Down the Rabbit Hole: Revisiting the Shrinking Method

Vivien Dubois

DGA Rennes, France
vivien.dubois@m4x.org

Abstract. The paper is about methodology to detect and demonstrate impossible differentials in a block cipher. We were inspired by the *shrinking* technique proposed by Biham *et al.* [2,3] in 1999 which recovered properties of scalable block cipher structures from numerical search on scaled down variants. Attempt to bind all concepts and techniques of impossible differentials together reveals a view of the search for impossible differentials that can benefit from the computational power of a computer. We demonstrate on generalized Feistel networks with internal permutations an additional clustering layer on top of shrinking which let us merge numerical data into relevant human-readable information to be used in an actual proof. After that, we show how initial analysis of scaled down TEA-like schemes leaks the relevant part of the design and the length and ends of the impossible differentials. We use that initial profiling to numerically discover 4 15-round impossible differentials (beating the current 13-round) and thousands of shorter ones.

Keywords: impossible differentials, block cipher, shrinking, generalized feistel networks with internal permutations, TEA.

1 Introduction

Consider a block cipher and assume its input and output space are both equipped with well-defined notions of difference between two elements. An impossible differential of this block cipher is a pair of input-output differences that never occur over queries (or inverse queries) to it. Impossible differentials started as a standalone cryptanalytic technique in the work of Biham [1,3] and Knudsen [10,9]. The exhibition of impossible differentials in a block cipher usually proceeds along the following lines. Particular input and output differences are first selected. The choice of these extremal differences is usually not explained, however they would be understood as being ones that are the least quickly destroyed by the operation of the block cipher (respectively in the direct and reciprocal querying direction). Each difference is then formally propagated through the block cipher up to some intermediate point. One then observes that e.g. for a certain number of rounds the extremal differences contradict at the intermediate point. This is called the miss-in-the-middle method of demonstrating an impossible differential. Most of the time, it was also the method of discovery of the impossible differential. When

that is the case, there is no real guarantee that the exhibited impossible differential is the longest or the only one, because the number of rounds was actually chosen to make it work.

Sometimes one may still be happy with that: the assumptions to derive the mathematical propagation model may seem fairly reasonable and one may be content that this should indeed gives the best results. However, ignoring the fact that such a model is usually not experimentally assessed at all, sometimes in an attack one does not just want one longest impossible differentials but a handful of them, were they slightly shorter. The current methodology favors mathematical simplicity first and working things by hand, so it restricts from the start the amount of discovery one might be able to derive from it.

There were attempts to overcome these limitations. Some work still defines a formal propagation system but let a computer find a contradiction [8,12,15]. This indeed increases the search space but the results are still limited by the assumption of a propagation model. Another approach actually dates back to 1999 and was used by Biham, Biryukov and Shamir in their attacks against Skipjack and Khufu [2,3]. They used scale models of these block ciphers to experimentally discover the impossible extremal patterns. They called this technique *Shrinking*. Although they discovered the impossible differential patterns experimentally, these patterns were formally demonstrated by using the miss-in-the-middle technique.

In this paper, we essentially raise the following question: In the Shrinking method, why just use the scale model to discover the extremal patterns and not also derive an impossible trail? (that is, actually, derive the propagation model of these extremal differences). While this may look like an ambituous goal, it is a fact that if one can actually exhaust on the scale model to find the extremal differences, one can find at the same time all the intermediate differences. But arises the technical problem that intermediate sets of differences are just to large to let the human eye capture their formal structures. In this paper, we propose one technique to ease off this problem. It is a simple value clustering technique that we call *Rectangle Partitioning*; it simply arranges a set of values having coordinates into *rectangles i.e.* sets that can be described as Cartesian products of sets of coordinate values. As we shall see this makes it far easier to grasp the structure of a set of values. The propagation of extremal differences into the intermediate states is just a sequence of sets that can all be arranged into sets of rectangles. Then, we can choose a splice point, where the concatenation of the forward trail and the backward trail has the least complexity in terms of rectangles. In cases when each intermediate set is actually one big rectangle, it is trivial to derive a formal miss-in-the-middle trail. So we let the shrinking technique go all the way down to finding the most obvious impossible trail for us.

