MANAS Journal of Engineering
ISSN 1694-7398 | e-ISSN 1694-7398

Volume, Issue, (2024) Pages 192-197
https://doi.org/1524490

A Modified Key Generation Algorithm to Rebalanced-RSA and RPower-RSA

İsrafil Okumuş[1], Ercan Çelik[2,*]

[1] Erzincan Binali Yildirim University, Erzincan, Türkiye, iokumus@erzincan.edu.tr, ORCID: 0000-0003-1193-3984

[2] Kyrgyz-Turkish Manas University, Bishkek, Kyrgyz Republıc, ercan.celik@manas.edu.kg, ORCID: 0000-0001-5971-7653

ABSTRACT

This paper presents a novel key generation algorithm for Rebalanced-RSA and RPower-RSA that accelerates encryption and decryption processes by utilizing smaller modular exponents. Subsequently, two of these variants are modified by altering the key generation process. A theoretical analysis of all variants, including the proposed modifications, demonstrates that these key generation algorithms improve the encryption process compared to the original variants, and since the encryption key size and one of the decryption key sizes are approximately equal in length, they achieve a more balanced computational effort between encryption and decryption.

ARTICLE INFO

Research article

Received: 29.07.2024
Accepted: 03.12.2024

Keywords:
RSA,
Variation of RSA,
Rebalanced-RSA,
RPower-RSA,
Key Generation Algorithm

*Corresponding author

1. Introduction

In cryptography, there are two types of cryptosystems: secret-key and public-key. Secret-key cryptosystems use a single key, referred to as the secret key, for both encryption and decryption processes. In contrast, public-key cryptosystems utilize two distinct keys: a public key and a private key. The public key is used for the encryption, while the private key is used for the decryption process. This allows secure communication over a insecure communication channel without the need to share the private key. Therefore, public key cryptosystems are widely used for information security and digital signatures worldwide.

The RSA algorithm and its variants have played a significant role in modern cryptography, providing a foundation for secure communication and data protection. The RSA cryptosystem, first introduced by Rivest, Shamir, and Adleman [1], is one of the most widely recognized public-key cryptosystems. Many researchers have proposed varius variants to enhance the efficiency of RSA [2]. Most of these efforts focus on reducing computational cost to shorten the duration of encryption or decryption.

The variants enhance decryption efficiency by speeding up the process through the use of the Chinese Remainder Theorem or the selection of shorter decryption keys.

Quisquater and Couvreur [3] proposed an RSA variant known as CRT-RSA, which utilizes the Chinese Remainder Theorem (CRT) to accelerate the decryption process. Fiat [4] introduced Batch RSA, a variant that employs a small public exponent for the same common modulus enabling the decryption of the two ciphertext at the cost of one. Wiener [5] suggested a key generation algorithm called Rebalanced-RSA, which uses small private keys to the decryption speed of CRT-RSA,

Collins et al. [6] introduced MultiPrime-RSA, where the modulus n is modified using k primes instead of just two, as in CRT-RSA. Takagi [7] proposed MultiPower-RSA, which defines the common modulus $n := p^k q$ and uses the Hensel-Lifting method for decryption [17]. Paixão and Filho [8] introduced an efficient variant called Rprime RSA, which combines MultiPrime-RSA and Rebalanced-RSA to improve the efficiency of MultiPrime-RSA. Grag and Verma [9] further combined Rebalanced-RSA and MultiPower-RSA to create a variant called RPower-RSA.

The Rebalanced-RSA, RPrime-RSA, and RPower-RSA variants expedite the decryption process by utilizing shorter decryption keys. However, these algorithms increase the encryption time by generating larger-sized encryption keys. This is the primary motivation for modifying these algorithms is to improve encryption performance, making them more efficient in terms of computational costs. This paper focuses on proposing a novel key generation algorithm tailored for Rebalanced-RSA and RPower-RSA variants.

Initially, a concise overview of Standard RSA and its efficiency-enhancing variants is provided.

