Quantum Rotational Cryptanalysis for Preimage Recovery of Round-Reduced Keccak

Runsong Wang[1], Xuelian Li[1], Juntao Gao[2], Hui Li[1] and Baocang Wang[2]

[1] School of Mathematics and Statistics, Xidian University, Xian, Shaanxi, China
[2] School of Telecommunication and Engineering, Xidian University, Xian, Shaanxi, China

Abstract. This paper considers the capability of 4-round Keccak-224/256/384/512 against the cryptanlysis involved by the quantum algorithm. In order to effectively find the corresponding rotational number for the rotational counterpart of preimage, we first establish a probabilistic algorithm based on the Grover search to guess a possible rotational number by using the fixed relations of bits pairs in some coordinates. This is committed to achieving that each iteration of searching the rotational counterparts contains only one run of 4-round Keccak variant applied for the verification, which can reduce the attack complexity in the quantum setting. Based on finding the rotational number under an acceptable randomness, we construct two attack models to focus on the recovery of preimage. In the first model, the Grover's algorithm serves as finding out a rotational counterpart of the preimage. Through 64 attempts of checking the correct rotational number, the desired preimage can be obtained. In the second model, we abstract the finding of rotational counterparts into searching vertexes on a hypercube, and then, the SKW quantum algorithm is used to deal with the finding of the vertexes acted as rotational counterparts. Compared to the recent works in the classical setting, we greatly reduce the attack complexity of preimage recovery. Furthermore, the first attack model is superior to the generic quantum preimage attack for 4-round Keccak-224/256/384/512, and the second model has slightly lower attack effect but more practicality on the 4-round Keccak-512/384, that is, the model is exponentially easier to implement in quantum circuit than both our first attack model and the generic quantum preimage attack.

Keywords: Keccak · Rotational Cryptanalysis · Preimage Attack · Grover's Algorithm · SKW Algorithm

1 Introduction

Since the security flaws of SHA-1 and SHA-2, the National Institute of Standards and Technology (NIST) announced a public contest aiming at the standardization of next generation cryptographic hash function in 2007. The Keccak sponge function family [BDPVA11] became a candidate for the SHA-3 competition in 2008 [Sha] and eventually was selected as the winner of competition in 2012. In 2015, Keccak has been standardized as the Federal Information Processing Standards (FIPS) by NIST [Dwo15].

In recent years, there are many works of preimage attack on round-reduced Keccak in the field of classical cryptanalysis. Among these preimage attacks, Bernstein et al. proposed an 8-round preimage attack [Ber10], which has the higher computational complexity and more memory consumption compared with the parallel exhaustive search. By introducing the cross-linear structures, Li et al. proposed a preimage attack on 3-round Keccak in [LSLW17]. With the aid of rotational cryptanalysis, Morawiecki et al. mounted a preimage attack on 4-round Keccak [MPS13]. Guo et al. linearized the underlying permutation of Keccak and gave a preimage attack on 4-round Keccak in [GLS16]. The allocating

approach is introduced in preimage attack on 4-round Keccak by Li et al. [LS19], and improved by He et al. [HLY21]. The main idea of this attack is to allocate the complexity into two stages in order that fewer constraints are considered and the complexity is lowered in each stage. More recently, with the help of solving the Boolean multivariate quadratic (MQ) system, Wei et al. [WWF+21] obtained the best attack complexity so far for the preimage attacks on the 4-round Keccak-256 and 4-round Keccak-224.

As for the quantum cryptanalysis, the known attacks are mainly classified into two models [HS18, KLLNP15, Gag17]: Q1 and Q2 models. In the Q1 model, it is assumed that the attacker is equipped with a quantum computer to perform any offline computation and can only have an access to make online classical queries. In the Q2 model, on top of the help of quantum computer, the attacker is allowed to make online superposition queries to a cryptographic primitive. In recent years, many interesting results have appeared under the Q2 model. For instance, Simon's algorithm was first used by Kaplan et al. for the forgery attack on MACs in [KLLNP16]. Leander et al. combined the Grover's algorithm and Simon's algorithm to greatly reduce the effective key-length of FX-construction in [LM17]. The works described above are based on the following assumption that cryptographic primitives in the quantum setting will provide a superposition query interface for the attacker. In practice, the designers of cryptographic primitives do not have enough motivation to expose such powerful interfaces to attacker. Therefore, Bonnetain et al. proposed the improvement of Grover meets Simon in [BHNP+19] to bring the work of Leander et al. into the Q1 model. More recently, cryptographers increasingly payed attention to the keyless primitives which are intrinsically belong the Q1 model. Hosoyamada et al. mounted a quantum rebound attack to find the collision of AES-Like hash function in [HS20] and then a better version was given by Dong et al. in [DSS+20]. Subsequently, Hosoyamada et al. further proposed a quantum collision attack for SHA-2 in [HS21]. However, previous to our achievements presented in this paper, there had been no preimage attack on Keccak based on the quantum algorithms. The latest works of SHA-3 is proposed in [ADMG+16] which estimated the cost of generic quantum preimage attacks on SHA-3.

Our contributions. Firstly, we show the original preimage attack in [MPS13] cannot be simply generalized to the quantum setting. To determine the correctness of the guessed rotational counterpart for the certain preimage, the attacker needs to find the corresponding rotational number by verifying whether the fixed relations at some positions are all satisfied. However, it is noteworthy that this satisfaction is accompanied by some randomness, which is deeply related to the number of these positions. Consequently, for a guessed rotational counertpart, the attacker needs to check all the 64 values of rotational number one by one. Once for some value that leads to the satisfaction of all the fixed relations, a verification with a high-cost running of Keccak must be required to verify the correctness of the guessed rotational counterpart, resulting in the increase of attack complexity. In the quantum setting, owing to the generally used quantum technology called *the uncompute trick*, this heavy workload about these verifications needs to be doubled, which is absolutely unbearable.

In order to show that the rotational cryptanalysis can still derive a better quantum preimage attack compared with the generic quantum preimage attack, we need to greatly reduce the number of the high-cost verifications in the quantum setting. We design a unitary oracle operator by using the fixed relations of bits pairs at some positions, which can be used to mark a possible rotational number for a given guessed rotational counterparts. This allows us to guess the corresponding rotational number of a rotational counterpart with a high probability by using the Grover's algorithm. Therefore, our core idea is to change the checking of rotational numbers one by one into a unique detection of this high-prob value, which can reduce the number of verifications by 64 times. Under the foundation of this probabilistic algorithm, the Grover's algorithm and the SKW

algorithm are introduced to search the rotational counterparts respectively, so as to give a quadratic speedup on the searching of rotational counterparts, which are interpreted as two preimage attack models: The Grover-with-Grover model and the SKW-with-Grover model. The first attack model lays particular emphasis on improving the attack effect of our algorithm. The second attack model is dedicated to reducing the implementation threshold of quantum preimage attack.

The following Table 1 summarizes the best known preimage attacks on the round-reduced Keccak. Compared to the recent works in the classical setting, our attack effects on 4-round Keccak-512 and 4-round Keccak-384 are roughly 2^{251} and 2^{186} times better than that given by Morawiecki et al. in [MPS13]. Meanwhile, For the 4-round Keccak-256 and 4-round Keccak-224, our attack effects are still roughly 2^{86} and 2^{70} times better than that given by Wei et al. in [WWF+21]. Compared to the generic quantum preimage attack, our quantum preimage attack on 4-round Keccak-512 has roughly 4 times better attack effect. Even for the 4-round Keccak-224, our preimage attack still reduces the attack complexity by at least 25%.

Table 1: Summary of best-known preimage attacks on the Keccak variants.

Setting	Rounds	Variants	Time	RAM	qRAM	Reference
Classical	6/7/8	512	$2^{506}/2^{507}/2^{511.5}$	$2^{176}/2^{320}/2^{508}$	0	[Ber10]
	4	512/384	$2^{506}/2^{378}$	Negligible	0	[MPS13]
	4	256/224	2^{214} (for 256) / 2^{182} (for 224)	Negligible	0	[WWF+21]
	4	256	1^{st}block:2^{193} 2^{nd}block:2^{218}	Negligible	0	[HLY21]
	4	224	1^{st}block:2^{129} 2^{nd}block:2^{192}	Negligible	0	[HLY21]
Quantum	4	512/384/256/224	$2^{257}/2^{193}/2^{129}/2^{113}$	0	Negligible	Grover Search
	4	512/384/256/224	$2^{255.08}/2^{191.49}/2^{128.57}/2^{112.57}$	0	Negligible	Sect. 4

2 Preliminaries

In this section we introduce some backgrounds in order that the next section unfolds smoothly. We first review Keccak, the winner of SHA-3. Then an explicit description is given about the idea of preimage attack proposed in [MPS13]. As two quantum search algorithms, the Grover's algorithm and the SKW algorithm are also depicted in this section. Finally, a brief discussion about the accessibility of qRAM is presented.

2.1 SHA-3 Keccak

A concise description of SHA-3 Keccak is provided in this section. The complete version of specification can be obtained in [Dwo15].

