Improved Generic Attacks on Unbalanced Feistel Schemes with Expanding Functions

Emmanuel Volte[1], Valérie Nachef[1], and Jacques Patarin[2]

[1] Department of Mathematics, University of Cergy-Pontoise, CNRS UMR 8088
2 avenue Adolphe Chauvin, 95011 Cergy-Pontoise Cedex, France
[2] PRISM, University of Versailles
45 avenue des Etats-Unis, 78035 Versailles Cedex, France
emmanuel.volte@u-cergy.fr, valerie.nachef@u-cergy.fr
jacques.patarin@prism.uvsq.fr

Abstract. "Generic" Unbalanced Feistel Schemes with Expanding Functions are Unbalanced Feistel Schemes with truly random internal round functions from n bits to $(k-1)n$ bits with $k \geq 3$. From a practical point of view, an interesting property of these schemes is that since $n < (k-1)n$ and n can be small (8 bits for example), it is often possible to store these truly random functions in order to design efficient schemes (example: CRUNCH cf [6]). Attacks on these generic schemes were studied in [7] and [18]. As pointed in [7] and [18], there are surprisingly much more possibilities for these attacks than for generic balanced Feistel schemes or generic unbalanced Feistel schemes with contracting functions. In fact, this large number of attack possibilities makes the analysis difficult. In this paper, we shall methodically analyze again these attacks. We have created a computer program that systematically analyze all the possible attacks and detect the most efficient ones. We have detected a condition on the internal variables that was not clearly analyzed in [18], and we have found many new improved attacks by a systematic study of all the "rectangle attacks" when $k \leq 7$, and then we have generalized these improved attacks for all k. Many simulations on our improved attacks have also been done and they confirm our theoretical analysis.

Key words: Unbalanced Feistel permutations, pseudo-random permutations, generic attacks on encryption schemes, Block ciphers.

1 Introduction

A classical way to construct permutation $\{0,1\}^N$ to $\{0,1\}^N$ is to use Feistel schemes with d rounds built with round functions $f_1, \ldots, f_d$. In order to get "Random Feistel Scheme", these round functions need to be randomly chosen. "Generic attacks" on these schemes are attacks that are valid for most of the round functions.

The most usual Feistel schemes are when $N = 2n$ and the functions f_i are from $\{0,1\}^n$ to $\{0,1\}^n$. Such schemes are called "balanced Feistel Schemes" and

they have been studied a lot since the famous paper by M.Luby and C.Rackoff [12]. Many results have been obtained on the security of such classical Feistel schemes (see [13] for an overview of these results). When the number of rounds is lower than 5, we know attacks with less than $2^N (= 2^{2n})$ operations: for 5 rounds, an attack in $O(2^n)$ operations is given in [16] and for 3 or 4 rounds an attack in $\sqrt{2^n}$ is given in [1],[14]. When the functions are permutations, similar attacks for 5 rounds are given in [8] and [10]. Therefore, for security, at least 6 rounds are recommended, i.e. each bit will be changed at least 3 times.

When $N = kn$ and when the round functions are from $(k-1)n$ bits to n bits, we obtain what is called an "Unbalanced Feistel Scheme with contracting functions". In [13], M.Naor and O.Reingold give security when for the first and the last rounds pairwise independent functions are used instead of random contracting functions. In [20] security proofs for these schemes are also proved. At Asiacrypt 2006 ([17]) generic attacks on such schemes have been studied.

When $N = kn$ and when the round functions are from n bits to $(k-1)n$ bits, we obtain what is called an "Unbalanced Feistel Scheme with expanding functions", also called "complete target heavy unbalanced Feistel networks"(see [19]). Generic attacks on Unbalanced Feistel Schemes with expanding functions is the theme of this paper. One advantage of these schemes is that it requires much less memory to store a random function of n bits to $(k-1)n$ bits than a random function of $(k-1)n$ bits to n bits. Unbalanced Feistel Schemes with expanding functions together with the Xor of random permutations have been used in the construction of the hash function CRUNCH for the cryptographic hash algorithm competition organized by NIST in 2008 (cf [6]). Our results give a lower bound for the number of rounds used to construct this hash function.

Other kinds of Feistel Schemes are used for well known block ciphers. For example, BEAR and LION [2] are two block ciphers which employ both expanding and contracting unbalanced Feistel networks. The AES-candidate MARS is also using a similar structure.

Attacks on Unbalanced Feistel Schemes with expanding functions have been previously studied by C.S. Jutla ([7]) and improved attacks were given in [18]. However some of the attacks presented in [18] need too many conditions on the internal variables. These attacks work, but with weak keys. In this paper, we make a systematic study of the equations between the internal variables to avoid unlikely collisions on the round functions. Thus we get additional conditions. Nevertheless, with more conditions, we show that it is still possible to attack the same number of rounds as in [18]. In Known Plaintext Attacks (KPA), we obtain the same complexity except for $d = 3k-1$ where our complexity is slightly greater than in [18] but we do not have too many conditions on the internal variables. For Non-Adaptive Chosen Plaintext Attacks (CPA-1), we give a general method to obtain CPA-1 from KPA. Then we get complexities that are, most of the time, better than the ones in [18]. We also show that the best CPA-1 are not derived from the best KPA. For $k \leq 7$, we have generated all the possible attacks, thus the attacks presented here are the best possible attacks. We believe that the

generalization of these attacks for any k still gives the best possible attacks. We also provide simulation results for $k = 3$.

The paper is organized as follows. First we introduce some notation and definitions. Then we give an overview of the attacks. In Section 4, we show how we have generated all the possible attacks for $k \leq 7$. In Section 5, we introduce the different kinds of attacks we will used. These attacks named TWO, R1, R2, R3 and R4 generalize the attacks of [18]. Then in Section 6, we present R1, R2 KPA attacks. In Section 7, we show how to get CPA-1 from KPA. In Section 8, we study R1 and R2 CPA-1 and we give the results of our simulations. Finally, all the results are summarized in Section 9.