We describe two applications of Shrinking and Rectangle Partitioning:

- We first return to the original target of the Shrinking technique: generalized Feistel networks with internal permutations. It served as a test bench to the rectangle partitioning technique and we demonstrate it on this case to compute automatically impossible trails. (the longest impossible differentials were known already since Biham *et al.*)

– After that, we apply the shrinking technique to a block cipher that is not strictly scalable: TEA. Of course, it is easy to let the particular parameters vary to obtain a generalized family of TEA block ciphers, members of which are scalable. We use such variants at varying scales to learn the connection between parameters and properties of the design such as the length of the longest impossible differentials and non-trivial supersets of the associated extremal differences and their propagations. Computing short description trails of small scale variants let us observe that the optimal splice point overlaps with trails that could obtained by miss-in-the-middle, with a propagation model that is suggested by the trails computed at small scale. This provides us with the ingredient to settle a miss-in-the-middle search successful up to finding the longest impossible differentials. Indeed we find 4 15-round impossible differentials of TEA (beating the current 13-round impossible differentials of [5]). The same search also provide thousands of impossible differentials over 14 or 13 rounds (but obviously more may exist by relaxing the extremal supersets which were profiled for the longest impossible differentials). As always, for each of them, we can use the rectangle partitioning technique to let us see the "structure" of the intermediate sets of differences, sometimes trivially overlapping with an immediately verifiable trail.

The paper is organized as follows. The first half of the paper is a theoretical discussion about impossible differentials. The reader mostly interested in the results should just skip it and jump directly to section 5. Since the results have just been reviewed, we just pay a moment to describe the structure of the discusssion part. The goal of Section 2 is to revisit the usual concepts on a fresh basis. The goal of Section 3 is to move the discussion about the search for impossible differentials to a concrete *circuit*-oriented model of practicality; we also put the usual search methodology in this view. Section 4 presents an alternate search paradigm which includes the Shrinking approach. Section 5 and Section 6 presents the results. Section 7 integrates the results to the theoretical discussion.

2 The Idea of Impossible Differentials

Differential Cryptanalysis. Let $\mathcal{C} = \{E_k,\ k \in \mathcal{K}\}$ denote a block cipher, that is, a family of permutations indexed by a parameter $k \in \mathcal{K}$. The value of k will not be of importance and one will just address a generic element of $\mathcal{C}$ simply by E. Given oracle-access to a permutation E, differential cryptanalysis considers pairs of parallel evaluations $x \mapsto E(x)$ and $x' \mapsto E(x')$ for random x, x' prepared under some prescribed binary (symmetric) relation $\mathcal{R}_{in}$ (a symmetric subset of pairs, for instance defined by a fixed difference). A particular relation $\mathcal{R}_{in}$ is chosen that would let an instance of the cipher exhibit non random behaviour. That would be captured by another binary relation $\mathcal{R}_{out}$ that holds on $(E(x), E(x'))$ with some probability p for random $(x, x') \in \mathcal{R}_{in}$ and $E \in \mathcal{C}$. A particular behaviour is observed on $\mathcal{C}$ under $\mathcal{R}_{in}$ when p deviates noticeably from the probability that random y, y' satisfy $\mathcal{R}_{out}$. At this point arises a nice dichotomy.

1. Either $(\mathcal{C}, \mathcal{R}_{in}, \mathcal{R}_{out})$ is a deterministic event, that is, $p = 0$ or 1.
2. Or $(\mathcal{C}, \mathcal{R}_{in}, \mathcal{R}_{out})$ is probabilistically biased, that is, p is either high or low.

Up to changing $\mathcal{R}_{out}$ into its complement, the alternatives within each case are just equivalent to each other. Now let us introduce the following notation. We denote by $\mathcal{C}[\mathcal{R}_{in}]$ the set of pairs $(E(x), E(x'))$ over all $(x, x') \in \mathcal{R}_{in}$ and $E \in \mathcal{C}$.

$$\mathcal{C}[\mathcal{R}_{in}] = \{(E(x), E(x')),\ (x, x') \in \mathcal{R}_{in},\ E \in \mathcal{C}\}.$$

Then, the first case purely corresponds to a set-theoretic relationship: either $\mathcal{R}_{out}$ contains $\mathcal{C}[\mathcal{R}_{in}]$ ($p = 1$) or it belongs to the complement ($p = 0$). The second case addresses distribution biases within $\mathcal{C}[\mathcal{R}_{in}]$. Therefore, the two cases deal with somehow orthogonal aspects of the block cipher. Case 1 purely deals with the matter of **coverage**. Case 2 purely deals with *unevenness* within whatever is covered. These two cases therefore correspond to two distinct cryptanalytic approaches, respectively termed *impossible* and *statistical* differential cryptanalysis. In this paper, we are concerned with issues of impossible differential cryptanalysis. Our concern is on methodology to detect and demonstrate incomplete coverage in a block cipher.

