Fuzzy Controller for a Dynamic Window in Elliptic Curve Cryptography Wireless Networks for Scalar Multiplication

Xu Huang
Faculty of Information Sciences and Engineering
University of Canberra
Canberra, Act 2601 Australia
Xu.Huang@canberra.edu.au

Dharmendra Sharma
Faculty of Information Sciences and Engineering
University of Canberra
Canberra, Act 2601 Australia
Dharmendra.Sharma@canberra.edu.au

Abstract—**The rapid progress of wireless communications and embedded micro-electro-mechanical systems technologies has made wireless sensor networks (WSN) possible. However, the security of the WSN becomes one of the major concerns in its applications. Elliptic curve cryptography (ECC) prominently provides solid potential for wireless sensor network security due to its small key size and its high security strength. However, there is a urgent need to reduce key calculation time to satisfy the full range of potential applications, in particularly for those applications involved wireless sensor networks (WSN). It is well known that scalar multiplication operation in ECC accounts for about 80% of key calculation time on wireless sensor network motes. In this paper we present a fuzzy controller for a dynamic window sizing to allow the calculation processing to run under optimum conditions by balanced case allocating available RAM and ROM at the sensor node within a wireless sensor network. The whole quality of Service (QoS) is improved, in particular the power consuming is more efficiently. The simulation results showed that the average calculation time decreased by approximately 15% in comparison to traditional algorithms in an ECC wireless sensor network**

Keywords- ***Elliptic curve cryptography (ECC), scalar multiplication, non-adjacent form, slide window, fuzzy controller***

I. INTRODUCTION

The rapid progress of wireless communications has become popular in our daily life, together with rapid growth in very large scale integrated (VLSI) technology, embedded systems and micro electro mechanical systems (MEMS) has enabled production of inexpensive sensor nodes which can communicate information over shorter distances with efficient use of power [1]. In the WSN systems, the sensor node will detect the interested information, processes it with the help of an in-built microcontroller and communicates results to a sink or base station. Normally the base station is a more powerful node, which can be linked to a central station via satellite or internet communication to form a network. There are many deployments for wireless sensor networks depending on various applications including environmental monitoring e.g. volcano detection [2,3], distributed control systems [4], agricultural and farm management [5], detection of radioactive sources [6], and computing platform for tomorrows' internet[7].

Contrast to traditional networks, a wireless sensor network normally has many resource constraints [4] due to the limited size. For example, the MICA2 mote consists of an 8 bit ATMega 128L microcontroller working on 7.3 MHz. As a result nodes of WSN have limited computational power. Radio transceiver of MICA motes can normally achieve maximum data rate of 250 Kbits/s, which restricts available communication resources. The flash memory that is available on the MICA mote is only 512 Kbyte. Apart from these limitations, the onboard battery is 3.3.V with 2A-Hr capacity. Therefore, the above restrictions with the current state of art protocols and algorithms are expensive for sensor networks due to their high communication overhead.

Elliptic Curve Cryptography was first introduced by Neal Koblitz [9] and Victor Miller [10] independently in the early eighties. The advantage of ECC over other public key cryptography techniques such as RSA, Diffie-Hellman is that the best known algorithm for solving elliptic curve discrete logarithm problem (ECDLP) which is the underlying hard mathematical problem in ECC which will take the fully exponential time. On the other hand the best algorithm for solving RSA and Diffie-Hellman takes sub exponential time [11]. In summary, the ECC problem can only be solved in exponential time and, to date, there is a lack of sub exponential methods to attack ECC.

An elliptic curve E over $GF(p)$ can be defined by $y^2 = x^3 + ax + b$ where $a, b \in GF(p)$ and $4a^3 + 27b^2 \neq 0$ in the $GF(p)$.

The point (x, y) on the curve satisfies the above equation and the point at infinity denoted by ∞ is said to be on the curve.

