Pollard's ρ and λ Methods for Discrete Logarithms

Halil İbrahim Kanpak

Koç University

May 9, 2025

Motivation: The Need for Pollard's ρ

- The Discrete Logarithm Problem (DLP): Given $P, Q \in G$ (a group of order N), find k such that $Q = kP$.
- Baby Step, Giant Step (BSGS) algorithm solves DLP in $O(\sqrt{N})$ time.
- **Memory Problem with BSGS:**
  - BSGS requires storing $O(\sqrt{N})$ group elements.
  - This can be prohibitive for large N.
- Pollard's ρ method aims to achieve similar $O(\sqrt{N})$ time complexity but with significantly less storage.

The Setup for Pollard's ρ

- Let G be a finite group of order N. We want to solve $Q = kP$.
- Choose a function $f : G \to G$ that behaves "randomly".
- Start with a random element $P_0 \in G$.
- Compute the sequence: $P_{i+1} = f(P_i)$ for $i = 0, 1, 2, \ldots$.
  - $P_0, P_1 = f(P_0), P_2 = f(P_1), \ldots$

Why a "Match" Happens

- Since G is a finite set (with N elements), the sequence P_i must eventually repeat.
- There exist indices $i_0 < j_0$ such that $P_{i_0} = P_{j_0}$.
- Once a match occurs, the sequence becomes periodic:
  - $P_{i_0+1} = f(P_{i_0}) = f(P_{j_0}) = P_{j_0+1}$
  - $P_{i_0+\ell} = P_{j_0+\ell}$ for all $\ell \geq 0$.
- The sequence looks like the Greek letter ρ: a "tail" $P_0, \ldots, P_{i_0-1}$ followed by a "cycle" $P_{i_0}, \ldots, P_{j_0-1}$.

Why Expect a Match in $O(\sqrt{N})$ Steps?

- This is related to the "Birthday Paradox".
- If f is a randomly chosen function, we expect to find a match $P_{i_0} = P_{j_0}$ with j_0 being at most a constant times $\sqrt{N}$.
- The length of the tail (i_0) and the cycle ($j_0 - i_0$) are also expected to be $O(\sqrt{N})$.
- Total steps to find a match: $j_0 = O(\sqrt{N})$.

Naive Implementation: Space Issue

The Obvious (but inefficient) Way

- Store all computed points $P_0, P_1, P_2, \ldots$ in a list or hash table.
- After computing each P_i, check if it's already in the list.
- If P_i is found, we have a match.

Problem: $O(\sqrt{N})$ Space Complexity

- We expect to store approximately $j_0 = O(\sqrt{N})$ points before a match.
- This leads to $O(\sqrt{N})$ space complexity.
- This is the same space complexity as BSGS, which we wanted to avoid!

Tortoise and Hare: Finding a Match with $O(1)$ Space

- Idea: Use two "pointers" (the tortoise and the hare) moving through the sequence at different speeds.
  - Tortoise: $X_t \leftarrow f(X_t)$ (moves one step at a time)
  - Hare: $X_h \leftarrow f(f(X_h))$ (moves two steps at a time)
- Using the sequence P_i: compute pairs (P_i, P_{2i}).

$$
\begin{aligned}
P_{i+1} &= f(P_i) \\
P_{2(i+1)} &= f(f(P_{2i}))
\end{aligned}
$$

- A match $P_i = P_{2i}$ will be found for some i.
- This occurs when i is a multiple of the cycle length $d = j_0 - i_0$, and $i \geq i_0$.
- The first such i is $\leq j_0$. So, $O(\sqrt{N})$ computations.
- Crucially, only the current pair (P_i, P_{2i}) needs to be stored: $O(1)$ space.

Distinguished Points

- Define a "distinguished property": a property that points satisfy with some probability p.
  - E.g., for a point $P_i = (x, y)$, require the last k bits of x to be 0.
  - Probability $p \approx 1/2^k$.
- Only store points P_i that are "distinguished".
  - On average, store one out of every $1/p$ (e.g., 2^k) points.
  - Storage reduced by factor $1/p$.
- If a match $P_i = P_j$ occurs:
  - P_i and P_j might not be distinguished.
  - However, $P_{i+\ell} = P_{j+\ell}$ for $\ell \geq 0$.
  - We expect $P_{i+\ell}$ to be a distinguished point for some small ℓ (approx. $1/p$ on average).
  - So, a match between distinguished points will be found with a little more computation.

