정 스키마가 다음 세대에도 존재할 확률을 예측할 수 있다.
  - 유전 알고리즘의 행동을 특정 스키마 인스턴스 개수의 증감으로 나타낼 수 있다.
  - 적합도가 평균보다 높은 스키마가 확실히 다음 세대의 염색체 중에서 더 자주 나타나는 경향이 있다.
  - 적합도가 낮은 스키마는 덜 나타나는 경향이 있다.

- 교차와 변이에 따라 일어나는 효과
  - 교차와 변이는 둘 다 스키마의 인스턴스를 생성하고 파괴할 수 있다.

- 스키마 정의 길이
  - 스키마에서 가장 바깥쪽에 있는 두 특정 비트 사이의 거리를 정의 길이라 한다.

    예) | * | * | * | * | 1 | 0 | 1 | 1 | - 정의 길이 : 3

    | 1 | * | * | * | * | * | * | 0 | - 정의 길이 : 7
  - 교차가 정의 길이 안쪽에서 일어나면 스키마 H는 파괴되어 H의 인스턴스가 아닌 자식이 만들어 질 수 있다.

04_유전 알고리즘의 작동 원리

- 스키마 정의 길이
  - 스키마 H가 교차 후에 살아남을 확률을 식(7.4)와 같이 정의할 수 있다.

$$P_H^{(c)} = 1 - p_c\left(\frac{l_d}{l-1}\right) \qquad (7.4)$$

  교차율 l과 l_d는 각각 스키마 H의 길이와 정의 길이다.
  - 교차 후에 살아남을 확률은 긴 스키마타보다 짧은 스키마타가 높다.
  - 스키마 H가 변이 후에 살아남을 확률은 식(7.5)와 같다.

$$P_H^{(m)} = (1 - p_m)^n \qquad (7.5)$$

  p_m을 스키마 H의 비트당 변이율, n을 스키마H의 차수라 하자. 그러면 $(1 - p_m)$은 그 비트가 변이 후에 바뀌지 않을 확률이 된다.
  - 변이 후에 살아남을 확률은 차수가 높은 스키마타보다는 차수가 낮은 스키마타가 놓다.

- 스키마 정의 : 식(7.6)
  - 식(7.4)와 (7.5)를 이용하여 식(7.6)과 같이 스키마가 성장할 확률을 구할 수 있다.

$$m_H(i+1) = \frac{\hat{f}_H(i)}{\hat{f}(i)} m_H(i)\left[1 - p_c\left(\frac{l_d}{l-1}\right)\right](1 - p_m)^n \qquad (7.6)$$

  - 교차와 변이의 효과만 고려하였기 때문에 다음 세대에 존재하는 스키마 H의 인스턴스 개수의 하한을 제시한다.

05_사례 연구: 유전 알고리즘으로 푸는 정비 스케줄링

- 유전 알고리즘을 성공적으로 응용한 분야 중 하나는 자원 스케줄링 문제다.
- 자원 스케줄링 문제를 풀 때는 보통 담색 기법과 휴리스틱을 결합한 방법을 사용한다.

❖ 스케줄링 문제가 어려운 이유

- 스케줄링은 NP-완비 문제다.
  - NP-완비 문제는 다루기 힘들고, 조합 탐색 기법으로는 풀 수 없다.
  - 휴리스틱만으로는 최적해를 보장할 수 없다.
- 제약 조건이 많다.
  - 스케줄링 문제는 자원이 제한되어 있어야 경쟁을 해야 하는데, 수많은 제약 조건이 걸려 복잡하다.

❖ 전력 시스템의 정비 스케줄

- 전력 시스템 정비 시 고려사항
  - 이 작업은 고장, 발전 설비의 사고정지, 교환 부품을 준비하는 지연 시간 등 여러 제약 조건과 불확실성을 고려하며 진행해야 한다.
  - 스케줄은 급히 수정될 때도 많다.

05_사례 연구: 유전 알고리즘으로 푸는 정비 스케줄링

❖ 전력 시스템의 정비 스케줄

- 인간 전문가가 스케줄링 할 때의 문제점.
  - 인간 전문가는 보통 손으로 정비 계획을 세우는데, 이 방법은 최적 스케줄을 보장하지 않는다.
  - 최적에 가까운 스케줄을 세운다는 보장도 할 수 없다.

- 유전 알고리즘을 이용하여 정비 스케줄을 세우는 과정.
  - 문제 영역을 명확히 제시하고 제약 조건과 최적해의 기준을 정한다.
  - 문제 영역을 염색체로 나타낸다.
  - 염색체의 적합도를 평가할 적합도 함수를 정의한다.
  - 유전 연산자를 고안한다.
  - 유전 알고리즘을 실행하고 인자를 조정한다.

- 정비 스케줄의 목적
  - 일정한 기간(보통 1년)동안 발생할 전력 장치의 정지를 파악하여 전력 시스템의 안전성을 최대화하는 것이다.

- 1단계: 문제 영역을 명확히 제시하고 제약 조건과 최적해의 기준을 정한다.
  - 이 단계를 정확하게 정리하지 못하면 실용적인 스케줄을 얻을 수 없기 때문에 가장 중요한 단계다.

05_사례 연구: 유전 알고리즘으로 푸는 정비 스케줄링

- 1단계: 문제 영역을 명확히 제시하고 제약 조건과 최적해의 기준을 정한다.
  - 전력 시스템의 구성 요소는 예방 정비를 통해 가용 기간 동안 멈추기 않고 작동해야 한다.
  - 모든 전력 시스템의 정지는 안전성의 감소를 초래한다.
  - 안전성(security margin)은 전력 시스템의 예지 전력에 따라 결정된다.
  - 예비 전력은 전력 시스템의 총 발전 용량에서 계획된 정지에 따른 전력 손실을 빼고, 정비 기간 동안 예상되는 최도 부하까지 뺀 것이다.
  - 예) 각 장치의 발전 용량과 정비 수요

[표 7-2] 전력 장치의 발전 용량과 정비 요구사항

장치 번호	발전 용량(MW)	1년 동안 장치 정비에 필요한 기간 수
1	20	2
2	15	2
3	35	1
4	40	1
5	15	1
6	15	1
7	10	1

- 동일한 네 기간 중에 정비할 전력 장치가 7개 있다고 가정하자.
- 각각의 기간에 예상되는 최대 부하는 80, 90, 65, 70MW다.