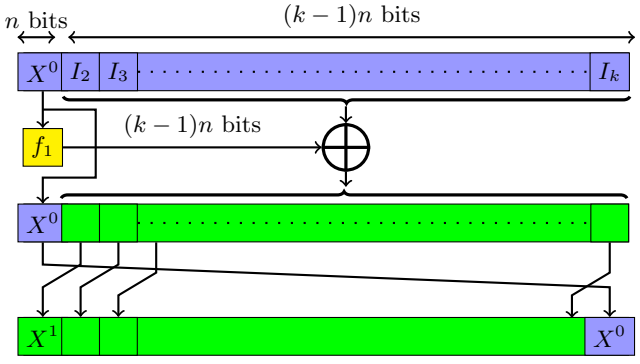
2 Notation

2.1 Unbalanced Feistel Schemes Notation

We first describe Unbalanced Feistel Scheme with Expanding Functions F_k^d and introduce some useful notations. F_k^d is a Feistel scheme of d rounds that produces a permutation from kn bits to kn bits. At each round j, we denote by f_j the round function from n bits to $(k-1)n$ bits. f_j is defined as $f_j = (f_j^{(1)}, f_j^{(2)}, \ldots, f_j^{(k-1)})$, where each function $f_j^{(i)}$ is defined from $\{0,1\}^n$ to $\{0,1\}^n$. On some input $[I_1, I_2, \ldots, I_k]$, F_k^d produces an output denoted by $[S_1, S_2, \ldots, S_k]$ by going through d rounds. At round j, the first n bits of the round entry are called X^{j-1}. We can notice that $I_1 = X^0$. We compute $f_j(X^{j-1})$ and obtain $(k-1)n$ bits. Those bits are xored to the $(k-1)n$ last bits of the round entry and the result is rotated by n bits.

The first round is represented on Figure 1 below:

Fig. 1. First Round



We have

$$\begin{aligned}
X^0 &= I_1 \\
X^1 &= I_2 \oplus f_1^{(1)}(I_1) \\
X^2 &= I_3 \oplus f_1^{(2)}(I_1) \oplus f_2^{(1)}(X^1) \\
X^3 &= I_4 \oplus f_1^{(3)}(I_1) \oplus f_2^{(2)}(X^1) \oplus f_3^{(1)}(X^2)
\end{aligned}$$

More generally, we can express the X^j recursively:

$$\forall \xi < k,\ X^\xi = I_{\xi+1} \bigoplus_{i=1}^{\xi} f_i^{(\xi-i+1)}(X^{i-1})$$

$$\forall \xi \geq 0,\ X^{k+\xi} = X^\xi \bigoplus_{i=2}^{k} f_{\xi+i}^{(k-i+1)}(X^{\xi+i-1})$$

After d rounds ($d \geq k+1$), the output $[S_1, S_2, \ldots, S_k]$ can be expressed by using the introduced values X^j:

$$\begin{aligned}
S_k &= X^{d-1} \\
S_{k-1} &= X^{d-2} \oplus f_d^{(k-1)}(X^{d-1}) \\
S_{k-2} &= X^{d-3} \oplus f_{d-1}^{(k-1)}(X^{d-2}) \oplus f_d^{(k-2)}(X^{d-1}) \\
&\ldots \\
S_\xi &= X^{d-1-k+\xi} \bigoplus_{i=d-k+\xi}^{d-1} f_{i+1}^{(\xi+d-i-1)}(X^i) \\
&\ldots \\
S_1 &= X^{d-k} \bigoplus_{i=d-k+1}^{d-1} f_{i+1}^{(d-i)}(X^i)
\end{aligned}$$

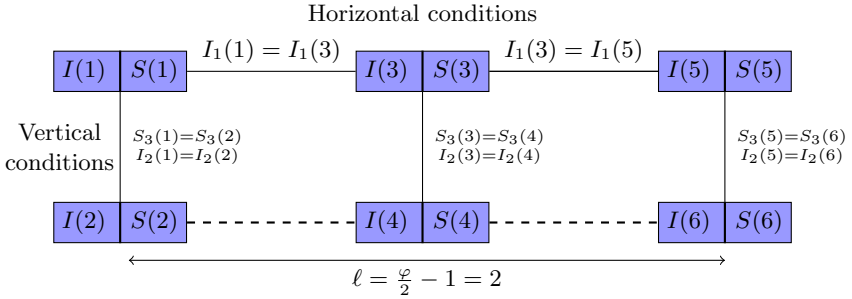
We don't need another notation, but for a better understanding we introduce a notation for the intermediate values. After round p, we obtain $[M_1^p, M_2^p, \ldots, M_k^p]$. So we have $M_1^p = X^p$, and for all $i \in \{1, 2, \ldots, k\}$ $M_i^0 = I_i$ and $M_i^d = S_i$.

2.2 Differential Attack Notation

Our attacks use sets of points. A point is a plaintext/ciphertext pair. The total number of points gives us the complexity of the attack. From the set of points we extract all the φ-tuple of distinct points $P(1)$, $P(2)$ $\ldots$ $P(\varphi)$, and we count how many φ-tuple verify some equalities (see Figure 2 for an example).

Now, we can describe an attack with a differential path. With the path we can explain why the number of φ-tuples that match the conditions is more important for a F_k^d scheme than for a random permutation. We introduce more definition. After p rounds, we define "horizontal equalities" on part M_i of the output M as $M_i^p(1) = M_i^p(3) = \ldots = M_i^p(\varphi - 1)$ and $M_i^p(2) = M_i^p(4) = \ldots = M_i^p(\varphi)$. Let $\ell = \frac{\varphi}{2} - 1$. "Vertical equalities" on part M_i are given by $\forall j,\ 0 \leq j \leq \ell$, $M_i^p(2j+1) = M_i^p(2j+2)$. We also define "differential equalities" on part M_i

Fig. 2. Example of equalities for $\varphi = 6$



by $\forall j,\ 0 \leq j \leq \ell - 1,\ M_i^p(2j+1) \oplus M_i^p(2j+2) = M_i^p(2j+3) \oplus M_i^p(2j+4)$. Notice that when we have the differential equalities, in order to get the horizontal equalities, it is enough to have the first sequence of equalities, and for the vertical equalities, it is enough to get only the first one. When we impose some equalities, we call them **conditions** (they are satisfied with probability $\frac{1}{2^n}$). This may imply that other equalities will be satisfied with probability 1. On the input and output variables we will always have ℓ differential conditions and either horizontal or vertical conditions. On the internal variables, we will get horizontal or vertical equalities and moreover we will impose more vertical or horizontal conditions. We need to always have differential equalities. When we impose new conditions on the internal variables, we must check that we do not add too many of them. We now give an example with an attack over the F_3^6 scheme. See Table 1.

Table 1. F_3^6 attack

i (round)	$M_1^i(2j+1) \oplus M_1^i(2j+2)$	$M_2^i(2j+1) \oplus M_2^i(2j+2)$	$M_3^i(2j+1) \oplus M_3^i(2j+2)$
0	$\mathbf{0}$	$\mathbf{0}$	Δ_1
1	0	Δ_1	0
2	$\bullet\Delta_1$	$\bullet 0$	0
3	$.\Delta_2$	Δ_3	$.\Delta_1$
4	$\mathbf{0}$	$.\mathbf{0}$	$.\Delta_2$
5	0	Δ_2	0
6	Δ_2	0	0

The "." in this table means that there are horizontal equalities or conditions. The "0" in the table means that there are vertical equalities or conditions. This notation will be used for any attack. We can count the total number of conditions for the different part: $n_I = 3\ell + 2$ (number of input conditions), $n_X = 2\ell + 2$ (number of internal conditions), $n_S = 3\ell + 2$ (number of output conditions). If a φ-tuple follow the path, i.e. if it satisfies both the input and the internal conditions, then it will verify the output conditions. But there exist other ways

to verify both these output conditions and the input conditions. So, we can prove that the number of φ-tuple will be greater for a F_k^d permutation.