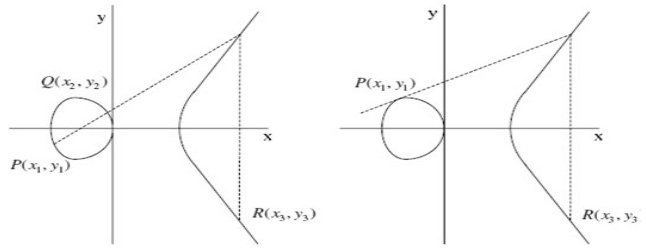
If there are two points on the curve namely, $P(x_1, y_1)$, $Q(x_2, y_2)$ and their sum is given by point $R(x_3, y_3)$ the algebraic formulas for point addition and point doubling are given by following equations:

We have: $x_3 = \lambda^2 - x_1 - x_2$

$$y_3 = \lambda(x_1 - x_3) - y_1$$

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1}, \quad \text{if } P \neq Q$$

$$\lambda = \frac{3x^2 + a}{2y_1}, \quad \text{if } P = Q$$



(a) Addition: $P + Q = R$ (b) Doubling: $P+P = R$

Figure 1: Point addition and point doubling on elliptic curve.

Where the addition, subtraction, multiplication and inverse are the arithmetic operations over $GF(p)$, which can be shown in Figure 1.

II. ELLIPTIC Curve Diffie-Hellman Scheme (ECDH) Proposed for WSN

Before we get into our innovation method, we need to have a closer look at the popular legacy scheme for WSN. As per [13] the original Diffie-Hellman algorithm with RSA requires a key of 1024 bits to achieve sufficient security but *Diffie Hellman based on ECC* can achieve the same security level with only 160 bit key size.

The classical Elliptic Curve Diffie Hellman scheme operates as shown in the Fig. 2

Initially Alice and Bob agree on a particular curve with base point P. They generate their public keys by multiplying P with their private keys namely K_A and K_B. After sharing public keys, they generate a shared secret key by multiplying public keys by their private keys. The secret key is $R = K_A*Q_A = K_B*Q_A$. With the known values of Q_A, Q_B and P, it is computationally intractable for an eavesdropper to calculate K_A and K_B which are the private keys of Alice and Bob. As a result, adversaries cannot calculate R the shared secret key.

In ECC two heavily used operations are involved: scalar multiplication and modular reduction. Gura et. al. [14] showed that 85% of execution time is spent on scalar multiplication. Scalar Multiplication is the operation of multiplying point P on an elliptic curve E defined over a field $GF(p)$ with positive integer k which involves point addition and point doubling. Operational efficiency of kP is affected by the type of coordinate system used for point P on the elliptic curve and the algorithm used for recoding of integer k in scalar multiplication.

This research paper proposes an innovative algorithm based on one's complement for representation of integer k which accelerates the computation of scalar multiplication in wireless sensor networks.

The number of point doubling and point addition operations in scalar multiplication depends on the recoding of integer k. Expressing integer k in binary format highlight this dependency.

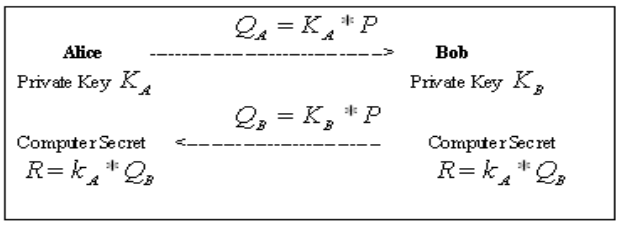


Figure 2. Diffie-Hellman protocol based on ECC

The number of zeros and number of ones in the binary form, their places and the total number of bits will affect the computational cost of scalar multiplications. The Hamming weight as represented by the number of non-zero elements, determines the number of point additions and bit length of integer K determines the number of point doublings operations in scalar multiplication.

One point addition when $P \neq Q$ requires one field inversion and three field multiplications [13]. Squaring is counted as regular multiplication. This cost is denoted by $1I + 3M$, where I denotes the cost of inversion and M denotes the cost of multiplication.

One point doubling when $P = Q$ requires $1I + 4M$ as we can neglect the cost of field additions as well as the cost of multiplications by small constant 2 and 3 in the above formulae.

