# MATH 314 - Class Notes

5/11/2017

Scribe: Parker Norwood

**Summary:** Dixon's Factoring Algorithm, Discrete Logarithm(with Baby-step-Giant-Step), Index Calculus, Digital Signatures, DSA, Birthday Attack

**Dixon's Factoring Algorithm:**

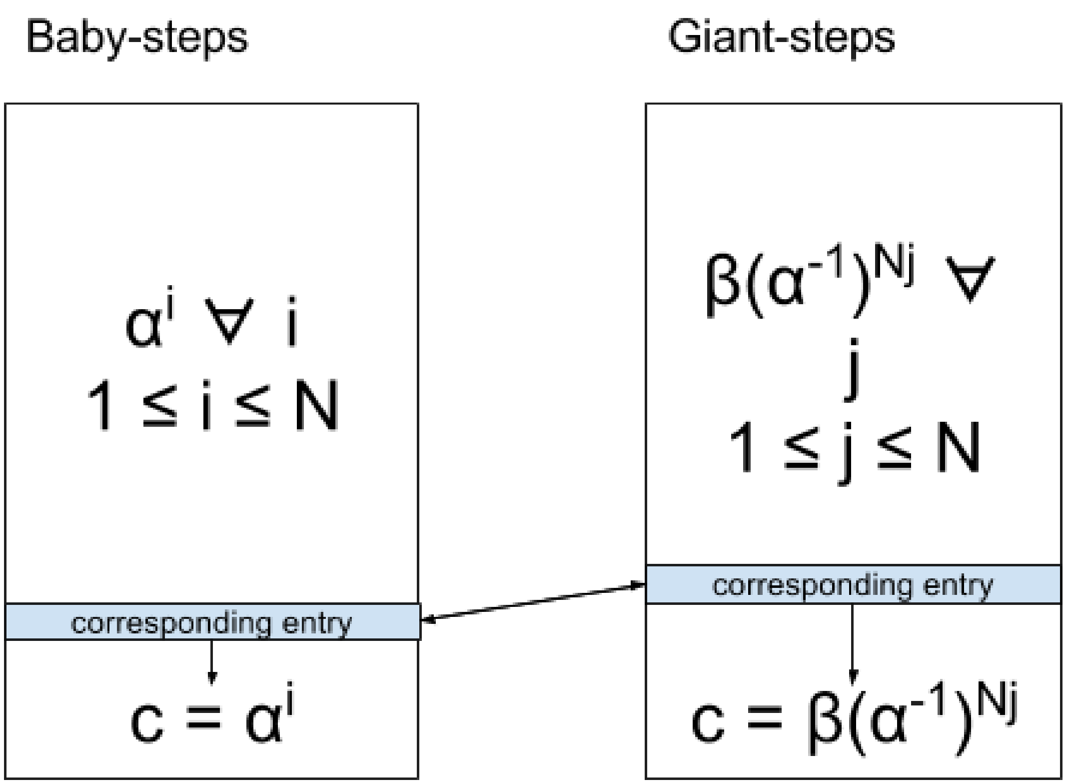
Factor $n = pq$ much faster than naively.

**Discrete Logarithm:**

$\beta = \alpha^x (mod\ p)$
Solve for $x$.

Examples:

- $1^{st}$ Attempt: Try all possible exponents.
  Check if $\alpha^n \equiv \beta(mod\ p)\ \forall\ n \leq p-1$
  If $p$ has 100 digits this is impossibly long.
- $2^{nd}$ Attempt: Baby-step, Giant-step:
  Given $\beta = \alpha^x(mod\ p)$, solve this for $x$.
  Set $N = \lceil \sqrt{p} \rceil$
  Create two tables:



Let's suppose $c$ appears in both tables.
$c \equiv \alpha^i \equiv \beta(\alpha^{-1})^{Nj} (mod\ p)$
So, $x = i + Nj$
This requires $2\sqrt{p}$ steps, much faster than $p - 1$ steps.

Why does this work?