A Feeling of Impossible Differentials. So let $\mathcal{C}$ be a block cipher. Our goal is to find $\mathcal{R}_{in}$ such that $\mathcal{C}[\mathcal{R}_{in}]$ is incomplete. Consider the map

$$D :\ \mathcal{R}_{in} \longmapsto \#\{(y, y')\} - \#\mathcal{C}[\mathcal{R}_{in}]$$

that to any relation $\mathcal{R}_{in}$ maps the cardinality of the coverage defect. We are interested in those $\mathcal{R}_{in}$'s such that $D(\mathcal{R}_{in}) > 0$. We are especially interested in those $\mathcal{R}_{in}$'s such that $D(\mathcal{R}_{in})$ is maximal. With insight in the cipher's operation, we may have a strong suspicion on which those $\mathcal{R}_{in}$'s could be. These would minimize mixing with x-dependent or E-dependent information accross the cipher. Indeed the more transitions are dependent on x or E the more the information of $\mathcal{R}_{in}$ is dissipated and irrelevant, and the more pairs can actually be reached. This however does not say whether these $\mathcal{R}_{in}$'s do indeed induce a coverage defect after the whole operation of the cipher. That is, even with the right guess of $\mathcal{R}_{in}$, we still need ways to demonstrate incomplete coverage.

A Reflection. Observe that, because $\mathcal{C}$ is a family of permutations, we could just as well work from the other end. We may then try $\mathcal{R}_{out}$'s out on $\mathcal{C}^{-1} = \{E^{-1},\ E \in \mathcal{C}\}$, although alone it does not seem to bring any particular avail. Now let us assume that some $\mathcal{R}_{in}$ indeed induces a coverage defect on $\mathcal{C}$. For any $\mathcal{R}_{out}$ in the complement of this (unknown) coverage, *i.e.* $\mathcal{C}[\mathcal{R}_{in}] \cap \mathcal{R}_{out} = \emptyset$, we also of course have $\mathcal{R}_{in} \cap \mathcal{C}^{-1}[\mathcal{R}_{out}] = \emptyset$. That is, existence of a coverage defect for $\mathcal{C}$ implies existence of a coverage defect for $\mathcal{C}^{-1}$. This is nice consistency. Furthermore, when we expect only the most liable $\mathcal{R}_{in}$'s to actually expose a coverage defect, we similarly only expect the most liable $\mathcal{R}_{out}$'s from the other end, which therefore, would be their actual counterparts. This heuristic gives a

hint of what the ultimate incomplete coverages can be. By the way, we may now state the definition: $(\mathcal{C}, \mathcal{R}_{in}, \mathcal{R}_{out})$ is said **impossible** when the two relations are mutually exclusive through the cipher, that is, $\mathcal{C}[\mathcal{R}_{in}] \cap \mathcal{R}_{out} = \emptyset$. Of course, the ability to compute $\mathcal{C}[\mathcal{R}_{in}]$ gives all the impossible counterparts of $\mathcal{R}_{in}$ at once.

Half the Way. Assume the block cipher has two independent halves, that is $\mathcal{C} = \mathcal{H}' \circ \mathcal{H}$. In that case, we may first more easily compute $\mathcal{C}[\mathcal{R}_{in}]$ by first computing $\mathcal{H}[\mathcal{R}_{in}]$, then decomposing $\mathcal{H}[\mathcal{R}_{in}]$ into several components $\mathcal{R}^0_{mid} \cup \ldots$ where the computation of each $\mathcal{H}'[\mathcal{R}^i_{mid}]$ is easy, and finally get $\mathcal{C}[\mathcal{R}_{in}] = \cup_i \mathcal{H}'[\mathcal{R}^i_{mid}]$. This would be one approach.