2. The RSA Cryptosystem

The RSA cryptosystem, like other asymmetric encryption algorithms, consists of three fundamental components: key generation, encryption, and decryption. An overview of the original RSA scheme, commonly referred to as standard RSA, follows the general procedure for key generation in the following order:

Firstly, two distinct large strong prime numbers [10], p and q of equal bit lengths are chosen, and $n := pq$ and $\varphi := (p-1)(q-1)$ are computed. Second, a random integer e is selected such that $1 < e < \varphi$ and $gcd(e,\varphi) = 1$. Finally, the integer d is computed such that $d := e^{-1} \bmod \varphi$ using the Extended Euclidean Algorithm. The integer n serves as the common modulus, the public key e is used for encryption, and the private key d is used for decryption. The formula (1.1) is used to encrypt the plaintext $M \in Z_n$.

$$C := M^e \bmod n \tag{1.1}$$

The formula (1.2) gives the decrypted text from the ciphertext C.

$$M := C^d \bmod n \tag{1.2}$$

It's easy to see that the cost of both encryption and decryption processes depends roughtly on the bit size of the keys and the modulus since it includes operations of modular exponentiation. [11-13]. For example, Fast Modular Exponention alorithm uses $(\log e)^2 \log n$ bit operations to find $a^e \bmod n$ [22].

To significantly speed up RSA encryption, one may try to use a much smaller public key. In this case, However, various threat assumptions are presented by Coppersmith [14].

Similarly, to speed up RSA decryption one may try to use a much smaller private key. However, Wiener [5] showed that when $d < \frac{1}{3}\sqrt[4]{n}$ public modulus n can be factored easily. Later, Boneh and Durfee [15] presented a vulnerability called short private key exponent attack, and suggested an increase in this bound up to $d < n^{0.292}$.

3. Overview of RSA Variants and Their Improvements

All variants of RSA primarily aim to reduce the computational cost and shorten the encryption or decryption times of the RSA cryptosystem. We can classify these developed variants by categorizing into two classes [16,19]. The first class uses the Chinese Remainder Theorem (CRT) to enhance the speed of the RSA. The algorithms in this class include CRT-RSA, MultiPrime-RSA, and MultiPower-RSA. The second class speeds up the encryption process by adjusting the key size used in the algorithms from the first class. The variants in this class include Rebalanced-RSA, RPrime-RSA, and RPower-RSA. In all variants, the encryption process resembles the standard RSA.

3.1. CRT-based RSA variants

To speed up the RSA decryption, the key generation process has been modified in this variants. The variants in this class improve the decryption process by implementing the Chinese Remainder Theorem (CRT) algorithm, which accellerates decryption compared to standard RSA. However, the encryption operations remains like that of standard RSA, resulting in encryption time being equivalent to that of standard RSA.

CRT-RSA: This variant of the RSA cryptosystem utilizes the CRT to accelerate of the RSA decryption. During key generation,the public key e is selected first, o speed up decryption, the private keys d_p and d_q are computed such that

$$\begin{aligned} d_p &:= e^{-1} \bmod (p-1) \\ d_q &:= e^{-1} \bmod (q-1) \end{aligned} \tag{3.1}$$

To determine the plaintext , in decryption process are used the CRT or the Garner algorithm as:

$$M := M_p + p\left[(M_q - M_p)p^{-1} \bmod q\right] \tag{3.2}$$

such that

$$C_p := C \bmod p \tag{3.3}$$

$$M_p := {C_p}^{d_p} \bmod p \tag{3.4}$$

$$C_q := C \bmod q \tag{3.5}$$

$$M_q := {C_q}^{d_q} \bmod q \tag{3.6}$$

Its decryption approximately about four times faster than the standard RSA in theoretically.

MultiPrime-RSA: In this variant are used k primes and the common modulus are calculated as $n := p_1.p_2.\ldots.p_k$. Similar to CRT-RSA, Like CRT-RSA, the key generation process involves selecting the public key e first, and the private keys are computed as $d_{p_i} := e^{-1} \bmod (p_i - 1)$. The decryption preccess is an extension like CRT-RSA, to determine the plaintext M are computed as

$$M := \sum_{i=1}^{k} \left[M_i . \left(\frac{n}{p_i}\right) . \left(\left(\frac{p_i}{n}\right) \bmod p_i \right) \right] \bmod n$$

such that $C_{p_i} := C \bmod p_i$, $M_i := {C_{p_i}}^{d_i} \bmod p_i$. The speed of decryption is approximately about $2k$ times faster than the standard RSA in theoretically.