05_사례 연구: 유전 알고리즘으로 푸는 정비 스케줄링

- 1단계: 문제 영역을 명확히 제시하고 제약 조건과 최적해의 기준을 정한다.
  - 제약 조건을 다음과 같이 정리할 수 있다.
    - 장치의 정비는 기간의 처음에 시작하고, 그 기간이나 인접한 바로 다음 기간의 끝에 끝난다. 정비를 중단 할 수 없으며, 계획보다 일찍 끝낼 수도 없다.
    - 전력 시스템의 예비 전력은 항상 0 이상이어야 한다.

- 2단계: 문제 영역을 염색체로 나타낸다.
  - 전력 시스템 정비 스케줄 문제는 작업을 특정 순서래도 나열하는 문제다. 전체 스케줄은 중복된 작업들로 이루어질 수 있다.
  - 동시에 장치를 정비할 수도 있고, 정비에 걸리는 시간도 스케줄에 포함시켜야 하므로 장치 정비 스케줄의 순서를 정한다.
  - 장치 스케줄은 4비트 문자열로 나타낸다.

0	1	0	0

    - 각 비트는 정비 가간을 뜻한다. 특정 기간에 정비한다면 그에 대응하는 비트는 1, 그렇지 않은 비트는 0으로 설정한다.
    - 예에서는 둘째 기간에 정비되며, 이 장치를 정비하는 데 필요한 기간이 1이라는 것을 보여 준다.
  - 완전한 정비 스케줄은 28비트 염색체로 나타낼 수 있다.

05_사례 연구: 유전 알고리즘으로 푸는 정비 스케줄링

- 2단계: 문제 영역을 염색체로 나타낸다.
  - 교차와 변이 연산 적용 시 문제점
    - 교차와 변이 연산을 적용하면 어떤 장치는 한 번 이상 실행하고, 어떤 장치는 한 번도 정비하지 않는 이진 문자열이 만들어 질 수 있다.
    - 장치 정비에 필요한 기간 수를 초과해서 정비 기간을 요구하는 경우도 생길 수 있다.
  - 염색체 구성을 바꾸어 교차와 변이 연산을 적용한다.
    - 기본적으로 유전자 하나는 1비트로 나타내고 더 작은 요소로 나눌 수 없다.
    - 염색체를 더 이상 나눌 수 없는 가장 작은 단위인 4비트 문자열로 나타낸다.
  - 유전 알고리즘은 유전자 7개로 된 염색체를 앞에서 임의로 선택한 유전자로 채워 초기 해집단을 생성할 수 잇다.

장치 1:	1 1 0 0	0 1 1 0	0 0 1 1	
장치 2:	1 1 0 0	0 1 1 0	0 0 1 1	
장치 3:	1 0 0 0	0 1 0 0	0 0 1 0	0 0 0 1
장치 4:	1 0 0 0	0 1 0 0	0 0 1 0	0 0 0 1
장치 5:	1 0 0 0	0 1 0 0	0 0 1 0	0 0 0 1
장치 6:	1 0 0 0	0 1 0 0	0 0 1 0	0 0 0 1
장치 7:	1 0 0 0	0 1 0 0	0 0 1 0	0 0 0 1

05_사례 연구: 유전 알고리즘으로 푸는 정비 스케줄링

- 2단계: 문제 영역을 염색체로 나타낸다.
  - 유전 알고리즘은 유전자 7개로 된 염색체를 앞에서 임의로 선택한 유전자로 채워 초기 해 집단을 생성한다.

[그림 7-9] 스케줄링 문제에 사용하는 염색체

- 3단계: 염색체의 적합도를 평가할 적합도 함수를 정의한다.
  - 짝지을 염색체는 적합도에 비례하여 선택되기 때문에 염색체 평가는 유전 알고리즘에서 반드시 필요한 부분이다.
  - 적합도 함수
    - 적합도 함수는 무엇이 정비 스케줄을 사용자에게 좋게 하고 무엇이 나쁘게 하는지를 포착해야 한다.
    - 정비 시스템의 스케줄 문제에서는 제약 조건 위반 및 각 기간의 예비 전력과 관련 함수를 사용한다.
  - 염색체 평가 1 : 각 기간에 정비하기로 한 장치의 발전용량 더하기.

$$\text{기간}1: 0\times 20+0\times 15+0\times 35+1\times 40+0\times 15+0\times 15+1\times 10=50$$

$$\text{기간}2: 1\times 20+0\times 15+0\times 35+0\times 40+1\times 15+0\times 15+0\times 10=35$$

$$\text{기간}3: 1\times 20+1\times 15+0\times 35+0\times 40+0\times 15+1\times 15+0\times 10=50$$

$$\text{기간}4: 0\times 20+1\times 15+1\times 35+0\times 40+0\times 15+0\times 15+0\times 10=50$$

02_샘플

- 3단계: 염색체의 적합도를 평가할 적합도 함수를 정의한다.
  - 염색체 평가 2 : 전력 시스템의 총 발전 용량(150MW)에서 이 값을 뺀다.

$$\text{기간}\,1: 150 - 50 = 100$$

$$\text{기간}\,2: 150 - 35 = 115$$

$$\text{기간}\,3: 150 - 50 = 100$$

$$\text{기간}\,4: 150 - 50 = 100$$

  - 염색체 평가 3 : 각 기간에 예상되는 최대 부하를 빼서 예비 전력을 구한다.