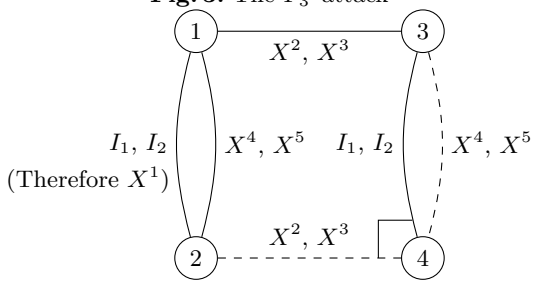
3 Example: CPA-1 attack on F_3^6

We present here a first example where we have obtained a new and better attack than previously known for F_3^6. In the next sections a complete analysis will be given for more general parameters. This attack is the one described in Table 1 with $\varphi = 4$ and so $\ell = 1$. Figure 3 illustrates this attack. It explains the terms of horizontal and vertical equalities. Moreover, conditions are represented by a solid edge and equalities that are automatically satisfied by a dotted edge.

We will generate all the possible messages $[I_1, I_2, I_3]$ such that $I_1 = 0$ and the first $n/2$ bits of I_2 are 0. So, we will generate exactly $m = 2^{3n/2}$ messages. How many 4-tuple of points will verify the input conditions ? For the first message we have m possibilities. For the second we have only 2^n possibilities because I_1 and I_2 are imposed by the first message. For the third point we have again m possibilities, and then we have no choice for the last point. Therefore there are $m^2 \times 2^n = 2^{4n}$ 4-tuple of points that satisfy all the input conditions. For a F_3^6 scheme, each of these tuple will satisfy at random the 4 internal conditions with a probability equal to $1/2^{4n}$. So, the expected number of 4-tuples that satisfy also the output conditions will be approximatively 1. Since there are 5 output conditions, the expected number of 4-tuple that satisfy the input conditions and the output conditions will be much lower for a random permutation. So, this CPA-1 attack will succeed with a high probability. We have found here a CPA-1 attack with $O(2^{3n/2})$ complexity and $O(2^{3n/2})$ messages. This is better than the $O(2^{5n/3})$ found in [18]. To find this complexity we can also use Table 3 with $r = 2$, $n_X = 4$, $\ell = 1$ and $k = 3$.

Moreover we have checked that all the other path conditions are verified (see Section 4) and this attack has been simulated by computer. For example, with $n = 10$ and 1000 attacks, we were able to distinguish 575 F_3^6 schemes from a random permutation, so the percentage of success is about 57.5%.

Fig. 3. The F_3^6 attack



4 Generation of all possible attacks for $k \leq 7$

In this section we describe the way we generate all the possible attacks for $k \leq 7$. First we choose a value for k, then we increase the value of d, beginning with $d = 1$, until we find no possible attacks. All the attacks (or sometimes only the best attacks when the number is too much important) are put in a specific file corresponding to the values of k and d.
To find an attack, we need to construct all the differential paths. There are two constraints for this construction:

- In the same round, it's not possible to have k vertical conditions, because it leads to a collision between the points, i.e. $P(1) = P(3) = \cdots = P(\varphi - 1)$ and $P(2) = P(4) = \cdots = P(\varphi)$.
- In the same round, it's not possible to have k horizontal conditions, because it also lead to a collision between the points, i.e. $P(1) = P(2)$ and $P(3) = P(4)$ and ... $P(\varphi - 1) = P(\varphi)$.

When the path is constructed, we look if the attack is valid. To be valid, an attack must overcome five constraints.

1. The complexity of the attack must be smaller than the total number of possible messages: $\frac{n_I + n_X}{2\ell + 2} \leq k$.
2. There must be less internal conditions than output conditions: $n_X \leq n_S$
3. If $n_X = n_S$ then n_S must be different from the number of final consecutive vertical conditions in the output conditions. If not, it is easy to prove that the output conditions are completely equivalent to the internal conditions. So, the output conditions will not happen more often than for a random permutation.
4. The number of equalities inside the path must be smaller than the number of variables included in them. Moreover we do not consider equalities when a variable occurs only once for all the equalities.
   Let us take an example. The F_3^6 attack given in section 2.2. The equations are: $f_3^{(1)}(X_2 \oplus \Delta_1) \oplus f_3^{(1)}(X_2) = \Delta_2$, $f_3^{(2)}(X_2 \oplus \Delta_1) \oplus f_3^{(2)}(X_2) = \Delta_3$, $f_4^{(1)}(X_3 \oplus \Delta_2) \oplus f_4^{(1)}(X_3) = \Delta_3$, $f_4^{(2)}(X_3 \oplus \Delta_2) \oplus f_4^{(2)}(X_3) = \Delta_1$. We have 4 equations and 5 variables X_2, Δ_1, Δ_2, Δ_3, X_3. All the variables are used at least in 2 equalities, so we cannot simplify.
5. There is no bottleneck in the equalities, i.e. any subset of equalities must have a greater number of variables. If it is not the case, the attack will only work with very particular functions (weak keys). This last point is very difficult to carry out without the help of a computer.

Finally, all the possible attacks are sorted in function of their complexity (KPA or CPA-1). For example there is 71 different attacks on the F_3^6 scheme, and 20 attacks with a CPA-1 complexity equal to $2^{3n/2}$.

All possible attacks are given in an extended version of this paper. In the next sections, we generalize for any k the best attacks (KPA and CPA-1) obtained for $k \leq 7$.

5 Different kinds of attacks: TWO, R_1, R_2, $R3$ and $R4$

5.1 TWO Attacks

The TWO attack consists in using m plaintext/ciphertexts pairs and in counting the number $\mathcal{N}_{F_k^d}$ of couples of these pairs that satisfy the relations between the input and output variables. We then compare $\mathcal{N}_{F_k^d}$ with $\mathcal{N}_{perm}$ where $\mathcal{N}_{perm}$ is the number of couples of pairs for a random permutation instead of F_k^d. The attack is successful, i.e. we are able to distinguish F_k^d from a random permutation if the difference $|E(\mathcal{N}_{F_k^d}) - E(\mathcal{N}_{perm})|$ is much larger than the standard deviation σ_{perm} and than the standard deviation $\sigma_{F_k^d}$, where E denotes the expectancy function.

These attacks give the best attacks from 1 round to $k+2$ rounds. They are studied in [18]. Their complexity is summarized in Section 9.