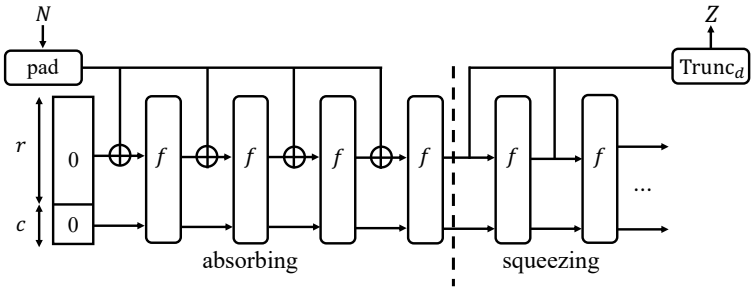

Fig. 1: Sponge construction.

The sponge construction which equips absorbing phase and squeezing phase is the most distinctive feature of Keccak. Figure 1 shows the details of the construction. We

pay close attention to the sponge construction for cryptographic hashing in the paper (This construction can also act as a stream cipher or a pseudorandom bit generator). There are two main parameters bitrate r and capacity c in hash function Keccak. The relation among the state size b, bitrate r and capacity c is given as follows: $b = r + c$, where $b \in \{25, 50.100, 200, 400, 800, 1600\}$. Different values of bitrate r are corresponding to a different security levels of Keccak. If a higher security is desired, a lower bitrate r is required. On the contrary, if a greater efficiency is noticed, a lower capacity c is more useful. For the SHA-3 proposal, the state size is restricted to $b = 1600$, and there are four different variants as the SHA-3 candidates: SHA3-224, SHA3-256, SHA3-384 and SHA3-512. For each case, the capacity c is twice the length of digest L(also called the hash length), i.e., $c = 2L$. Note that the default variants referred through the whole paper satisfy $b = 1600$, in order to keep one with the SHA-3 standard.

For a given resulting input N (The relation between the resulting input N and message M is given as follows: $N = M\|01$), a padding operation to N is required at first such that the length of the input string is a positive multiple of bitrate r. The purpose of the absorbing phase is to ensure all the message blocks are processed by the Keccak-f permutation which consists of 24 iterations of the round function R. In the squeezing phase, the first r bits of output for each Keccak-f permutation are returned as a part of hash value. As long as the L bits have been obtained as the L long hash value, the squeezing phase is finished.

The round function R consists of 5 sub-step θ, ρ, π, χ and τ.

$$R = \tau \circ \chi \circ \pi \circ \rho \circ \theta,$$

$$\theta : A_{(x,y,z)} = A_{(x,y,z)} \oplus \bigoplus_{j \in \{0,\cdots,4\}} \left(A_{(x-1,j,z)} \oplus A_{(x+1,j,z-1)}\right),$$

$$\rho : A_{(x,y,z)} = A_{\left(x,y,z-r_{(x,y)}\right)},$$

$$\pi : A_{(y,2x+3y,z)} = A_{(x,y,z)},$$

$$\chi : A_{(x,y,z)} = A_{(x,y,z)} \oplus \left(A_{(x+1,y,z)} \oplus 1\right) \cdot A_{(x+2,y,z)},$$

$$\tau : A_{(0,0,z)} = A_{(0,0,z)} \oplus RC_z.$$

where "$A_{(x,y,z)}$" denotes a bit value in the state array A, which is indexed by the coordinates (x, y, z), "$\oplus$" denotes the bitwise XOR operation, "$\cdot$" denotes the bitwise AND operation and "RC" denotes the round constant of the corresponding round.

Linear operation θ can provide diffusion for the state array A, which plays an essential role in the round function R. After the operation ρ, each lane of the state array A deploys a bitwise rotation operation with a certain rotational number which called *offset*. The operation π only rearranges the position of the lanes. Step χ is the unique non-linear operation which can be treated as a layer of 5-bits Sboxes and we call the one layer 5-bits χ operation as *One-Row χ Operation*. Although the last step τ is the simplest operation among the five processes, which xores round constant with the first lane, the classical preimage attack based on rotational cryptanalysis have a deep relation with this step. For the five steps described above, the operations on both x-coordinate and y-coordinate are modulo-5 and the operations on z-coordinate are modulo-ω.

2.2 4-Round Preimage Attack in Classical Setting

An important technique which called rotational analysis is a relatively new type of attack improved by Morawiecki et al. [MPS13]. This novel method which mainly follows the evolution of rotational pair of states can be applied to mount a preimage attack on 4-round Keccak with a lower computational complexity than the classical exhaustive search.

Definition 1. [MPS13] A pair of b-bit states $(A, A^{\leftarrow})$ is called a rotational pair if and only if each lane of state $A^{\leftarrow}$ is created by the rotation operation of corresponding lane of state A with a fixed number n called rotational number.

It is notable that for mostly given state arrays A, there are always ω rotational counterparts $n \in \{0, \cdots, \omega - 1\}$ in total. If the state array A is a cyclic pattern, the number of rotational counterparts is smaller than ω.

Definition 2. An m-bit string is called cyclic pattern if and only if the m-bit string can be regenerated by a n-bitwise rotation operation($n < m$).

A simple case of cyclic pattern is alternating '01' and '10'.

Definition 3. [MPS13] Rotational set S_n consists of 2^b pairs of state $(B, B^{\leftarrow})$ which can be generated by applying operations α and β on A and $A^{\leftarrow}$ of all possible rotational pairs respectively.

The "all possible rotational pairs" means that given a rotational number n, there are 2^b rotational pairs.

Definition 4. [MPS13] $P_{(x,y,z)}$ is the probability that for a pair of state $(B, B^{\leftarrow})$ selected from rotational set S_n randomly, we have $B_{(x,y,z)} \neq B^{\leftarrow}_{(x,y,z+n)}$.

For each bit of the state array B, the corresponding probability $P_{(x,y,z)}$ is used to describe the bits relation between state B and $B^{\leftarrow}$. If $P_{(x,y,z)} = 0$, the corresponding bits are equal. If $P_{(x,y,z)} = 1$, the corresponding bits have opposite values and if $P_{(x,y,z)} = \frac{1}{2}$, the bits are independent. Suppose operations α and β are applied on A and $A^{\leftarrow}$ respectively, then there are $25 \cdot \omega$ probabilities $P_{(x,y,z)}$ in total and the bits relations between states B and $B^{\leftarrow}$ are defined.

Definition 5. For any rotational pair $(A, A^{\leftarrow})$, we assume that the original Keccak and modified Keccak are applied on state A and state $A^{\leftarrow}$ respectively. By arranging these $25 \times \omega$ values $P_{(x,y,z)}$ in a fixed order, we can quickly obtain the *Table of Probability Propagation.*

In the modified Keccak, only the round function R does not equip the operation τ. We use M-Keccak to represent the modified Keccak and TPP to represent the *Table of Probability Propagation* in the rest of this paper. Both the original Keccak and M-Keccak only involve bitwise XOR operation, bitwise AND operation, bitwise NOT operation and bitwise rotation operation. With the previous work [MPS13], there have been concrete conclusions on how the probability $P_{(x,y,z)}$ is changed by these bitwise operations described above:

Proposition 1. *[MPS13] Given the input bits a and b, and denote the output bit as "out". After the bitwise XOR operation, probability $P_{(out)}$ can be acquired as follows:*

$$P_{out} = P_a + P_b - 2P_aP_b.$$

Proposition 2. *[MPS13] Given the input bits a and b, and denote the output bit as "out". After the bitwise AND operation, probability $P_{(out)}$ can be acquired as follows:*

$$P_{out} = \frac{1}{2}\left(P_a + P_b - P_aP_b\right).$$

Because the bitwise NOT operation flips the input bits meanwhile, the probability $P_{(x,y,z)}$ is not affected. Throughout the bitwise rotation operation, the positions of two input bits remain relatively stationary, so the probability $P_{(x,y,z)}$ remains unchanged. Recall that each step of the round function R consists of four bitwise operations mentioned

above. Therefore we can obtain the TPP directly. It is notable that when the step τ is applied to the state B and nothing is done on the state $B^{\leftarrow}$, the probability $P_{(x,y,z)}$ is changed if and only if the corresponding bit of round constant is '1', and we can obtain the probability $P_{(x,y,z)}$ as follows:

$$P_{(x,y,z)} = 1 - P_{(x,y,z)}.$$

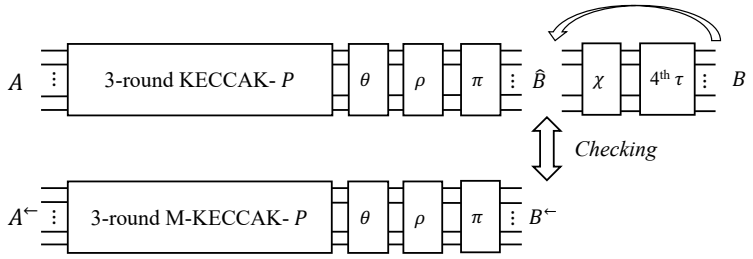

Fig. 2: The Core of 4-round preimage attack.