$$\text{기간}\,1: 100 - 80 = 20$$

$$\text{기간}\,2: 115 - 90 = 25$$

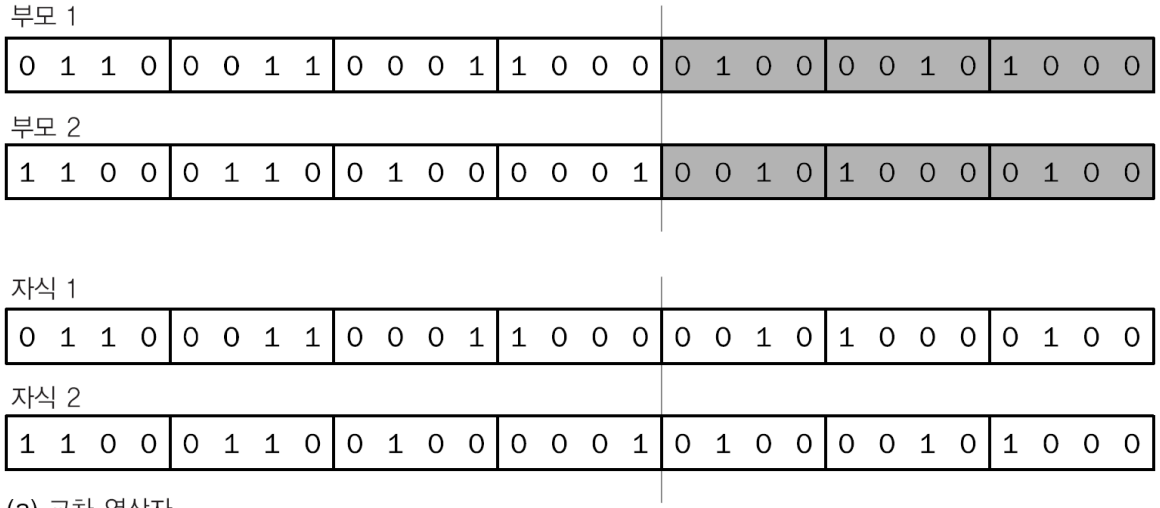
$$\text{기간}\,3: 100 - 65 = 35$$

$$\text{기간}\,4: 100 - 70 = 30$$

  - 결과 평가
    - 예비 전력 결과가 모두 양수다. 염색체 제약 조건을 위배하지 않았으므로 스케줄은 유효하다.
    - 염색체의 적합도는 가장 적은 예비 전력으로 결정한다. 이 예에서는 20이다.
    - 예비 전력 값이 음수인 기간이 있다면 스케줄은 유효하지 않다. 그 결과 적합도 함수의 결과값은 0이 된다.

- 3단계: 염색체의 적합도를 평가할 적합도 함수를 정의한다.
  - 결과 평가
    - 실행을 시작할 때, 임의로 만든 초기 해집단의 스케줄이 모두 유효하지 않을 수 있다. 이 경우 염색체의 적합도 값은 그래로 두고 실제 적합도 값에 따라 선택한다.

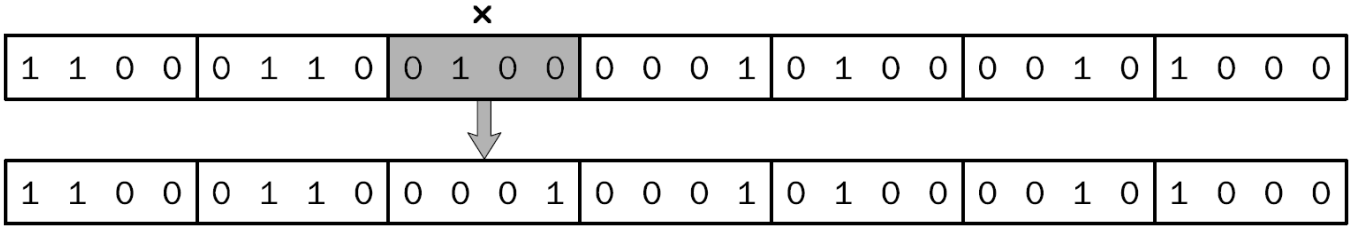
- 4단계: 유전 연산자를 고안한다.
  - 유전 연산자를 고안하는 것은 애무 어려운 작업으로 교차와 변이가 올바르게 작동하도록 실험해야 한다.
  - 염색체는 문제에서 적법한 방식으로 나뉘어야 한다.
  - 유전 알고리즘을 한 번 실행하는 동안 교차를 적용한 예: [그림 7-10] 의 (a)
    - 부모 염색체내 임의의 지점에 선을 긋고, 그 지점 뒤의 유전자를 교환하여 지식을 만듦.



(a) 교차 연산자

[그림 7-10] 스케줄링 문제를 위한 유전 연산자

- 4단계: 유전 연산자를 고안한다.
  - 변이를 적용한 예: [그림 7-10]의 (b)
    - 변이 연산자는 염색체에서 4비트 유전자 하나를 임의로 선택해 그에 대응하는 집합에서 임의로 선택한 유전자로 교체한다.
    - (b)를 보면, 염색체의 세 번째 유전자가 변이되어 장치 3에 대응하는 유전자 집합에서 선택된 유전자 0 0 0 1 로 교체된다.



(b) 변이 연산자

[그림 7-10] 스케줄링 문제를 위한 유전 연산자

- 5단계: 유전 알고리즘을 실행하고 인자를 조정한다.
  - 해집단 크기와 실행할 세대수를 정한다.
  - 해집단이 크면 더 좋은 해를 얻을 수 있으나 작동 속도가 느리다.
  - 가장 효율적인 해집단 크기는 문제를 인코딩하는 방법에 좌우된다.
  - 유전 알고리즘은 해를 얻을 때까지 한정된 세대만 실행한다.

06_진화 전략

- 독일 베를린 공과대학교 학생인 잉고 레켄베르크와 한스-파울 슈베펠이 제안.
- 유전 알고리즘과는 달리 기술적 최적화 문제를 풀기 위한 방법.
- 자연의 돌연변이를 본받아 형태를 정의하는 인자를 임의로 바꿈.
- 공학자의 직관을 대신하는 용도로 개발됨.

❖ 진화 전략의 구현

- 구현 조건
  - 가장 단순한 형태인 (1 +1) 진화 전략에서는 정규 분포 변이를 적용하여 한 세대당 부모 하나가 자식 하나를 생성한다.

(1 +1) 진화 전략

- 1단계: 문제를 나타낼 인자의 수 N을 정하고, 인자마다 유효한 범위를 정한다.
  - 각 인자의 최적화할 함수의 표준 편차를 정의한다.

$$\{x_{1min}, x_{1max}\}, \{x_{2min}, x_{2max}\}, \ldots, \{x_{Nmin}, x_{Nmax}\}$$

- 2단계: 인자 각각의 초기값을 유효한 범위 내에서 임의로 선택한다.
  - 인자 집합은 부모 인자의 초기 해집단을 구성한다.

$$x_1, x_2, \ldots, x_N$$

02_샘플

- 3단계: 부모 인자와 연관된 해를 계산한다.

$$X = f(x_1, x_2, \ldots, x_N)$$

- 4단계: 평균이 0이고, 미리 선택한 표준편차가 δ인 정규 분포를 따르는 확률 변수 a를 각 부모인자에 더해 새로운(자식) 인자를 만든다.
  - 평균이 0인 정규 분포를 따르는 변이는 커다란 변화보다 작은 변화가 자주 일어나는 자연의 진화 과정을 반영한 것이다.