Suppose $\beta = \alpha^x (mod\ p)\ \forall\ x < p - 1 < (\sqrt{p})^2 \leq N^2$
Write $x$ in base $N$
$x = a + bN$
Then let $i = a$, and $j = b$
$c$ should show up in these tables at positions $a$ and $b$ respectively.

**Index Calculus:**

Index Calculus uses the same basic idea as Dixon's Factoring Algorithm.
*Smooth Numbers*: Only using numbers with small prime factors
Using this we get a running time of $O(e^{\sqrt{\ln(p)\ln(\ln(p))}})$
This means you want to use primes with $\approx 200$ digits to be secure.

**Digital Signatures:**

Digital Signatures using the discrete log problem.
ElGamal backwards gives a way to sign messages used in practice

**DSA (Digital Signature Algorithm):**

Introduced by NIST in 1991.
Use two different prime numbers,
$p >> q$, things are kept secure using $p$, but the arithmetic is $(mod\ q)$.

Setup DSA:

Pick $q$ prime (160 bits).
Pick $p$ to be a prime with about 1000 bits.
$p = aq + 1$, Need $p \equiv 1 (mod\ q)$
Let $\alpha$ be a primitive root $(mod\ p)$.
Let $\beta = \alpha^{\frac{p-1}{q}} (mod\ p)$. Note: $a = \frac{p-1}{q}$
Pick a secret $k \in 1 \leq k \leq q - 1$
Let $r = (\alpha^k (mod\ p)(mod\ q))$
Alice's public key: $(p, q, \beta)$

Example:

Alice wants to use this to send a message $m$.
Compute $s = k^{-1}(m + ar)(mod\ q)$
She sends $(m, r, s)$, $r$ and $s$ together are her signature.
Bob wants to verify if Alice's signature is valid
Bob computes:
$U_1 \equiv s^{-1}m(mod\ q)$
$U_2 \equiv s^{-1}r(mod\ q)$
$V = (\alpha^{U_1}\beta^{U_2}(mod\ p))(mod\ q)$
If $V \equiv r(mod\ q)$, Accept Signature, otherwise, it is invalid.

Check that this works:

Since $s \equiv k^{-1}(m + ar)(mod\ q)$
$sk \equiv m + ar(mod\ q)$
$m \equiv sk - ar(mod\ q)$
Multiply both sides by $s^{-1}$
$U_1 : S^{-1}$, and $aU_2 : s^{-1} - ar$
$U_1 \equiv k - aU_2(mod\ q)$
$s^{-1}m \equiv k - s^{-1}ar(mod\ q)$
$k = U_1 + aU_2(mod\ q)$
Now, $r \equiv \alpha^k(mod\ q)$
$\equiv \alpha^{(U_1+aU_2)}(mod\ q)$
$\equiv \alpha^{U_1}(\alpha^a)^{U_2}(mod\ q)$
$\equiv \alpha^{U_1}\beta^{U_2}(mod\ q)$
$\equiv V$
$r \equiv V(mod\ q)$

**Birthday Attack:**

Way to get what appears to be someones signature on a different document.
How many people do you need in a room before two share a birthday?
Suppose we had two people,
Probability that they don't have the same birthday is $1 - \frac{1}{365}$ or $\frac{364}{365}$
With three people: $\frac{364}{365} \cdot \frac{363}{365}$
In general, if we have $k$ people, then the probability that no two people share a birthday is:
$\frac{364}{365} \cdot \frac{363}{365} \cdot \frac{362}{365} \cdot \ldots \cdot \frac{(365-k+1)}{365}$
If $k = 23$, then this probability is: 0.497
$\frac{364}{365} \cdot \frac{363}{365} \cdot \ldots \cdot \frac{343}{365} = 0.497$
If we have $k$ people and $n$ days in the year, then this is approximately $e^{\frac{k^2}{2n}}$
Probability that no two share a birthday is $1 - e^{\frac{k^2}{2n}}$ if $k$ and $n$ are large.