The Best Biclique Cryptanalysis of the Lightweight Cipher FUTURE

G. C. de Carvalho[1], **L. A. B. Kowada**[1]

[1]Instituto de Computação - Universidade Federal Fluminense (UFF) - Niterói - RJ

Abstract. *In the past decade, lightweight cryptography has been of much interest in the academy, especially in regards to the cryptanalysis of such ciphers. The National Institute of Standards and Technology (NIST) is one of the responsible for this interest, given that they promoted in 2019 a public process to choose the American standard for lightweight cryptography. In 2022, the FUTURE cipher was published and has since been the target of much cryptanalysis, including integral, meet-in-the-middle and differential cryptanalysis in a very short period of time. Earlier this year, a biclique attack for FUTURE was published. We show in this paper, a biclique attack that is better than the one previously published, both in time, memory and data complexities, obtained through semi-automatic search and bicliques based on distinct generator sets. It requires* $2^{125.18}$ *full computations of the cipher to run, while requiring only* 2^{20} *data pairs and negligible memory. Prior to June of this year when an integral attack on FUTURE was published, it was also the fastest attack without using the full code-book of data. Still, when compared to it, our attack uses much less data while being only slightly slower, which presents a good trade-off.*

1. Introduction

A Lightweight cipher is a cipher designed to be used in very constrained environments, such as embedded systems, radio devices and sensor networks [McKay et al. 2016]. FUTURE [Gupta et al. 2022] is one of the most recent developments in this field and one of the most academically relevant to the field of block cipher cryptanalysis. It is a Substitution Permutation Network (SPN), using very similar structures to the AES [Daemen and Rijmen 2013] with 10 rounds, 64 bit block size and secret key of size 128 bits.

FUTURE has been the target of several cryptanalysis in a very short period of time [Xu et al. 2024, Schrottenloher and Stevens 2023a, Roy et al. 2024, Mondal et al. 2024, İlter and Selçuk 2022] since it's publishing in 2022. Some of them are applied to round reduced versions of FUTURE, while others are applied to the full round cipher. Since our attack is on the full version, the ones that also attack this version are more relevant to this work.

Schrottenloher and Steve published a simple meet-in-the-middle attack on many ciphers, FUTURE being one of them. Their attack uses the full code-book and recovers the secret key in 2^{126} full computations of the FUTURE cipher. The code-book is the set of all plaintext-ciphertext pairs for the cipher with the secret key [Schrottenloher and Stevens 2023a].

Roy, Dey, Mondal and Adhikari published in the end of 2023 the first biclique attacks to FUTURE. They presented two of them: one using a simple balanced biclique

and the other using a new variation proposed by them, which uses two bicliques simultaneously in the attack. The simplest one uses a $32-$dimensional balanced biclique to create an attack with time complexity of $2^{125.8875}$, while the second one uses $2^{125.5365}$ full computations of the cipher to perform the attack. Both use 2^{48} for data and close to 4GB of memory [Roy et al. 2024].

Xu, Cui, Hu and Wang published in June of this year the fastest attacks to FUTURE currently available, both using the full code-book and less than it. One of the attacks requires $2^{123.7}$ full computations of the cipher using 2^{63} data and another, even faster, requires 2^{112}, but also requires the full code-book (2^{64}) [Xu et al. 2024]. Throughout the paper, the term "faster" is used to compare the time complexity between attacks.

Biclique cryptanalysis is most famous for its application to the AES cipher [Bogdanov et al. 2011], being the first attack to its full round version. The following years, many others improved on their work, such as [Tao and Wu 2015, de Carvalho et al. 2023a, Bogdanov et al. 2015]. It was also applied to many others in the past decade, such as ARIA [Chen and Xu 2014], Serpent [de Carvalho and Kowada 2020], HIGHT [Hong et al. 2011] and, most recently, FUTURE [Roy et al. 2024].

The use of software tools for automatic or semi-automatic searches to assist the cryptanalyst has become common over time. Most cryptanalysis nowadays uses tools in some capacity. Modeling attacks as a MILP or SAT problem is one of most used ways of automating part of the work. The works of [Schrottenloher and Stevens 2023b, Shi et al. 2023, de Carvalho et al. 2023b] are recent examples of using MILP modeling for the problem of searching for MITM attacks on many ciphers, while [Bellini et al. 2024] published earlier this year a Python library called **CLAASP** that enables the automation of various forms of cryptanalysis for different types of block ciphers.

Our Contributions. All of the current attacks on FUTURE have the same issue: the amount of data is too big for it to be relevant. This paper presents an attack that is fast, when compared to others, and requires as little data as possible. Therefore, we describe a biclique attack based on distinct generator sets (GS-Biclique) on full round FUTURE, using a balanced biclique obtained from the automatic search for related-key differentials, a variation introduced by [de Carvalho et al. 2022]. It results in the fastest biclique attack against full round FUTURE and the one that requires by far the least amount of data to be carried. This is arguably the attack with the best time-data trade-off using less than the full code-book. This attack was carried out through a tool that does an automatic search for bicliques. It can be found at `https://github.com/Clique33/BicliqueFinder`. Table 1 shows the comparison between our attack and the other in literature on the full round FUTURE.