$$x'_i = x_i + a(0, \delta) \qquad i = 1, 2, \ldots, N \qquad (7.7)$$

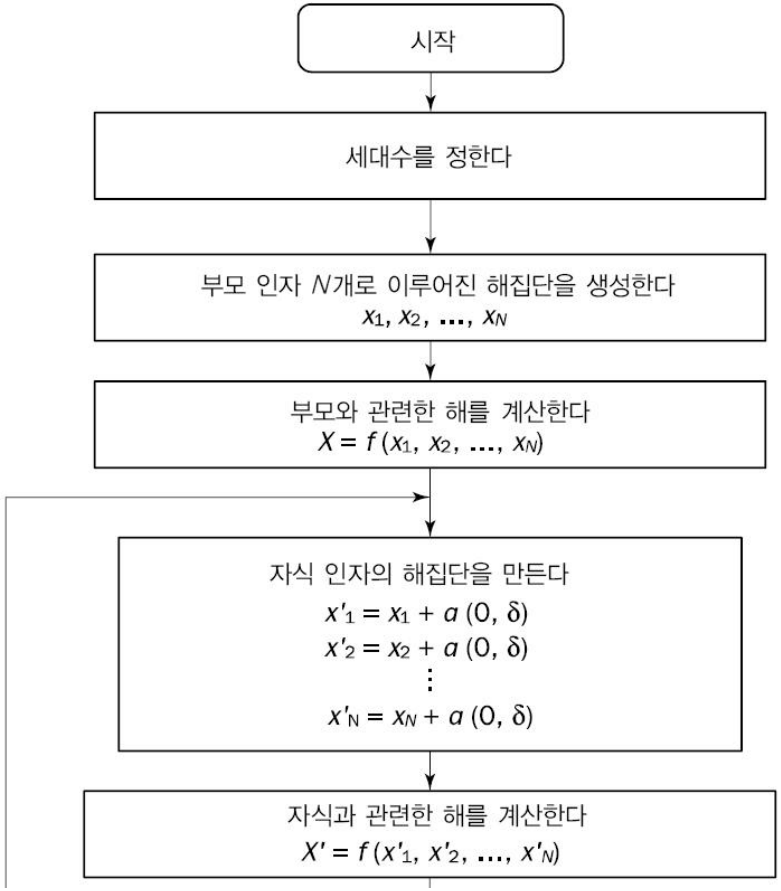
- 5단계: 자식 인자와 관련된 해를 계산한다.

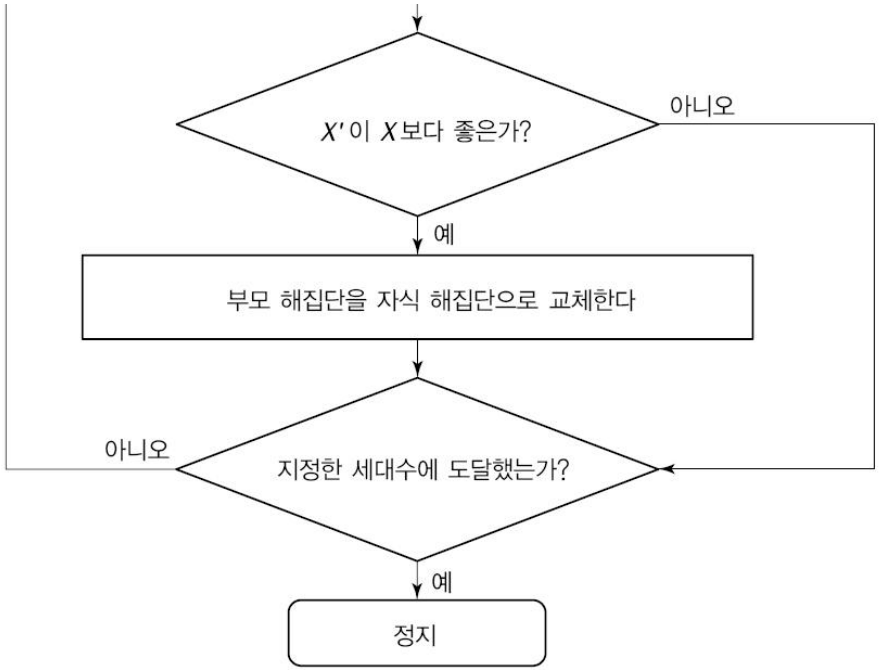
$$X' = f(x'_1, x'_2, \ldots, x'_N)$$

- 6단계: 부모 인자와 관련된 해와 지식 인자와 관련된 해를 비교한다.
  - 자식과 관련된 해가 부모와 관련된 해보다 더 좋으면 부모 해집단을 자식 해집단으로 교체한다. 그렇지 않으면 부모 인자를 유지한다.

- 7단계: 4단계로 돌아가서 만족스러운 해를 얻거나 지정한 세대수에 도달할 때까지 이 과정을 반복한다.

▪ 진화 전략 구현 블록 다이어그램





[그림 7-13] (1 + 1) 진화 전략의 블록 다이어그램

02_샘플

❖ 진화 전략의 특징

- 새로운 해를 생성할 때 모든 인자가 동시에 변함
  - 진화 전략은 염색체의 본성을 반영한다.
  - 한 유전자가 유기체의 여러 특징에 동시에 영향을 줄 수 있다.
  - 여러 유전자가 동시에 상호 작용하여 개체 특성 하나를 결정할 수 있다.
  - 자연 선택은 유전자 하나에 독립으로 작용하지 않고 집단에 작용한다.

- 진화 전략의 해결 범위
  - 제약 조건이 있는 것과 없는 것은 모두 포함한 광범위한 비선형 최적화 문제를 풀 수 있다.

- 유전 알고리즘과 진화 전략의 차이
  - 유전 알고리즘이 교차에 변이를 사용하는 반면, 진화 전략은 변이만 사용한다는 것이 주요 차이다.
  - 유전 알고리즘은 문제를 인코딩된 형태로 나타내야 하지만 진화 전략은 문제를 인코딩된 형태로 나타낼 필요가 없다.

07_유전 프로그래밍

- 유전 프로그래밍은 진화 연산 중에서 최근에 발전한 분야다.
- 1990년대 존 코자가 크게 발전 시켰다.

❖ 유전 프로그래밍

- 컴퓨터 과학 분야의 문제
  - 프로그램을 구체적으로 제시하지 않은 채 컴퓨터가 문제를 풀 수 있는 방법을 찾는 것이다.

