Journal of Discrete Mathematical Sciences & Cryptography
ISSN 0972-0529 (Print), ISSN 2169-0065 (Online)
Vol. 29 (2026), No. 10-B, pp. 4321–4331
DOI : 10.47974/JDMSC-3054

Efficient scalar multiplication techniques on elliptic curves

Bhawani Sankar Panigrahi [†]
Department of Computer Science & Engineering
GITAM School of Computer Science and Engineering
GITAM Deemed to be University
Vishakhapatnam 530045
Andhra Pradesh
India

Rahul Kumar Singh *
Department of Computer Science & Engineering
Noida International University
Greater Noida 203201
Uttar Pradesh
India

Praney Madan [§]
Centre for Research Impact and Outcome
Chitkara University
Rajpura 140417
Punjab
India

Nyalam Ramu [‡]
Centre for Multidisciplinary Research
Anurag University
Hyderabad 500088
Telangana
India

[†] *E-mail:* bspanigrahi.cse@gmail.com
* *E-mail:* rahul.singh1@niu.edu.in (Corresponding Author)
[§] *E-mail:* praney.madan.orp@chitkara.edu.in
[‡] *E-mail:* nyalamramu.nr@proton.me

© Taru Publications

Anisha Chaudhary [@]
Quantum University Research Center
Quantum University
Roorkee 247662
Uttarakhand
India

Manoj I. Patel [#]
Faculty of Computer Science and Applications
Gokul Global University
Sidhpur 384151
Gujarat
India

Abstract

The elliptic curve cryptography is based upon the efficient scalar multiplication, which directly affects the performance and security of a calculation. The article is an in-depth analysis of the advanced techniques of multiplication of scalars to reduce time complexity and resource consumption. It analyzes algorithmic optimization strategies, including, but not limited to, precomputation techniques, windowing techniques and memorycomputation trade-offs. The parallel processing and pipelining are addressed to enhance the throughput features of hardware and software programs. The use of the GLV and GLS methods of endomorphisms are examined with the aim of hastening the breakdown of scalars. The binary method and its computational complexity is also taken into consideration along with the optimized point addition and doubling algorithms in the context of the group law. According to the experimental findings, the time spent on execution, the amount of energy used, and the scalability have improved positively, which demonstrates the applicability of the proposed optimizations to the recent cryptographic systems.

Subject Classification: *68P25.*

Keywords: *Elliptic curve cryptography, Scalar multiplication, GLV/GLS methods, Parallel processing, Point doubling, Precomputation techniques.*

I. Introduction

Elliptic Curve Cryptography (ECC) has become a critical component of the existing secure communication systems, being able to provide a high degree of security with an exceptionally small key sizes [1, 2]. The simplest operation of ECC is scalar multiplication, the computation of kP, with k an integer, and P a point on the elliptic curve [3, 4, 5]. This operation is the most common in the overall computation cost of cryptographic algorithms such as key exchange, digital signature and encryption [6, 7]. In this way,

[@] *E-mail:* **anisha.chaudhary@quantumeducation.in**
[#] *E-mail:* **manoj.patel@gokuluniversity.ac.in**

improving the performance of scalar multiplication is critical in improving the performance of a system, particularly with resource-constrained systems such as embedded systems and Internet of Things devices [8, 9]. A number of algorithms have been proposed to maximize scalar multiplication, including binary algorithms, window algorithms, and precomputation algorithms, which are memory-speed tradeoffs [10, 11].