Now, the core idea of 4-round preimage attack on round-reduced Keccak can be presented clearly: Recall that we want to find the preimage A from a given hash value B output by the original 4-round Keccak. Owing to the step χ and τ are reversible and can be calculated independently for each lane of state array, we first obtain the state value $\hat{B}$ from the hash value B. As shown in Figure 2, suppose that the preimage A is applied to the original Keccak and the rotational counterpart $A^{\leftarrow}$ is applied to M-Keccak, and then we calculate the TPP to ensure that there are still enough coordinates whose probability value $P_{(x,y,z)}$ is equal to 0 or 1, which implies the bits pairs between the state $\hat{B}$ and $B^{\leftarrow}$(the state pair $\left(\hat{B}, B^{\leftarrow}\right)$) have fixed relations at these positions. We call these coordinates satisfying the above properties eigenpoints. In practice, the preimage attack focuses on finding the rotational counterpart $A^{\leftarrow}$ which can form a rotational pair with the preimage A. For a guess of rotational counterpart $A^{\leftarrow}$, we run the 3-round M-Keccak and step θ, ρ, π of 4^{th} round, then check out whether the state pair $\left(\hat{B}, B^{\leftarrow}\right)$ satisfies all the relations of eigenpoints defined by the probability values $P_{(x,y,z)}$ for a certain rotational number n. Once all the relations of eigenpoints are satisfied, we rotate back the guess value of $A^{\leftarrow}$ by n bits. By running 4-round Keccak on the rotated state to check whether the result is equal to the hash value B, we can verify the correctness of the rotational counterpart $A^{\leftarrow}$ and the rotational number n, so as to finally derive the preimage A from the correct $A^{\leftarrow}$ and n. For a given hash value B, since there are ω rotational counterparts of the preimage A, the attack complexity of the preimage attack is always expected lower than the classical exhaustive search.

2.3 Grover's Algorithm

Grover's problem [Gro96] Given a database X, approximately $\lceil \log_2(|X|) \rceil$ qubits are required to generate the equal superposition $\frac{1}{\sqrt{|X|}} \sum_{x=1}^{|X|} |x\rangle$ to represent all the elements of X. Given an oracle accessing to a function $f : X \to \{0,1\}$ which the corresponding oracle operator U_f can act f on $\sum_{x=1}^{|X|} |x\rangle$ in $O(1)$ time, the solutions $x \in X$ such that $f(x) = 1$ are expected to be obtained.

In the classical setting, suppose there are $M = 2^{\lambda}$ target elements $x \in X$ to satisfy $f(x) = 1$, the exhaustive search takes an average of accessing $|X|/M$ times function f to get one. However, in the quantum setting, the principle of search is totally different. With

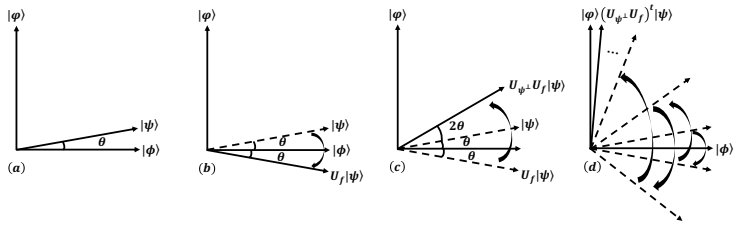

Fig. 3: The iterations of Grover search.

the foundation of quantum parallel, Grover's algorithm can increase the probability amplitudes of target elements and decrease the probability amplitudes of non-target elements. As shown in the Figure 3, the equal superposition state $|\psi\rangle = \frac{1}{\sqrt{|X|}} \sum_{x \in X} |x\rangle$ is divided into the sub-superpositions $|\varphi\rangle$ and $|\phi\rangle$, where $|\varphi\rangle$ denotes the superposition of all the M target elements and $|\phi\rangle$ denotes the superposition of remaining elements. Both two sub-superposition $|\varphi\rangle$ and $|\phi\rangle$ form a set of orthogonal basis in two-dimensional Hilbert space. The $|\psi\rangle$ and $|\phi\rangle$ are at an angle θ in the space. The Grover iterate G consists of a diffusion operator $U_{\psi^\perp} = -I + 2|\psi\rangle\langle\psi|$ and an oracle operator U_f as follows: $G = U_{\psi^\perp} U_f$, where the function of U_f and $U_{\psi^\perp}$ is to flip the phase of a certain superposition state along state $|\phi\rangle$ and $|\psi\rangle$ respectively. It is noteworthy that the diffusion operator $U_{\psi^\perp}$ is applied to the 2^n-dimensional search space. Every time applying Grover iterate G on intermediate state, the current state is closer to $|\varphi\rangle$. After roughly $t = \sqrt{|X|/M}$ times Grover iterations, the amplitude of the superposition of all the target elements approximates 1. Further, we can measure the appropriate result with the probability $P \to 1$.

2.4 SKW Algorithm

The SKW algorithm [SKW03], which is also known for providing a quadratic speedup, is always applied to search the marked nodes in an undirected graph. Concretely, it is based on the quantum discrete time random walk [MR02, Kem02] on the n-cube, i.e. the hypercube of dimension n. As a graph with $N = 2^n$ nodes, the hypercube is the generalization of 3-cube and has $M = 2^\lambda$ marked nodes in total. Each node can be labelled by a n-bit binary string. Two nodes on the hypercube can be connected by an edge if and only if the corresponding bit strings $\overrightarrow{x}$ and $\overrightarrow{y}$ satisfy $|\overrightarrow{x} - \overrightarrow{y}| = 1$, where $|\overrightarrow{x}|$ is the Hamming weight of $\overrightarrow{x}$. Therefore, each node has a degree n i.e. it is connected to n other nodes, and the Hilbert space of this algorithm can be denoted as: $\mathcal{H} = \mathcal{H}^n \otimes \mathcal{H}^{2^n}$. Each sate of $\mathcal{H}^n$ is used to describe the direction d which specifies the state of the coin, and

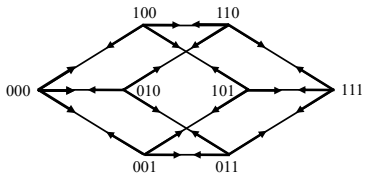

Fig. 4: The $n = 3$ hypercube.

each state of $\mathcal{H}^{2^n}$ is used to describe the current position $\overrightarrow{x}$ on the hypercube. Similar to Grover's algorithm, this algorithm also equips an iterative operator U called the unitary evolution operator which consists of the shift operator S and the coin operator C as follows: $U = SC$.

Specifically, the shift operator S maps the state $|d, \overrightarrow{x}\rangle$ onto the state $|d, \overrightarrow{x} \otimes \overrightarrow{e_d}\rangle$, and can be written as: $S = \sum_{d=0}^{n} \sum_{\overrightarrow{x}} |d, \overrightarrow{x} \otimes \overrightarrow{e_d}\rangle \langle d, \overrightarrow{x}|$, where $\overrightarrow{e_d}$ is the d-th basis vector on the hypercube. The coin operator C applies a marking coin $C_1 = -I$ to the marked nodes and a different coin $C_0 = U_{\psi^\perp}$ to the unmarked nodes, where I denotes the identity operator and $U_{\psi^\perp}$ denotes the diffusion operator applied only to the n-dimensional coin space. For instance, without loss of generality we assume one of the marked nodes is $|\overrightarrow{x}_{marked}\rangle = \overrightarrow{0}$, and the coin operator C becomes: $C = U_{\psi^\perp} \otimes I + \left(-I - U_{\psi^\perp}\right) \otimes |\overrightarrow{0}\rangle \langle \overrightarrow{0}|$. Then the evolution operator U is essentially a perturbed coin quantum walk on the hypercube. After roughly $t = \sqrt{N/M}$ times iterations, we can expect to get one of the marked nodes with probability $P = \frac{1}{2} - O(1/n)$.

2.5 The Uncompute Trick

The uncompute trick [DHM+18] is a common technique in quantum algorithms, which is used to carry out a computation and then retrieve the initial state $|0\rangle$. According to the no-deleting theorem [PB99], there is no single-qubit unitary operator that sets an arbitrary qubit state to $|0\rangle$. In practice, both the Grover's algorithm and the SKW algorithm require the implementation of oracle opertor U_f to map the state $|x\rangle |0\rangle |0\rangle$ to $|x\rangle |g(x)\rangle |f(x)\rangle$ in each iteration, where the $|g(x)\rangle$ is a *garbage state* in a *working register*. To make the oracle operator U_f on the identical *working register* still work in the next iteration of quantum search as if the maping the state $|x\rangle |0\rangle |0\rangle$ to $|x\rangle |g(x)\rangle |f(x)\rangle$, we perform the uncompute tirck to reset and reuse the *working register*.

2.6 The Accessibility of qRAM

The assumption that qRAM can be implemented is highly controversial in the field of quantum cryptanalysis. The recent progress of qRAM is presented by Giovannetti et al. in which the author proposes an architecture that exponentially reduces the requirements for a memory call [GLM08]. The current bottleneck of research on qRAM is very similar to the research on RAM decades ago. In the realm of cryptanalysis, attackers are usually assumed to equip excellent capabilities and powerful computing resources. That's why there are many recent papers such [HS20,DSS+20,BHNP+19] based on the assumption that qRAM is available have made many outstanding contributions in the quantum cryptanalysis. In our work, the qRAM is also assumed to be accessible and we only require a negligible cost of qRAM to store the information about the eigenpoints.