- 유전 프로그래밍 특징
  - 유전 프로그래밍은 자연 선택 방법으로 컴퓨터 프로그램을 진화시킴으로써 컴퓨터 과학 분야의 문제를 해결하였다.
  - 유전 프로그래밍은 기존 유전 알고리즘을 확장한 것이다.
  - 유전 알고리즘은 해를 나타내는 이진 문자열을 만드는 반면, 유전 프로그래밍은 해로서 컴퓨터 프로그램을 만든다.

- 유전 프로그래밍의 목표
  - 어떤 문제의 이진 문자열 표현을 진화시키는 게 아니라 문제를 푸는 컴퓨터 코드를 진화시키는 것이 유전 프로그래밍의 목표다.

❖ 유전 프로그래밍

- 유전 프로그래밍의 연산 및 고려사항
  - 유전 프로그래밍은 컴퓨터 프로그램 공간에서 당장 문제를 풀기에 적합한 프로그램을 찾는다. (코자, 1992)
  - 모든 컴퓨터 프로그램은 값(인자)에 적용되는 연산자(함수)의 순서열이다.
  - 프로그래밍 언어마다 사용하는 구문, 연산자, 문법 제약이 모두 다르다.
  - 유전 프로그래밍은 유전 연산자를 적용하여 프로그램을 조작하기 때문에 프로그래밍 언어가 컴퓨터 프로그램을 데이터로 조작하고 새로운 만든 데이터를 프로그램으로 실행할 수 있어야 한다.

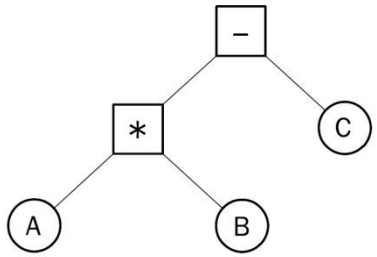
- 리스프
  - 유전 프로그래밍을 구현하는 언어다.
  - 가장 오래된 고급 언어 중 하나다(1950년대 말에 개발한 언어).
  - 리스프 구조는 기호 지향적이다.
  - 리스프의 기본 자료 구조는 아톰(atom)과 리스트(list)다.
    - 아톰은 리스트 문법에서 나눌 수 없는 가장 작은 원소다.
    - 숫자 21, 기호 X, 문자열 '이것은 문자열이다'같은 것이 아톰의 예다.
    - 리스트는 아톰이나 다른 리스트로 이루어진 객체다.
  - 리스프의 리스트는 괄호 안에 기호를 순서대로 모아서 적는다.

    예) $(-(* \ A \ B) \ C)$

07_유전 프로그래밍

- 리스프
  - (- (*A B) C)는 리스트 와 아톰 C에 뺄셈 함수 (-)를 적용하라는 뜻이다. 먼저 리스프는 곱셈 함수 ( )를 아톰 A와 B에 적용한다. 리스트 가 평가되면 리스프는 뺄셈 함수 (-)를 두 인자에 적용하여 전체 리스트를 평가한다.
  - 리스프 프로그램은 자기 자신을 수정할 수도 있고, 심지어 다른 리스프 프로그램을 작성할 수도 있다. 이러한 리스프의 비범한 특징은 유전 프로 그래밍의 관점에서 매우 매력적이다.

- 리스프 S-표현식(S-expression)
  - 아톰과 리스트 모두를 기호 표현식(symbolic expression) 또는 S-표현식이라 한다.
  - 리스프의 S-표현식은 모두 루트(root)가 있고 가지에 순서가 있는 트리로 표현할 수 있다. : [그림 7-14] - S-표현식 (- (*A B) C)에 대응하는 트리.



[그림 7-14] 리스프 S-표현식 (- (* A B) C)의 그래프 표현

  - 그래프를 표현할 때는 가지에 순서를 정한다. 함수가 많을 때는 인자의 순서가 결과에 직접 영향을 주기 때문이다.

❖ 유전 프로그래밍의 활용

▪ 문제 적용 방법

- 유전 프로그래밍을 문제에 적용하기에 앞서 다섯 가지 준비 단계를 완료해야 한다.
  - 말단 집합(set of terminals)을 정한다.
  - 기본 함수 집합을 선택한다.
  - 적합도 함수를 정의한다.
  - 실행을 제어할 인자를 결정한다.
  - 실행 결과를 나타낼 방법을 선택한다.

▪ 문제 상황

- 유전 프로그래밍의 목표는 피타고라스의 정리에 맞는 프로그램을 발견하는 것이다.
- 피타고라스 정리 : 대각선 c와 밑변 a, b의 관계를 나타내는 정리.

$$c = \sqrt{a^2 + b^2}$$

- 피타고라스 정리 적합도 예 : [표 7-3]

[표 7-3] 피타고라스 정리 적합도 예제 10가지

밑변 a	밑변 b	대각선 c	밑변 a	밑변 b	대각선 c
3	5	5.830952	12	10	15.620499
8	14	16.124515	21	6	21.840330
18	2	18.110770	7	4	8.062258
32	11	33.837849	16	24	28.844410
4	3	5.000000	2	9	9.219545

07_유전 프로그래밍

- 피타고라스 정리 적합도 예 : [표 7-3]
  - [표 7-3]은 직각삼각형의 여러 개를 사용하여 얻은 결과 값이다.
  - 발견하기 못한 컴퓨터 프로그램의 성능을 측정하기 위해 서로 다른 적합도 예제 여러 개를 사용한다.
  - 적합도 예제는 변수 a와 b 값의 범위에서 임의로 선택된다.

[유전 프로그래밍의 적합도 구하기]

- 1단계: 말단 집합을 정한다.
  - 말단을 발견한 컴퓨터 프로그램의 입력에 대응한다. 여기서는 입력 두 개 a와 b가 있다.

- 2단계: 기본 함수의 집합을 선택한다.
  - 함수는 산술 연산자, 프로그래밍 연산자, 수학 함수, 논리 함수, 영역에 특화된 함수가 될 수 있다.
  - 여기서는 사칙연산(+, -, *, /)과 수학 함수 sqrt(제곱근)를 사용한다.

- 3단계: 적합도 함수를 정의한다.
  - 적합도 함수는 특정 컴퓨터 프로그램이 얼마나 문제를 잘 풀 수 있는지를 평가한다.
  - 적합도 함수는 문제에 따라 달라지며, 각 문제에 따라 크게 달라질 수 있다.