Constructing the Function f

- We need f to not only create a collision but also allow us to extract k.
- To solve $Q = kP$ in group G of order N.
- Divide G into s disjoint subsets $S_1, \ldots, S_s$ (e.g., $s \approx 20$).
- Choose $2s$ random integers $a_j, b_j \pmod N$.
- Define "step" elements $M_j = a_j P + b_j Q$.
- Define $f(g)$ based on which subset g belongs to:

$$f(g) = g + M_j \quad \text{if } g \in S_j$$

- Choose random a_0, b_0 and start with $P_0 = a_0 P + b_0 Q$.
- Keep track of how each P_i is expressed in terms of P and Q: If $P_i = u_i P + v_i Q$, and $P_{i+1} = P_i + M_j$ (because $P_i \in S_j$), then $P_{i+1} = (u_i P + v_i Q) + (a_j P + b_j Q)$. So, $u_{i+1} = (u_i + a_j) \pmod N$ and $v_{i+1} = (v_i + b_j) \pmod N$.

Using the Match to Find k

- Suppose we find a match $P_{j_0} = P_{i_0}$ (with $i_0 < j_0$).
- We have:

$$P_{i_0} = u_{i_0} P + v_{i_0} Q$$
$$P_{j_0} = u_{j_0} P + v_{j_0} Q$$

- Since $P_{j_0} = P_{i_0}$:

$$u_{j_0} P + v_{j_0} Q = u_{i_0} P + v_{i_0} Q$$

- Rearranging gives:

$$(u_{j_0} - u_{i_0})P = (v_{i_0} - v_{j_0})Q$$

- Substitute $Q = kP$:

$$(u_{j_0} - u_{i_0})P = (v_{i_0} - v_{j_0})kP$$

- This implies (modulo N, the order of P):

$$(u_{j_0} - u_{i_0}) \equiv (v_{i_0} - v_{j_0})k \pmod{N}$$

Using the Match to Find k

- Let $U = u_{j_0} - u_{i_0}$ and $V = v_{i_0} - v_{j_0}$. We need to solve $U \equiv Vk \pmod{N}$.

  $$k \equiv V^{-1}U \pmod{N/d}$$

  where $d = \gcd(V, N)$.
- This gives d possible values for k. Usually d is small.
- If N is prime (common in cryptography):
  - If $V \not\equiv 0 \pmod{N}$, then $d = 1$, unique k.
  - If $V \equiv 0 \pmod{N}$:
    - If $U \equiv 0 \pmod{N}$ too, it's a trivial relation. Start over.
    - If $U \not\equiv 0 \pmod{N}$, no solution for k. This implies P is identity, usually not the case. (Or error in calculation).

Example of Pollard's ρ

- Group $G = E(\mathbb{F}_{1093})$, elliptic curve $y^2 = x^3 + x + 1$.
- $P = (0, 1)$, $Q = (413, 959)$. Order of P is $N = 1067$ (prime). Find k for $Q = kP$.
- Use $s = 3$ subsets. Let M_0, M_1, M_2 be predefined linear combinations of P, Q.
  - E.g., $M_0 = 3P + 5Q$, $M_1 = 4P + 3Q$, $M_2 = 9P + 17Q$.
- Define $f(X, Y) = (X, Y) + M_i$ if $X \equiv i \pmod 3$.
- Start with $P_0 = a_0P + b_0Q$. (Text: $P_0 = (326, 69)$).
- Sequence $P_0, P_1 = f(P_0), \ldots$. A match $P_5 = P_{58}$ is found.
  - $P_5 = (1006, 951)$
  - $P_{58} = (1006, 951)$

Example of Pollard's ρ

- Coefficients are tracked:

$$P_5 = 88P + 46Q$$
$$P_{58} = 685P + 620Q$$

- From $P_{58} = P_5$, we get $P_{58} - P_5 = \mathcal{O}$ (identity element):

$$(685 - 88)P + (620 - 46)Q = \mathcal{O}$$

$$597P + 574Q = \mathcal{O}$$

- So, $574Q = -597P \pmod{1067}$.

$$kP = Q \implies 574kP = -597P \pmod{1067}$$

$$574k \equiv -597 \pmod{1067}$$

- $k \equiv (-597) \cdot (574^{-1}) \pmod{1067}$.