Binary Method

Scalar multiplication is the computation of the form $Q = kP$, where P and Q are the elliptic curve points and k is positive integer. This is obtained by repeated elliptic curve point addition and doubling operations. In binary method the integer k is represented in binary form:

$$k = \sum_{j=0}^{l-1} K_j 2^j, \; K_j \in \{0,1\}$$

The binary method scans the bits of K either from left-to-right or right-to-left. The binary method for the computation of kP is given in the following *algorithm 1*, as shown below:

Algorithm 1: Left to right binary method for point multiplication

---

Input: A point $P \in E(\mathcal{F}_q)$, an

l bits integer $k = \sum_{j=o}^{l-1} K_j 2^j, \; K_j \in \{0,1\}$

Output: $Q = kP$

1. $Q \leftarrow \infty$
2. For $j = l - 1$ to 0 do:
   - 2.1 $Q \leftarrow 2Q$,
   - 2.2 if $k_j = 1$ the $Q \leftarrow Q + P$.
3. Return Q.

The cost of multiplication when using binary method depends on the number of non-zero elements and the length of the binary representation of k. If the representation has $k_{l-1} \neq 0$

then binary method require ($l - 1$) point doublings and (W-1) where l is the length of the binary expansion of k, and W is the Hamming weight of k (ie, the number of non-zero elements in expansion of k). For example, if $k = 629 = (1001110101)_2$, it will require (W-1) = 6 – 1 = 5 point additions and $l - 1 = 10 - 1 = 9$ point doublings operations.

Signed Digit Representation Method

The subtraction has virtually the same cost as addition in the elliptic curve group. The negative of point (x, y) is (x, $-y$) for odd characters. This leads to scalar multiplication methods based on addition –subtraction chains, which help to reduce the number of curve operations. When integer k is represented with the following form, it is a *binary signed digit representation*.

$$k = \sum_{j=0}^{l} S_j 2^j, \quad S_j \in \{1,0,-1\}$$

When a signed-digit representation has no adjacent non zero digits, i.e. $S_j S_{j+1} = 0$ for all $j \geq 0$ it is called a non-adjacent form (NAF).

The following *algorithm 2* computes the NAF of a positive integer given in binary representation.

Algorithm 2: Conversion from Binary to NAF

Input: An integer $k = \sum_{j=0}^{l-1} K_j 2^j, \ K_j \in \{0,1\}$

Output: NAF $k = \sum_{j=0}^{l} S_j 2^j, \ S_j \in \{1,0,-1\}$

1. $C_0 \leftarrow 0$
2. *For j = 0 to l do:*
3. $C_{j+1} \leftarrow [(K_j + K_{j+1} + C_j)/2]$
4. $S_j \leftarrow K_j + C_j - 2C_{j+1}$
5. *Return* (S_l ... S_0)

NAF usually has fewer non-zero digits than binary representations. The average hamming weight for NAF form is ($n - 1$)/3.0. So generally it requires ($n - 1$) point doublings and ($n - 1$) /3.0 point additions. The binary method can be revised accordingly and is given another algorithm for NAF, and this modified method is called the *Addition Subtraction* method.

III. Dynamitic Window with Fuzzy Controller in ECC Proposed Algorithm Based

We are going to use the algorithm based on *subtraction by utilization of the 1's* complement is most common in binary arithmetic. The 1's complement of any binary number may be found by the following equation [19-22]:

$$C_1 = (2^a - 1) - N \qquad (1)$$

where C_1 = 1's complement of the binary number, a = number of bits in N in terms of binary form, N = binary number

From a closer observation of the equation (I), it reveals the that any positive integer can be represented by using minimal non-zero bits in its 1's complement form provided that it has a minimum of 50% Hamming weight. The minimal non-zero bits in positive integer scalar are very important to reduce the number of intermediate operations of multiplication, squaring and inverse calculations used in elliptical curve cryptography as we have seen in previous sections.