MultiPower-RSA: In this variant, the common modulus is calculated as $n := p^k q$, where $k \geq 2$ is an integer. When

generating keys, the public key e and private keys d_p and d_q are calculated like CRT-RSA. While decrypting the chipper text C, to begin with, the values of M_p and M_q are calculated and then the values of M_{p^k} and M_q are combined via CRT algorithm as

$$M = M_{p^k} + p^k.[((M_q - M_{p^k}).p^{-k}) \bmod q] \quad (3.7)$$

such that $M_{p^k} \equiv M \bmod p^k$ which is can be calculated by using the Hensel-Lifting method was introduced by Takagi [17]. For example, it is calculated using the following formula (for k=2)

$$M_{p^2} = M_p + [(C - M_p{}^e \bmod p^2)(e^{-1} \bmod p)(C_p{}^{d_p-1} \bmod p)] \bmod p^2 \quad (3.8)$$

This approach speeds up the decryption process by leveraging the power of the modulus about 3 times faster than CRT-RSA [6].

3.2. RSA variants using shorter decryption keys

The algorithms in this class shorten the decryption process by selecting shorter decryption keys in CRT-based RSA variants, but, it is important to note that the encryption process becomes more time-consuming compared to both the standrad RSA and the algorithm in the first class.

Rebalanced-RSA: To shorten the time of decryption process CRT-RSA. Initially, when generating keys, private keys d_p and d_q are chosen short size, satisfying $\gcd(d_p, p-1) = \gcd(d_q, q-1) = 1$. The integer d is computed using the Garner algorith, such that:

$$d = d_q + (q-1).[((d_q - d_p).(p-1)^{-1}) \bmod (q-1)]$$

to obtain the public key as $e = d^{-1} \bmod \varphi$.

In this algorithm, the encryption and decryption process are like standard RSA. The private keys are chosen to be shorter in size, which speeds up the decryption process and the decryption time is significantly faster. However, since the bit size of the encryption key is approximately equal to the bit size of φ, the encryption time increases compared to standard RSA.

.

Rprime-RSA (Rebalanced MultiPrime-RSA): for $n := p_1.p_2 \ldots .p_k$ (k primes) the private key generation is similar to Rebalanced-RSA to further improve the decryption speed of MultiPrime-RSA and the decryption process is considered as MultiPrime-RSA. To shorten the decryption are chosen short private keys d_i. The integer d is calculated with CRT such that $d := d_{p_i} \bmod (p_i - 1)$ to get $e := d^{-1} \bmod \varphi$. Note that, we can calculate the private key as d_{k-1} by applying Garner algorithm successively as follows:

$$d_1 = d_{p_1} + (p_1 - 1)[(d_{p_2} - d_{p_1})(p_1 - 1)^{-1} \; mod \; (p_2 - 1)]$$

$$d_2 = d_1 + (p_1 - 1)(p_2 - 1)[(d_{p_3} - d_1)((p_1 - 1)(p_2 - 1))^{-1} \; mod \; (p_3 - 1)]$$

$$d_3 = d_2 + \prod_{i=1}^{3}(p_i - 1)\left[(d_{p_4} - d_2)\left(\prod_{i=1}^{3}(p_i - 1)\right)^{-1} \; mod \; (p_4 - 1)\right]$$

...

$$d_{k-1} = d_{k-2} + \prod_{i=1}^{k-2}(p_i - 1)\left[(d_{p_{k-2}} - d_2)\left(\prod_{i=1}^{k-2}(p_i - 1)\right)^{-1} \; mod \; (p_{k-2} - 1)\right]$$

Its decryption process is approximately 8 times faster than the CRT-RSA and about 27 times faster than the standard RSA theoretically (for 2048 bits) [8]. However, like Rebalanced-RSA the encryption time increases compared to MultiPrime-RSA.

RPower-RSA (Rebalanced MultiPower-RSA): Similarly, in RPrime-RSA, the key generation process is modified to enhance the decryption performance of MultiPower-RSA. The private key generation is like that of Rebalanced-RSA, when generating keys, firstly, private keys d_p and d_q are chosen and the public key is calculated as $e = d^{-1} \bmod \lambda$ such that $\lambda = (p-1)(q-1)$ and the integer d is calculated with the CRT.

The decryption process is the same as MultiPower-RSA, and it achives a speed gain of approximately 56 times over the RSA cryptosystem [18].