3 Quantum Preimage Attack on Round-Reduced Keccak

We begin with new observations on the preimage attack of 4-round Keccak in the classical setting, and then elaborate two efficient quantum preimage attack models on the 4-round Keccak. In the first model, we only focus on the attack effect achieved by using the Grover's algorithm. In the second model, we propose an idea that the SKW algorithm serves as searching on a hypercube to avoid the application of diffusion operator on an exponentially large space and lower the implementation threshold of our algorithm.

3.1 New Observations

When taking the rotational cryptanalysis based preimage attack into the quantum setting, thanks to the special mechanism of quantum computing, we can not only apply the quantum search algorithm to deal with the process of finding rotational counterpart $A^{\leftarrow}$, so as to obtain the quadratic speedup on the main loop, but also the quantum algorithm can

be used to greatly reduce the verifications of passing all the relations of eigenpoints. Recall that the aforementioned algorithm described in Section 2.2 provides an efficient idea that the attacker can transform the process of finding the preimage A into the process of finding some special bit strings $A^{\leftarrow}$ which have a definite relation with the preimage A. The attack complexity Θ of the algorithm can be roughly divided into two parts. $\mathbb{A}$: Running the Keccak in each main loop. $\mathbb{B}$: Running the Keccak for the case that all the relations of eigenpoints are satisfied. In the theoretical attack, the attack complexity Θ is mainly determined by the second part $\mathbb{B}$, which means the second part $\mathbb{B}$ will heavily affect the attack effect of the aforementioned algorithm.

When guessing the rotational counterpart $A^{\leftarrow}$, there are only two cases to happen here: either the guessed string is indeed a rotational counterpart $A^{\leftarrow}$ or the guessed string is meaningless to the preimage A. Specifically, in the first case, suppose we guessed one of the ω rotational counterparts $A^{\leftarrow}$ with the corresponding rotational number $n = i$, and this rotational counterpart can be denoted as $A_i^{\leftarrow}$, where $i \in \{0, \cdots, \omega - 1\}$. However, the attacker can ensure the correctness of $A_i^{\leftarrow}$ only by running the Keccak to obtain the corresponding state $B^{\leftarrow}$ and then testing the rotational number i times untill in the turn of $n = i$ by checking all the eigenpoints. In this procedure, it could be the case that for this rotational counterpart $A_i^{\leftarrow}$, there is a pseudo rotational number $n = j < i$ such that $A_i^{\leftarrow}$ passes the checking of all the relations of eigenpoints, but can not obtain the preimage by rotating $A_i^{\leftarrow}$ back j bits. A simplified version of the checking is shown in the Figure

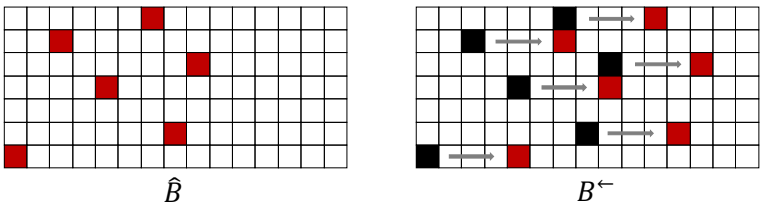

Fig. 5: The checking for the rotational number.

5. When checking whether the state pair $(\hat{B}, B^{\leftarrow})$ meets all the relations of eigenpoints, the attacker needs to match the bit values of red blocks in the state $\hat{B}$ with the bit values of red blocks in the state $B^{\leftarrow}$. It is noteworthy that the red blocks shown in $\hat{B}$ or the black blocks shown in $B^{\leftarrow}$ can be used to describe the positions of eigenpoints, and the red blocks(the positions of matched bit values) in $B^{\leftarrow}$ are obtained by rotating the black blocks(the positions of eigenpoints) back n bits. Without loss of generality, we use $(\overline{x}, \overline{y}, \overline{z})$ to denote a eigenpoint and $P_{(x,y,z)} = 0$. Therefore the bit value of $\hat{B}$ in position $(\overline{x}, \overline{y}, \overline{z})$ is 100% equal to the bit value of $B^{\leftarrow}$ in position $(\overline{x}, \overline{y}, \overline{z} + i)$. Now, we need to disscuss the probability that the bit value of $\hat{B}$ in position $(\overline{x}, \overline{y}, \overline{z})$ is equal to the bit value of $B^{\leftarrow}$ in position $(\overline{x}, \overline{y}, \overline{z} + j)$, $j < i$. Obviously, the probability can also be used to describe the case where the bit value of $B^{\leftarrow}$ in position $(\overline{x}, \overline{y}, \overline{z} + i)$ is equal to the bit value of $B^{\leftarrow}$ in position $(\overline{x}, \overline{y}, \overline{z} + j)$, which is completely random. Suppose there are ξ eigenpoints, for the $n = j < i$, the probability that the state pair $(\hat{B}, B^{\leftarrow})$ satisfies all the relations of eigenpoints is equal to $1/2^{\xi}$. In the second case, suppose we couldn't guess a rotational counterpart $A^{\leftarrow}$ which means there is no rotational number for the guessed random string. Therefore, the corresponding state $B^{\leftarrow}$ which is derived from $A^{\leftarrow}$ makes no sense to the state $\hat{B}$. For each $n \in \{0, \cdots, \omega - 1\}$, the probability that the bit values of $\hat{B}$ in the ξ positions (x, y, z) defined by all the eiginpoints are matched with the bit values of $B^{\leftarrow}$ in the positions $(x, y, z + n)$ is also equal to $1/2^{\xi}$. In the theoretical preimage attack, heavy calculations of running the 4-round Keccak $\omega \times \frac{(1+\omega)\cdot\omega}{2} \times 2^{-\xi} + (2^L - \omega) \times \omega \times 2^{-\xi}$ times on average are usually required to verify these cases where all the relations of eigenpoints are met, as a result of the less number ξ of eigenpoints.

Fortunately, these heavy calculations can be 64 times decreased in the quantum setting in order that the affect of $\mathbb{B}$ on the attack complexity can be reduced. Different from the traversal of classical exhaustive search, the keys of quantum search are parallel computing and iteration. Therefore, we use the Grover's algorithm to find the corresponding rotational number n for a rotational counterparts $A^{\leftarrow}$ and return an expected value with a relatively high probability in order that each iteration of searching rotational counterparts $A^{\leftarrow}$ contains only one verification which requires 4-round Keccak to determine the correctness of the rotational counterpart $A^{\leftarrow}$ and rotational number n. The randomness on the return value of searching the corresponding rotational number n results in the less marked rotational counterparts in each iteration of searching the $A^{\leftarrow}$, which requires a little additional iterations on the $A^{\leftarrow}$ such that the desired preimage can still be obtained and always keeps a lower cost compared with the quantum exhaustive search on the preimage A. Laterly, we will show that even in the worst case, attacking the variant Keccak-224, the attack complexity with respect to our new algorithm is still better than the generic quantum preimage attack.

3.2 The Detailed Settings and Precomputation

Detailed settings. Since the most expensive step of round function R is the unique non-linear operation χ which can be treated as a layer of 5-bit Sboxes (Note that the implementation of χ on the quantum circuit does not require the access to Sboxes which requires additional cost of qRAM), a reasonable assumption can be obtained that the $5 \cdot \omega \cdot r$ times *One-Row* χ *Operations* can be treated as one time computation of r-round Keccak. The consumption of the remaining steps are considered negligible.

Now, we first give a quantum version of precomputation to store the information of eigenpoints. It is worthy to mention that if there still are eigenpoints after $r-1$ round R and θ, ρ, π operations of r^{th} round, we can mount an r-round preimage attack.

Precomputation for the eigenpoints. We perform the following precomputation and store the results of eigenpoint in the qRAM in order that these data are accessible in the form of quantum superposition later to execute the checking of eigenpoints. The pseudocode is shown below:

Algorithm 1 Precomputation for the eigenpoints.

Output: l

1: Generate the TPP: T;
2: Record all the eigenpoints (x, y, z) for which $\omega(5y + x) + z < L$ and $P_{(x,y,z)}$ equals 0 or 1;
3: Let l be empty list of size $\xi \cdot (3 + 3 + \varepsilon + 1)$;
4: **for all** ξ coordinates **do**
5: Convert the coordinates (x, y, z) and the corresponding values of $P_{(x,y,z)}$ to a binary string $x_1x_2x_3 \| y_1y_2y_3 \| z_1z_2 \cdots z_\varepsilon \| p$ and store string in l ;
6: **end for**
7: **return** l .

where ξ denotes the number of eigenpoints, and ε denotes the logarithm of ω (The lane size of state array in bits). Since the values of state $\hat{B}$ matched later are in the first L/ω lanes of state arrays, we require the restriction about $\omega(5y + x) + z < L$ described in the third step of precomputation to ensure the collected eigenpoints are also in the first L/ω lane of state arrays.

Complexity analysis for the precomputation. Even in the classical setting, the computational complexity is negligible. Therefore, we also believe that this workload is still negligibly small in the quantum setting. The only quantum memory cost of precompu-

tation is to allocate memory for l. We require roughly $\xi \cdot (3+3+\varepsilon+1) \approx 2^{\zeta}$ qRAM to store the information of eigenpoints which are converted into bit strings.