5.2 R1 Attacks

Here we have vertical conditions on the input and output variables. These attacks are more general than the attacks named R1 in [18] since we allow more vertical conditions on the input and output variables. These attacks were first described by Jutla ([7]). With our differential notation, we have:

	I_1	$\dots$	I_r	I_{r+1}	$\dots$	I_k		S_1	$\dots$	S_{k-v}	S_{k-v+1}	$\dots$	S_k
Round 0	0	$\dots$	0	Δ^0_{r+1}	$\dots$	Δ^0_k	Round d	Δ^d_1	$\dots$	Δ^d_{k-v}	0	$\dots$	0

Thus, $n_I = k\ell + r$, $n_X = t\ell + w$, $n_S = k\ell + v$. Here n_I denotes the conditions on the input variables. $\ell = \frac{\varphi}{2} - 1$. The number of vertical conditions on the input variables is r. n_X denotes the number of conditions on the internal variables. We use t for horizontal conditions and w for vertical conditions. Similarly, n_S and v denote respectively the number of conditions and the number of vertical conditions on the output variables. Then the number of rounds is given by $r + t + w$. When $n_X \leq n_S$, we can easily obtain a sufficient condition of success (without computing the standard deviation), since in that case we will have for most permutation about 2 times more solutions with F_k^d than with a random permutation. Here this gives the condition: $(k-t)\ell \geq w - v$. In order to avoid weak keys, the number of equations with the internal variables must be smaller than or equal to the number of internal variables. This condition was not always satisfied in [18]. For R1 attacks, it is easy to check that the number of equations is given by $t(k-1)$ and the number of variables is $k(t+1) - r - w$. Thus we get the condition: $r + w \leq t + k$. The complexity of such an attack is $2^{\frac{n_I + n_X}{\varphi} n}$. This implies $\frac{n_I + n_X}{\varphi} \leq k$, i.e. $\frac{(k+t)\ell + r + w}{2\ell + 2} \leq k$.

5.3 R2 Attacks

Here we have horizontal conditions on the input variables and vertical conditions on the output variables. Again these attacks are more general than the attacks named R2 in [18] since we allow more horizontal conditions on the input variables and more vertical conditions on the output variables. We have:

	I_1	$\ldots$	I_u	I_{u+1}	$\ldots$	I_k		S_1	$\ldots$	S_{k-v}	S_{k-v+1}	$\ldots$	S_k
Round 0	$.\Delta_1^0$	$\ldots$	$.\Delta_u^0$	Δ_{u+1}^0	$\ldots$	Δ_k^0	Round d	Δ_1^d	$\ldots$	Δ_{k-v}^d	0	$\ldots$	0

Thus, $n_I = (k+u)\ell$, $n_X = t\ell + w$, $n_S = k\ell + v$. The number of horizontal conditions on the input variables is denoted by u. The number of rounds is given by $u + t + w$. The condition $n_X \leq n_S$ is equivalent to $(k-t)\ell \geq w - v$. For $R2$ attacks, it is easy to check that the number of equations is given by $(t+1)(k-1)$ and the number of variables is $k(t+2) - w$. Thus we get the condition: $w \leq t + k + 1$. The complexity of such an attack is $2^{\frac{n_I+n_X}{\varphi}n}$. This implies $\frac{n_I+n_X}{\varphi} \leq k$, i.e. $\frac{(k+t+u)\ell+w}{2\ell+2} \leq k$.

5.4 R3 and R4 Attacks

We describe briefly, R3 and R4 attacks. It is easy to get the number of rounds and the conditions on the number of equations and variables.
For R3 attacks, we have vertical conditions on the input variables and horizontal conditions on the output variables. This gives:

	I_1	$\ldots$	I_r	I_{r+1}	$\ldots$	I_k		S_1	$\ldots$	S_{k-s}	S_{k-s+1}	$\ldots$	S_k
Round 0	0	$\ldots$	0	Δ_{r+1}^0	$\ldots$	Δ_k^0	Round d	Δ_1^d	$\ldots$	Δ_{k-s}^d	$.\Delta_{k-s+1}^d$	$\ldots$	$.\Delta_k^d$

and $n_I = k\ell + r$, $n_X = t\ell + w$, $n_S = (k+s)\ell$.
For R4 attacks, we have horizontal conditions on the input and output variables. This gives:

	I_1	$\ldots$	I_u	I_{u+1}	$\ldots$	I_k		S_1	$\ldots$	S_{k-s}	S_{k-s+1}	$\ldots$	S_k
Round 0	$.\Delta_1^0$	$\ldots$	$.\Delta_u^0$	Δ_{u+1}^0	$\ldots$	Δ_k^0	Round d	Δ_1^d	$\ldots$	Δ_{k-s}^d	$.\Delta_{k-s+1}^d$	$\ldots$	$.\Delta_k^d$

and $n_I = (k+u)\ell$, $n_X = t\ell + w$, $n_S = (k+s)\ell$.

6 Best KPA attacks: R_1, R_2

In this section we describe the best attacks we have found. As mentioned before, we know that for $k \leq 7$, they are the best possible attacks. We will mostly describe one example of R2 attacks since for any round there are many possible R2 attacks that give the best complexity. It can be noticed that in KPA, there is a symmetry between R2 and R3 attacks. Thus there always exist R2 and R3 attacks with the same complexity. Sometimes, it is also possible to have R1 attacks. Most of the time, R4 attacks are worse. We give attacks from $k+3$ rounds to $3k-1$ rounds since from 1 to $k+2$ rounds, TWO attacks are most of the time better and they are described in [18]. In all our attacks, it is easily checked that the conditions given in the previous section are satisfied. Moreover, we always look for attacks where the number of points is minimum. Our best R2 KPA attacks are summarized in Table 2:

Remarks:

1. We have the following R1 attacks:

Table 2. Best known KPA on F_k^d, for any $k \geq 3$

d values	n_I	n_X	n_S	ℓ	Complexity
$k+2q \in [k{+}3, 2k{-}2]$	$(2k{-}1)\ell$	$q\ell+q+1$	$k\ell{+}q{+}1$	1	$2^{\frac{k+d}{4}n}$
$k+2q+1 \in [k{+}3, 2k{-}2]$	$(2k{-}1)\ell$	$q\ell+q+2$	$k\ell{+}q{+}2$	1	$2^{\frac{k+d}{4}n}$
$k+2q \in [2k{-}1, 3k{-}2]$	$(2k{-}1)\ell$	$(q-\lfloor\frac{k-1}{2}\rfloor)\ell+q+\lfloor\frac{k+1}{2}\rfloor$	$k\ell{+}k{-}1$	1	$2^{\frac{k+d}{4}n}$
$k+2q+1 \in [2k{-}1, 3k{-}2]$	$(2k{-}1)\ell$	$(q-\lfloor\frac{k-3}{2}\rfloor)\ell+q+\lfloor\frac{k+1}{2}\rfloor$	$k\ell{+}k{-}1$	1	$2^{\frac{k+d}{4}n}$
$3k-1$	$k\ell{+}\ell$	$k\ell+2k-1$	$k\ell{+}k{-}1$	k	$2^{(k-\frac{1}{2k+2})n}$

(a) When $k+3 \leq d \leq 2k-2$ and $d = k+2q$, we set

$$n_I = k\ell+k-1, \quad n_X = q\ell+q+1, \quad n_S = k\ell+q+1$$

It is possible to choose $\ell = 1$ and the complexity is also $2^{\frac{k+d}{4}n}$

(b) When $2k-1 \leq d \leq 3k-2$ and $d = k+2q$, we set

$$n_I = k\ell+2, \quad n_X = q\ell+k+q-2, \quad n_S = k\ell+k-1$$

The complexity is still $2^{\frac{k+d}{4}n}$, but ℓ is greater than 1.