07_유전 프로그래밍

- 3단계: 적합도 함수를 정의한다.
  - 피타고라스 정리 문제에서는 컴퓨터 프로그램의 적합도를 프로그램의 실제 결과와 적합도 예제로 주어진 결과 사이의 오차로 측정할 수 있다.
  - 오차는 적합도 예제 하나만 가지고 재는 것이 아니라, 여러 적합도 예제에 대한 오차의 절대값의 합으로 계산한다.
  - 합계가 0에 가까울수록 훌륭한 컴퓨터 프로그램이다.

- 4단계: 실행을 제어할 인자를 결정한다.
  - 실행을 제어하려면 유전 알고리즘에서 사용하는 기본 인자를 사용해야 한다.
  - 피타고라스 정리 문제에서는 해집단 크기와 한 번 실행할 때 필요한 최대 세대수가 포함된다.

- 5단계:
  - 실행의 결과로 생성된 프로그램 중 가장 적합도가 높은 것을 선정한다.

- 유전 프로그래밍의 실행
  - 적합도를 구하기 위한 5단계까지 마치면 유전 프로그래밍을 실행할 수 있다.
  - 컴퓨터 프로그램으로 이루어진 초기 해집단을 임의로 생성하여 실행을 시작한다.
  - 프로그램은 +, -, , /, sqrt 함수와 말단 a와 b로 이루어진다.

[교차 연산자와 컴퓨터 프로그램]

- 교차 연산자를 컴퓨터 프로그램에 적용하기.
  - 교차 연산자는 적합도에 따라 선택한 두 컴퓨터 프로그램에 작용한다.
  - 두 프로그램의 크기와 모양은 각자 다를 수 있다.
  - 부모 프로그램에서 임의로 선택한 부분을 재조합하여 자식 프로그램 두 개를 만든다.
  - 예) 리스프 S-표현식 두 개를 생각해 보자.

$$(/\ (-\ (sqrt\ (+\ (*\ a\ a)\ (-\ a\ b)))\ a)\ (*\ a\ b)) \quad \text{- (1)}$$

$$(+\ (-\ (sqrt\ (-\ (*\ b\ b)\ a))\ b)\ (sqrt\ (/\ a\ b))) \quad \text{- (2)}$$

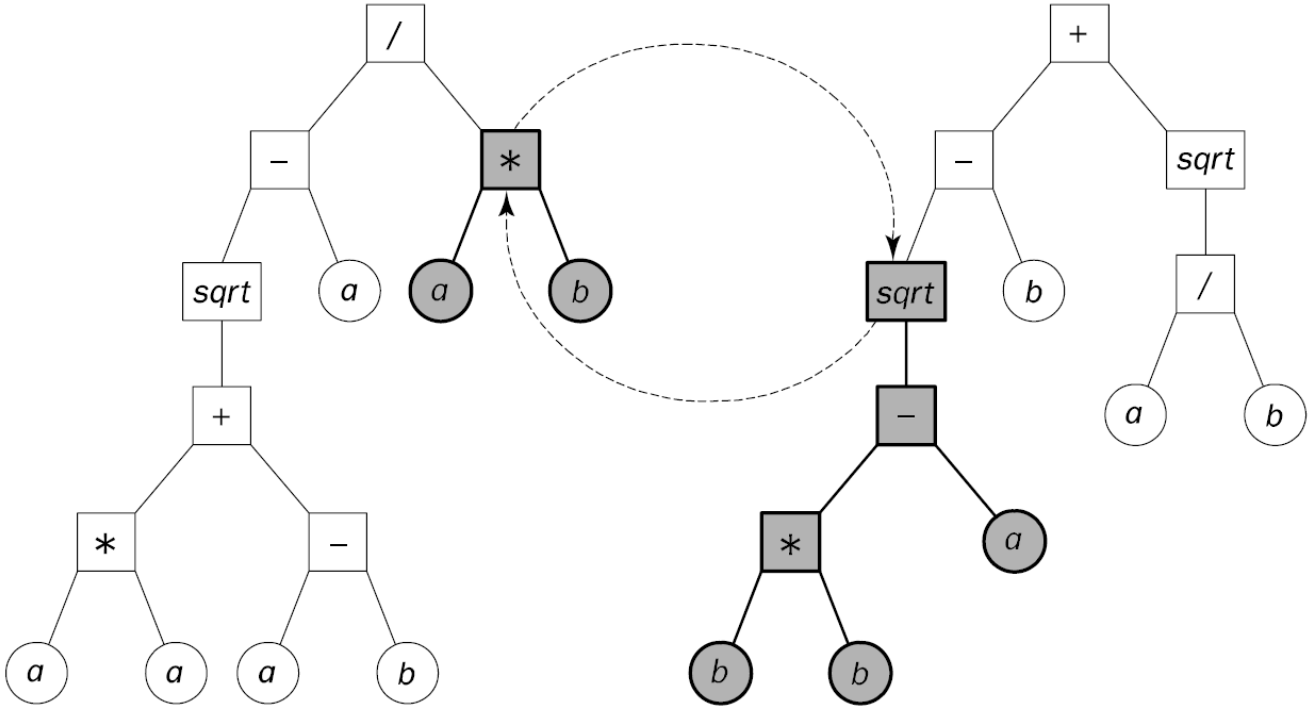
  - 이를 수식으로 표현하면 다음과 같다.

$$\frac{\sqrt{a^2 + (a-b)} - a}{ab} \qquad \left(\sqrt{b^2 - a} - b\right) + \sqrt{\frac{a}{b}}$$

  - 내부든 외부든 모든 노드를 교차점으로 선택할 수 있다.
  - 예) 첫째 부모(왼쪽)의 교차점을 ( ) 함수, 둘째 부모(오른쪽)의 교차점을 sqrt 함수라 하자. 그 결과 [그림 7-15]의 (a)와 같이 선택한 교차점 을 루트로 하는 두 교차 부분을 얻는다.

07_유전 프로그래밍

- 교차 연산자를 컴퓨터 프로그램에 적용하기.