Paper structure. Section 2 describes the biclique cryptanalysis the concept of generator sets for key bits. The FUTURE cipher and its notation are described in Section 3. Both the attack description and its analysis are discussed in Section 4. Finally, Section 6 concludes the paper.

Table 1. Time, data and memory complexity the attacks on full round FUTURE. The ≈ 0 sign symbolizes a negligible amount.

Attack	Time	Data	Memory	Reference
Meet-in-the-Middle	2^{126}	2^{64}	2^{36}	[Schrottenloher and Stevens 2023a]
Balanced Biclique	$2^{125.8875}$	2^{48}	2^{32}	[Roy et al. 2024]
Multiple Biclique	$2^{125.5365}$	2^{48}	2^{32}	[Roy et al. 2024]
Integral	$2^{123.7}$	2^{63}	$\approx \mathbf{0}$	[Xu et al. 2024]
Integral	$\mathbf{2^{112}}$	2^{64}	$\approx \mathbf{0}$	[Xu et al. 2024]
GS-Biclique	$2^{125.18}$	$\mathbf{2^{20}}$	$\approx \mathbf{0}$	Section 4

2. Biclique Cryptanalysis

In this section we present the basic steps associated with the balanced biclique attack with the biclique being built in the last rounds of the cipher, as used in this paper to attack the FUTURE cipher.

- **Preparation phase.** An adversary partitions the key space into groups with 2^{2d} keys for some d. Each key group is associated with a $2^d \times 2^d$ matrix K, where each element $K[i, j]$ represents a key in the group. Let k be the number of bits of the secret key. In this case we have 2^{k-2d} groups. Also, the cipher $\Gamma = f \circ g \circ h$ being attacked is a composition of three subciphers f, g and h. This phase is the most important of the attack, since most of the complexities derive from the choices made here. For this reason, Section 2.1 is dedicated to this phase. The three steps below are then done for each key group.

1. **Building the biclique.** A biclique structure is built over the subcipher f, resulting in a structure that satisfies the condition

$$\forall i, j : S_j \xrightarrow[f]{K[i,j]} C_i,$$

   where $0 \leq i, j < 2^d$, S_j are internal states of the cipher and C_i are ciphertexts. Section 2.2 presents the fundamentals for building such structures.
2. **Obtain plaintexts.** This is a chosen ciphertext attack. Consequently, we have at our disposal a decryption oracle, which is used to obtain the plaintext P_i for each ciphertext C_i.

$$\forall i : C_i \xrightarrow[E^{-1}]{decryption\ oracle} P_i.$$

3. **Meet-in-the-Middle.** For each key $K[i, j]$ in the group it is tested if

$$\exists i, j : P_i \xrightarrow[g]{K[i,j]} S_j.$$

   If one of the $K[i, j]$ is the secret key, then the above condition is satisfied. Therefore, every key that satisfies it is a candidate to the secret key. This can be done through any meet-in-the-middle method. Here, we use the Matching with precomputations technique, explained in Section 2.3.

2.1. Preparation Phase

Here we informally define the core concepts of the preparation phase from the biclique cryptanalysis.

Let $E = f \circ g \circ h$ be a cipher, with s subkeys of k' bits each. The total key bits is then $m = s \cdot k'$. A *generator set of key bits* is any set of key bits that is enough to generate all m key bits of cipher E through some algorithm. Each family of differentials in the attack can be defined over a different generator set, if it is necessary. This concept was created by [de Carvalho et al. 2022].

The biclique attack can be seen as an optimization of an exhaustive search. This is due to the fact that every single key is tested, except that it is done in a way that is faster than just simply testing each possible key through the cipher. For that to happen, it is necessary to partition the whole key space into disjoint sets in a way that each set is tested separately through the biclique. The improvement in speed comes from this choice.

Each group has a representative, called a *base key*. The base key of the group has some bits fixed to 0 while all other vary from group to group. The bits that are fixed to 0 are the ones that go through every possible value when the key difference of each related-key differential in the biclique is applied to them.

For example, suppose a base key in which bytes 0 and 1 are fixed to 0. This means that there has to be a total of 2^{16} related-key differentials in the cipher, and every one of them influence one or both of those bytes in a way that no key is repeated nor is not tested. The base key must also be a generator set (usually a subkey or consecutive subkeys), so that all the key bits can be generated to carry out the attack.

2.2. Building the Biclique

Here, we take a look at the biclique structure built on the subcipher f of the target cipher $E = f \circ g \circ h$. Let f be the subcipher that maps an state S to the ciphertext C using the key K thorugh the subcipher f (*i.e.* $f_K(S) = C$). A *dimension d biclique over f* is the 3-tuple $(\{S_j\}, \{C_i\}, \{K[i,j]\})$, where $0 \leq i, j < 2^d$ such that

$$\forall i, j : f_{K[i,j]}(S_j) = C_i.$$

One way to achieve this condition is using related-key differentials. It is important to highlight the fact that this is a single key attack. The related-key model is used only within the key groups. Let $K[0,0]$ be the *base key*, *i.e.* the key that maps the internal state S_0 to the ciphertext C_0. This is called the *base computation*