4. A Modified Key Generation Algorithm for Rebalanced-RSA and Rpower-RSA

In this section, we present two new modified key generation algorithms for Rebalanced-RSA and RPower-RSA to minimize the gap between encryption and decryption times. The encryption and decryption operations are performed in the same way as in the variants under consideration.

In standard RSA, key generation begins with selecting an integer e, followed by computing d. In CRT-RSA, e is selected first, and then d_p and d_q are computed. Rebalanced-RSA, on the other hand, reverses the process: d_p and d_q are chosen first, followed by the computation of e. The modified Rebalanced-RSA algorithm changes this sequence slightly, starting with the selection of d_p, then computing e, and finally d_q. This adjustment ensures that the bit size of e and one of the decryption keys is approximately equal, balancing encryption and decryption times. A similar approach is applied to RPower-RSA, achieving comparable improvements.

4.1. Modified key generation algorithm for Rebalanced-RSA

To achieve the encryption and decryption times close for Rebalanced-RSA we first select d_p with a very short bit size such that $\gcd(d_p, p-1) = 1$. Then, the integer e and d_q are

calculated consecutively. The key generation scheme is as follows:

A distinct prime number p and q are chosen, and $n := p \cdot q$ is calculated. A random short integer d_p is chosen such that

$$\gcd(d_p, p-1) = 1 \quad (4.1)$$

.

The integer e is then calculated as follows:

$$e := {d_p}^{-1} \bmod (p-1) \quad (4.2)$$

If $\gcd(e, q-1) \neq 1$, the integer d_p is reselected in the previous step. Next, the integer d_q is calculated as:

$$d_q := e^{-1} \bmod (q-1) \quad (4.3)$$

The values $\{e, n\}$represent the public keys, and the values $\{\mathrm{p}, \mathrm{q}, \mathrm{d_p}, \mathrm{d_q}\}$ represent the private keys. The encryption process follows the same steps as in the standard RSA, and the decryption process is identical to that of CRT-RSA.

We can validate the keys for the presented algorithms as follows:

From equation (4.1) and (4.2), we have:

$$e^{-1} := \left({d_p}^{-1}\right)^{-1} \bmod (p-1)$$

As a result, we obtain:

$$d_q \equiv e^{-1} \bmod (q-1) \quad (4.4)$$

Therefore, from equations (4.2) and (4.3), we can deduce that the key generation process of the proposed algorithm is equivalent to the private key generation process in CRT-RSA, as described in section 3.1.

Numerical Example:

Key generation:

$p =$ 170141183460469231731687303715884105757 (128 bits)

$q =$ 255211775190703847597530955573826158773 (128 bits)

$n =$ 43422033463993573283839119378257965483172939936490554038560393687183405356161(255 bits)

$d_p =$ 49157 (16 bits)

Using the formula provided in equation (4.2),

$e =$ 708088030440075582880741066261073099 57 (126 bits)

Using the formula provided in equation (4.3),

$d_q =$ 938525285308498399011302239357892058 5 (123 bits)

Encryption:

$M =$ 100000000000000000000000000000000000000000

Using the formula provided in equation (1.1),

$C =$ 2867984525195645912899553497828015699893872953197401515209673598356851516618

Decryption:

Using the formula provided in equations (3.3), (3.4), (3.5) and (3.6).

$M_p =$ 14929408269765384134156348142057947121 5

$M_q =$ 2343646744278884572074071332785215236 8

Finally, the plain text M is obtained using equation (3.2).

4.2. Modified key generation algorithm for RPower-RSA

Additionally, the key generation scheme described in Section 4.1 is applicable to RPower-RSA. For $\boldsymbol{n := p^k . q}$, the integer $\boldsymbol{d_p}$ is first chosen such that $\gcd(\boldsymbol{d_p}, \boldsymbol{p-1}) = \boldsymbol{1}$, and then the integers $\boldsymbol{e}$ and $\boldsymbol{d_q}$ are calculated sequentially using equations (4.2) and (4.3). This key generation process ensures the encryption and decryption efficiency for RPower-RSA, like the modifications made for Rebalanced-RSA.