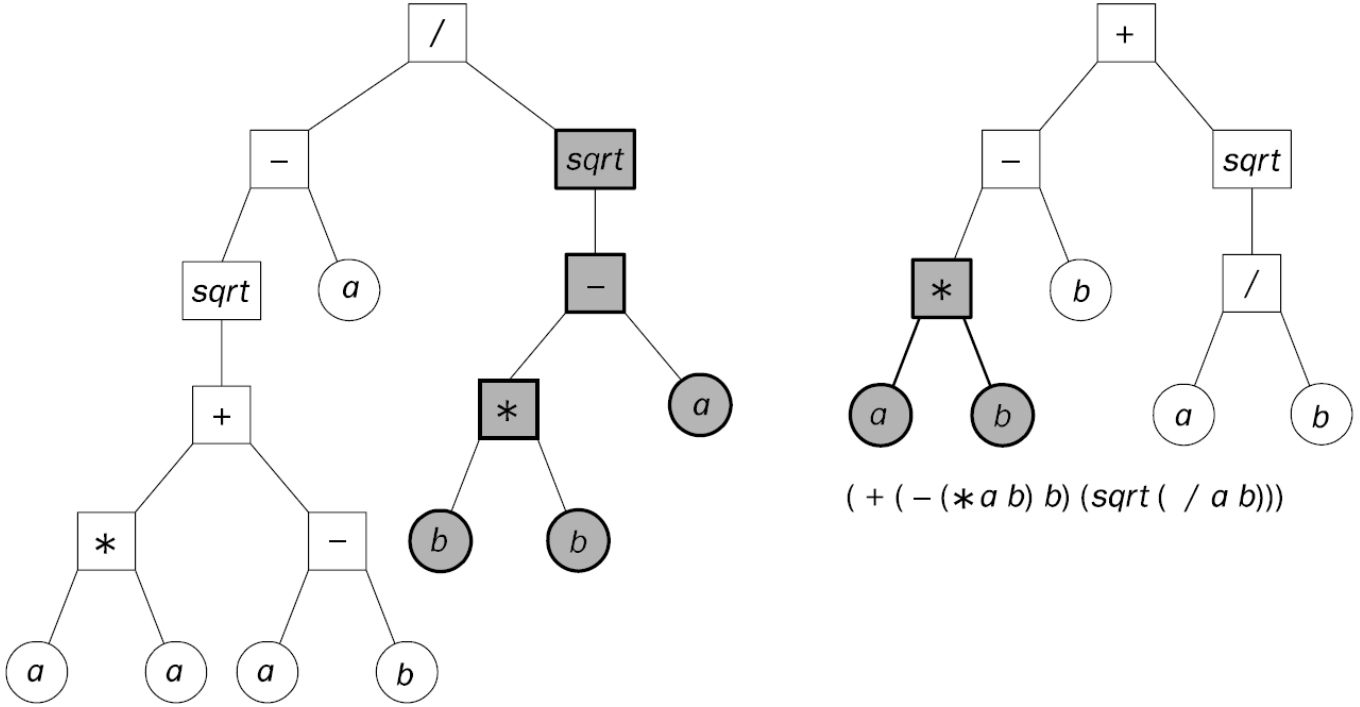
(a) 부모 둘의 S-표현식

[그림 7-15] 유전 프로그래밍의 교차

- 교차 연산자는 부모 둘의 교차 부분을 교환하여 자식 둘을 만든다. 따라서 둘째 부모의 교차 부분을 첫째 부모의 교차 부분이 있던 자리에 넣어 첫 번째 자식을 만든다.

07_유전 프로그래밍

- 교차 연산자를 컴퓨터 프로그램에 적용하기.
  - 첫째 부모의 교차 부분을 둘째 부모의 교차 부분이 있던 자리에 넣어 둘째 자식을 만든다.
  - [그림 7-15]의 (b)는 부모 둘을 교차시켜 나온 두 개의 자식이다.



(b) 자식 둘의 S-표현식

[그림 7-15] 유전 프로그래밍의 교차

07_유전 프로그래밍

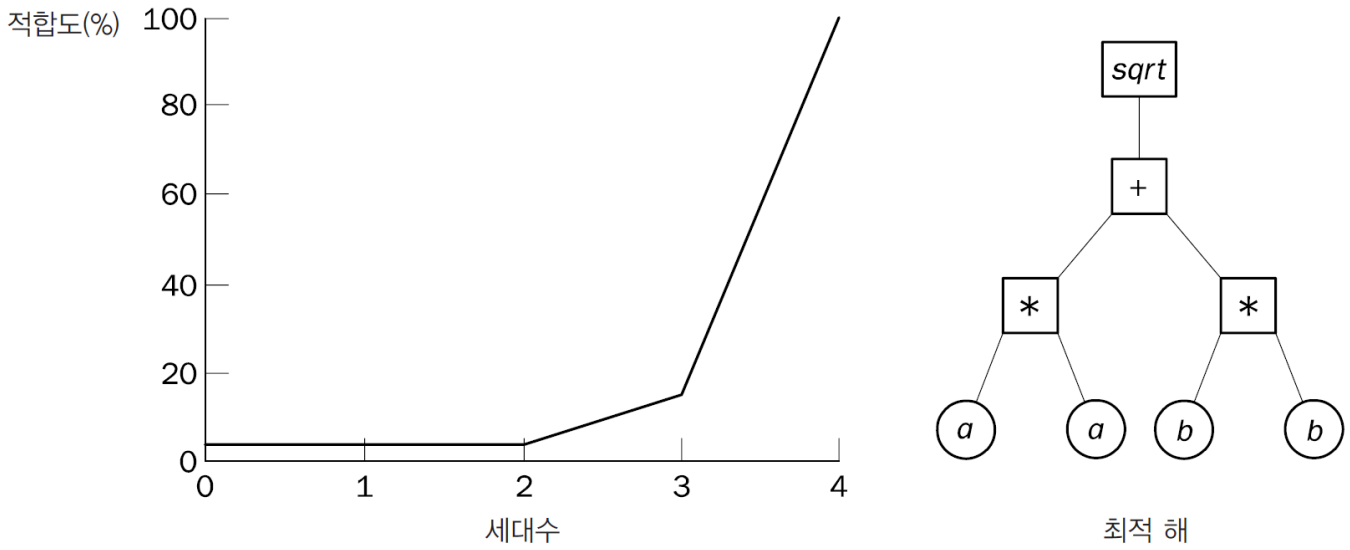
[유전 프로그래밍으로 컴퓨터 프로그램 만들기]

- 1단계
  - 실행할 최대 세대수, 복제율, 교차율, 변이율을 정한다. 복제율, 교차율, 변이율을 모두 더하면 1이 되어야 한다.
- 2단계
  - 선택한 함수와 말단을 임의로 결합하여 컴퓨터 프로그램 N개로 이루어진 초기 해집단을 생성한다.
- 3단계
  - 해집단의 컴퓨터 프로그램을 각각 실행하고, 적합도 함수를 적용하여 적합도를 계산한다.
  - 현재 적합도가 가장 높은 개체를 실행 결과로 선정한다.
- 4단계
  - 정해진 확률대로 유전 연산자를 선택하여 복제, 교차, 변이를 수행한다.
- 5단계
  - 복제 연산자가 선택되면, 현재 프로그램 해집단에서 컴퓨터 프로그램을 하나 선택하여 새로운 해집단에 복사해 넣는다.

