Raising all group elements to a common power

HUI-TING CHEN, CHING-LUEH CHANG
Department of Computer Science and Engineering
Yuan Ze University
No. 135, Yuandong Rd., Zhongli Dist., Taoyuan City Taiwan (R.O.C.)
TAIWAN

Abstract: - We give a deterministic $O(|G|)$-time algorithm that, given the multiplication table of a finite group $(G, \cdot)$ and nonzero p, $q \in \mathbb{Z}$, finds all solutions (if any) to $x^p = g^q$ for all $g \in G$.

Key-Words: - inverting element, group, multiplication table and power.

Received: July 4, 2022. Revised: January 5, 2023. Accepted: February 4, 2023. Published: March 2, 2023.

1 Introduction

Many properties of a group-like structure can be discovered from its multiplication table. Zumbrägel et al., [1] , consider the problem of learning the multiplication table of a groupoid $(G, \cdot)$ by making the minimum number of queries, each for a product $a \cdot b$, with $a, b \in G$. An interesting problem is to determine algebraic properties of a finite group G from $\Psi(G) = \sum_{g \in G} o(g)$, where $o(g)$ denotes the order of $g \in G$, [2]–[5]. Jahani et al., [6], find a pair of finite groups G and S of the same order such that $\Psi(G) < \Psi(S)$, with G solvable and S simple.

Now we are interested in efficiently finding a given power of all elements simultaneously. By convention, the multiplication in G costs $O(1)$ time. Let G be a group with n elements. If we want to calculate the qth power of each element, how long does it take? The brute force method takes $O(q)$ time to calculate the qth power of an element. So the total time is $O(nq)$.

Recursive doubling method reduces the time required to calculate the qth power of an element to $O(\log q)$, so the total time can be reduced to $O(n \log q)$. Kavitha, [7], presents an $O(n)$ algorithm that determines if two Abelian groups with n elements each are isomorphic. Similar research can see, [8] and [9]. The main ingredient in this result is an $O(|G|)$-time algorithm that finds the orders of all elements in any finite group G given as input the multiplication table of G. Inspired by Kavitha's result, we give a deterministic $O(|G|)$-time algorithm that, given the multiplication table of a finite group $(G, \cdot)$ and nonzero p, $q \in \mathbb{Z}$, finds all solutions (if any) to $x^p = g^q$ for all $g \in G$.

Primitive roots are elements of order $|G|$ and have been extensively studied. See, e.g., [10]. To find the solutions to $x^p = g^q$ for each $g \in G$, it suffices to do the following:

(1) Calculate g^q for each $g \in G$.

(2) Find a primitive root r and calculate $r^1, r^2, \ldots, r^{|G|}$. When some r^j matches any value calculated in step 1, a solution for $x^p = g^q$ is found.

Unlike in our result, however, the above procedure takes $\omega(|G|)$ time.

2 Preliminaries

We refer to some basic definitions in algebra, [11].For more detail, please see, [12] and [13].

Definition 1. *A nonempty set G endowed with a binary operation $\cdot$, $G \cdot G \to G$ is called a groupoid. An element $e \in G$ is an identity if and only if for all $x \in G$, $x \cdot e = e \cdot x = x$. If y has a unique inverse, it's denoted y^{-1}.*

Definition 2. *A groupoid $(G, \cdot)$ is*

- *Abelian if $x \cdot y = y \cdot x$ for all $x, y \in G$.*
- *associative if $x \cdot (y \cdot z) = (x \cdot y) \cdot z$ for all $x, y, z \in G$.*
- *a quasigroup if for all $x, y \in G$, there are unique elements $a, b \in G$ such that $x \cdot a = y$ and $b \cdot x = y$.*
- *a loop if $(G, \cdot)$ is a quasigroup with an identity.*

Definition 3. *The order of a finite group $(G, \cdot)$ refers to the number of elements of G. The order of an element a in a finite group $(G, \cdot)$ refers to the least positive integer h which satisfies $a^h = e$, where e is the identity of $(G, \cdot)$.*