The equation (I) can therefore be modified as per below:

$$N = (2^a - C_1 - 1) \qquad (2)$$

For example, we may take N =1788 then it appears
$N = (11011111100)_2$ in its binary form
C_1= 1's Complement of the number of $N = (00100000011)_2$
a is in binary form so we have $a = 11$
After putting all the above values in the equation (2) we have:
$1788 = 2^{11} - 00100000011 - 1$, this can be reduced as below:
$1788 = 100000000000 - 00100000011 - 1$ (3)
So we have
$1788 = 2048 - 256 - 2 - 1 - 1$

As is evident from equation (3), the Hamming weight of scalar N has reduced from 8 to 5 which will save 3 elliptic curve addition operations. One addition operation requires 2 Squaring, 2 Multiplication and 1 inverse operation. In this case a total of 6 Squaring, 6 Multiplication and 3 Inverse operations will be saved.

The above recoding method based on one's complement subtraction combined with sliding window method provides a more optimized result.

Let us compute [763] P (in other words k = 763) as an example, with a *sliding window algorithm* with K recoded in binary form and window sizes ranging from 2 to 10. It is observed that as the window size increases the number of pre-computations also increases geometrically. At the same time number of additions and doubling operations decrease.

Now we present the details for the different window size to find out the optimal window size using the following example:

Window Size w = 2

$763 = (1011111011)_2$
No of precomputations = $2^w - 1 = 2^2 - 1 = [3]\ P$
763 = 10 11 11 10 11
The intermediate values of Q are

P, 2P 4P, 8P, 11P, 22P, 44P, 47P, 94P, 95P, 190P, 380P, 760P, 763P

Computational cost = 9 doublings, 4 additions, and 1 pre-computation.

Window Size w = 3

No of pre-computations = $2^w - 1 = 2^3 - 1 = [7]\ P$
So all odd values: [3]P, [5]P, [7]P
763 = 101 111 101 1
= [5]P [7]P [5]P [1]P
The intermediate values of Q are

5P, 10P, 20P, 40P, 47P, 94P, 188P, 376P, 381P, 762P, 763P

Computational cost = 7 doublings, 3 additions, and 3 pre-

computations.

Algorithm for sliding window scalar multiplication on elliptic curves.

1. $Q \leftarrow P_{\infty}$ *and* i $\leftarrow l$ - 1
2. while i $\geq$ 0 do
3. if $n_i = 0$ then Q $\leftarrow$ [2]Q and i $\leftarrow$ i - 1
4. else
5. s $\leftarrow \max(i - k + 1, 0)$
6. *while* $n_s = 0$ do s $\leftarrow$ s + 1
7. for h = 1 to i - s + 1 do Q $\leftarrow$ [2]Q
8. u $\leftarrow (n_i \ldots\ldots n_s)_2$ $[n_i = n_s = 1$ and i - s + 1 $\leq$ k]
9. Q $\leftarrow$ Q $\oplus$ [u]P [u is odd so that [u]P is precompute d]
10. i $\leftarrow$ s - 1
11. return Q

We continue to derive the remaining calculations for Window Size w = 6, Window Size w = 7, Window Size w = 8, Window Size w = 9, and Window Size w = 10. The results for all calculations are presented in Table 1.

The effects of "doublings" and "additions" as shown in Table 1 are further considered below.

TABLE I. WINDOW SIZE VS NO OF DOUBLINGS, ADDITIONS AND PRE COMPUTATIONS

Window Size	No of Doublings	No of Additions	No of Pre computations
2	9	4	1
3	7	3	3
4	6	2	7
5	5	1	15
6	4	1	31
7	3	1	61
8	3	1	127
9	1	1	251
10	0	0	501

IV. Fuzzy Controller System in ECC

It is clear, from above description that there is a tradeoff between the computational cost and the window size as shown in Table 1. However, this tradeoff is underpinned by the balance between computing cost (or the RAM cost) and the pre-computing (or the ROM cost) of the node in the network.

It is also clear that, from above description that the variety of wireless network working states will make this control complex and calculations could be relatively more expensive.

Therefore, we propose a fuzzy dynamic control system, to provide dynamic control to ensure the optimum window size is obtained by tradeoff between pre-computation and computation cost.