2. In [7], Jutla gave a R1 attack on $3k-3$ rounds but the complexity that we obtain with a R2 attack here is better. It is possible to perform a R1 attack on $3k-2$ rounds just by adding a vertical condition on the input variables to the attack on $3k-3$ rounds and the we obtain the same complexity as the one we get with a R2 attack. Due to the conditions between the number of equations and internal variables, it is not possible to use the same idea for $3k-1$ rounds. In this last case, we have R2 (and of course R3) attacks.

7 Way to transform KPA Attacks into CPA-1 Attacks

We have analyzed all the possible situations and we are now able to present formulas that give us directly the CPA complexity depending on the initial conditions. We call u the number of horizontal conditions, r the number of vertical condition and $\delta = |u-r|$. So we can distinguish four cases:

Case 1 $u = 0$: $\underbrace{0\ 0\ \ldots\ 0}_{r\text{ “0”}}\ \Delta_1\ \ldots\ \Delta_{k-r}$

Case 2 $r = 0$: $\overbrace{.\Delta_1\ .\Delta_2\ \ldots\ .\Delta_u}^{u\text{ “.”}}\ \Delta_{u+1}\ldots\ \Delta_{k-r}$

Case 3 $u \leq r$: $\underbrace{\overbrace{.0\ .0\ \ldots\ .0}^{u\text{ “.”}}\ \overbrace{0\ \ldots\ 0}^{\delta=r-u}}_{r\text{ “0”}}\ \Delta_1\ \ldots\ \Delta_{k-r},$

Case 4 $u \geq r$: $\underbrace{\overbrace{.0\ .0\ \ldots\ .0}^{u\text{ “.”}}}_{r\text{ “0”}}\ .\underbrace{\Delta_1\ \ldots\ .\Delta_\delta}_{\delta=u-r}\ \Delta_{\delta+1}\ldots\ \Delta_{k-r}$

Table 3. KPA to CPA

Conditions			$\log_{2^n}(CPA)$
$u = 0$	$\frac{n_X}{\ell+2} \leq k - r$		$\frac{n_X}{\ell+2}$
r vertical conditions	$\frac{n_X}{\ell+2} > k - r$		$\frac{n_X - k + r}{\ell+1}$
$r = 0$	$\frac{n_X}{\ell+2} \leq k - u$		$\frac{n_X}{\ell+2}$
u horizontal conditions	$\frac{n_X}{\ell+2} > k - u$		$\frac{n_X - \ell(k-u)}{2}$
$r \neq 0$ and $u \neq 0$	$u \leq r$	$(\ell+2)(k-r) > n_X$	$\frac{n_X}{\ell+2}$
		$(\ell+2)(k-r) + (\ell+1)\delta > n_X$ and $(\ell+2)(k-r) \leq n_X$	$\frac{n_X - k + r}{\ell+1}$
		$(\ell+2)(k-r) + (\ell+1)\delta \leq n_X$	$n_X - k + r - \ell(k-u)$
u horizontal conditions and r vertical conditions	$u > r$	$(\ell+2)(k-u) > n_X$	$\frac{n_X}{\ell+2}$
		$(\ell+2)(k-u) + 2\delta > n_X$ and $(\ell+2)(k-u) \leq n_X$	$\frac{n_X - \ell(k-u)}{2}$
		$(\ell+2)(k-u) + 2\delta \leq n_X$	$n_X - k + r - \ell(k-u)$

We can notice that the best CPA-1 attacks do not always come from the best KPA attacks. Nevertheless, if we want to express the CPA complexity with the KPA complexity, we can use the following formula: $\log_{2^n}(KPA) = \frac{r + (u+k)\ell + n_X}{2\ell + 2}$.
For all the CPA-1 Attacks we found, we prove that the best choice is to keep the first bits constant and generate all the possible messages with the same first bits.

Let's show how we prove it for Case 1. The best way to choose messages is to keep some of the bits constant (for example equal to zero) and consider all the possible combination for the other bits. We call b the number of varying bits among the first rn bits, and we call β the number of varying bits among the last $(k-r)n$ bits. So we have $0 \leq b \leq rn$ and $0 \leq \beta \leq (k-r)n$, and this allow us to generate $2^{b+\beta}$ points (plaintext/cyphertext pair). Now we count how many φ-tuples $M^0(1), \ldots, M^0(\varphi)$ of points will verify the input conditions. For $M^0(1)$ we have $2^{b+\beta}$ possibilities, for $M^0(2)$ only $2^\beta - 1 \approx 2^\beta$ possibilities, because the first rn bits are imposed by $M^0(1)$. For $M^0(3)$ we have again $2^{b+\beta} - 2 \approx 2^{b+\beta}$ possibilities. For $M^0(4)$ only one possibility : $M^0(4) = M^0(3) \oplus (M^0(1) \oplus M^0(2))$. We continue like this until we reach the last two points. For $M^0(\varphi - 1)$ we have again almost $2^{b+\beta}$ possibilities, and for $M^0(\varphi)$ only one possibility. So, the total number of φ-tuples is $\left(2^{b+\beta}\right)^{\varphi/2} \times 2^\beta = 2^{(b+\beta)(\ell+1)+\beta}$. The complexity of the CPA-1 is equal to $2^{b+\beta}$. We want this number to be as small as possible, and

at the same time we want to generate a maximum of φ-tuples that satisfy the input conditions. So, we want to have β as large as possible. Each φ-tuple has a probability equal to $1/2^{n_X \cdot n}$ to satisfy the internal conditions. In order to have a reasonable chance to realize these conditions, we must have $(b+\beta)(\ell+1)+\beta = n_X \cdot n$. If $b = 0$ we get $\beta = \frac{n_X}{\ell+2}n$. But this is possible only if $\beta \leq (k-r)n$, i.e. if $\frac{n_X}{\ell+2} \leq k-r$. If we have $\frac{n_X}{\ell+2} > k-r$ then we must take the maximum possible value for β: $\beta = (k-r)n$ and that gives us a CPA-1 complexity equal to $2^{b+\beta} = 2^{\frac{n_X-k+r}{\ell+1}n}$.