Input: The multiplication table of a group $(G, \cdot)$ and $q \in \mathbb{Z}^+$

1: Compute g^{-1} for all $g \in G$;
2: Compute the order of g, denoted order(g), for all $g \in G$;
3: **for all** $g \in G$ **do**
4:   ans$[g] \leftarrow \perp$;
5: **end for**
6: **for** $\ell = 1, 2, \ldots, |G|$ **do**
7:   $g \leftarrow$ the ℓth element of G;
8:   **if** ans$[g] = \perp$ **then**
9:     $k \leftarrow \min\{q \bmod \text{order}(g)\} \cup \{i \geq 2 \mid (\text{ans}[g^{i-1}] \in G) \wedge (\text{ans}[g^i] \in G)\}$;
10:     Calculate $g, g^2, \ldots, g^k$;
11:     **if** $k = (q \bmod \text{order}(g))$ **then**
12:       ans$[g] \leftarrow g^k$;
13:     **else**
14:       ans$[g] \leftarrow$ ans$[g^k] \cdot (\text{ans}[g^{k-1}])^{-1}$;
15:     **end if**
16:     **for** $j = 2, 3, \ldots, k-1$ **do**
17:       ans$[g^j] \leftarrow$ ans$[g^{j-1}] \cdot$ ans$[g]$;
18:     **end for**
19:   **end if**
20: **end for**

Figure 1: Algorithm All Powers outputting g^q, stored in ans$[g]$, for all $g \in G$

Definition 4. *For any finite group $(G, \cdot)$, we say $(H, \cdot)$ is a subgroup of $(G, \cdot)$ if $H \subseteq G$ and for any $x, y \in H$, $x \cdot y \in H$.*

3 Raising powers

To begin with, we check that ans$[g^k] \in G$ and ans$[g^{k-1}] \in G$ in line 14 of algorithm All Powers in Fig. 1; hence line 14 tries neither to invert $\perp$ nor to multiply a group element with $\perp$.

Lemma 5. *In line 14 of* All Powers*,* ans$[g^{k-1}] \in G$ *and* ans$[g^k] \in G$*.*

Proof. Clearly, $k \neq q$ in line 14. So line 9 implies the lemma. □

Lemma 6. *At any time,* ans$[a] = a^q$ *for all* $a \in G$ *satisfying* ans$[a] \neq \perp$*.*

Proof. Assume as induction hypothesis that the lemma is true up to the $(\ell-1)$th iteration of the **for** loop in lines 6–20, where $\ell \geq 1$. In the ℓth iteration:

- As $g^{q \bmod \text{order}(q)} = g^q$, line 12 maintains the lemma.
- Upon reaching line 14, ans$[g^{k-1}] \in G$ and ans$[g^k] \in G$ by Lemma 5, implying ans$[g^{k-1}] = (g^{k-1})^q$ and ans$[g^k] = (g^k)^q$ by the induction hypothesis (note that ans$[g^{k-1}]$ and ans$[g^k]$ are not yet modified in the current iteration). So line 14 calculates ans$[g]$ as g^q.
- Upon reaching Line 17, we must have just run line 12 or line 14, resulting in ans$[g] = g^q$ by the analyses above. So lines 16–18 calculate ans$[g^j]$ as $(g^j)^q$ for all $2 \leq j \leq k-1$.

In summary, the lemma remains true after the ℓth iteration.