- 5단계
  - 교차 연산자가 선택되면, 현재 해집단에서 컴퓨터 프로그램 쌍을 선택하여 자식 프로그램 쌍을 만들고, 새로운 해집단에 넣는다.
  - 변이 연산자가 선택되면, 현재 해집단에서 컴퓨터 프로그램 하나를 선택하고 변이를 수행하여 변이된 것을 새로운 해집단에 넣는다.
  - 모든 프로그램을 적합도에 근거한 확률에 따라 선택한다.

❖ 피타고라스 정리 결과

- [그림 7-18]은 컴퓨터 프로그램 500개로 이루어진 해집단에서 가장 좋은 S-표현식의 적합도 이력을 보여준다.



[그림 7-18] 가장 좋은 S-표현식의 적합도 이력

❖ 피타고라스 정리 결과

- 결과 분석 : [그림 7-18]
  - 임의로 생성된 초기 해집단에서는 가장 좋은 S-표현식도 적합도가 매우 형편없다.
  - 적합도는 매우 급격히 향상되고, 넷째 세대에서는 올바른 S-표현식이 재생산된다.
  - 예제는 유전 프로그래밍이 컴퓨터 프로그램을 진화시키는 일반적이고 믿을 수 있는 방법임을 보여준다.

❖ 유전 프로그래밍의 특징

- 유전 알고리즘과 유전 프로그래밍
  - 유전 프로그래밍은 유전 알고리즘과 동일한 진화 방법을 적용한다.
  - 유전 프로그래밍은 이진 문자열을 다루지 않고, 특정 문제를 푸는 완전한 컴퓨터 프로그램으로 해를 나타낸다.
  - 유전 알고리즘과 다른 점은 문제 표현, 즉 고정 길이 인코딩에 있다. 나쁜 표현은 유전 알고리즘의 위력을 제한하고, 심한 경우에는 잘못된 해를 이끌어낼 수도 있다.
  - 고정 길이 인코딩은 다소 인공적이다. 길이의 동적 다양성을 제공하지 못하기 때문에 낭비가 많고 유전 탐색의 효율성을 떨어뜨린다.
  - 유전 프로그래밍은 길이가 변하는 고차 빌딩 블록을 사용한다. 그 크기와 복잡도는 프로그램 수행 중에 변할 수 있다.
  - 유전 프로그래밍은 매우 다양한 문제를 잘 해결하고 있고, 앞으로도 다양하게 응용할 수 있다

❖ 유전 프로그래밍의 특징

- 유전 프로그래밍의 단점
  - 유전 프로그래밍은 이미 여러 차례 응용에 성공했지만, 아직까지 대형 컴퓨터 프로그램이 필요할 정도로 복잡한 문제를 다룰 수 있다는 사실은 증명하지 못했다.
  - 대형 컴퓨터 프로그램을 다룰 수 있다고 해도 수행 시간이 무척 길 것이다.

08_요약

❖ 진화 연산 요약

- 진화 연산
  - 인공지능에 대한 진화적 접근법을 진화 연산이라고 하며, 자연 선택과 유전학의 계산 모델에 기초한다. 진화 연산은 유전 알고리즘, 진화 전략, 유전 프로그래밍을 포함한다.
  - 진화 연산의 모든 방식은 개체 집단을 만들어 적합도를 평가하고, 유전 연산자를 적용하여 새로운 해집단을 생성하는 과정을 일정 횟수만큼 반복한다.

- 유전 알고리즘
  - 유전 알고리즘은 존 홀랜드가 1970년대 초에 개발했다. 홀랜드의 유전 알고리즘은 인공적인 '염색체'를 한 세대에서 또 다른 세대로 옮기는 절차적인 단계다. 유전 알고리즘은 '자연' 선택과 유전학에서 영감을 얻은 교차와 변이를 사용한다. 염색체는 여러 '유전자'로 이루어지며, 각 유전자는 0과 1로 표현한다.
  - 유전 알고리즘은 재생산할 때 개별 염색체의 적합도 값을 사용한다. 교차 연산자는 두 염색체의 일부를 교환하고, 변이 연산자는 염색체에서 임의로 선택한 유전자 값을 바꾼다. 여러 세대에 걸쳐 재생산을 수행하면 적합도가 낮은 염색체는 멸종하고, 적합도가 가장 높은 염색체는 점점 해집단 전체를 지배한다.
  - 유전 알고리즘으로 문제를 풀려면 먼저 제약 조건과 최적해의 기준을 정의하고, 문제의 해를 염색체로 인코딩한다. 그런 다음 염색체의 성능을 평가할 적합도 함수를 정의하고, 적절한 교차와변이 연산자를 만든다.

08_요약

- 진화 전략
  - 진화 전략은 해석적 목적 함수와 전통적인 최적화 방법이 없이 공학자의 직관에만 의존해야 하는 기술적 최적화 문제에 사용된다.
  - 진화 전략은 순수한 수치 최적화 절차로, 집중된 몬테카를로 탐색과 비슷하다. 유전 알고리즘과 달리 진화 전략은 변이 연산자만 사용한다. 또 문제를 인코딩한 표현은 필요 없다.

- 유전 프로그래밍
  - 유전 프로그래밍은 유전 알고리즘과 같은 진화적 방법을 적용한다. 그러나 더 이상 해를 인코딩한 이진 문자열을 다루지 않고, 당장의 문제를 푸는 완전한 컴퓨터 프로그램을 다룬다.
  - 유전 프로그래밍으로 문제를 풀려면 먼저 인자 집합을 결정하고, 함수 집합을 선택한다. 그런 다음 만들어진 컴퓨터 프로그램을 평가할 적합도 함수를 정의하고, 실행 결과를 선정할 방법을 선택한다.
  - 유전 프로그래밍은 유전 연산자를 적용하여 프로그램을 조작하기 때문에 프로그래밍 언어가 컴퓨터 프로그램을 데이터로서 조작하고, 새로 만들어진 데이터를 프로그램으로 실행되게 해야 한다. 이런 이유로 유전 프로그래밍의 주 언어를 리스프로 선택했다.

Thank You !