BOJ 17603
Factorization

#number_theory #discrete_sqrt

난이도 – Diamond V

sorohue@sasa.hs.kr

BOJ 17603 Factorization

- $a_0 \equiv b_0 \cdot b_1,\ a_1 \equiv b_0 + b_1 \pmod p$ 를 만족하는 두 수 b_0, b_1을 구해야 합니다.

- 곱 또는 합 중 하나에 대해 가능한 모든 경우를 탐색해 보는 경우 테스트 케이스 당 최소 $O(p)$가 필요합니다.
- 문제를 해결하기 위해선 더 빠르게 계산해야 합니다.

BOJ 17603 Factorization

- 식을 정리해 봅시다. 모든 합동식은 mod p 입니다.
  - $a_1 \equiv b_0 + b_1$ 를 양변 제곱하면 $a_1^2 \equiv (b_0 + b_1)^2$
  - 양변에서 $4a_0$ 을 빼면 $a_1^2 - 4a_0 \equiv (b_0 + b_1)^2 - 4b_0 b_1 \equiv (b_0 - b_1)^2$

- 이제 $b_0 - b_1 \equiv a_2$ 라고 합시다. 그러면 $a_2^2 \equiv a_1^2 - 4a_0$ 입니다.
- a_2의 값만 구할 수 있다면 b_0 와 b_1 을 쉽게 구할 수 있습니다.
- a_2 를 빠르게 구하는 방법을 알아 봅시다.

BOJ 17603 Factorization

- $x^2 \equiv N \bmod p$ 꼴에서 x를 p에 대한 N의 이산 제곱근이라 부릅니다.
- Tonelli-Shanks 알고리즘으로 홀수 소수 p에 대한 어떤 수의 이산 제곱근을 빠른 시간 안에 계산할 수 있습니다.
- 유일한 짝수 소수인 2에 대해서는 전처리를 해 주어야 합니다.

BOJ 17603 Factorization

- Tonelli-Shanks 알고리즘은 다음과 같이 작동합니다.
- $x^2 \equiv N \bmod p$ 에서 x의 값을 구한다고 합시다.
  - Step I. 이산 제곱근의 존재성 확인
  - Step II. $p-1 = q \times 2^S$ 인 홀수 q, 음이 아닌 정수 S 계산
  - Step III. mod p에서 이산 제곱근이 존재하지 않는 자연수 z 검색
  - Step IV. 알고리즘에 사용할 파라미터 c, r, t 계산
  - Step V. $t \leq 1$이 될 때까지 특정 계산을 반복

BOJ 17603 Factorization

- Step I. 이산 제곱근의 존재성 확인
  - 만약 x가 존재한다면, 페르마의 소정리에 의해 $x^{p-1} \equiv 1 \bmod p$ 입니다.
  - 이때 p−1이 짝수이므로 $x^{p-1} \equiv N^{(p-1)/2} \equiv 1 \bmod p$ 이면 x가 존재하고,

    이외의 경우 x가 존재하지 않습니다.
  - 특히 자연수 N에 대해 $N^{(p-1)/2} \equiv 1$ 또는 $N^{(p-1)/2} \equiv -1$임을 기억합시다.
- Step I 의 시간 복잡도는 $O(\lg p)$ 입니다.

BOJ 17603 Factorization

- Step II. $p-1 = q \times 2^S$ 인 홀수 q, 음이 아닌 정수 S 계산
  - p−1을 더 이상 나누어지지 않을 때까지 2로 나누면 됩니다.
  - 비트 연산을 쓰는 쪽이 약간 더 빠릅니다.

    (이 문제를 푸는 데 있어서는 필요하지 않은 최적화입니다.)

- Step II 의 시간 복잡도는 O(lg p) 입니다.