3.3 Implementation of the Quantum Oracle U_Q for Finding Rotational Number

The core of verifying the correctness of the guessed rotational counterpart $A^{\leftarrow}$ is to judge whether there is a corresponding (pseudo)rotational number n that causes the guessed $A^{\leftarrow}$ to satisfy all the relations of eigenpoint. We use the Grover's algorithm to find the corresponding (pseudo)rotational number n for the derived $B^{\leftarrow}$ from the guessed rotational counterpart $A^{\leftarrow}$. We first define the oracle function $Q(n, B^{\leftarrow}) : \{0,1\}^{\varepsilon} \rightarrow \{0,1\}$ to support the marking of (pseudo)rotational number n. Note that function Q can be converted to the unitary operator U_Q implemented efficiently in the quantum circuit if there exists an efficient classical circuit to compute Q [NC00, DSS+20]. To build the quantum circuit of U_Q, we first construct an efficient reversible classical circuit of Q and then substitute quantum gates for each of the reversible gates involved. The implementation of oracle operator U_Q is shown in the following pseudo-codes:

Algorithm 2 Implementation of oracle operator U_Q.

Input: $|n\rangle |B^{\leftarrow}\rangle |y\rangle$
Output: $|n\rangle |B^{\leftarrow}\rangle |y \oplus Q(n, B^{\leftarrow})\rangle$

1: For the given state $B^{\leftarrow}$ derived from the guessed rotational counterpart $A^{\leftarrow}$, set $flag = 0$;
2: For the given n, check if these matched values of $B^{\leftarrow}$ satisfy all the ξ relations with the corresponding values of $\hat{B}$ by accessing the information of these eigenpoints stored in qRAM. If so, set $flag = 1$. Do nothing otherwise;
3: Return 1 as the value of oracle function $Q(n, B^{\leftarrow})$ if and only if $flag = 1$, and return 0 otherwise;
4: Uncompute the Steps 2.

Then, the Grover iteration operator $G_Q = U_{\psi^{\perp}} U_Q$ can be quickly obtained, and we will embed G_Q into the implementation of the next oracle operator which is dedicated to marking the rotational counterpart $A^{\leftarrow}$. Note that the $B^{\leftarrow}$ is a fixed value as the input of G in every execution of the Grover's algorithm performed on the oracle function Q. In the searching process, the diffusion operator $U_{\psi^{\perp}}$ is applied to the 2^{ε}-dimensional search space. Owing to the scale of 2^{ε} is always small than 2^6, we believe that the implementation of this diffusion operator $U_{\psi^{\perp}}$ on quantum circuits is relatively easy. However what needs to be explained is that this method doesn't work well since the eigenpoints are always not enough. The following Lemma 1 describes how the number of eigenpoints affects the correctness of finding the corresponding rotational number n of rotational counterpart $A^{\leftarrow}$.

Lemma 1. *If the guessed string $A^{\leftarrow}$ is a rotational counterpart, then the Grover search on oracle operator U_Q outputs the corresponding rotational number n with probability $\frac{2^{\xi}}{2^{\xi}+2^{\varepsilon}-1}$.*

Laterly, we will show how the acceptable randomness of finding n weakens the efficiency of marking the rotational counterparts $A^{\leftarrow}$.

3.4 Preimage Attack on Keccak Using Grover's Algorithm Only

Now, we turn to look for one of the ω rotational counterparts $A^{\leftarrow}$ and provide a detailed description about our first attack model: the Grover-with-Grover model. An intuitive

idea is to use the Grover search again to find the correct rotational counterpart $A^{\leftarrow}$. We define the oracle function $F(A^{\leftarrow}) : \{0,1\}^L \rightarrow \{0,1\}$ for the oracle operator U_F which is committed to marking the rotational counterparts $A^{\leftarrow}$. As shown below, we give the pseudo-codes of the implementation of oracle operator U_F:

Algorithm 3 Implementation of oracle operator U_F.

Input: $|A^{\leftarrow}\rangle |y\rangle$
Output: $|A^{\leftarrow}\rangle |y \oplus F(A^{\leftarrow})\rangle$

1: For the given guessed state $A^{\leftarrow}$, set $flag = 0$;
2: Run $r-1$ rounds R and the step θ on the state $A^{\leftarrow}$;
3: Run the Grover's algorithm with certainty on the oracle function $Q(n) : \{0,1\}^{\varepsilon} \rightarrow \{0,1\}$. Let n be the output;
4: Rotate back the guessed state $A^{\leftarrow}$ by n bits and run r rounds R on it to check whether the result is the hash value B. If so, set $flag = 1$. Do nothing otherwise;
5: Return 1 as the value of $F(A^{\leftarrow})$ if and only $flag = 1$. Return 0 otherwise;
6: Uncompute Steps $2-4$.

Note that, the reason why we do not operate the ρ and π of 4^{th} round is that the two operations can not change the value of TPP, and only the positions of coordinates are changed. We still can verify these matched values for the ξ eigenpoints by coding easily. So, the overall framework about our first attack model is clear: we nest the Grover search acting on the oracle operator U_Q into the design of oracle operator U_F to focus on the marking of rotational counterparts $A^{\leftarrow}$. Then, we can find one of the rotational counterparts $A^{\leftarrow}$ by performing the Grover's algorithm on the obtained oracle operator U_F. As shown in step 3 of Algorithm 3, each execution of the Grover iteration operator $G_F = U_{\psi^{\perp}} U_F$ contains a measurment to obtain a possible value of rotational number n. Benefit from the general deferred measurement principle of quantum computation [NC00], we can postpone the measurement in the Grover's algorithm acting on the oracle operator U_Q to the very end of our entire quantum algorithm to avoid the *collapse* driven by the observer effect. The detailed quantum circuit model is shown in the Figure 6.

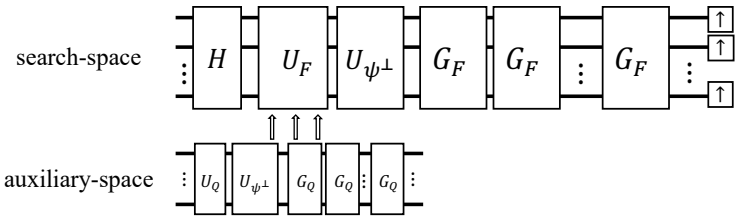

Fig. 6: The Grover-with-Grover model.

Complexity analysis for the Grover-with-Grover model. When we apply the Grover search on the oracle function $F(A^{\leftarrow})$, no additional qRAM consumption is required. And the factors affecting attack complexity for this process are mainly concentrated in the iteration of searching rotational counterparts $A^{\leftarrow}$. For a given cyclic-pattern preimage A , the number of the corresponding rotational counterparts $A^{\leftarrow}$ is always less than a normal one, and for some extreme instances there are only 2 corresponding counterparts. For these cases, the preimage attack based on the rotational cryptanalysis keeps little effect. However, the following lemma tells us that the number of b-bit cyclic patterns is negligible compared with 2^b possible state arrays.

Lemma 2. *For 2^n-bit string $\alpha \in S_\alpha = \{0,1\}^{2^n}, n \in \{1,2,\cdots\}$, there are $2^{2^{n-1}}$ cyclic patterns in total.*

Therefore, it is only necessary to carry out some additional calculations in advance to verify whether the preimage is in those cyclic patterns, and the complexity is far less than the non-cyclic one. If there is no preimage after these calculations in advance, we need to deploy our first attack model on the finding of preimage A. In the step 3 of Algorithm 3, we apply the Grover's algorithm on the oracle operator U_Q to return a possible rotational number n. For a given non-rotational counterpart $A^{\leftarrow}$, the return value n of this Grover search is completely meaningless even there is a value $n \in \{0, \cdots, \omega - 1\}$ that leads to the satisfaction of all the relations of eigenpoint. Therefore, this state $A^{\leftarrow}$ given can not be marked by the oracle operator U_F in the step 4. For a given rotational counterpart $A^{\leftarrow}$, as long as n is not the corresponding rotational number n, there is a $1/2^{\xi}$ probability that $A^{\leftarrow}$ passes all the checking of ξ eigenpoints, resulting in the oracle operator U_Q can not distinguish the possible pesudo rotational number with the really corresponding one. According to the Lemma 1, the return of correct rotational number n for the rotational counterpart $A^{\leftarrow}$ given with the probability $\frac{2^{\xi}}{2^{\xi}+2^{\varepsilon}-1}$. Consequently, in each iteration of G_F the oracle operator U_F can mark roughly $\omega \cdot \frac{2^{\xi}}{2^{\xi}+2^{\varepsilon}-1} = \frac{2^{\xi+\varepsilon}}{2^{\xi}+2^{\varepsilon}-1}$ of all the ω rotational counterparts $A^{\leftarrow}$. The details of cost estimation about our first preimage attack model can be given in the following theorem:

Theorem 1. *Given an L-bit hash value B, suppose there are ξ eigenpoints. With the cost of 2^{ζ}* qRAM*, the preimage A can be recovered with the attack complexity of $\frac{r-1}{r} \cdot \sqrt{\frac{2^{\xi}+2^{\varepsilon}-1}{2^{\xi}}} \cdot 2^{\frac{L-\varepsilon}{2}+2}$.*

Note that the attack complexity shown in Theorem 1 is a conservative estimate, and the concrete effect is better. All the proofs about the conclusions appeared in this section can be found in the Appendix A.