The base case that $\ell = 0$ is trivial because ans$[g] = \perp$ for all $g \in G$ before the first iteration. □

Lemma 7. *After running* All Powers*,* ans$[g] = g^q$ *for all* $g \in G$*.*

Proof. Lines 11–15 and Lemma 5 guarantee ans$[g] \neq \perp$. So the loop in lines 6–20 ends up guaranteeing ans$[g] \neq \perp$ for all $g \in G$. Now apply Lemma 6. □

Lemma 8. *Each execution of lines 8–19 of* All Powers *take $O(k)$ time, where k is as in line 8.*

Proof. Run line 9 by calculating g^i for an increasing $i \geq 1$ until either (1) $i = q \bmod \text{order}(g)$ or (2) ans$[g^{i-1}] \neq \perp$ and ans$[g^i] \neq \perp$. Because $g^i = g^{i-1} \cdot g$ for all i, line 8 takes $O(k)$ time. Similarly, line 9 also takes $O(k)$ time. Clearly, lines 11–15 and 16–18 take $O(1)$ and $O(h)$ time, respectively (note that the inverse $(\text{ans}[g^{k-1}])^{-1}$ in line 14 has been found in line 1). □

Lemma 9. *Each execution of lines 9–18 of* All Powers *turn $\Omega(k)$ entries of* ans$[\cdot]$ *from $\perp$ to non-$\perp$.*

Proof. By the minimality of k in line 9, the sequence $\{\text{ans}[g^j]\}_{j=1}^{k-1}$ does not contain two consecutive elements that are non-$\perp$ (when line 9 is executed). So $\perp$ appears for at least $\lfloor (k-1)/2 \rfloor$ times in $\{\text{ans}[g^j]\}_{j=1}^{k-1}$. But after lines 11–19, ans$[g^j] \neq \perp$ for all $j \in \{1, 2, \ldots, k-1\}$. Note that as $k < \text{order}(g)$ by line 9, g^1, g^2, …, g^{k-1} are distinct. In summary, lines 9–18 turn at least $\lfloor (k-1)/2 \rfloor$ distinct entries of ans$[\cdot]$ from $\perp$ to non-$\perp$. Unless $k \leq 2$, $\lfloor (k-1)/2 \rfloor = \Omega(k)$. When $k \leq 2$, the lemma still holds because lines 11–15 turn ans$[g]$ from $\perp$ to non-$\perp$. □

Lemma 10. All Powers *take $O(|G|)$ time.*

Proof. Appendix A proves the easy, probably folklore, result that line 1 takes $O(|G|)$ time. Kavitha [7] gives an $O(|G|)$-time algorithm for line 2. Clearly, once an entry of ans$[\cdot]$ becomes non-$\perp$, it remains non-$\perp$ forever. So by Lemmas 8–9, the running time is at most proportional to the total number of entries of ans$[\cdot]$, which is $|G|$. □

Lemma 11. *Given the multiplication table of a finite group $(G, \cdot)$ and a nonzero $q \in \mathbb{Z}$, it takes $O(|G|)$ time to find g^q and all qth roots (if any) of g, for all $g \in G$.*

Proof. There are several cases:

- $q \geq 2$: By Lemmas 7 and 10, finding g^q for all $g \in G$ takes $O(|G|)$ time. Create a list L_a for each $a \in G$. For each $g \in G$, put g into L_{g^q}. Then the qth roots of each $a \in G$ are just the elements of L_a.
- $q = 1$: Trivial.
- $q < 0$: Find g^{-1} for all $g \in G$ in $O(|G|)$ time, as in Appendix A. Replace q by $-q \geq 1$ and each $g \in G$ by g^{-1}. Then proceed as if $q > 0$.

□

Below is our main result.

Theorem 12. *Given the multiplication table of a finite group $(G, \cdot)$ and nonzero $p, q \in \mathbb{Z}$, it takes $O(|G|)$ time to find all solutions (if any) to $x^p = g^q$ for all $g \in G$.*

Proof. Use Lemma 11 twice to find g^q and all pth roots (if any) of g, for all $g \in G$. □

4 Conclusion

If we want to find the power of a finite group G given the multiplication table, we give the optimal algorithm that takes $O(|G|)$ time to find all solutions (if any) to $x^p = g^q$ for all $g \in G$. And we use this method to invert all elements in G.

