Chapter 1.1 Application: Verifying Polynomial Identities

다음과 같은 다항식이 주어졌을 때,

$$(x+1)(x-2)(x+3)(x-4)(x+5)(x-6)$$

계산하여 다음과 같이 결과를 내는 프로그램을 개발했다고 하자.

$$(x+1)(x-2)(x+3)(x-4)(x+5)(x-6) = x^6 - 7x^3 + 25$$

위 식이 참인지 알아보고 싶다면(좌변과 우변이 같은지),
좌변을 우변으로 계산해주는 프로그램을 새로 구현해서 double-checking 하면 된다.

이 방법은 다음과 같은 두 가지 단점이 있다.
1) 새로 구현한 코드가 기존의 코드와 다른 것이 없다.
  - 기존의 코드를 구현한 사람이 다시 개발한다면, 같은 버그가 존재할 가능성이 높다.
2) 속도가 느리다.
  - 좌변의 식을 우변으로 바꾸기 위해서는 O(d^2)의 시간복잡도가 소요된다.
  ($(x-6)$을 제외한 $(x+1)(x-2)(x+3)(x-4)(x+5)$식을 다 계산했다고 가정해보자.)
  계산된 식은 5 차항이다.
  (x-6)와 5 차항을 곱하기 위해서는 10(5*2)번의 계산이 필요하다.
  즉, 하나의 항을 계산할 때마다 최대 n-1 개의 계산이 필요하므로,
  시간복잡도는 O(d^2)

Deterministic 한 방법을 이용하여 checking 하는 것은 시간이 오래 걸리고, 에러를 찾지 못 할 가능성이 높기 때문에, Randomized 한 방법을 이용하여 체크하는 방법을 알아보자.

먼저,
$F(x) = (x+1)(x-2)(x+3)(x-4)(x+5)(x-6)$
$G(x) = x^6 - 7x^3 + 25$
라고 정의하자.

그리고 F(x)와 G(x)의 최고차항을 d 라고 했을 때, (여기서 d=6)

{ 1 … 100*d } 집합에서 하나의 수를 uniformly 선택한다.
선택된 수를 r 이라고 하자.

숫자 r 을 각각 F(x)와 G(x)에 대입하여,
F(r) = G(r) 이면 두 식은 같고 F(r) ≠ G(r) 두식은 다르다고 판단한다.

위의 Randomized Algorithm 은 다음과 같은 특성을 갖는다.

1) 알고리즘이 deterministic 알고리즘보다 빠르다.
   F(r)을 계산하는데, O(d)의 시간복잡도가 걸린다.(x 에 대입 후 d 항 곱셈연산)
   G(r)을 계산하는데, 역시 최대 O(d)의 시간복잡도가 걸린다. (최고차항 계산)
2) 알고리즘이 언제나 올바른 답을 주지 않는다.
   이 알고리즘을 수행하면 다음과 같이 3 가지 결과가 나올 수 있다.

   i) F(x) = G(x), F(r) = G(r)
   올바른 결과값이 나왔다.

   ii) F(x) ≠ G(x), F(r) ≠ G(r)
   i)와 마찬가지로 알고리즘이 적절한 결과를 생산했다.

   iii) F(x) ≠ G(x), F(r) = G(r)
   이 경우가 알고리즘에서 올바르지 않은 결과값을 발생하는 부분이다.
   H(x) = F(x) – G(x)라고 했을 때, 함수 H(x)의 해가 존재할 수 있는데, 그 해가 r 일 경우에 위와 같은 결과가 나올 수 있다.

Randomized 알고리즘은 위와 같이 언제나 올바른 해답을 주지 않지만, 그럼에도 불구하고 Deterministic 알고리즘을 사용하는 것보다 좋을 수 있다.

그 이유는 다음 장에서 설명하도록한다.