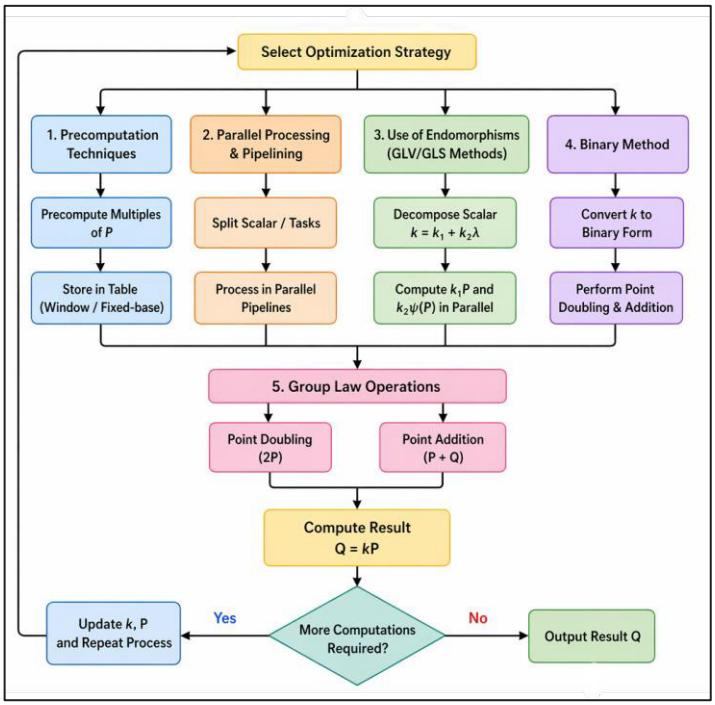


Figure 1

Flowchart of Efficient Scalar Multiplication Techniques on Elliptic Curves

A more sophisticated method such as endomorphism-based decompositions (GLV/GLS) is faster to compute the scalar, subdividing the scalar into smaller components. Figure 1 illustrates an optimized process of scalar multiplication, parallel processing. In addition, the good implementation of point addition and point doubling are also significant issues that influence performance.

II. Algorithmic Optimization Strategies

Algorithms Optimizations based on scalar multiplication, with windowing, recoding and efficient arithmetic operations make trade-offs between speed, storage and computational overhead between a set of elliptic curve cryptographic algorithms [12].

A. *Precomputation Techniques and Memory Trade-Offs*

Base points are pre-computed in multiples to accelerate repeated calculations, with the cost of increased memory consumption, and is particularly important when multiplication of fixed bases is required.

$$P_{set} = \{P, 2P, 3P, \dots, (2^w - 1)P\} = sqrt\left(\left(sum^2_{\{i=1\ to\ (2^w-1)\}||iP||}\right)^2 + \int_0^1((iP * t + t^2)^2 dt)\right) \quad (1)$$

Equation (1) defines this precomputation set, and calculates multiple times base point P, at the expense of additional storage and a reduced run time.

$$k = sum_{\{j=0\ to\ l-1\}(k_j * 2^{j*w})} = sqrt\left(\left(sum^2_{\{j=0\ to\ l-1\}(k_j * 2^{j*w})}\right)^2 + \int_0^1 \left(\left(k_j * 2^{j*w} * t + t^2\right)^2 dt\right)\right) \quad (2)$$

The scalar decomposition with windowing can be described as in equation (2) where k is broken down into smaller functions and it can be easily computed using fewer steps and more effective utilization of pre-calculated values.

$$T(M) = sqrt\left(\left(\frac{C_{comp}}{2^w}\right)^2 + \int_0^1((M * t + t^2)^2 dt)\right) \quad (3)$$

Equation (3) is the time-memory trade-off, which implies that the less the memory assignment, the less the expenses of the computation, and the performance efficiency of the optimized implementations are traded with the needs of storage.

$$O_{pre} = \min sqrt\left(((T(M)^2 + M^2_{cost})^2) + \int_0^1((T(M) * t + M_{cost} * t^2)^2 dt)\right) \quad (4)$$

The optimization problem is presented in equation (4) and is to maximize the total cost in terms of computational time and memory of elliptic curve cryptographic system to achieve an optimum tradeoff of an efficient scalar multiplication [13].