Numerical Example:

Key generation:

$p =$ 9223372036854775837 (64 bits size)

$q =$ 255211775190703847597530955573826158773 (128 bit size)

$n =$ 217110167319967867784465224337892895659474221181287517056927221853532099918 37(254 bit size)

$d_p =$ 49157 (16 bit size)

$e =$ 3321817412382304685 (62 bit size)

$d_q =$ 170151490439940554373010335073770611753 (128 bit size)

Encryption:

For $M =$ 100000000000000000000000000000000000000000

$C =$ 112961980218424752267915390599180942945021316132734991674616705352696093333 98

Decryption:

$C_p =$ 4957498867100872130

$M_p =$ 4788622440386139474

$C_q =$ 22020261606487972598132348770903966498z

$M_q =$ 46740767562549943696292732620779807853

Using the formula provided in equation 2.8,

$M_{p^2} =$ 467407675625498811064900905242710835 50

Finally, the plain text M is obtained using equation (3.7).

4.3. Security and Validity

If $M < p$ or $M < q$, then $M_p = M_q = M$. In this case, the proposed scheme becomes vulnerable to an active attack, as discussed in references [20-21]. Therefore, it is essential to ensure that M satisfies the conditions $M > p$ and $M > q$ to maintain the security of the scheme.

For the modified Rebalanced-RSA, from equations (3.1), (4.3), and (4.4), we can observe that the theoretical validity of the keys is equivalent to the validity of the Rebalanced-RSA key generation algorithm. Considering the validity of the Rebalanced-RSA encryption and decryption processes, the formal proof of their correctness in the presented algorithm can be easily established. Similarly, the validity of the modified RPower-RSA encryption and decryption processes is equivalent to RPower-RSA.

5. Comparison

The encryption and decryption times in Rebalanced-RSA and RPower-RSA are generally longer than in standard RSA due to the increased bit size of the public key exponent e, which is equal to the bit size of φ. However, the key generation algorithms presented in Section 4 address this by modifying the bit sizes of the keys. In the proposed method, the bit size of e is approximately equal to the bit size of p, resulting in a shorter encryption time compared to the original variants. Additionally, the bit size of one decryption key is approximately equal to the bit size of q, leading to encryption and decryption times that are closer to each other.
In the modified key generation algorithm for Rebalanced-RSA, the bit size of e being roughly equal to the bit size of p significantly reduces the encryption time compared to the original Rebalanced-RSA. Likewise, the bit size of one decryption key being approximately equal to q ensures encryption and decryption times are well-balanced. In the modified RPower-RSA key generation algorithm, the same principle is applied. Furthermore, when computing M_{p^k}, the exponent e is used, enhancing the efficiency of the encryption process compared to the original RPower-RSA. These modifications collectively improve the overall performance of both encryption and decryption operations.
To compare RSA and its variants with the two new key generation algorithms presented in this paper, a table containing the key lengths was created for $n = 1024$ bit size.

Table 5. 1: Key Length of Algorithms**(bit size)**

Algorithms	Primes	Public key	Private key(s)
RSA	$p = q = 512$	$e = 16$	$d = 1024$
CRT-RSA	$p = q = 512$	$e = 16$	$dp = dq = 512$
MultiPrime-RSA (for k=4 primes)	$p1 = p2 = p3 = p4 = 256$	$e = 16$	$d1 = d2 = d3 = d4 = 256$
MultiPower-RSA (for p^3q)	$p = q = 256$	$e = 16$	$dp = dq = 256$
ReBalanced-RSA	$p = q = 512$	$e = 1024$	$dp = dq = 16$
Modified Rebalanced-RSA	$p = q = 512$	$e = 512$	$dp = 16, dq = 512$
Rprime-RSA	$p1 = p2 = p3 = p4 = 256$	$e = 1024$	$d1 = d2 = d3 = d4 = 16$
Rpower-RSA (for p^3q)	$p = q = 256$	$e = 512$	$dp = dq = 16$
Modified Rpower-RSA (for p^3q)	$p = q = 256$	$e = 256$	$dp = 16, dq = 256$

As shown in Table 5.1, in the new algorithms, when one of the private keys is selected to be very short, the size of the public key is reduced. Additionally, the bit size of the other private key is approximately equal to the bit size of e in this scheme. As a result, the encryption and decryption times will be closer to each other.