Example of Pollard's ρ: Solving for k

- We arrived at the congruence:

$$574k \equiv -597 \pmod{1067}$$

- The inverse of 574 modulo 1067 is given or computed as:

$$574^{-1} \equiv 303 \pmod{1067}$$

- Multiply both sides of the congruence by 574^{-1}:

$$k \equiv (-597) \cdot (574^{-1}) \pmod{1067}$$

$$k \equiv (-597) \cdot 303 \pmod{1067}$$

So, we have:

$$k \equiv -180891 \pmod{1067}$$

- Reduce -180891 modulo 1067:

$$k \equiv 499 \pmod{1067}$$

Example of Pollard's ρ: Conclusion

- The discrete logarithm is $k = 499$. Thus $Q = 499P$.
- The key steps were:
  1. Generating a sequence $P_i = u_i P + v_i Q$.
  2. Finding a collision $P_{i_0} = P_{j_0}$.
  3. Using the coefficients to form $(u_{j_0} - u_{i_0})P + (v_{j_0} - v_{i_0})Q = \mathcal{O}$.
  4. Rearranging to $(u_{j_0} - u_{i_0})P = (v_{i_0} - v_{j_0})Q$.
  5. Substituting $Q = kP$ to get $(u_{j_0} - u_{i_0}) \equiv (v_{i_0} - v_{j_0})k \pmod{N}$.
  6. Solving the linear congruence for k.

Tortoise and Hare: Deriving the Relation from $P_{53} = P_{106}$

- Floyd's cycle-finding algorithm (Tortoise and Hare) searches for a match $P_i = P_{2i}$.
- The provided text states that for $i = 53$, a match $P_{53} = P_{106}$ is found.
- Let's derive the algebraic relation from this specific collision.
- The coefficients for these points are given in the text as:

$$P_{53} = 620P + 557Q$$
$$P_{106} = 1217P + 1131Q$$

- Since $P_{53} = P_{106}$, it follows that $P_{106} - P_{53} = \mathcal{O}$ (the identity element of the group).
- Substituting the expressions for P_{106} and P_{53}:

$$(1217P + 1131Q) - (620P + 557Q) = \mathcal{O}$$

- Now, group the terms involving P and Q:

$$(1217 - 620)P + (1131 - 557)Q = \mathcal{O}$$

- This yields the relation:

$$\mathbf{597P} + \mathbf{574Q} = \mathcal{O}$$

Tortoise and Hare: Deriving the Relation from $P_{53} = P_{106}$

- This is exactly the same algebraic relation ($597P + 574Q = \mathcal{O}$) that was obtained from the collision $P_5 = P_{58}$ in the naive storage method.
- Therefore, solving this relation for k (where $Q = kP$) will lead to the same result:

$$574Q = -597P \implies 574kP = -597P \implies 574k \equiv -597 \pmod{1067}$$

Which, as previously shown, gives $k = 499$.

Pollard's λ (Kangaroo) Method

- A variation of the ρ method.
- Uses the same type of random-looking function $f : G \to G$.
- Multiple random starting points: $P_0^{(1)}, P_0^{(2)}, \ldots, P_0^{(r)}$.
- Generate r sequences in parallel: $P_{i+1}^{(\ell)} = f(P_i^{(\ell)})$.
- Often uses distinguished points, which are reported to a central computer.
- When a match is found between points from *different* sequences (or one sequence with itself), say $P_i^{(a)} = P_j^{(b)}$, we get a relation.
- If $P_i^{(a)} = u_i^{(a)} P + v_i^{(a)} Q$ and $P_j^{(b)} = u_j^{(b)} P + v_j^{(b)} Q$, then

  $$(u_i^{(a)} - u_j^{(b)})P = (v_j^{(b)} - v_i^{(a)})Q$$

  From which k can be found.
- With two starting points ($r = 2$), the paths resemble the Greek letter λ when they collide.

Conclusion

Pollard's ρ and λ Methods

- Time Complexity: $O(\sqrt{N})$ group operations (expected).
- Space Complexity:
  - Naive: $O(\sqrt{N})$.
  - With Floyd's cycle-finding: $O(1)$.
  - With distinguished points: Tunable, e.g., $O(\sqrt[4]{N})$ or less.
- These methods are **probabilistic**:
  - High probability of success within the expected time.
  - Not guaranteed (unlike BSGS, which is deterministic).
  - May need to restart with a different f or P_0 if a trivial relation is found or it runs too long.
- Significant improvement over BSGS in terms of memory, making them practical for larger groups.