Remark 1. It is notable that the diffusion operator $U_{\psi^{\perp}}$ of G_F is applied to the 2^L-dimensional search space. In practice, even attacks on Keccak-224, the scale of 2^L is theoretically equal to 2^{224}, and the diffusion operator $U_{\psi^{\perp}}$ under this scale is universally considered to be difficult to implement [SKW03]. However, in the recent field of quantum cryptography the difficulty of large scale diffusion operator implementation is generally ignored. In order to improve the applicability of our algorithm, we take into account this difficulty and introduce the second attack model from another perspective based on quantum random walk.

3.5 Preimage Attack on Keccak Using SKW Algorithm

Our main idea is to abstract the process of searching the rotational counterparts $A^{\leftarrow}$ of preimage A as finding some marked nodes in an undirected graph. The oracle operator U_F defined in the Algorithm 3 still plays a key role to mark these nodes acted as rotational counterparts $A^{\leftarrow}$. Then, by using the SKW algorithm based on quantum random walk, we can expect to obtain a correct counterpart $A^{\leftarrow}$. We call the second attack model the SKW-with-Grover model.

Concretely, we use $\Gamma = \{0,1\}^L$ to denote the search space of rotational counterparts $A^{\leftarrow}$. Each element of Γ is an L-bit string representing a node of graph, and we connect two nodes if and only if the Hamming weight $|\overrightarrow{x} - \overrightarrow{y}|$ of corresponding strings $\overrightarrow{x}$ and $\overrightarrow{y}$ is equal to 1. Therefore, each node of Γ is connected to other L nodes, and we can obtain the hypercube of search space Γ. The Hilbert space of our second attack model can be described as $\mathcal{H} = \mathcal{H}^L \otimes \mathcal{H}^{2^L}$. Each state of $\mathcal{H}^{2^L}$ is used to describe the current node on the graph acted as a guessed rotational counterpart, and each state of $\mathcal{H}^L$ is used to determine one of the neighbor nodes. Then we can define the shift operator S for our

algorithm as:

$$S = \sum_{d=0}^{L} \sum_{A^{\leftarrow}=0}^{2^L} |d, A^{\leftarrow} \otimes \overrightarrow{e_d}\rangle \langle d, A^{\leftarrow}| .$$

Recall that the coin operator C applies the identity coin operator $C_1 = -I$ for the marked nodes, and applies the diffusion coin operator $C_0 = U_{\psi^\perp}$ for the unmarked nodes. Therefore we use the oracle operator U_F to assist the coin operator C in deploying the operator I ($U_{\psi^\perp}$) on the marked nodes (unmarked nodes), and obtain the unitary evolution operator $\Lambda = SCU_F$. The following theorem gives the analysis of cost about the SKW-with-Grover model, whose proof is similar to that of the Theorem 1.

Theorem 2. *Given an L-bit hash value B, suppose there are ξ eigenpoints. By costing 2^ζ* qRAM*, the preimage A has a $1/2$ probability of being recovered with the attack complexity of $\frac{r-1}{r} \cdot \sqrt{\frac{2^\xi + 2^\varepsilon - 1}{2^\xi}} \cdot 2^{\frac{L-\varepsilon}{2}+2}$.*

In the overall framework about our second attack model is clear: all the guessed rotational counterparts $A^{\leftarrow}$ works as some certain nodes to form a hypercube. Nest the Grover's algorithm performed on the oracle operator U_Q into the design of oracle operator U_F. The key unitary oracle operator U_F assists the coin operator C to distinguish which nodes acted as the rotational counterparts $A^{\leftarrow}$ and mark them. Then, the unitary evolution operator Λ provides a perturbed coin quantum walk on the obtained hypercube to search the rotational counterpart nodes. It is noteworthy that the use of diffusion operator $U_{\psi^\perp}$ throughout whole algorithm is the diffusion coin operator $C_0 = U_{\psi^\perp}$ which is applied to the L - dimensional coin space for the unmarked nodes. The scale of diffusion coin operator C_0 is reduced exponentially compared with 2^L of the first attack model. Therefore, we believe that the quantum circuit of second model is easier to realize than the the previous one. However, the better practicability and the efficient attacks, the two cannot have both in the second attack model. We need to execute the second algorithm twice on average in order to expect a correct rotational counterpart $A^{\leftarrow}$. The detailed quantum circuit model is shown in the Figure 7.

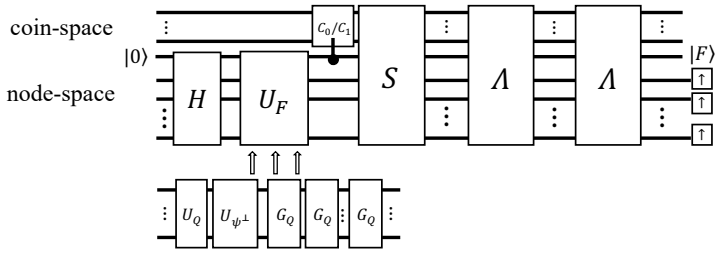

Fig. 7: The SKW-with-Grover model.

4 Preimage Attack on 4-Round Keccak

This section shows that our two quantum preimage attack models can be used to attack the 4-round Keccak-224 and 4-round Keccak-512. Owing to the least eigenpoints obtained in the case of 4-round Keccak-224, the Grover-with-Grover model is applied to this situation to show the performance of our first attack model. Then, we perform the SKW-with-Grover model on the 4-round Keccak-512 to show that our second attack model still achieves a good attack effect on the top of more practicality. The results of attacking 4-round Keccak-384 and 4-round Keccak-256 are also briefly given. Before the preimage attack, we need to calculate the TPP in advance. Note that the identical TPP for different

Keccak variants is shown in Appendix B. The only difference among these cases is that how many lanes are required to be checked.

4.1 Preimage Attack on Keccak-224 Under the First Attack Model

We first consider the padding function and the Keccak parameters of SHA-3 standard. For the Keccak-224, the bitrate $r = 1152$, capacity $c = 448$ and the hash length is set to 224 bits. In our preimage attack, the structure of message is restricted as follows: the message length is 1152 bits, of which only the first 3.5 lanes(224 bits) are unknown for us, and the remaining 14.5 lanes (928 bits) can be set to the known fixed bits value.

By accessing the TPP, we can find a total of 3 eigenpoints in the first 3.5 lanes. We suppose the claimed Algorithm 1 can help us store the information of the eigenpoints in the qRAM for the later superposition accessing. According to the Algorithm 2, we can expect to obtain the corresponding rotational number of a rotational counterparts with a probability of 8/71, which further means that, for the rotational counterpart, the oracle opertor U_F can mark it with the same probability. Recall that the certain preimage has 64 rotational counterparts. Therefore, each Grover iterate of searching the rotational counterparts has roughly $64 \times \frac{8}{71}$ items to be marked. With the help of Theorem 1, we know that the attack complexity is approximately equivalent to running of 4-round Keccak-224 $2^{112.57}$ times, and at most 2^6 qRAM are required. An obvious result is that the attack complexity of the generic qunautm preimage attack is approximately equivalent to running of 4-round Keccak-224 2^{113} times, which as a result of the uncompute trick on the corresponding oracle operator. That means our method reduces the attack complexity at least 25%.

The result in other variants. We also estimated the attack complexities against the 4-round Keccak-512/384/256 under the first attack model. Specifically, for the 4-round Keccak-512, the attack complexity is $2^{255.08}$, which reaches the best effect. For the 4-round Keccak-384, the attack complexity is $2^{191.49}$. The improvement of attack effect on the 4-round Keccak-256 over the generic quantum preimage attack is same to the case of 4-round Keccak-224. It is worth mentioning that the qRAM consumptions are negligible in all the cases.

4.2 Preimage Attack on Keccak-512 Under the Second Attack Model

For the Keccak-512, the bitrate $r = 576$, capacity $c = 1024$ and the hash length is set to 512 bits. Same as above-mentioned, we also restrict the structure of message as follows: the message length is 576 bits, of which only the first 8 lanes(512 bits) are unknown for us and the remaining 1 lane (64 bits) is set to the known bits value. By accessing the TPP, we can obtain 9 eigenpoints in the first 8 lanes, and store the eigenpoints with at most 2^7 consumption of qRAM.

According to the Algorithm 2 and Algorithm 3, we can use the unitary oracle operator U_F to mark a rotational counterpart successfully with a probability of 512/575, which allows us to distinguish the majority of rotational counterparts. Therefore, the coin operator C can efficiently perform an identity operator I on the nodes marked by the operator U_F in each iteration of the unitary evolution operator Λ. By using the SKW algorithm twice to search for the rotational counterparts, we can expect to obtain a correct one. Therefore, the attack complexity is approximately equivalent to running of 4-round Keccak-512 $2^{256.08}$ times, which is consistent with that described in the Theorem 2. Compared to the generic quantum preimage attack, this result reduces the attack complexity by about 2 times.

The result in other variants. We still estimated the attack complexities against the 4-round Keccak-384/256/224 under the second attack model. For the 4-round Keccak-384, with at most 2^7 qRAM costs, the attack complexity is $2^{192.49}$, which means our method

reduces the attack complexity roughly 29%. However, due to the SKW algorithm has success probability of only 1/2, the attack effects of the second attack model are not as good as those of the generic quantum preimage attack in the cases of 4-round Keccak-224/256.