$$S_0 \xrightarrow[f]{K[0,0]} C_0.$$

The next step is defining the *Δ_i-differentials* and *∇_j-differentials* using related-key differentials. *Δ_i-differentials* map the input difference 0 to the output difference Δ_i, using the key difference Δ_i^K, where $\Delta_0 = \Delta_0^K = 0$ and $0 \leq i < 2^d$

$$0 \xrightarrow[f]{\Delta_i^K} \Delta_i,\ \Delta_0 = \Delta_0^K = 0.$$

In contrast, ∇_j*-differentials* map the input difference 0 to the output difference ∇_j, using the key difference ∇_j^K on the opposite direction, from f^{-1}, where $\nabla_0 = \nabla_0^K = 0$ and $0 \leq j < 2^d$

$$\nabla_j \xleftarrow[f^{-1}]{\nabla_j^K} 0,\ \nabla_0 = \nabla_0^K = 0.$$

Two families of differentials are independent if there are no bits of states or subkeys of f in which ∇_j*-differentials* and Δ_i*-differentials* affect simultaneously. If both sets of differentials are independent, then it is possible to combine them into (Δ_i, ∇_j)*-differentials*

$$\nabla_j \xrightarrow[f]{\nabla_j^K \oplus \Delta_i^K} \Delta_i.$$

By definition, the base computation conforms to both sets of differentials and hence, it is possible to substitute it to the combined differentials

$$S_0 \oplus \nabla_j \xrightarrow[f]{K[0,0] \oplus \nabla_j^K \oplus \Delta_i^K} \Delta_i \oplus C_0.$$

By letting

$$S_j = S_0 \oplus \nabla_j,$$
$$C_i = \Delta_i \oplus C_0 \text{ and}$$
$$K[i,j] = K[0,0] \oplus \nabla_j^K \oplus \Delta_i^K$$

we have the definition of a dimension d biclique over f.

Building a biclique this way costs only 2^{d+1} computations of f, since it is possible to choose the key differences and base computation, and then, independently, compute the Δ_i*-differentials* and ∇_j*-differentials*.

2.3. Matching with Precomputations

This technique uses the knowledge that only parts of the cipher are affected by the differentials of the biclique to do the meet-in-the-middle step faster. It can be further exploited if instead of meeting an entire internal state, we meet in only a part of the state, namely v. This way we only look at the parts *affected by* the differentials *and* that *affect* v.

Let $E = f \circ g \circ h$, where the biclique was built over f. An adversary then computes and stores 2.2^d *full computations* of the cipher up to the variable v: 2^d computations of h and 2^d computations of g^{-1}

$$\forall i : P_i \xrightarrow[h]{K[i,0]} v_i^1 \text{ and } \forall j : v_j^2 \xleftarrow[g^{-1}]{K[0,j]} S_j.$$

This means that every internal state and subkeys of s and t up to v have to be stored. This is the *precomputation phase*.

Then comes the *recomputation phase*, where the parts that differ from the stored values must be recomputed. The cost of this method is rather variable depending on the diffusion properties of the cipher of interest, both the diffusion related to the key schedule and the states. The number of rounds also influences the cost.

2.4. Complexities

This attack can be seen as an improved exhaustive search, since every key will be tested, but not the whole cipher will be computed in each step. Three types of complexities are of interest: memory, data and time.

The memory complexity is dominated by the Precomputation Phase due to requiring the storage of whole states and keys of subciphers g and h. So if the biclique has dimension d, the memory complexity will be 2^{d+1} computations of $g \circ h$.

The data complexity depends only on how many bits of C_i are affected by the Δ_i-differentials, since all possible relevant C_i must be turned into P_i for the meet-in-the-middle step, which in turn depends essentially on the amount of rounds covered by the biclique, as well as on its dimension, and on the diffusion properties of the cipher.

Finally, the time complexity is where most of the analysis is necessary. It is basically the number of key groups times the time complexity of each iteration. each iteration builds the biclique and then does the matching with precomputations, which is divided into precomputation phase and recomputation phase. In the end, we have

$$C_{time} = 2^{k-2d}(C_{biclique} + C_{precomp} + C_{recomp} + C_{falspos}).$$

The false positives are the keys that pass on the test in the recomputation phase, meaning that they are secret key candidates. Thus it is necessary to check if they are the secret key.

3. The FUTURE Cipher

FUTURE [Gupta et al. 2022] is a 10 round Substitution-Permutation Network cipher (SPN). The key size is 128 bits while its block size is 64 bits. Each subkey has 64 bits, seen as a bitstring of 16 nibbles (chunks of 4 bits each).

$$FUTURE = AK_{10} \circ SR \circ SC \circ AK_9 \circ R_8 \circ R_7 \circ ... \circ R_1 \circ R_0$$

Each round $R_i = SR \circ MC \circ SC \circ AK_i$, for $0 \leq i < 9$. Rounds vary from 0 to 9 and there are 11 subkeys, derived from the secret key, indexed from K^0 (or \$0) to K^{10} (or \$10), and 41 states, where state jth is denoted as denoted as $\#j$, where $P = \#0$ and $C = \#40$. Each state $\#j$ can be graphically represented by a 4×4 matrix of nibbles (chunks of 4 consecutive bits). The nibbles are enumerated from the leftmost to the rightmost and then down to the next word. The words are enumerated from top to bottom. The h-th nibble of the state S é denoted as s_h. Next we have the graphic representation of an internal state of the cipher.