The fuzzy decision problem introduced by Bellman and Zadeh has as a goal the maximization of the minimum value of the membership functions of the objectives to be optimized. Accordingly, the fuzzy optimization model can be represented as a multi-objective programming problem as follows [21]:

$$Max: \min\{\mu_s(D)\} \& \min\{\mu_l(U_l)\} \quad \forall s \in S \& \forall l \in L$$

$$\text{such that } A_l \le C_l \qquad \forall l \in L,$$

$$\sum_{r \in R_p} x_{rs} = 1 \qquad \forall p \in P \& \forall s \in S,$$

$$x_{rs} = 0 \text{ or } 1 \qquad \forall r \in R \& \forall s \in S$$

In above equation, the objective is to maximize the minimum membership function of all delays, denoted by D, and the difference between the recommend value and the measured value, denoted by U.

The Fuzzy control system is extended from and shown in Figure 3. For accurate control, we designed a three inputs fuzzy controller. The first input is *storage room*, which has three statuses, showing storage room in one of the three, namely (a) low, (b) average, and (c) high. The second input is pre-computing working load (*PreComputing*) in one of three states, namely (a) low, (b) average, and (c) high. The third input is *Doubling*, expressing how much working load for the calculation "doubling" which has three cases, namely (a) low, (b) average, and (c) high. The output is one, called *WindowSize,* to express the next window size should be moved in which way, which has three states for the window sizes, namely (a) down, (b) stay, and (c) up.

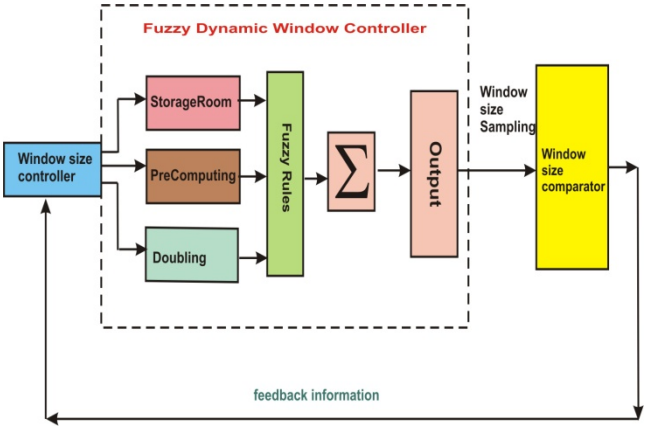


Figure 3: Three inputs fuzzy window control system

There are 26 *Fuzzy Rules* listed as follows (wight are unit):

1. If (StorgeRoom is low) and (PreComputing is low) and (Doubling is low) then (WindowSize is Up)
2. If (StorgeRoom is low) and (PreComputing is low) and (Doubling is average) then (WindowSize is Up)
3. If (StorgeRoom is low) and (PreComputing is low) and (Doubling is high) then (WindowSize is stay)
4. If (StorgeRoom is low) and (PreComputing is average) and (Doubling is low) then (WindowSize is Up)
5. If (StorgeRoom is low) and (PreComputing is average) and (Doubling is average) then (WindowSize is Up)
6. If (StorgeRoom is low) and (PreComputing is average) and (Doubling is high) then (WindowSize is stay)
7. If (StorgeRoom is low) and (PreComputing is high) and (Doubling is low) then (WindowSize is Up)
8. If (StorgeRoom is low) and (PreComputing is high) and (Doubling is average) then (WindowSize is stay)