5 Conclusion

In this work we have shown the quantum preimage attack based on the rotational cryptanalysis against the round-reduced Keccak. We have proposed a probabilistic algorithm to guess the rotational number of rotatoinal counterpart with a high probability such that the high-cost verifications have been reduced by 64 times in the quantum setting. With the aid of the probabilistic algorithm, we have obtained two effective models for the preimage attack on Keccak variants: The Grover-with-Grover model and the SKW-with-Grover model. The first attack model pays close attention to reach the lowest attack complexity, and achieves the advantage that the attack effect on 4-round Keccak-512 is 4 times better than the generic quantum preimage attack, and still works well in other variants. The second attack model is dedicated to reducing the implementation threshold of our algorithm. We have transformed the searching of the rotational counterpart into the finding of vertexes acted as rotational counterparts on a hypercube in order that the second model has better practicability. Fortunately, our second attack model still has pretty good attack effects against the 4-round keccak-512 and 4-round Keccak-384.

For the future, there is a lot of work for us to confront with. For the classical algorithms, there is a huge improvement with the application of classical Markov chains which providing new approximation and optimization algorithms. By analogy, we believe the quantum counterparts of classical Markov chains which called quantum random walk can acts as an indispensable role in the quantum cryptanalysis. The quantum walk based SKW algorithm has been shown that can be used to improve the implementation of our quantum preimage attack. We believe that any breakthrough about quantum random walk will greatly enhance the performance of our achievements.

Acknowledgement

This work is supported in part by the Key Research and Development Program of Shaanxi (No. 2021ZDLGY06-04), Guangxi Key Laboratory of Cryptography and Information Security (No. GCIS201802).

Appendix A

Proof of Lemma 1.

Recall that we hope to find the corresponding rotational number n of a rotational counterpart $A^{\leftarrow}$ by this quantum search. Given a rotational counterpart $A^{\leftarrow}$, suppose $n \in \{0, \cdots, \omega - 1\}$ is the corresponding rotational number and any other value $i \in \{0, \cdots, \omega - 1\}/n$ becomes a pseudo rotational number with a probability $1/2^{\xi}$. We can roughly obtain a total of $1 + (2^{\varepsilon} - 1) \times 1/2^{\xi}$ values, including the rotational number and the pseudo rotational numbers, which lead the rotational counterpart $A^{\leftarrow}$ to pass the checking of eigenpoints. It is notable that once the rotational counterpart $A^{\leftarrow}$ has the pseudo rotational numbers, the oracle operator U_Q can not distinguish the corresponding rotational number with the pseudo rotational numbers. Therefore, the running of the Grover's algorithm on the iteration operator G_Q will return one of the $1 + (2^{\varepsilon} - 1) \times 1/2^{\xi}$ values and hit the corresponding rotational number n with the probability of $\frac{2^{\xi}}{2^{\xi}+2^{\varepsilon}-1}$.

Proof of Lemma 2.

Recall that we have defined the cyclic pattern in Definition 2, and we need to find out the number of 2^n-bit cyclic patterns. Thanks to the special length of the bit string, all the possible 2^n-bit cyclic patterns can be divided into:

$$\mathcal{A}^i : \underbrace{\alpha_i \| \alpha_i \| \cdots \| \alpha_i}_{2^{n-i}}$$

where α_i is sub-string and $\alpha_i \in S_{\alpha_i} = \{0,1\}^{2^i}, i \in (0, 1, \cdots, n-1)$. For instance, if $i = 0$, there are 2^n sub-strings α_0 to form the 2^n-bit string. So, the number of $\alpha_0 : |S_{\alpha_0}|$ is equal to the number of cyclic patterns $\mathcal{A}^0 : |\mathcal{A}^0|$. For the case of i, there are 2^{n-i} sub-strings α_i to form the 2^n-bit string. The same as above, the number of $\alpha_i : |S_{\alpha_i}|$ is also equal to the number of cyclic patterns $\mathcal{A}^i : |\mathcal{A}^i|$. Namely, $|S_{\alpha_i}| = |\mathcal{A}^i| = 2^{2^i}$. However there are many overlapping values among them, we can not get the number of cyclic patterns by summing up all the $|\mathcal{A}^i|$ for i directly. For example, if $\alpha_1 = 01$, the $\alpha_2 = 0101 = 01 \| 01 = \alpha_1 \| \alpha_1; \alpha_2 \in S_{\alpha_2} = \{0,1\}^{2^2}$, and if $\alpha_2 = 0110$, the $\alpha_3 = 01100110 = 0110 \| 0110 = \alpha_2 \| \alpha_2; \alpha_3 \in S_{\alpha_3} = \{0,1\}^{2^3}$.

Therefore, we need to find the exclusive cyclic patterns for each $\mathcal{A}^i$ and denote the new set by $\mathcal{B}^i$. Fortunately, there is an interesting fact that the factor of 2^i can be listed as follows: $2^0, 2^1, \cdots, 2^{i-1}$. And the number of $\mathcal{B}^i : |\mathcal{B}^i|$ can be obtained easily as follows:

$$|\mathcal{B}^0| = |\mathcal{A}^0| = 2^{2^0};$$
$$|\mathcal{B}^i| = 2^{2^i} - |\mathcal{B}^{i-1}| - \cdots - |\mathcal{B}^1| - |\mathcal{B}^0|, \quad i = 1, \ldots, n-1.$$

Finally, we can get the number of cyclic patterns for S_α by: $\sum_{i=1}^{n-1} |\mathcal{B}^i| = 2^{2^{n-1}}$.

Proof of Theorem 1 and Theorem 2.

From the Lemma 2, we know that there are $2^{2^{\varepsilon-1}}$ 2^ε-bit cyclic patterns. So, the number of combinations of cyclic patterns in first L/ω lanes is:

$$\underbrace{2^{2^{\varepsilon-1}} \cdot 2^{2^{\varepsilon-1}} \cdot \cdots\cdot 2^{2^{\varepsilon-1}}}_{L/\omega \ times} = 2^{L/2}$$

Any of these cyclic patterns has only $\lambda \in \{1, 2, \cdots, \omega/2\}$ rotational counterparts, resulting in the lower efficiency about the using of preimage attack based on the rotational cryptanalysis. Therefore, we perform the quantum exhaustive search on these $2^{L/2}$ cyclic patterns in advance to check whether the desired preimage A is one of these cyclic patterns, and the computational complexity is $2 \cdot \sqrt{2^{L/2}} = 2^{L/4+1}$ times calculations of r round Keccak. If the preimage A is not obtained in the previous calculation, which means the corresponding value of A has all the ω rotational counterparts $A^{\leftarrow}$, our claimed attack models can be effective.

We first pay attention on the proof of Theorem 1. Recall that in each iteration of operator G_F, the oracle operator U_F can mark roughly $\frac{2^{\xi+\varepsilon}}{2^\xi+2^\varepsilon-1}$ of all the ω rotational counterparts $A^{\leftarrow}$ in the step 4 of Algorithm 3. Conservatively, we believe that there are roughly $\frac{2^{\xi+\varepsilon}}{2^\xi+2^\varepsilon-1}$ rotational counterparts in total. With the help of quadratic speed up, we can find one of the rotational counterparts $A^{\leftarrow}$ in roughly:

$$\sqrt{2^L/(\frac{2^{\xi+\varepsilon}}{2^\xi+2^\varepsilon-1})} = \sqrt{\frac{2^\xi+2^\varepsilon-1}{2^\xi}} \cdot 2^{\frac{L-\varepsilon}{2}}$$

times Grover iterations. Each iteration we do 4 times $r-1$ round Keccak and require $4 \cdot 5 \cdot \omega \cdot (r-1)$ times *One-Row χ Operations*. Therefore, the whole process of Grover

search needs around: $20\cdot\omega\cdot(r-1)\cdot\sqrt{\frac{2^{\xi}+2^{\varepsilon}-1}{2^{\xi}}}\cdot 2^{\frac{L-\varepsilon}{2}}$ times *One-Row* χ *Operations*. Owing to r round Keccak have $5\cdot\omega\cdot r$ *One-Row* χ *Operations*, the computational complexity of Grover search is equal to

$$\frac{r-1}{r}\cdot\sqrt{\frac{2^{\xi}+2^{\varepsilon}-1}{2^{\xi}}}\cdot 2^{\frac{L-\varepsilon}{2}+2}$$

times calculations of r round Keccak.

Note that we only obtain the rotational counterpart $A^{\leftarrow}$ of preimage A up to now. We still require nearly 2^{ε} times calculations of Keccak to determine the corresponding rotational number n, and further recover the preimage A for the given hash value B. Therefore, in the first attack model the upper bound of attack complexity is equal to

$$\frac{r-1}{r}\cdot\sqrt{\frac{2^{\xi}+2^{\varepsilon}-1}{2^{\xi}}}\cdot 2^{\frac{L-\varepsilon}{2}+2}+2^{L/4+1}+2^{\varepsilon}\approx\frac{r-1}{r}\cdot\sqrt{\frac{2^{\xi}+2^{\varepsilon}-1}{2^{\xi}}}\cdot 2^{\frac{L-\varepsilon}{2}+2}$$

times calculations of r round Keccak. It is notable that throughout the preimage attack we only use 2^{ζ} qRAM to store the information of ξ eigenpoints. The only difference between the second attack model and the first attack model is that the former has the lower success probability of $1/2$. The running of the second attack model twice ensures the recovery of desired preimage A, and the conclusion described in Theorem 2 is undoubtedly logical.