6. Conclusion

In this paper, we propose two novel key generation algorithms as modifications to Rebalanced-RSA and RPower-RSA. The first proposed algorithm reduces the encryption time compared to Rebalanced-RSA, while maintaining a balance between encryption and decryption times, ensuring a more efficient computational effort. Similarly, the second proposed algorithm reduces the encryption time relative to RPower-RSA, with encryption and decryption times remaining approximately equal, thereby demonstrating improved efficiency and a well-balanced computational performance. Also, when calculating the number $\boldsymbol{M_{p^2}}$ in eq. (3.8) for $\boldsymbol{k = 2}$, the number $\boldsymbol{e}$ is used as the exponent. So, the modified RPrime key generating algorithm further improves encryption process compared to original RPower-RSA.

REFERENCES

[1] R. L. Rivest, A. Shamir and L. Adleman, "A method for obtaining digital signatures and public key cryptosystem," Communications of the ACM, vol. 21, no. 2, pp. 120-126, 1978.

[2] D. Boneh and H. Shacham, "Fast Variants of RSA," CryptoBytes, vol. 5, no. 1, pp. 1-9, 2002.

[3] J. J. Quisquater and C. Couvreur, "Fast decipherment algorithm for RSA public-key cryptosystem," Electronics Letters, vol. 18, no. 21, pp. 905-907, 1982.

[4] A. Fiat, "Batch RSA," Journal of Cryptography, vol. 10, no. 2, pp. 175-195, 1995.

[5] M. J. Wiener, "Cryptoanalysis of sort RSA secret exponents," IEEE Transactions on Information Theory, vol. 36, pp. 553-558, 1990.

[6] T. Collins, D. Hopkins, S. Langford and M. Sabin, "Public Key Cryptographic Apparatus and Method". Us Patent #5,848,159, Jan. 1997.

[7] T. Takagi, "Fast RSA-Type Cryptosystem modulo p^k.q," Advances in Cryptography - CRYPTO'98 Springer, vol. 1462, pp. 318-326, 1998.

[8] C. A. M. Paixao and D. L. G. Filho, "An efficient variant of the RSA cryptosystem," 2005.

[9] S. Verma and D. Garg, "Improvement in RSA Cryptosystem," Journal of Advences Information Technology, vol. 2, no. 3, 2011.

[10] J. Gordon, "Strong RSA keys," Electronics Letters, vol. 20, no. 12, pp. 514-516, 1984.

[11] D. M. Gordon, "A Survey of Fast Exponentiation Methods," Journal of algorithms, vol. 27, no. 1, pp. 129-146, 1998.

[12] Ç. K. Koç, "High-Speed RSA Implementation," RSA Laboraties, Redwood City, CA 94065-1031, 1994.

[13] P. L. Montgomery, "Modular multiplication without trial division," Mathematics of Computation, vol. 44, no. 170, pp. 519-521, 1985.

[14] D. Coppersmith, "Small Solutions to Polynomial Equations and Low Exponent RSA Vulnerabilities," Journal of Cryptology, vol. 10, pp. 233-260, 1997.

[15] D. Boneh and G. Durfee, "Cryptoanalysis of RSA with Private Key d less than N0.292," IEEE Transactions on Information Theory, vol. 46, no. 4, pp. 1339-1349, 2000.

[16] İ. Okumuş and E. Çelik, "Classification of factors that affect the speed of the RSA cryptosystem," in Georgian Mathematical Union Third International Conference, Batumi, GEORGIA, 2012, September 2-9.

[17] T. Takagi, "A Fast RSA_Type Public-Key Primitive Modulo pkq Using Hensel Lifting," IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences, Vols. E87-A, no. 1, pp. 94-101, 2004.

[18] D. Garg and S. Verma, "Improvement over Public Key Cryptographic Algorithm," in 2009 IEEE International Advance Computing Conference (IACC 2009), Patiala, 2009.

[19] İ. Okumuş, The Factors Affecting of RSA Cryptosystem, Ataturk University: Ph. D. Thesis, 2012.

[20] A. Shamir, "RSA for paranoids," CryptoBytes (The technical Newsletter of RSA Laboratories), vol. 1, pp. 1-4, 1995.

[21] G. Horng, "Identification Scheme Based on Shamir's 'RSA for Paranoids," Electronic Letters, vol. 35, no. 22, pp. 1941-1942, 1999.

[22] Kenneth H. Rosen, "Discrete Mathematics Applications", Eighth Edition, New York, NY : McGraw-Hill, 2019.

.