B. *Parallel Processing and Pipelining Approaches*

Parallelism distributes computations and pipelining overlaps computation phases to implement much higher throughput and lower latency in hardware and high-performance ECC implementations.

$$k = sum_{\{i=1\ to\ p\}k_i} = sqrt\left(\left(sum_{\{i=1\ to\ p\}k_i^2}\right)^2 + \int_0^1((k_i * t + t^2)^2 dt)\right) \quad (5)$$

An example of scalar partitioning is given in equation (5): k can be decomposed into smaller, independent components, and can be computationally partitioned to save time on the overall computation time and to offer a more efficient processing.

$$Q_i = k_i * P = sqrt\left(((k_i * P)^2)^2 + \int_0^1 \left(\left((k_i * P) * t + t^3\right)^2 dt\right)\right) \quad (6)$$

Parallel point multiplication is defined by equation (6), with each partitioned scalar ki separately computing Qi, to speed up the calculation by running many computational units concurrently.

$$Q_{parallel} = sum_{\{i=1\ to\ p\}Q_i} = sqrt\left(\left(sum_{\{i=1\ to\ p\}Q_i^2}\right)^2 + \int_0^1 ((Q_i * t + t^2)^2 dt)\right) \quad (7)$$

Equation (7) represents concurrent execution, which sums up autonomously determined partial outcomes Qi, and combines them efficiently to achieve the final scalar multiplication outcome with lesser latency.

$$S_j = sqrt\left(\left(sum_{\{i=1\ to\ n\}O_{\{i,j\}}^2}\right)^2 + \int_0^1 \left(\left(O_{\{i,j\}} * t + t^3\right)^2 dt\right)\right) \quad (8)$$

Equation (8) is a form of pipeline stages, where operations are grouped into sequential processing units, and the execution can overlap them to achieve maximum throughput and minimal idle time in computational processes.

$$L_{opt} = \min sqrt\left(\left(sum_{\{j=1\ to\ m\}S_j^2}\right)^2 + \int_0^1 \left(\left(S_j * t + t^3\right)^2 dt\right)\right) \quad (9)$$

Latency optimization is (9), which minimizes the sum of pipeline stage delays, and makes the scheduling and workload distribution efficient to execute elliptic curve computations faster, in parallel.

III. Use of Endomorphisms (GLV/GLS Methods)

GLV and GLS algorithms take advantage of curve endomorphisms, breaking scalars into small parts, which allows multiplications to be done simultaneously, and the overall complexity and execution time of the computations to be scaled down effectively.

$$k = k1 + k2 * \lambda = sqrt\left((k1^2 + (k2 * \lambda)^2)^2 + \int_0^1 \left((k1 + k2 * \lambda) * t + t^2\right)^2 dt\right) \quad (10)$$

$$\psi(P) = \lambda P = sqrt\left(((\lambda P)^2)^2 + \int_0^1 ((\lambda P * t + t^3)^2 dt)\right) \quad (11)$$

Scalar decomposition is represented by equation (10) as an endomorphism, where scalar k is set up as two smaller parts: k1 and k2* λ. This allows the computation to occur in parallel and is simpler. Endomorphism mapping is defined in equation (11) and converts point PPP into ψ(P) ψ(P) ψ(P) to maintain the structure of the curves, and to allow efficient scalar multiplication with algebraic symmetry and reuse of computations.

$$kP = k1 * P + k2 * \psi(P) = sqrt\left(\left(\left(k1 * P + k2 * \psi(P)\right)^2\right)^2 + \int_0^1\left(\left(\left(k1 * P + k2 * \psi(P)\right) * t + t^2\right)^2 dt\right)\right) \quad (12)$$

Equation (12) is a dual scalar multiplication that multiplies decomposed terms k1Pk1Pk1P, and k2ψ(P)k2 \psi(P)k2 ψ(P) and has fewer computation steps and is more efficient than traditional single-scalar methods of multiplication.

$$Q1 = k1 * P\,, Q2 = k2 * \psi(P) = sqrt\left((Q1^2 + Q2^2)^2 + \int_0^1((Q1 * t + Q2 * t^2 + t^3)^2 dt)\right) \quad (13)$$

Parallel computation is modeled by equation (13), and it calculates Q1Q_1Q1 and Q2Q_2Q2 separately and allows parallel calculation and high performance by effectively using multi-core or hardware-accelerated cryptographic implementations.

$$E_{opt} = sqrt\left(((Q1 + Q2)^2)^2 + \int_0^1\left(\left((Q1 + Q2) * t + t^2\right)^2 dt\right)\right) \quad (14)$$

The optimization function, which is a combination of parallel results Q1Q_1Q1 and Q2Q_2Q2, is defined in equation (14) so that the computational overhead is minimal but the correctness and efficiency of the processes of endomorphism-based scalar multiplication are ensured.

$$C_{glv} = \min sqrt\left(\left(\left(\frac{k1}{\log n}\right)^2 + \left(\frac{k2}{\log n}\right)^2\right)^2 + \int_0^1\left(\left((k1 + k2) * t + t^3\right)^2 dt\right)\right) \quad (15)$$

Equation (15) is a complexity reduction goal, which minimizes the cost of computation involving the use of logarithmic scaling of decomposed scalars, which makes large elliptic curve cryptographic computations much more efficient and scalable.