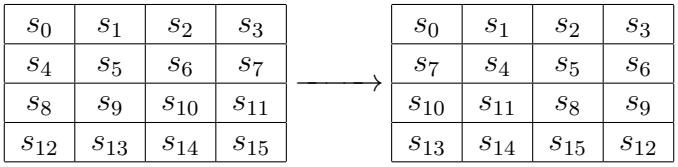
s_0	s_1	s_2	s_3
s_4	s_5	s_6	s_7
s_8	s_9	s_{10}	s_{11}
s_{12}	s_{13}	s_{14}	s_{15}

The *SubCell* operation SC looks up the S-box shown in Table 2 and substitutes each nibble of the state according to the it.

The *ShiftRows* SR is, similar to the AES, the rotation of nibbles of each row from the state. There are 4 rows, from 0 to 3. Row i is rotated i times to the right.

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
1	3	0	2	7	E	4	D	9	A	C	6	F	5	8	B

Table 2. S-box of the FUTURE cipher.



The *MixColumns* MC is, again very similar to the AES. Each of the four columns of the state are multiplied by a Maximum Distance Separable (MDS) matrix. This multiplication is done in $GF(2^4)$, with the primitive polynomial $x^4 + x + 1$. The matrix follows.

$$\begin{pmatrix} 8 & 9 & 1 & 8 \\ 2 & 2 & 9 & 9 \\ 2 & 3 & 8 & 9 \\ 9 & 9 & 8 & 1 \end{pmatrix}$$

The key scheduling for this cipher is remarkably simple: The 128 bit key is partitioned into two, the 64 leftmost bits become X and the other 64 become Y. Then, each K^i is equal to $X <<< (5 \cdot (\frac{i}{2}))$, if i is even and $Y <<< (5 \cdot (\frac{i}{2}))$ if i is odd. It does not use any S-boxes in its scheduling.

4. The Fastest Biclique Attack on FUTURE

In this section we present the fastest biclique attack on the FUTURE cipher to our knowledge. Due to space constraints,this section details the attack, while Section 5 only highlights the differences between that approach and this one. The attack also requires much less data than any other attack on the full round FUTURE. This biclique was found through the use of a tool created by us. It can be found at `https://github.com/Clique33/BicliqueFinder`.

4.1. Preparation Phase

We partition the key into $2^{128-2\cdot4} = 2^{120}$ groups, since FUTURE has a 128 bit secret key and our biclique is $4-$dimensional. We define FUTURE as the composition $FUTURE = f \circ g \circ h$, where h enciphers the plaintext into state $\#17$, g enciphers state $\#17$ into state $\#25$ and f enciphers state $\#25$ into the ciphertext.

Due to the key scheduling of the cipher, any two subkeys are a generator set for the key if the index of one is even and the index of the other is odd. Hence, we define both related-key differentials through subkeys $\$0$ and $\$1$. The nibble 3 of $\$0$ is the only active nibble of Δ^K and nibble 8 of $\$1$ is the only active nibble for ∇^K. These key differences create Δ_i-differentials and ∇_j-differentials that are independent from each other, as shown in Figure 1. It is a $4-$dimensional biclique because the differentials share no S-boxes and there are 2^4 possible C_i and S_j. This biclique covers the last 4 rounds of the cipher, which goes from state $\#25$ through to state $\#40$.

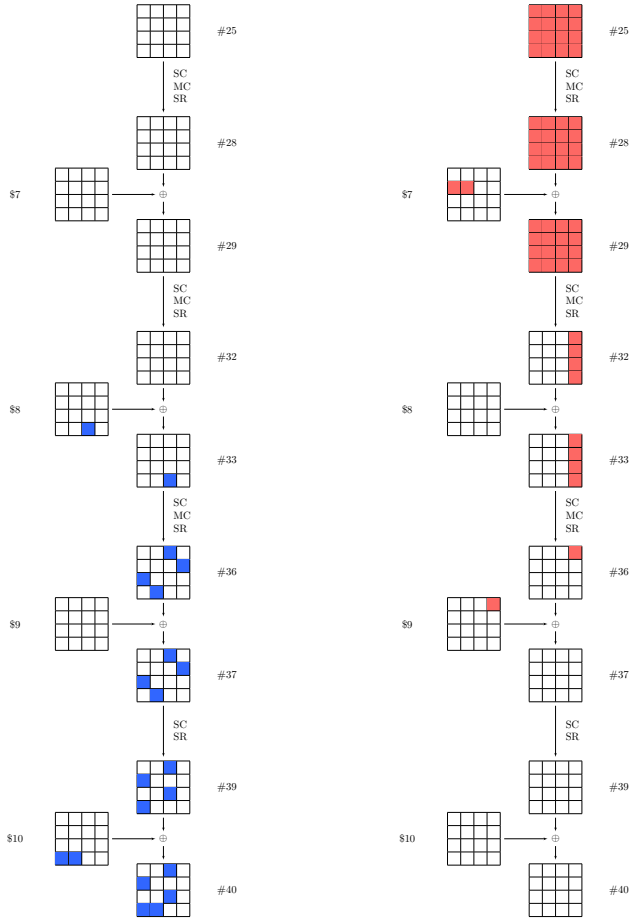


Figure 1. Δ_i-differentials and ∇_j-differentials for the balanced biclique.