Now, we'd like to explain why the concrete complexity can be lower than the given upper bound. In the original Grover search, the diffusion operator $U_{\psi^{\perp}}$ involved by the Grover iteration operator: $G=U_{\psi^{\perp}}\cdot U_f$ plays a vital role in the whole algorithm. For any superposition state $|\phi\rangle$: $|\phi\rangle=\sum_{x=0}^{2^n-1}\alpha_x\cdot|x\rangle$, the average value of amplitudes is denoted by $\mu=\frac{1}{2^n}\sum_{x=0}^{2^n-1}\alpha_x$. After applying the diffusion operator $U_{\psi^{\perp}}$ on the superposition state $|\phi\rangle$, we have $U_{\psi^{\perp}}|\phi\rangle=\sum_{x=0}^{2^n-1}(2\mu-\alpha_x)\cdot|x\rangle$. If the oracle operator U_f has flipped the phases of several states $|x\rangle$ in the superposition state $|\phi\rangle$, the diffusion operator $U_{\psi^{\perp}}$ can increase the amplitudes of these target elements. Suppose the oracle operator U_f can flip the phases of all the target elements effectively, the average value of amplitudes μ will decrease as the accumulation of iterations, as a result of the amplitudes of target elements increase gradually and the amplitudes of the non-target elements decrease gradually. That's why the efficiency of amplitude amplification decays in the original Grover's algorithm.

In practice, the oracle operator U_F defined in Algorithm 3 flips roughly $\frac{2^{\xi+\varepsilon}}{2^{\xi}+2^{\varepsilon}-1}$ of all the 2^{ε} rotational counterparts $A^{\leftarrow}$, and the marked rotational counterparts $A^{\leftarrow}$ are different in each iteration with a certain probability. Let $\mu_{F,i}$ and $\mu_{f,i}$ denote the average values of amplitudes in the i^{th} iteration under the practice and the conservative case, respectively. After the first Grover iterate, the values of $\mu_{F,1}$ and $\mu_{f,1}$ are obviously equal. However, in the second Grover iterate, the oracle operator U_F doesn't flip the identical states marked in the last iteration with a probability of at least $1-(\frac{2^{\xi}}{2^{\xi}+2^{\varepsilon}-1})^{\lfloor\frac{2^{\xi+\varepsilon}}{2^{\xi}+2^{\varepsilon}-1}\rfloor}$. So, in the practice case, the rotational counterpart states that have not been marked by the U_F in the last iteration can be expected to be marked in this iteration, and resulting in the value of $\mu_{F,2}$ is higher than $\mu_{f,2}$. By analogy, as the increase of the iterations, the attenuation of value $\mu_{F,i}$ is slower than that of value $\mu_{f,i}$ with a certain probability related to the number of eigenpoints. Therefore, we believe the practice attack effect is better than the given upper bound.

Appendix B

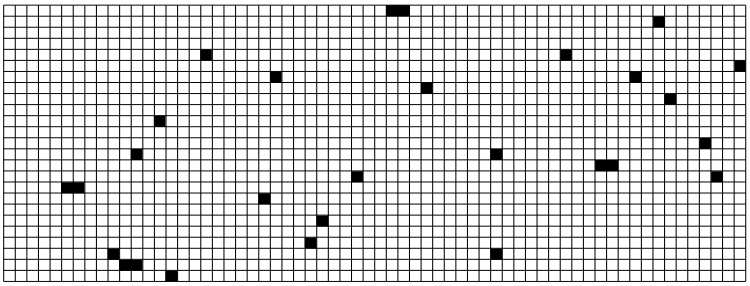

Fig. 8: The TPP of our preimage attacks.

It is notable that the 4-th round only applys the steps: θ, ρ and π. We use the black blocks to represent the positions of the eigenpoints.

References

[ADMG+16] Matthew Amy, Olivia Di Matteo, Vlad Gheorghiu, Michele Mosca, Alex Parent, and John Schanck. Estimating the cost of generic quantum pre-image attacks on sha-2 and sha-3. In *International Conference on Selected Areas in Cryptography*, pages 317–337. Springer, 2016.

[BDPVA11] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Cryptographic sponges. *online] http://sponge. noekeon. org*, 2011.

[Ber10] Daniel J Bernstein. Second preimages for 6 (7?(8??)) rounds of keccak. *NIST mailing list*, 2010.

[BHNP+19] Xavier Bonnetain, Akinori Hosoyamada, María Naya-Plasencia, Yu Sasaki, and André Schrottenloher. Quantum attacks without superposition queries: the offline simons algorithm. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 552–583. Springer, 2019.

[DHM+18] Danial Dervovic, Mark Herbster, Peter Mountney, Simone Severini, Naïri Usher, and Leonard Wossnig. Quantum linear systems algorithms: a primer. *arXiv preprint arXiv:1802.08227*, 2018.

[DSS+20] Xiaoyang Dong, Siwei Sun, Danping Shi, Fei Gao, Xiaoyun Wang, and Lei Hu. Quantum collision attacks on aes-like hashing with low quantum random access memories. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 727–757. Springer, 2020.

[Dwo15] Morris Dworkin. Sha-3 standard: Permutation-based hash and extendable-output functions, 2015-08-04 2015.

[Gag17] Tommaso Gagliardoni. Quantum security of cryptographic primitives. *arXiv preprint arXiv:1705.02417*, 2017.

[GLM08] Vittorio Giovannetti, Seth Lloyd, and Lorenzo Maccone. Quantum random access memory. *Physical review letters*, 100(16):160501, 2008.

[GLS16] Jian Guo, Meicheng Liu, and Ling Song. Linear structures: Applications to cryptanalysis of round-reduced keccak. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 249–274. Springer, 2016.

[Gro96] Lov K Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 212–219, 1996.

[HLY21] Le He, Xiaoen Lin, and Hongbo Yu. Improved preimage attacks on 4-round keccak-224/256. *IACR Transactions on Symmetric Cryptology*, pages 217–238, 2021.

[HS18] Akinori Hosoyamada and Yu Sasaki. Cryptanalysis against symmetric-key schemes with online classical queries and offline quantum computations. In *Cryptographers Track at the RSA Conference*, pages 198–218. Springer, 2018.

[HS20] Akinori Hosoyamada and Yu Sasaki. Finding hash collisions with quantum computers by using differential trails with smaller probability than birthday bound. *Advances in Cryptology–EUROCRYPT 2020*, 12106:249, 2020.

[HS21] Akinori Hosoyamada and Yu Sasaki. Quantum collision attacks on reduced sha-256 and sha-512. *IACR Cryptol. ePrint Arch.*, 2021:292, 2021.

[Kem02] Julia Kempe. Quantum random walks hit exponentially faster. *arXiv preprint quant-ph/0205083*, 2002.

[KLLNP15] Marc Kaplan, Gaëtan Leurent, Anthony Leverrier, and María Naya-Plasencia. Quantum differential and linear cryptanalysis. *arXiv preprint arXiv:1510.05836*, 2015.

[KLLNP16] Marc Kaplan, Gaëtan Leurent, Anthony Leverrier, and María Naya-Plasencia. Breaking symmetric cryptosystems using quantum period finding. In *Annual international cryptology conference*, pages 207–237. Springer, 2016.

[LM17] Gregor Leander and Alexander May. Grover meets simon–quantumly attacking the fx-construction. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 161–178. Springer, 2017.

[LS19] Ting Li and Yao Sun. Preimage attacks on round-reduced keccak-224/256 via an allocating approach. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 556–584. Springer, 2019.

[LSLW17] Ting Li, Yao Sun, Maodong Liao, and Dingkang Wang. Preimage attacks on the round-reduced keccak with cross-linear structures. *IACR Transactions on Symmetric Cryptology*, pages 39–57, 2017.

[MPS13] Paweł Morawiecki, Josef Pieprzyk, and Marian Srebrny. Rotational cryptanalysis of round-reduced keccak. In *International Workshop on Fast Software Encryption*, pages 241–262. Springer, 2013.

[MR02] Cristopher Moore and Alexander Russell. Quantum walks on the hypercube. In *International Workshop on Randomization and Approximation Techniques in Computer Science*, pages 164–178. Springer, 2002.

[NC00] Michael A Nielsen and Isaac L Chuang. Quantum computing and quantum information, 2000.

[PB99] Arun Kumar Pati and Samuel L Braunstein. Impossibility of deleting an unknown quantum state. *arXiv preprint quant-ph/9911090*, 1999.

[Sha] NIST Sha. competition, 2007-2012. http://csrc.nist.gov/groups/ST/hash/sha-3/index.html. November 6, 2021.

[SKW03] Neil Shenvi, Julia Kempe, and K Birgitta Whaley. Quantum random-walk search algorithm. *Physical Review A*, 67(5):052307, 2003.

[WWF+21] Congming Wei, Chenhao Wu, Ximing Fu, Xiaoyang Dong, Kai He, Jue Hong, and Xiaoyun Wang. Preimage attacks on 4-round keccak by solving multivariate quadratic systems. Cryptology ePrint Archive, Report 2021/732, 2021. https://ia.cr/2021/732.