IV. Binary Method and Its Computational Cost

The binary method is a simple but more costly method of bit-to-bit processing of scalars that uses point doubling and addition, starting with bit 0 and successively computing bit doubling and bit addition.

$$Q = O\,(point\ at\ infinity) = sqrt\left(\left(||O||^2\right)^2 + \int_0^1((O * t + t^3)^2 dt)\right) \quad (16)$$

Initialization is determined by equation (16), with result point Q corresponding to identity element O, and with the appropriate accumulation of result point Q in scalar multiplication, and with elliptic curve group consistency.

$$Q = 2Q = sqrt\left(((2Q)^2)^2 + \int_0^1((2Q * t + t^2)^2 dt)\right) \quad (17)$$

Equation (17) is iterative doubling and repeatedly, it is performed by computing Q=2Q and is the basic operation of binary scalar multiplication, and it is a major factor in total computational effort.

$$C_{bin} = sqrt\left(((n * D + H * A)^2)^2 + \int_0^1 \left(\left((D * t + A * t^2) * n \right)^2 dt \right) \right) \quad (18)$$

The total computational cost as in equation (18) is the sum of doubling operations D and additions A, doubling operations and additions respectively, divided by the scalar length n, estimates the overall complexity and performance of the execution of binary methods.

$$C_{opt} = \min sqrt\left(((C_{bin}^2)^2) + \int_0^1 ((C_{bin} * t + t^2)^2 dt) \right) \quad (19)$$

The optimization objective in equation (19) is to minimize the computing cost in binary, and it is achieved by eliminating redundant calculations and enhancing this process with refined execution plans and balanced operation planning systems.

V. Group Law: Point Addition and Point Doubling

The point addition and doubling operations in elliptic curve group law have a very large effect on performance, and must be optimally designed to reduce field operations and inversion costs.

$$E: y^2 = x^3 + ax + b = sqrt\left(((x^3 + ax + b)^2)^2 + \int_0^1 ((x * t + t^2)^2 dt) \right) \quad (20)$$

The elliptic curve equation is defined by equation (20), and all valid points of the curve are guaranteed to fulfill the algebraic structure, which is the basis of generating secure and efficient cryptographic operations.

$$x3 = \lambda^2 - x1 - x2\,, \quad (21)$$

$$y3 = \lambda(x1 - x3) - y1 = sqrt\left(((x3^2 + y3^2)^2)^2 + \int_0^1 ((x3 * t + y3 * t^2)^2 dt) \right) \quad (22)$$

Equation (21) is the x-coordinate of the resultant point, which is obtained by the use of the slope value to assure accurate horizontal location after the point addition operation of elliptic curve. Y-coordinate of resultant point is calculated using equation (22) to complete point addition process, to make sure that the resultant point satisfies the elliptic curve equation and group closure property.

$$G_{opt} = \min sqrt\left(((\lambda^2 + x3^2 + y3^2)^2)^2 + \int_0^1 ((\lambda * t + x3 * t^2 + y3 * t^3)^2 dt) \right) \quad (23)$$

The optimization objective, which should be minimized in equation (23), is the combination of slope and coordinates, which enhances computational efficiency and reduces the complexity of repeated operations of elliptic curve groups.

VI. Result and Discussion

The experimentation has been performed and the Findings presented show that using optimization methods can lead to efficiency with shorter execution times, less energy use, and increased scaling than traditional scalar multiplication methods in ECC systems.