4.2. Matching with Precomputations over 6 rounds

In this part of the attack, we check if the secret key belongs to the group $\{K[i,j]\}$, *i.e.* if $K_{secret} \in \{K[i,j]\}$. First we precompute 2^5 values of v, which we define as being the nibble 3 of state $\#17$, and save them, together with all internal states and subkeys involved in these precomputations. Then, we have

$$P_i \xrightarrow[h]{K[i,j]} v^1_{i,j} \text{ and } v^2_{i,j} \xleftarrow[g]{K[0,j]} S_j$$

for each i and j, recomputing only those parts that differ from the ones saved in memory. If $v^1_{i,j} = v^2_{i,j}$, then $K[i,j]$ is a key candidate.

Next, we observe the recomputation that has to be done in the matching with precomputations step. Figure 2 shows graphically the nibbles that need to be recomputed.

In the forward direction we are interested in the difference between the computation of $P_i \xrightarrow[h]{K[i,j]} v$ and the precomputed values of $P_i \xrightarrow[h]{K[i,0]} v^1_i$, given by the influence of ∇^K_j on the subkeys from $\$0$ to $\$4$. Since FUTURE does not use S-boxes in the key schedule, they are irrelevant for the recomputation step. Only 1 nibble of $\#5$ is influenced by ∇^K_j. Next, states $\#9$ and $\#13$ require the recomputation of 4 nibbles each. Therefore, we only have to recompute 9 S-boxes for the states in the forward direction.

Similarly to the forward recomputation, we look at the difference between $v_j^2 \xleftarrow[g^{-1}]{K[i,j]} S_j$ and the precomputed $v_j^2 \xleftarrow[g^{-1}]{K[0,j]} S_j$, given by the influence of Δ_i^K in the subkeys $\$5$ to $\$6$. It is possible to see that only one nibble of both states $\#21$ and $\#17$, since these are the only ones that affect variable v, and thus, only 2 S-boxes must be recomputed.

Therefore, only $9 + 2 = 11$ S-boxes out of 160 total in the cipher must be recomputed.

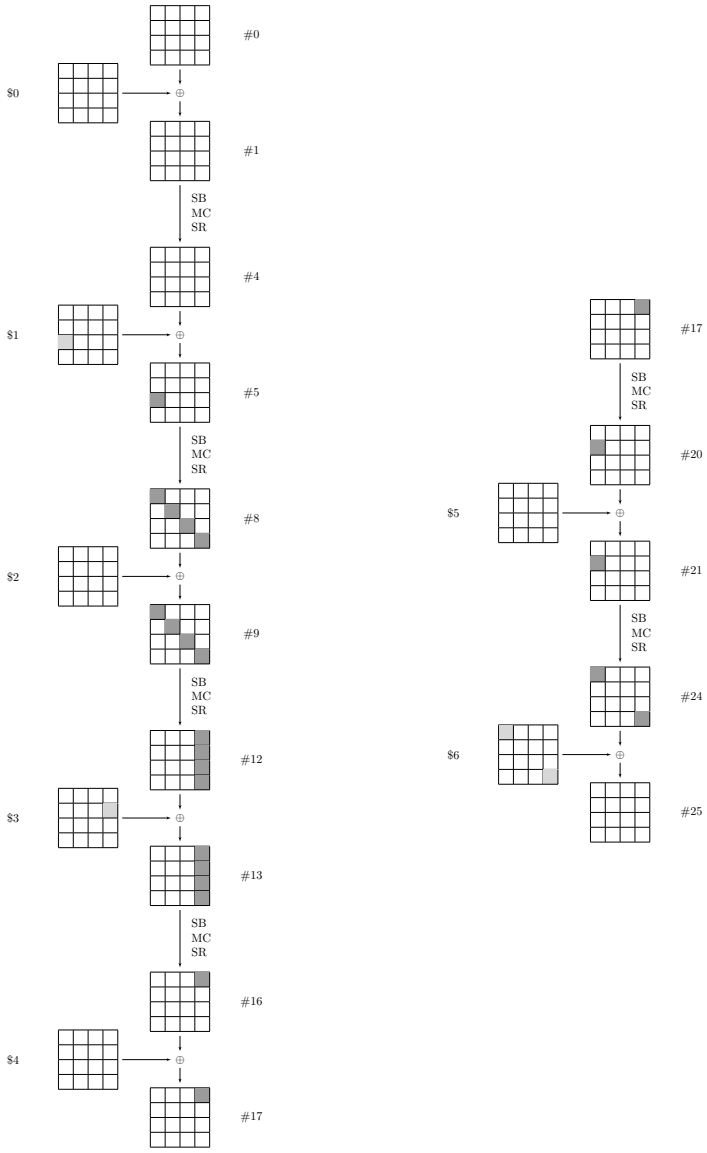


Figure 2. Forward and backward recomputations of the biclique cryptanalysis.

4.3. Complexities

Firstly we have

$$C_{total} = 2^{k-2d}(C_{biclique} + C_{precomp} + C_{recomp} + C_{falpos}).$$

The default approach to time complexity in biclique cryptanalysis is the percentage of S-boxes required to perform the attack, compared to the total number of S-boxes in the cipher. This is due to it usually being the bottleneck for time complexity in S-box based cipher. For the FUTURE cipher the total number of S-boxes is 160.