All the cases are summarized in Table 3.

8 Best CPA-1 Attacks: R_1, R_2. Simulation.

8.1 CPA-1 attacks

In this section, we describe the best CPA-1 that we have obtained. Again for $k \leq 7$ we know that we have the best possible attacks. Except for $3k-1$ rounds, we obtain a better complexity than in [18]. The best CPA-1 are generally R2 attacks. Sometimes R1 attacks exist with the same complexity. It is interesting to note that the best CPA-1 do not come from the best KPA. We will use the study of CPA-1 made in Section 7. We will describe CPA-1 for $k+3 \leq d \leq 3k-1$ since for $d \leq k+2$, the best attacks are the TWO attacks given in [18]. Again we will give an example of such an attack for each round. We notice that for the same conditions on the input and output variables, we can find several attacks: the horizontal and vertical conditions on the internal variables can be displayed differently inside the attack, but we must respect the conditions between the number of equations and variables at each step of the attack. An example is given at the end of this section. Our best R2 CPA-1 attacks are summarized in the following table:

Table 4. Best known CPA-1 on F_k^d, for any $k \geq 3$

d values	n_I	n_X	n_S	ℓ	Complexity
$k+3$	$k\ell+(k-1)\ell$	$\ell+3$	$k\ell+1$	1	$2^{\frac{3n}{2}}$
$k+4$	$k\ell+(k-2)\ell$	$2\ell+4$	$k\ell+2$	1	2^{2n}
$k+5$	$k\ell+(k-2)\ell$	$2\ell+5$	$k\ell+2$	1	$2^{5n/2}$
$k+2q \in [k+6, 3k-4]$	$k\ell+(k-q)\ell$	$(q-1)\ell+2q+1$	$k\ell+1$	$q-1$	$2^{\frac{q^2+2}{q+1}n}$
$k+2q+1 \in [k+7, 3k-5]$	$k\ell+(k-q)\ell$	$(q-1)\ell+2q+2$	$k\ell+1$	$q-1$	$2^{\frac{q^2+3}{q+1}n}$
$3k-3$	$k\ell+\ell$	$(k-2)\ell+2k-2$	$k\ell+1$	$k-1$	$2^{\frac{(k-1)k}{k+1}n}$
$3k-2$	$k\ell+(k-1)\ell$	$(k-\lfloor\frac{k}{2}\rfloor)\ell+2k-\lfloor\frac{k-1}{2}\rfloor$	$k\ell+k-1$	1	$2^{(k-1)n}$
$3k-1$	$k\ell+\ell$	$(k-1)\ell+2k-1$	$k\ell+k-1$	k	$2^{(k-\frac{1}{2})n}$

Remark. For $d = k+3, k+4, k+5$ and $3k-2$, there exist R1 attacks with the same complexity and the same number of points.

8.2 Overview of the R2 CPA-1 Attack on F_k^{3k-1}

We did a simulation of our best CPA-1 Attack. The input and output conditions were the following:

	I_1	I_2	...	I_k		S_1	S_2	...	S_k
Round 0	$.\Delta_1^0$	Δ_2^0	...	Δ_k^0	Round d	Δ_1^d	0	...	0

Several different differential paths match with these input and output conditions. For example let's see all the R2 path for the F_3^8 and F_4^{11}permutations. See Table 5 and Table 9 in Appendix A.

Table 5. All the paths for the R2 attack against F_3^8, $\varphi = 8$

	0	$.\Delta_1$	Δ_2	Δ_3		0	$.\Delta_1$	Δ_2	Δ_3
	1	0	0	$.\Delta_1$		1	0	Δ_4	$.\Delta_1$
	2	0	Δ_1	0		2	$.\Delta_4$	Δ_1	0
	3	$.\Delta_1$	0	0		3	0	0	$.\Delta_4$
Path 1:	4	0	Δ_4	$.\Delta_1$	Path 2:	4	0	Δ_4	0
	5	$.\Delta_4$	Δ_1	0		5	$.\Delta_4$	0	0
	6	0	0	$.\Delta_4$		6	0	0	$.\Delta_4$
	7	0	Δ_4	0		7	0	Δ_4	0
	8	Δ_4	0	0		8	Δ_4	0	0

We counted the number of paths for $k \leq 7$:

k	3	4	5	6	7
# path	2	8	27	89	296

We will see that, the greater k is, the better the attacks work.

8.3 Experimental results

We did simulations of these CPA-1 attacks. For each simulation, we generate a random Feistel scheme with 20 rounds, and a F_k^{3k-1} scheme. For both schemes, we compute $2^{(k-1/2)n}$ ciphertext/plaintext pairs, by varying only the last $(k - 1/2)n$ bits. After this, we extract all the couples of points that satisfy both input and output conditions. We sort these couples of points in order to count how many φ-tuples of points match the input and output condition. If we found q couples of points that satisfy all these conditions with $q \geq \varphi/2$, we count as if we have found $\frac{q!}{(q-\varphi/2)!}$ φ-tuples, because this is the number of φ-tuples we can take out these points, by changing the position of the couple of points. Once this is finished, we compare the number found for each permutation. Most of the time, that enables us to distinguish between them. See Table 6.

9 Summary of the attacks

In Tables 7 and 8, we give the complexity of the attacks we have found. For $k \leq 7$, since we have generated all the attacks, these are the best possible attacks.

Table 6. Experimental results for F_k^{3k-1}

n	k	kn	% of success	% of false alarm	# iteration
2	3	6	29,09%	0,35%	100000
2	4	8	61,6%	0,06%	10000
2	5	10	98,37%	0%	10000
2	6	12	99,99%	0%	10000
2	7	14	100%	0%	10000
2	8	16	100%	0%	1000
2	9	18	100%	0%	500
2	10	20	100%	0%	100
4	3	12	21,15%	1,12%	10000
4	4	16	42,5%	0%	1000
4	5	20	93%	0%	100
4	6	24	100%	0%	100
6	3	18	8%	1,2%	500
8	3	24	2%	0%	100

Then we have generalized the results for $k > 7$ and we believe that the attacks presented here are also the best possible attacks. For $d \leq k + 2$, we have TWO attacks. For $d \geq k + 3$, we have rectangle attacks. As mentioned before, in KPA, there are always R2 and R3 attacks that give the best complexity sometimes there is also a R1 attacks (for $3k - 2$ rounds for example). In CPA-1, the best complexity is given by R2 attacks, and sometimes R1 attacks.

Table 7. Best known TWO and Rectangle attacks on F_3^d. Details about the parameters in this table: (new) means that we have found a better attack than previously known.