BOJ 17603 Factorization

- Step III. mod p에서 이산 제곱근이 존재하지 않는 자연수 z 검색
  - 이산 제곱근의 존재성을 판별하는 방법은 Step I 과 동일합니다.
  - 그런데 가능한 z를 어떻게 찾을 수 있을까요?
  - 최악의 경우 z가 존재하지 않을 수도 있지 않을까요?

BOJ 17603 Factorization

- 다행히 mod p에서 이산 제곱근을 갖는 자연수는 (p−1)/2개 존재합니다.
- 즉 무작위로 z를 골랐을 때 이산 제곱근을 가질 확률은 50% 정도입니다.
- 그래서 그냥 z를 무작위로 골라 확인해 보면 평균적으로 2번의 시행에 이산 제곱근이 존재하지 않는 z를 찾을 수 있습니다.

- Step III의 시간 복잡도는 O(lg p)의 상수 배입니다.

BOJ 17603 Factorization

– Step IV. 알고리즘에 사용할 파라미터 c, r, t 계산

- $c = z^q$, $r = N^{(q+1)/2}$, $t = N^q$ 로 초기화 해 줍니다.

- Step IV의 시간 복잡도는 $O(\lg q)$ 입니다.

  $q < p$이므로 $\lg q < \lg p$ 입니다.

BOJ 17603 Factorization

- c, r, t에 관해 다음의 성질이 성립함을 기억합시다.
  - $c^{1\ll(S-1)} \equiv z^{q(1\ll(S-1))} \equiv z^{(p-1)/2} \equiv -1 \bmod p$
  - $t^{1\ll(S-1)} \equiv N^{q(1\ll(S-1))} \equiv N^{(p-1)/2} \equiv 1 \bmod p$
  - $r^2 \equiv N^{q+1} \equiv Nt \bmod p$

BOJ 17603 Factorization

- Step V. $t \le 1$이 될 때까지 특정 계산을 반복
  - 특정 계산은 다음을 의미합니다.
    - $t^{1 \ll k} \equiv 1$ 을 만족하는 가장 작은 자연수 k를 구합니다.
    - $v \equiv c^{1 \ll (S-k-1)}$ 로 두고, S, c, r, t에 각각 k, v^2, vr, tv^2 을 대입해 줍니다.
    - v를 계산할 때, 지수가 long long 범위를 초과할 수 있습니다. 이때는 지수에 mod p−1을 해 주면 됩니다. (∵페르마의 소정리)

BOJ 17603 Factorization

- 그러면 새로 계산된 수를 각각 k, c', r', t'라고 했을 때 다음이 성립합니다.
  - $c'^{1 \ll (k-1)} \equiv v^{1 \ll k} \equiv c^{1 \ll (S-1)} \equiv -1 \bmod p$
  - $t'^{1 \ll (k-1)} \equiv t^{1 \ll (k-1)} v^{1 \ll k} \equiv (-1)\times(-1) \equiv 1 \bmod p$
  - $r'^2 \equiv r^2v^2 \equiv Ntv^2 \equiv Nt' \bmod p$
- $t^{1 \ll (S-1)} \equiv 1 \bmod p$ 이므로 k의 최댓값은 S−1입니다.

  즉 S는 계산을 실행하면 반드시 감소합니다.
- t'이 1이 되었을 때 $r'^2 \equiv Nt' \equiv N \bmod p$ 입니다.

BOJ 17603 Factorization

- 즉 Step V가 완료되었을 때, r의 값이 바로 N의 이산 제곱근이 됩니다!
- Step V의 계산이 한 번 실행될 때마다 S가 줄어듭니다.
  따라서 계산은 최대 S번 실행됩니다.
- k를 찾는 연산 또한 매번 최대 S번 실행되므로
  최대 $O(S^2)$ 번의 연산이 필요합니다. 이때 $S \leq \lg p$ 입니다.
- Step V의 시간 복잡도는 $O(\lg^2 p)$ 입니다.
- 따라서, 알고리즘의 총 시간 복잡도는 $O(\lg^2 p)$ 입니다.