A Inverting all elements

We begin by verifying that algorithm All Inverses in Fig. 2 performs only reasonable operations. In particular, line 12 does not try to multiply a group element with $\perp$.

Lemma 13. *In line 12 of* All Inverses, inv$[g^h] \in G$.

Proof. By lines 9 and 11, $g^h \neq 1$ in line 12. So line 7 implies the lemma. □

Input: The multiplication table of a group $(G, \cdot)$
1: **for all** $g \in G$ **do**
2: inv$[g] \leftarrow \perp$;
3: **end for**
4: **for** $\ell = 1, 2, \ldots, |G|$ **do**
5: $g \leftarrow$ the ℓth element of G;
6: **if** inv$[g] = \perp$ **then**
7: $h \leftarrow \min\{i \geq 1 \mid (g^i = 1) \vee (\text{inv}[g^i] \in G)\}$;
8: Calculate $g, g^2, \ldots, g^h$;
9: **if** $g^h = 1$ **then**
10: inv$[g] \leftarrow g^{h-1}$;
11: **else**
12: inv$[g] \leftarrow g^{h-1} \cdot \text{inv}[g^h]$;
13: **end if**
14: **for** $j = 2, 3, \ldots, h-1$ **do**
15: inv$[g^j] \leftarrow \text{inv}[g^{j-1}] \cdot \text{inv}[g]$;
16: **end for**
17: **end if**
18: **end for**

Figure 2: Algorithm All Inverses outputting g^{-1}, stored in inv$[g]$, for all $g \in G$

Lemma 14. *At any time,* inv$[a] = a^{-1}$ *for all $a \in G$ satisfying* inv$[a] \neq \perp$.

Proof. Assume as induction hypothesis that the lemma is true up to the $(\ell - 1)$th iteration of the **for** loop in lines 4–18, where $\ell \geq 1$. In the ℓth iteration:

- Line 10 clearly maintains the lemma.
- Upon reaching line 12, inv$[g^h] \in G$ by Lemma 13, implying inv$[g^h] = (g^h)^{-1}$ by the induction hypothesis. So line 12 calculates inv$[g]$ as g^{-1}.
- Upon reaching Line 15, we must have just run line 10 or line 12, resulting in inv$[g] = g^{-1}$ by the analyses above. So lines 14–16 calculate inv$[g^j]$ as $(g^j)^{-1}$ for all $2 \leq j \leq h-1$.

In summary, the lemma remains true after the ℓth iteration.

The base case that $\ell = 0$ is trivial because inv$[g] = \perp$ for all $g \in G$ before the first iteration. □

Lemma 15. *After running* All Inverses*,* inv$[g] = g^{-1}$ *for all $g \in G$.*

Proof. Lines 9–13 and Lemma 13 guarantee inv$[g] \neq \perp$. So the loop in lines 4–18 ends up guaranteeing inv$[g] \neq \perp$ for all $g \in G$. Now apply Lemma 14. □

Lemma 16. *Each execution of lines 7–16 of* All Inverses *take $O(h)$ time, where h is as in line 7.*

Proof. Run line 7 by calculating g^i for an increasing $i \geq 1$ until either (1) $g^i = 1$ or (2) $\text{inv}[g^i] \neq \perp$. Because $g^i = g^{i-1} \cdot g$ for all i, line 7 takes $O(h)$ time. Similarly, line 8 also takes $O(h)$ time. Clearly, lines 9–13 and 14–16 take $O(1)$ and $O(h)$ time, respectively. □

Lemma 17. *Each execution of lines 7–16 of* All Inverses *turn* $\Omega(h)$ *entries of* $\text{inv}[\cdot]$ *from* $\perp$ *to non-*$\perp$.