	KPA	CPA-1
F_3^1	1	1
F_3^2	$2^{\frac{n}{2}}$, TWO	2
F_3^3	2^n, TWO	2
F_3^4	$2^{\frac{3}{2}n}$, TWO	$2^{\frac{n}{2}}$, TWO
F_3^5	2^{2n}, TWO	2^n, TWO
F_3^6	$2^{\frac{9}{4}n}$, R2, R3	$2^{\frac{3}{2}n}$, R2 (new)
F_3^7	$2^{\frac{5}{2}n}$, R1, R2, R3	2^{2n}, R2
F_3^8	$2^{\frac{23}{8}n}$, R2, R3	$2^{\frac{5}{2}n}$, R2

Table 8. Best known TWO and Rectangle attacks on F_k^d, for any $k \geq 3$. Details about the parameters in this table: (new) means that we have found a better attack than previously know.

	KPA	CPA-1
F_k^1	1	1
F_k^2	$2^{\frac{n}{2}}$, TWO	2
F_k^3	2^n, TWO	2
F_k^d, $2 \leq d \leq k$	$2^{\frac{d-1}{2}n}$,TWO	2
F_k^{k+1}	$2^{\frac{k}{2}n}$, TWO	$2^{\frac{n}{2}}$, TWO
F_k^{k+2}	$2^{\frac{k+1}{2}n}$, TWO	2^n, TWO
F_k^{k+3}	$2^{\frac{2k+3}{4}n}$, R2, R3	$2^{3n/2}$, R2 (new)
F_k^{k+4}	$2^{\frac{k+2}{2}n}$, R1, R2, R3	2^{2n}, R2 (new)
F_k^{k+5}	$2^{\frac{2k+5}{4}n}$, R2, R3	$2^{5n/2}$, R2 (new)
$\vdots$	$\vdots$	$\vdots$
F_k^d, d= $k+2q$, $3 \leq q \leq k-2$	$2^{\frac{d+k}{4}n}$, R1, R2, R3	$2^{\frac{q^2+2}{q+1}n}$, R2 (new)
F_k^d, d= $k+2q+1$, $3 \leq q \leq k-3$	$2^{\frac{d+k}{4}n}$, R2, R3	$2^{\frac{q^2+3}{q+1}n}$, R2 (new)
$\vdots$	$\vdots$	$\vdots$
F_k^{3k-3}	$2^{(k-\frac{3}{4})n}$, R2, R3	$2^{\frac{(k-1)k}{k+1}n}$, R2 (new)
F_k^{3k-2}	$2^{(k-\frac{1}{2})n}$, R1, R2, R3	$2^{(k-1)n}$, R2 (new)
F_k^{3k-1}	$2^{(k-\frac{1}{2k+2})n}$, R2, R3, (*)	$2^{(k-\frac{1}{2})n}$, R2

In these tables,"new" means that the complexity that we obtain is better than the complexity given in [18]. $(*)$ means that for $3k-1$ rounds our complexity is worse than the complexity in [18]. This comes from the fact, as we mentioned earlier, that the conditions between the equations and the internal variables were not all considered in [18].

10 Conclusion

In this paper we make a systematic study of rectangle generic attacks on unbalanced Feistel schemes with expanding functions. Although these attacks were

already analyzed in [7] and [18], this paper brings many improvements. Generation of all possible rectangle attacks for $k \leq 7$ was performed thanks to a computer program and the most efficient ones were selected. Then the generalization for any k was possible. This gives attacks for which conditions between equations and internal variables are satisfied. This was not detected in [18]. We also provide a complete description of the way to obtain CPA-1 from KPA. This shows how to get the best CPA-1 and we improved the CPA-1 complexity of [18]. Also many simulations confirm our theoretical results.

There are still some open problems. It would be interesting to complete the program in order to generate all the attacks for any k. This seems to be a memory space problem. Also, in this paper, we did not study attacks with complexity greater than kn. In that case, we need to attack permutations generators and not only one single permutation. In [18], attacks called "multi-rectangle attacks" were introduced, but so far no significant results have been obtained on these attacks. It might give a new way to study generic attacks on unbalanced Feistel schemes with expanding functions. As we mentioned in Section 3, when we have exactly the same condition on the input and output variables, there are many possible CPA-1 attacks (for $k = 7$, there exist 286 attacks on F_7^{20}, with the same conditions on the input and output variables). An estimation for any k will strengthen the attack.

References

1. W. Aiello and R. Venkatesan. Foiling Birthday Attacks in Length-Doubling Transformations - Benes: A Non-Reversible Alternative to Feistel. In Ueli M. Maurer, editor, *Advances in Cryptology – EUROCRYPT '96*, volume 1070 of *Lecture Notes in Computer Science*, pages 307–320. Springer-Verlag, 1996.
2. R. J. Anderson and E. Biham. Two Practical and Provably Secure Block Ciphers: BEARS and LION. In Dieter Gollman, editor, *Fast Software Encryption*, volume 1039 of *Lecture Notes in Computer Science*, pages 113–120. Springer-Verlag, 1996.
3. D. Coppersmith. Another Birthday Attack. In Hugh C. Williams, editor, *Advances in Cryptology – CRYPTO '85*, volume 218 of *Lecture Notes in Computer Science*, pages 14–17. Springer-Verlag, 1985.
4. D. Coppersmith. Luby-Rackoff: Four rounds is not enough. Technical Report RC20674, IBM Research Report, December 1996.
5. M. Girault, R. Cohen, and M. Campana. A Generalized Birthday Attack. In C. G. Guenther, editor, *Advances in Cryptology – EUROCRYPT '88*, volume 330 of *Lecture Notes in Computer Science*, pages 129–156. Springer-Verlag, 1988.
6. L. Goubin, M. Ivasco, W. Jalby, O. Ly, V. Nachef, J. Patarin, J. Treger, and E. Volte. CRUNCH. Technical report, Submission to NIST, October 2008.
7. C. S. Jutla. Generalized Birthday Attacks on Unbalanced Feistel Networks. In Hugo Krawczyk, editor, *Advances in Cryptology – CRYPTO '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 186–199. Springer-Verlag, 1998.
8. L. R. Knudsen. DEAL - A 128-bit Block Cipher. Technical Report 151, University of Bergen, Department of Informatics, Norway, february 1998.
9. L. R. Knudsen, X. Lai, and B. Preneel. Attacks on Fast Double Block Length Hash Functions. *J. Cryptology*, 11(1):59–72, 1998.