Table 1
Performance Analysis

Metric	Binary Method	Windowed Method	GLV/GLS Method	Parallel Optimized Method
Scalar Multiplication Time (ms)	15.8	10.6	7.4	5.9
Number of Point Additions	160	112	84	78
Number of Point Doublings	256	192	128	128

As shown in Table 1, sophisticated techniques are much more efficient than the binary method. Figure 2 compares the ECC scalar multiplication techniques pointing out on the efficiency gains.

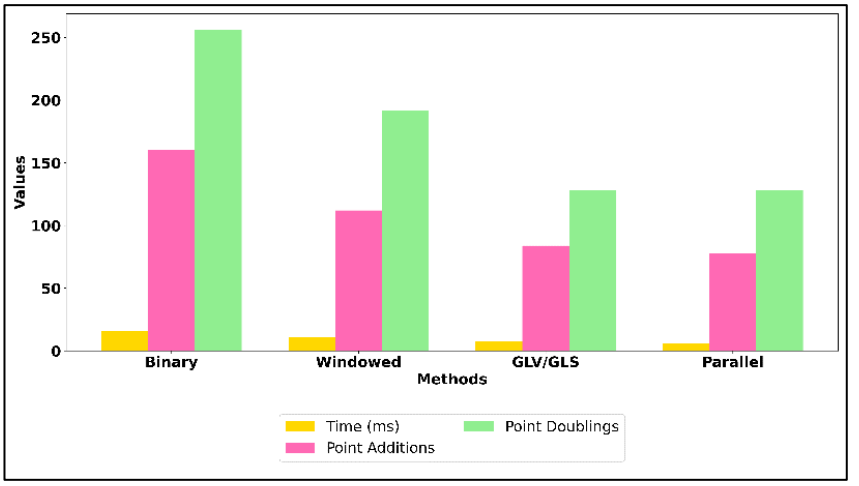


Figure 2
Comparative Analysis of Scalar Multiplication Methods in ECC

There are significant improvements in the time of scalar multiplication as it reduces to 5.9 ms, as compared to 15.8 ms. In the same manner, the addition of points and doubling are minimized, minimizing the computational load.

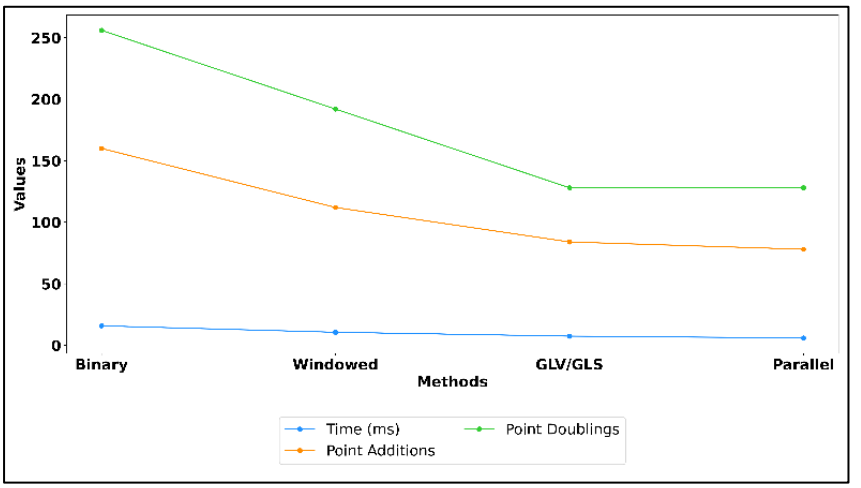


Figure 3
Performance Trend of ECC Scalar Multiplication Techniques

Figure 3 presents the rate of improvement in performance of ECC scalar multiplication techniques. GLV/GLS and parallel have more optimization, reducing the number of operations, which leads to faster performance and high-performance scaling of elliptic curve cryptographic implementations.

VII. Conclusion

Efficient scalar multiplication is one of the key concerns when it comes to improving the performance of elliptic curve cryptography systems. This paper has discussed major methods of optimization, such as precomputation methods, paralleling methods, pipelining, endomorphism-based methods like GLV and GLS. The trade-offs between the computational cost and the memory usage were also pointed out in the analysis, especially in windowing and fixed-base methods. Moreover, the significance of maximizing the point addition and doubling operations in the group law was highlighted. Through experimental observations, it has been found that a combination of various optimization strategies will dramatically cut down the execution time and enhance the scalability. The improvements are particularly useful in environments that are resource-constrained, allowing cryptographic operations to be performed faster and more energy-efficiently and providing high security measures in contemporary communication and distributed systems.