The complexity of building the biclique is given by the computation of the base, then 2^4 computations of Δ as well as 2^4 computations of ∇, which means counting the amount of S-boxes needed in each. For Δ it is only 5 S-boxes, while ∇ requires 36 S-boxes. For the base, it is enough to calculate the percentage of the cipher needed $C_{biclique} = (16/41) + (2^4 - 1) \cdot (5/160) + (2^4 - 1) \cdot (36/160) = 2^{2.0820}$.

For the precomputation it suffices to calculate how many times the subcipher $g \circ h$ is calculated, $C_{precomp} = 2^4 \cdot (25/41) = 2^{3.2863}$. On average, there will be 2^4 false positives per iteration, while the cost to test it is given by the total recomputation from P_i to $\#17$ and from S_j to $\#17$, instead of only meeting in v. This is about 37 S-boxes in the forward direction (1 S-box in $\#5$, 4 S-boxes in $\#9$, 16 S-boxes in $\#13$ and 16 S-boxes in $\#17$) and 20 S-boxes in the backward direction (4 S-boxes in $\#21$ and 16 S-boxes in $\#17$), Totalling 57 S-boxes. Hence, $C_{falpos} = 2^{2d}/2^{|v|} \cdot \text{cost} = 2^8/2^4 \cdot (57/160) = 2^{2.5110}$).

It remains to find C_{recomp}. As discussed in the last section, the total number of S-boxes to be recomputed is only 11. Then $C_{recomp} = (2^8 - 2^4) \cdot (11/160) = 2^{4.0444}$.

In the end, we obtain approximately

$$C_{total} = 2^{120}(2^{2.0820} + 2^{3.2863} + 2^{4.0444} + 2^{2.5110}) = 2^{125.18}$$

FUTURE full computations.

The data complexity is given by how many active nibbles are in the ciphertext state. It is possible to notice in Figure 1 that only 5 nibbles of C_i are affected and thus, only 2^{20} pairs of plaintexts/ciphertexts are necessary for the attack, *i.e.* there are only 2^{20} possibilities for the ciphertexts.

In terms of memory, the attack is upper limited by 2^4 computations of $g \circ h$ The full computation of $g \circ h$ consists of 25 states and 6 subkeys, with 16 nibbles each. Therefore the memory complexity is $2^4 \cdot (25 + 16) \cdot 16 = 10496$ nibbles, which is 5248 bytes. That is a negligible amount.

5. Fastest Attack with Minimum Data

Due to space constraints, this section only highlights the differences between that approach and this one. For more details, see Section 4. This attack uses a variation of the biclique attack called *star attack*. This is a different kind of biclique, in which there are two or more related-key differentials of the same kind (either both delta or both nabla). They are usually constructed in the beginning of the cipher, and so there are only $\Delta-$differentials. The most important characteristic of the star attack is that it requires only one pair of texts to be carried out. This is the case due to the fact that, when constructed this way, no words are active in the plaintext nor the ciphertext, making it so that only one pair of data is necessary to carry the attack.

Similarly to the attack presented in Section 4, we use a $4-$dimensional balanced biclique with two families of differentials created from $\Delta_{i_0}^K$ and $\Delta_{i_1}^K$. The generator sets

for both key differences are the subkey $\$0$ and $\$1$, where nibble 0 of $\$1$ is active in $\Delta^K_{i_0}$, and nibble 2 of $\$1$ is active in $\Delta^K_{i_1}$.

Our star is built in the beginning of the cipher, from the plaintext up to $\#10$. All other steps are executed in a similar way, where one of the deltas is treated as nabla for the precomputation step, and only the nibbles affected by both need to be recomputed. The variable v is chosen as nibble 3 from state $\#21$. The biclique is independent as show in Figure 3.

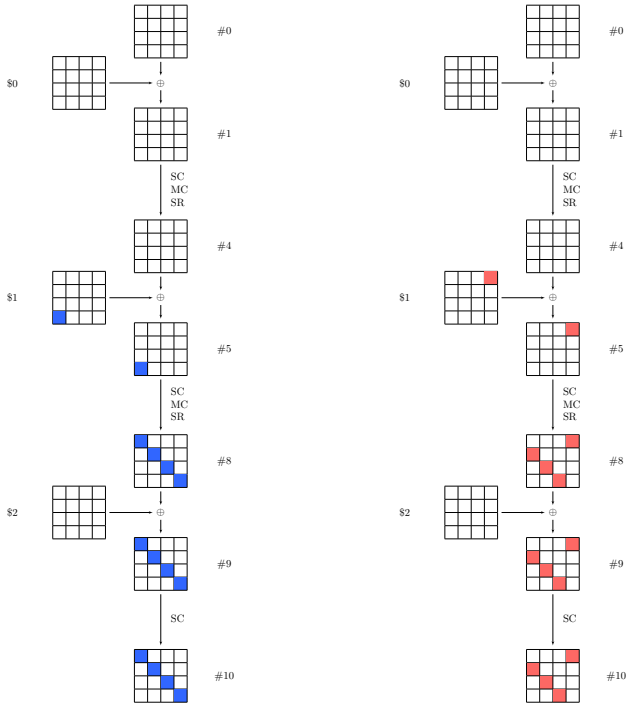


Figure 3. Δ_{i_0}-differentials and Δ_{i_1}-differentials in the star.