10. L. R. Knudsen and V. Rijmen. On the Decorrelated Fast Cipher (DFC) and Its Theory. In Lars R. Knudsen, editor, *Fast Software Encrytion – FSE '99*, volume 1636 of *Lecture Notes in Computer Science*, pages 81–94. Springer-Verlag, 1999.
11. M. Luby. *Pseudorandomness and Cryptographic Applications*. Princeton University Press, 1996.
12. M. Luby and C. Rackoff. How to Construct Pseudorandom Permutations from Pseudorandom Functions. *SIAM J. Comput.*, 17(2):373–386, 1988.
13. M. Naor and O. Reingold. On the Construction of Pseudorandom Permutations: Luby-Rackoff Revisited. *J. Cryptology*, 12(1):29–66, 1999.
14. J. Patarin. New Results on Pseudorandom Permutation Generators Based on the DES Scheme. In Joan Feigenbaum, editor, *Advances in Cryptology – CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 301–312. Springer-Verlag, 1991.
15. J. Patarin. Generic Attacks on Feistel Schemes. In Colin Boyd, editor, *Advances in Cryptology – ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 222–238. Springer-Verlag, 2001.
16. J. Patarin. Security of Random Feistel Schemes with 5 or More Rounds. In Matthew K. Franklin, editor, *Advances in Cryptology – CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 106–122. Springer-Verlag, 2004.
17. J. Patarin, V. Nachef, and C. Berbain. Generic Attacks on Unbalanced Feistel Schemes with Contracting Functions. In Xuejia Lai and Kefei Chen, editors, *Advances in Cryptology – ASIACRYPT 2006*, volume 4284 of *Lecture Notes in Computer Science*, pages 396–411. Springer-Verlag, 2006.
18. J. Patarin, V. Nachef, and C. Berbain. Generic Attacks on Unbalanced Feistel Schemes with Expanding Functions. In Kaoru Kurosawa, editor, *Advances in Cryptology – ASIACRYPT 2007*, volume 4833 of *Lecture Notes in Computer Science*, pages 325–341. Springer-Verlag, 2007.
19. B. Schneier and J. Kelsey. Unbalanced Feistel Networks and Block Cipher Design. In Dieter Gollmann, editor, *Fast Software Encrytion – FSE '96*, volume 1039 of *Lecture Notes in Computer Science*, pages 121–144. Springer-Verlag, 1996.
20. A. Yun, J. H. Park, and J. Lee. Lai-Massey Scheme and Quasi-Feistel Networks. *Cryptology ePrint archive: 2007/347: Listing for 2007.*

A All the paths for the R2 attack against F_4^{11}, $\varphi = 10$

Table 9. All the paths for the R2 attack against F_4^{11}, $\varphi = 10$

0	$.\Delta_1$	Δ_2	Δ_3	Δ_4	$.\Delta_1$	Δ_2	Δ_3	Δ_4	$.\Delta_1$	Δ_2	Δ_3	Δ_4	$.\Delta_1$	Δ_2	Δ_3	Δ_4
1	0	Δ_5	Δ_6	$.\Delta_1$	0	Δ_5	Δ_6	$.\Delta_1$	0	Δ_5	Δ_6	$.\Delta_1$	0	0	Δ_5	$.\Delta_1$
2	$.\Delta_5$	Δ_6	Δ_1	0	$.\Delta_5$	Δ_6	Δ_1	0	$.\Delta_5$	Δ_6	Δ_1	0	0	Δ_5	Δ_1	0
3	0	0	Δ_7	$.\Delta_5$	0	Δ_7	Δ_8	$.\Delta_5$	0	0	0	$.\Delta_5$	$.\Delta_5$	Δ_1	0	0
4	0	Δ_7	Δ_5	0	$.\Delta_7$	Δ_8	Δ_5	0	0	0	Δ_5	0	0	Δ_6	Δ_7	$.\Delta_5$
5	$.\Delta_7$	Δ_5	0	0	0	0	Δ_9	$.\Delta_7$	0	Δ_5	0	0	$.\Delta_6$	Δ_7	Δ_5	0
6	0	Δ_8	Δ_9	$.\Delta_7$	0	Δ_9	Δ_7	0	$.\Delta_5$	.0	0	0	0	Δ_8	Δ_9	$.\Delta_6$
7	$.\Delta_8$	Δ_9	Δ_7	0	$.\Delta_9$	Δ_7	0	0	$.\Delta_7$	Δ_8	Δ_9	$.\Delta_5$	$.\Delta_8$	Δ_9	Δ_6	0
8	0	0	0	$.\Delta_8$	0	0	0	$.\Delta_9$	0	0	.0	$.\Delta_7$	0	0	0	$.\Delta_8$
9	0	0	Δ_8	0	0	0	Δ_9	0	0	0	Δ_7	0	0	0	Δ_8	0
10	0	Δ_8	0	0	0	Δ_9	0	0	0	Δ_7	0	0	0	Δ_8	0	0
11	Δ_8	0	0	0	Δ_9	0	0	0	Δ_7	0	0	0	Δ_8	0	0	0

0	$.\Delta_1$	Δ_2	Δ_3	Δ_4	$.\Delta_1$	Δ_2	Δ_3	Δ_4	$.\Delta_1$	Δ_2	Δ_3	Δ_4	$.\Delta_1$	Δ_2	Δ_3	Δ_4
1	0	0	0	$.\Delta_1$	0	0	0	$.\Delta_1$	0	0	Δ_5	$.\Delta_1$	0	0	Δ_5	$.\Delta_1$
2	0	0	Δ_1	0	0	0	Δ_1	0	0	Δ_5	Δ_1	0	0	Δ_5	Δ_1	0
3	0	Δ_1	0	0	0	Δ_1	0	0	$.\Delta_5$	$.\Delta_1$	0	0	$.\Delta_5$	Δ_1	0	0
4	$.\Delta_1$	.0	0	0	$.\Delta_1$	0	0	0	$.\Delta_6$	Δ_7	Δ_8	$.\Delta_5$	0	0	Δ_6	$.\Delta_5$
5	$.\Delta_5$	Δ_6	Δ_7	$.\Delta_1$	0	Δ_5	Δ_6	$.\Delta_1$	0	0	$.\Delta_9$	$.\Delta_6$	0	Δ_6	Δ_5	0
6	0	Δ_8	$.\Delta_9$	$.\Delta_5$	$.\Delta_5$	$.\Delta_6$	Δ_1	0	0	Δ_9	Δ_6	0	$.\Delta_6$	$.\Delta_5$	0	0
7	$.\Delta_8$	Δ_9	Δ_5	0	$.\Delta_7$	Δ_8	Δ_9	$.\Delta_5$	$.\Delta_9$	Δ_6	0	0	$.\Delta_7$	Δ_8	Δ_9	$.\Delta_6$
8	0	0	0	$.\Delta_8$	0	0	.0	$.\Delta_7$	0	0	0	$.\Delta_9$	0	0	.0	$.\Delta_7$
9	0	0	Δ_8	0	0	0	Δ_7	0	0	0	Δ_9	0	0	0	Δ_7	0
10	0	Δ_8	0	0	0	Δ_7	0	0	0	Δ_9	0	0	0	Δ_7	0	0
11	Δ_8	0	0	0	Δ_7	0	0	0	Δ_9	0	0	0	Δ_7	0	0	0