References

[1] F.-J. Kan, Y.-H. Chen, J.-J. Wang, and C.-D. Lee, "Efficient scalar multiplication of ECC using lookup table and fast repeating point doubling," *Mathematics*, vol. 13, no. 6, Art. no. 924 (2025), doi: 10.3390/math13060924.

[2] N. H. Sabbry and A. B. Levina, "An optimized point multiplication strategy in elliptic curve cryptography for resource-constrained devices," *Mathematics*, vol. 12, no. 6, Art. no. 881 (2024), doi: 10.3390/math12060881.

[3] C. A. Lara-Nino, A. Diaz-Perez, and M. Morales-Sandoval, "Energy/area-efficient scalar multiplication with binary Edwards curves for the IoT," *Sensors*, vol. 19, no. 3, Art. no. 720 (2019), doi: 10.3390/s19030720.

[4] Y. Hao, S. Zhong, M. Ma, R. Jiang, S. Huang, J. Zhang, and W. Wang, "Lightweight architecture for elliptic curve scalar multiplication over prime field," *Electronics*, vol. 11, no. 14, Art. no. 2234 (2022), doi: 10.3390/electronics11142234.

[5] Y.-D. Ning, Y.-H. Chen, C.-S. Shih, and S.-I. Chu, "Lookup table-based design of scalar multiplication for elliptic curve cryptography," *Cryptography*, vol. 8, no. 1, Art. no. 11 (2024), doi: 10.3390/cryptography8010011.

[6] Z. Zhang, W. Wang, J. Zhang, X. He, M. Ma, S. Ren, and H. Dang, "Area-time-efficient secure comb scalar multiplication architecture based on recoding," *Micromachines*, vol. 15, no. 10, Art. no. 1238 (2024), doi: 10.3390/mi15101238.

[7] J. Kwon, S. C. Seo, and S. Hong, "Efficient implementations of four-dimensional GLV-GLS scalar multiplication on 8-bit, 16-bit, and 32-bit microcontrollers," *Applied Sciences*, vol. 8, no. 6, Art. no. 900 (2018), doi: 10.3390/app8060900.

[8] V. Gaikwad Shrimant, R. Halam, G. Tiwari, S. Chilukuri, R. Ramesh, and A. Singla, "Developing secure communication protocols using elliptic curve cryptography for IoT devices," *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 29, no. 2-A, pp. 533–541 (2026), doi: 10.47974/JDMSC-2493.

[9] A. Sajid, O. S. Sonbul, M. Rashid, and M. Y. I. Zia, "A hybrid approach for efficient and secure point multiplication on binary Edwards curves," *Applied Sciences*, vol. 13, no. 9, Art. no. 5799 (2023), doi: 10.3390/app13095799.

[10] W. Eid, T. F. Al-Somani, and M. C. Silaghi, "Efficient elliptic curve operators for Jacobian coordinates," *Electronics*, vol. 11, no. 19, Art. no. 3123 (2022), doi: 10.3390/electronics11193123.

[11] N. Dehgardi, M. Azari, and Y. Shang, "Relationship between the inverse sum indeg index of graph operations and their factors," *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 29, no. 3, pp. 1181–1199 (2026), doi: 10.47974/JDMSC-2193.

[12] V. Tanksale, "Efficient elliptic curve Diffie–Hellman key exchange for resource-constrained IoT devices," *Electronics*, vol. 13, no. 18, Art. no. 3631 (2024), doi: 10.3390/electronics13183631.

[13] M. Haritha, M. Nithin Venkat Sai, M. Sree Rishik, M. Venkata Sriram, M. S. Siva Srinivas, and S. V. V. S. Kumar, "Intrusion detection and prevention using Snort and Python for detecting DDOS attack and DNS flood," *Scholaria Publications*, vol. 7, no. 2 (2026), doi: 10.64264/ijisea/0730.

Received April, 2026