This attack is slower when compared to the previous one to compensate its lack of data requirements. The forward recomputation requires the recomputation of all 16 S-boxes in state $\#13$, 4 S-boxes in state $\#17$ and only one in state $\#21$. In the Backward direction, all 16 S-boxes in state $\#29$ and 4 S-boxes in state $\#25$ have to be recomputed. This results in 41 S-boxes.

The time complexity for the recomputation is $C_{recomp} = (2^8 - 2^4) \cdot (41/160) = 2^{5.9425}$. The precomputation complexity is $C_{precomp} = 2^4 \cdot (32/41) = 2^{3.6424}$, since the matching step goes from state $\#10$ up to the ciphertext. The number of false positives does not change, but the cost of each test is higher since the star covers less states than our other attack. Hence, $C_{falpos} = 2^{2d}/2^{|v|} \cdot \text{cost} = 2^8/2^4 \cdot (80/160) = 2^3$). Lastly, $C_{biclique} = (11/41) + (2^4 - 1) \cdot (5/160) + (2^4 - 1) \cdot (5/160) = 2^{0.26998}$, since the biclique covers 10 states and each Δ involves only 5 S-boxes.

In the end, we obtain approximately

$$C_{total} = 2^{120}(2^{0.26998} + 2^{3.6424} + 2^{5.9425} + 2^3) = 2^{126.38}$$

FUTURE full computations.

6. Concluding Remarks

We presented here the fastest biclique attack and the one with by far the lowest data complexity on the full round FUTURE cipher. It is a $4-$dimensional balanced GS-biclique, requiring only 2^{20} pairs of plaintext-ciphertext to be carried out and has $2^{125.18}$ of time complexity and uses negligible memory. It was made possible thanks to our tool that automates the search for GS-bicliques. All of the complete diagrams can be found at `https://github.com/Clique33/BicliqueFinder`.

Future work on this cipher involves the search for star bicliques and the implementation of multiple bicliques to the tool we used. An unbalanced biclique approach can also produce results. The same process can also be applied to other lightweight ciphers, such as the GIFT family.

References

Bellini, E., Gerault, D., Grados, J., Huang, Y. J., Makarim, R., Rachidi, M., and Tiwari, S. (2024). Claasp: A cryptographic library for the automated analysis of symmetric primitives. In Carlet, C., Mandal, K., and Rijmen, V., editors, *Selected Areas in Cryptography – SAC 2023*, pages 387–408, Cham. Springer Nature Switzerland.

Bogdanov, A., Chang, D., Ghosh, M., and Sanadhya, S. K. (2015). Bicliques with minimal data and time complexity for aes. In *Information Security and Cryptology-ICISC 2014: 17th International Conference, Seoul, South Korea, December 3-5, 2014, Revised Selected Papers 17*, pages 160–174. Springer.

Bogdanov, A., Khovratovich, D., and Rechberger, C. (2011). Biclique cryptanalysis of the full aes. In *Advances in Cryptology–ASIACRYPT 2011: 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings 17*, pages 344–371. Springer.

Chen, S.-z. and Xu, T.-m. (2014). Biclique key recovery for ARIA-256. *IET Information Security*, 8(5):259–264.

Daemen, J. and Rijmen, V. (2013). *The design of Rijndael: AES-the advanced encryption standard*. Springer Science & Business Media.

de Carvalho, G. et al. (2022). Generator sets for the selection of key differences in the biclique attack. In *Anais do XXII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais*, pages 1–14. SBC.

de Carvalho, G. et al. (2023a). Revisiting the biclique attack on the aes. In *Anais do XXIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais*, pages 153–166. SBC.

de Carvalho, G. C. and Kowada, L. A. (2020). The first biclique cryptanalysis of serpent-256. In *Anais do XX Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais*, pages 29–42. SBC.

de Carvalho, G. C., Neto, T. S., and do Rêgo Sousa, T. (2023b). Automated security proof of square, led and clefia using the milp technique. In *Anais do XXIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais*, pages 445–455. SBC.

Gupta, K. C., Pandey, S. K., and Samanta, S. (2022). Future: a lightweight block cipher using an optimal diffusion matrix. In *International Conference on Cryptology in Africa*, pages 28–52. Springer.

Hong, D., Koo, B., and Kwon, D. (2011). Biclique attack on the full HIGHT. In *International Conference on Information Security and Cryptology*, pages 365–374. Springer.

İlter, M. B. and Selçuk, A. A. (2022). Milp-aided cryptanalysis of the future block cipher. In *International Conference on Information Technology and Communications Security*, pages 153–167. Springer.

McKay, K., Bassham, L., Sönmez Turan, M., and Mouha, N. (2016). Report on lightweight cryptography. Technical report, National Institute of Standards and Technology.

Mondal, S. K., Rahman, M., Sarkar, S., and Adhikari, A. (2024). Yoyo cryptanalysis on future. *International Journal of Applied Cryptography*, 4(3-4):238–249.