9. If (StorgeRoom is low) and (PreComputing is high) and (Doubling is high) then (WindowSize is stay)
10. If (StorgeRoom is average) and (PreComputing is low) and (Doubling is low) then (WindowSize is Up)
11. If (StorgeRoom is average) and (PreComputing is low) and (Doubling is average) then (WindowSize is Up)
12. If (StorgeRoom is average) and (PreComputing is low) and (Doubling is high) then (WindowSize is stay)
13. If (StorgeRoom is average) and (PreComputing is average) and (Doubling is low) then (WindowSize is Up)
14. If (StorgeRoom is average) and (PreComputing is average) and (Doubling is average) then (WindowSize is stay)
15. If (StorgeRoom is average) and (PreComputing is average) and (Doubling is high) then (WindowSize is Down)
16. If (StorgeRoom is average) and (PreComputing is high) and (Doubling is average) then (WindowSize is stay)
17. If (StorgeRoom is average) and (PreComputing is high) and (Doubling is high) then (WindowSize is stay)
18. If (StorgeRoom is high) and (PreComputing is low) and (Doubling is low) then (WindowSize is stay)
19. If (StorgeRoom is high) and (PreComputing is low) and (Doubling is average) then (WindowSize is stay)
20. If (StorgeRoom is high) and (PreComputing is low) and (Doubling is high) then (WindowSize is Down)
21. If (StorgeRoom is high) and (PreComputing is average) and (Doubling is low) then (WindowSize is stay)
22. If (StorgeRoom is high) and (PreComputing is average) and (Doubling is average) then (WindowSize is stay)
23. If (StorgeRoom is high) and (PreComputing is average) and (Doubling is high) then (WindowSize is Down)
24. If (StorgeRoom is high) and (PreComputing is high) and (Doubling is low) then (WindowSize is Down)
25. If (StorgeRoom is high) and (PreComputing is high) and (Doubling is average) then (WindowSize is Down)
26. If (StorgeRoom is high) and (PreComputing is high) and (Doubling is high) then (WindowSize is Down)

The number at each fuzzy condition with a bracket is the weight number, currently it is unit. Later we shall change it with different number according to the running situations as described in the next.

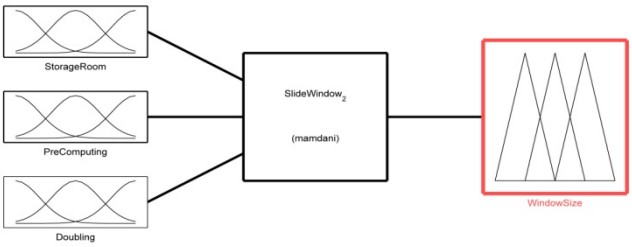


Figure 4: The block diagram of current fuzzy controller

The three inputs with 26 fuzzy rules in *Mamdani model* running fuzzy controller part is shown in Figure 4. The three inputs are *StorageRoom*, *PreComputing* and *Doubling*. The output is *WindowSize*.

The output with *StorageRoom* and *PreComputing* is shown in Figure 5. The surface *StorageRoom* vs. *Doubling* is shown in Figure 6.

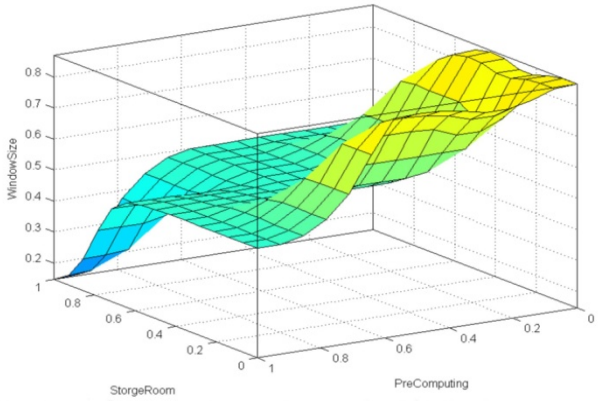


Figure 5: The output of the surface for the *StorageRoom* vs. *PreComputing*.

The surface *StorageRoom* vs. *PreComputing* is shown in Figure 7.

From above figures, it is clearly observed that in the low window size side, if the storage room is low the dominated function of "doubling" will play role as Figure 5 shown but if the window size is at the high side, the storage room will be fairly stay at the middle either for PreComputing or Doubling, which is the doubling will sharply increased when window size a little bit larger that also can be shown from Table 1. From Figure 6 it is clearly to show when the storage room is getting big, it would be nice to have larger window size for the "doubling".