Proof. By the minimality of h in line 7, $\text{inv}[g^j] = \perp$ for $1 \leq j \leq h-1$ (when line 7 is executed). But after lines 9–16, $\text{inv}[g^j] \neq \perp$ for all $j \in \{1, 2, \ldots, h-1\}$. So lines 7–16 turn at least $h-1$ entries of $\text{inv}[\cdot]$ from $\perp$ to non-$\perp$. Unless $h \leq 1$, $h-1 = \Omega(h)$. When $h \leq 1$, the lemma still holds because lines 9–13 turn $\text{inv}[g]$ from $\perp$ to non-$\perp$. □

Lemma 18. All Inverses *take* $O(|G|)$ *time.*

Proof. Clearly, once an entry of $\text{ans}[\cdot]$ is non-$\perp$, it remains non-$\perp$ forever. So by Lemmas 16–17, the running time is at most proportional to the total number of entries of $\text{ans}[\cdot]$, which is $|G|$. □

Lemmas 15 and 18 yield the following.

Theorem 19. *Finding* g^{-1} *for all* $g \in G$ *takes* $O(|G|)$ *time.*

References

[1] N. Suvorov and N. Kryuchkov, “Examples of some quasigroups and loops admitting only discrete topologization,” *Siberian Mathematical Journal*, vol. 17, no. 2, pp. 367–369, 1976.

[2] H. Amiri and S. Jafarian Amiri, “Sum of element orders of maximal subgroups of the symmetric group,” *Communications in Algebra*, vol. 40, no. 2, pp. 770–778, 2012.

[3] H. Amiri and S. Jafarian Amiri, “Sum of element orders on finite groups of the same order,” *Journal of Algebra and its Applications*, vol. 10, no. 02, pp. 187–190, 2011.

[4] Y. Marefat, A. Iranmanesh, and A. Tehranian, “On the sum of element orders of finite simple groups,” *Journal of Algebra and its Applications*, vol. 12, no. 07, p. 1 350 026, 2013.

[5] M. Tărnăuceanu and D. G. Fodor, “On the sum of element orders of finite abelian groups,” *arXiv preprint arXiv:1805.11693*, 2018.

[6] M. Jahani, Y. Marefat, H. Refaghat, and B. Vakili Fasaghandisi, “The minimum sum of element orders of finite groups,” *International Journal of Group Theory*, vol. 10, no. 2, pp. 55–60, 2021.

[7] T.Kavitha, “Linear time algorithms for Abelian group isomorphism and related problems,” *Journal of Computer and System Sciences*, vol. 73, no. 6, pp. 986–996, 2007.

[8] C. D. Savage, *An* $O(n^2)$ *algorithm for abelian group isomorphism*. Computer Studies [Program], North Carolina State University, 1980.

[9] N. Vikas, “An o(n) algorithm for abelianp-group isomorphism and an o(nlogn) algorithm for abelian group isomorphism,” *journal of computer and system sciences*, vol. 53, no. 1, pp. 1–9, 1996.

[10] V. Edemsky and W. CHENHUANG, “On the linear complexity of binary sequences derived from generalized cyclotomic classes modulo (2n)(pm),” *WSEAS Transactions on Mathematics*, vol. 18, pp. 197–202, 2019.

[11] D. S. Dummit and R. M. Foote, *Abstract algebra*. Prentice Hall Englewood Cliffs, NJ, 1991, vol. 1999.

[12] M. Hall, *The theory of groups*. Courier Dover Publications, 2018.

[13] I. N. Herstein, *Topics in algebra*. John Wiley & Sons, 2006.

Contribution of individual authors to the creation of a scientific article (ghostwriting policy)

Ching-Lueh Chang carried out the conceptualization and is the supervisor.
Hui-Ting Chen did the data curation and has writing and editing.

Sources of funding for research presented in a scientific article or scientific article itself

This work was Supported in part by the Ministry of Science and Technology of Taiwan under grant 111-2221-E-155-035-MY2.

Conflict of Interest

The authors have no conflicts of interest to declare that are relevant to the content of this article.

Creative Commons Attribution License 4.0 (Attribution 4.0 International , CC BY 4.0)