Roy, H. S., Dey, P., Mondal, S. K., and Adhikari, A. (2024). Cryptanalysis of full round future with multiple biclique structures. *Peer-to-Peer Networking and Applications*, 17(1):397–409.

Schrottenloher, A. and Stevens, M. (2023a). Simplified modeling of mitm attacks for block ciphers: New (quantum) attacks. *IACR Transactions on Symmetric Cryptology*, 2023:146–183.

Schrottenloher, A. and Stevens, M. (2023b). Simplified modeling of mitm attacks for block ciphers: new (quantum) attacks. *Cryptology ePrint Archive*.

Shi, D., Sun, S., Song, L., Hu, L., and Yang, Q. (2023). Exploiting non-full key additions: Full-fledged automatic demirci-selcuk meet-in-the-middle cryptanalysis of skinny. Cryptology ePrint Archive, Paper 2023/255. `https://eprint.iacr.org/2023/255`.

Tao, B. and Wu, H. (2015). Improving the biclique cryptanalysis of aes. In *Information Security and Privacy: 20th Australasian Conference, ACISP 2015, Brisbane, QLD, Australia, June 29–July 1, 2015, Proceedings 20*, pages 39–56. Springer.

Xu, Z., Cui, J., Hu, K., and Wang, M. (2024). Integral attack on the full future block cipher. *Tsinghua Science and Technology*.

A. Complete Differentials used on the attack.

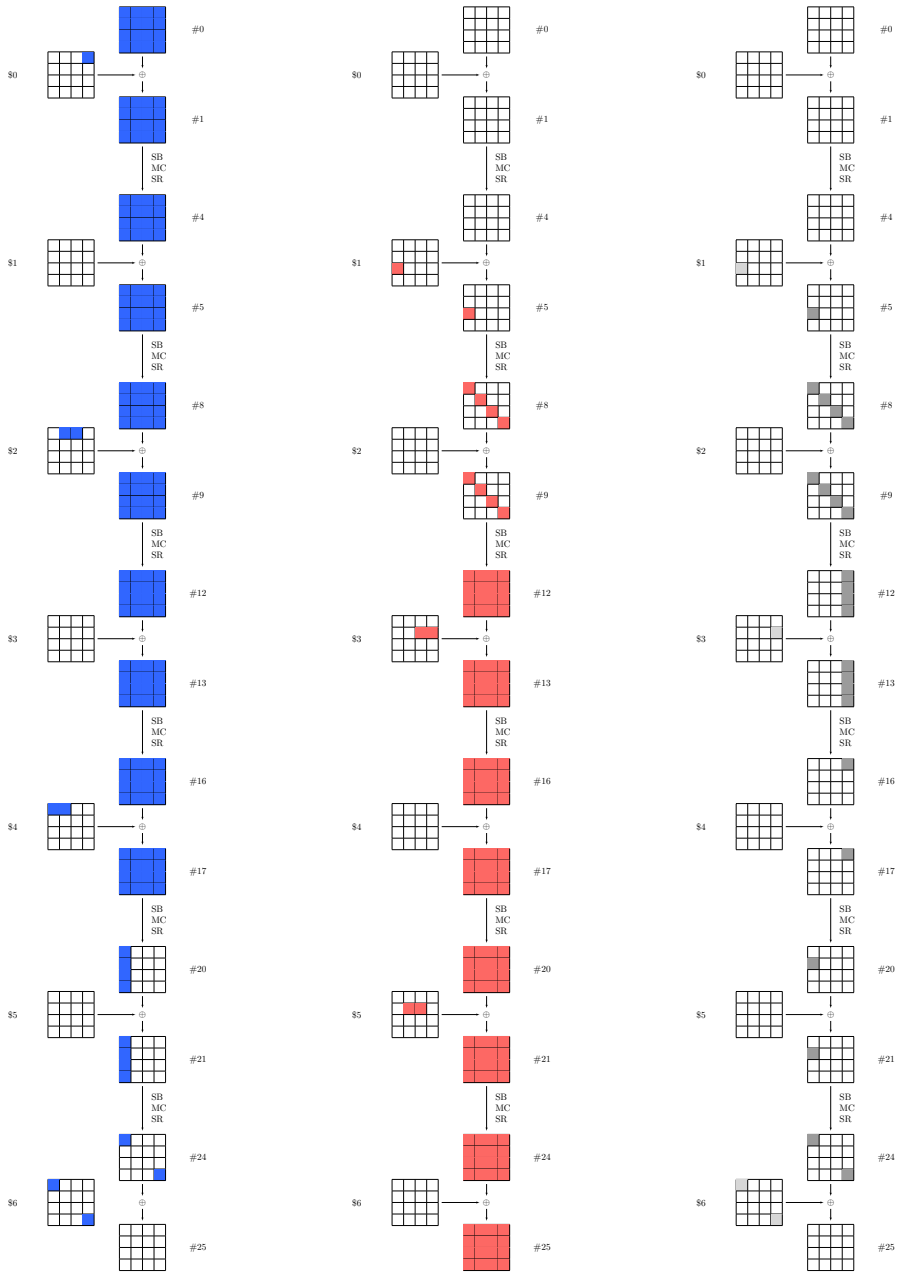


Figure 4. The rest of the Δ_i-differentials and ∇_j-differentials, excluding the biclique. The recomputation is also included here.