Now if we change the weight for above fuzzy rules as such the rules 1,5 10, 13, 14, 15, 16, 18, 20 ,21, 22, 23, 25, and 26 are set in 0.5 (the rest will keep the same) due to the major functions are controlled by the storage room, and doubling will rapidly increasing by the window size larger. The outputs will changed as the average storage room will increased 0.04% and the other two inputs are decreased by 0.02% the output become window staying a little wider side by 0.003%.

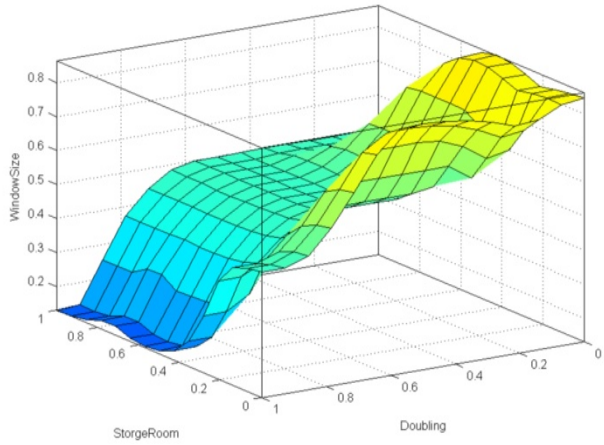


Figure 6: The output of the surface for the *StorageRoom* vs. *Doubling*.

It is clear that this fuzzy controller for the dynamic window is also involved a tradeoff between accuracy and control costs. For example the same system may go further for the second order parameters, not just check the changes about the input variables but also check the change tendencies of the variables, which will be discussed in our another paper.

If we keep the storage constant and the situation shown by Figure 7 is how those two major factors shown in Table 1 to impact on the output.

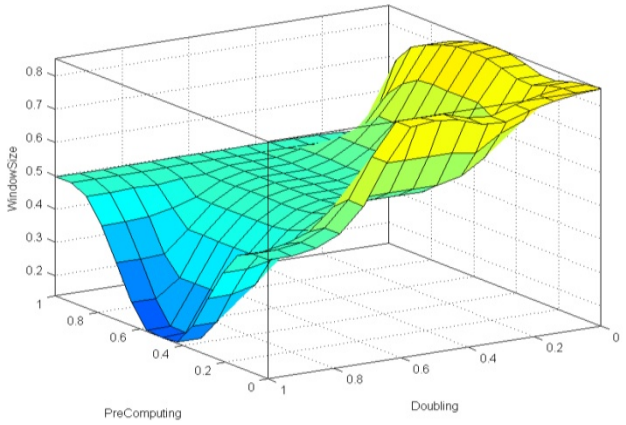


Figure 7: The output of the surface for the *StorageRoom* = constant and *PrecComputing* vs. *Doubling*.

The simulations of the example described in above were implemented. With equation (2), the computational cost has been reduced from 3 additions as in the binary method to only 1 addition in one's complement subtraction form. The number of pre-computations has remained the same. This can be proved for different window sizes.

In our simulations, the proposed method together with a fuzzy window size controller makes the ECC calculation almost 15% more efficient than traditional methods in ECC wireless sensor network.

V. CONCLUSION

The positive integer in point multiplication may be recorded with *one's complement subtraction* to reduce the computational cost involved in this heavy mathematical operation for wireless sensor network platforms. As the NAF method involves modular inversion operation to get the NAF of binary number, the one's complement subtraction can provide a very simple way of recoding the integer. There is always decision between pre-computing and computing, the former is related to the storage and the latter is associated with computing capability and capacity. The window size may be the subject of trade-off between the available RAM and ROM at a particular instance on a sensor node, which can be controlled by fuzzy controller. The final simulation in a sensor wireless network shows that about 15% more efficient than transitional method can be obtained with ECC.

REFERENCE

[1]. I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks,* vol. 38, pp. 393-422, 2002.

[2]. C. Chung-Kuo, J. M. Overhage, and J. Huang, "An application of sensor networks for syndromic surveillance," 2005, pp. 191-196.

[3]. G. Werner-Allen, K. Lorincz, M. Ruiz, O. Marcillo, J. Johnson, J. Lees, and M. Welsh, "Deploying a wireless sensor network on an active volcano," *IEEE Internet Computing,* vol. 10, pp. 18-25, 2006.

[4]. B. Sinopoli, C. Sharp, L. Schenato, S. Schaffert, and S. S. Sastry, "Distributed control applications within sensor networks," *Proceedings of the IEEE,* vol. 91, pp. 1235-1246, 2003.

[5]. P. Sikka, P. Corke, P. Valencia, C. Crossman, D. Swain, and G. Bishop-Hurley, "Wireless ad hoc sensor and actuator networks on the farm," 2006, pp. 492-499.

[6]. D. L. Stephens, Jr. and A. J. Peurrung, "Detection of moving radioactive sources using sensor networks," *Nuclear Science, IEEE Transactions on,* vol. 51, pp. 2273-2278, 2004.

[7]. Z. Feng, "Wireless sensor networks: a new computing platform for tomorrow's Internet," 2004, pp. I-27 Vol.1.

[8]. I. F. Akyildiz, Weilian Su, Yogesh Sankarasubramaniam, and E. Cayirci, "A Survey on Sensor Networks" in *IEEE Communication Magazine*. vol. 40, August 2002, pp. 102-116.

[9]. N.Koblitz, "Elliptic Curve Cryptosystems," *Mathematics of Computation,* vol. 48, pp. 203-209, 1987.

[10]. V. S. Miller, "Use of Elliptic Curves in Cryptography," in *Advances in Cryptology - CRYPTO '85: Proceedings*. vol. 218: Springer-Verlag, 1986, pp. 417-426.

[11]. J. Lopez and R. Dahab., " An overview of elliptic curve cryptography," Technical report ,Institute of Computing, Sate University of Campinas, Sao Paulo, Brazil, May 2000.

[12]. K. Lauter, "The advantages of elliptic curve cryptography for wireless security," *Wireless Communications, IEEE [see also IEEE Personal Communications],* vol. 11, pp. 62-67, 2004.

[13]. H. Wang, B. Sheng, and Q. Li, "Elliptic curve cryptography-based access control in sensor networks," *Int. J. Security and Networks,,* vol. 1, pp. 127-137, 2006.

[14]. N. Gura, A. Patel, and A. Wander, "Comparing elliptic curve cryptography and RSA on 8-bit CPUs," in *Proceedings of the 2004 Workshop on Cryptographic Hardware and Embedded Systems (CHES)* August 2004.

[15]. http://csrc.nist.gov/CryptoToolkit/dss/ecdsa/NISTReCur.pdf."

[16]. D. J. Malan, M. Welsh, and M. D. Smith, "A public-key infrastructure for key distribution in TinyOS based on elliptic curve cryptography," in *2nd IEEE International Conference on Sensor and Ad Hoc Communications and Networks (SECON 2004)2nd IEEE International Conference on Sensor and Ad Hoc Communications and Networks (SECON 2004)*, 2004, pp. 71-80.

[17]. I. Blake, G. Seroussi, and N. Smart, *Elliptic Curves in Cryptography* vol. 265, 1999.

[18]. D. Hankerson, J. L. Hernandez, and A. Menezes, "Software Implementation of Elliptic Curve Cryptography over Binary Fields, CHES," 2000.

[19]. Angelo C Gillie, "Binary Arithmetic and Boolean algebra," McGRAW-HILL Book Company, 1965. pp53.

[20]. H R. Bellman and L.A. Zadeh, Decision-making in a fuzzy environment, Management Science 17 (1970),141-164.

[21]. Xu Huang, Shirantha Wijesekera, and Dharmendra Sharma, "Fuzzy Dynamic Switching in Quantum Key Distribution for Wi-Fi Networks," 6th International Conference on Fuzzy Systems and Knowledge Discovery, 14-16 August 2009, Tianjin, China. Proceeding pp302-306.

[22]. Xu Huang, Pritam Gajkumar Shah and Dharmendra Sharma, "Multi-Agent System Protecting from Attacking with Elliptic Curve Cryptography," the 2nd International Symposium on Intelligent Decision Technologies, Baltimore, USA, 28-30 July 2010. Accepted to be published.