A second approach would, as before, assume $\mathcal{R}_{out}$ and try to demonstrate that $(\mathcal{C}, \mathcal{R}_{in}, \mathcal{R}_{out})$ is impossible. From $\mathcal{C}[\mathcal{R}_{in}] \cap \mathcal{R}_{out} = \emptyset$,

$$\mathcal{H}[\mathcal{R}_{in}] \cap \mathcal{H}'^{-1}[\mathcal{R}_{out}] = \emptyset.$$

Therefore, if we can do the computation of $\mathcal{H}[\mathcal{R}_{in}]$ and $\mathcal{H}'^{-1}[\mathcal{R}_{out}]$ and it turns out these sets are disjoint, we get our demonstration. This is the root of the *miss-in-the-middle* strategy [9,3]: if we have target $\mathcal{R}_{in}$ and $\mathcal{R}_{out}$, we need the ability to compute the halfway coverages only for these two. On the other hand, we then never get the full $\mathcal{C}[\mathcal{R}_{in}]$.

An Example. We may illustrate the above two strategies on the simple example of 5-round Feistel networks with (independent) permutations (and final swap). One round makes $(L, R) \mapsto (R, L \oplus \pi(R))$ where π is a permutation. We split the cipher into the 3 first ($\mathcal{H}$) and the 2 last ($\mathcal{H}'$) rounds. Now we choose $\mathcal{R}_{in}$ to be the line $\mathcal{L}(a, 0) = (u, u \oplus (a, 0))$ for some fixed $a \neq 0$, because it propagates one round independently of $x = (L, R)$ and π to $\mathcal{L}(0, a)$ and looks promising as such. In particular, that shows that $\mathcal{L}(0, a)$ is a good candidate for $\mathcal{R}_{out}$ as it has the same property in the backward direction. Below, we shall simply denote $\pi(a)$ for short of $\pi(R \oplus a) \oplus \pi(R)$ since it will bear no ambiguity, and (L, R) will be considered fixed all the time. The computation of $\mathcal{H}[\mathcal{R}_{in}]$ follows the transitions

$$(a, 0) \to (0, a) \to (a, b = \pi_2(a)) \to (b, c = a \oplus \pi_3(b)).$$

There, b is a random non zero value when π_2 is a random permutation, and c is a random value distinct from a when π_3 is random. Therefore

$$\mathcal{H}[\mathcal{R}_{in}] = \cup_{b \neq 0, c \neq a} \mathcal{L}(b, c). \tag{1}$$

If we pursue the computation of $\mathcal{C}[\mathcal{R}_{in}]$, we may split $\mathcal{H}[\mathcal{R}_{in}]$ into

$$\mathcal{R}^0_{mid} = \cup_{b \neq 0} \mathcal{L}(b, 0) \quad \text{and} \quad \mathcal{R}^1_{mid} = \cup_{b \neq 0, c \neq \{0, a\}} \mathcal{L}(b, c).$$

From $(b, 0) \to (0, b) \to (b, d = \pi_5(b))$, we get

$$\mathcal{H}'[\mathcal{R}^0_{mid}] = \cup_{b \neq 0, d \neq 0} \mathcal{L}(b, d).$$

Call it $\mathcal{A}$ for short. On the other hand, the computation of $\mathcal{H}'[\mathcal{R}^1_{mid}]$ gives $(b, c) \to (c, e = b \oplus \pi_4(c))$ after one round, where $c \notin \{0, a\}$ and a random $e \neq b$. If e is non-zero, after one more round, we get $(e, c \oplus \pi_5(e))$. That intersects $\mathcal{A}$ but for $(e, 0)$. If e is zero, we get $(0, c)$, and that intersects $\mathcal{A}$ but for $(0, \neq 0, a)$. Finally

$$\mathcal{C}[\mathcal{R}_{in}] = \cup_{b\neq 0, d\neq 0}\mathcal{L}(b, d) \bigcup \cup_{e\neq 0}\mathcal{L}(e, 0) \bigcup \cup_{c\neq 0, a}\mathcal{L}(0, c).$$

The complement is indeed $\mathcal{L}(0, a)$ (and the 0-line), as we had suspected.

If, on the other hand, we only care to show that $\mathcal{R}_{out} = \mathcal{L}(0, a)$ is indeed impossible, we can compute its image by $\mathcal{H}'^{-1}$. Propagating backward $(0, a) \to (a, 0) \to (b' = \pi_4(a), a)$, we find

$$\mathcal{H}'^{-1}[\mathcal{R}_{out}] = \cup_{b'\neq 0}\mathcal{L}(b', a).$$

By confronting with (1), we indeed see that the two sets are disjoint. That was incredibly less painful. On the other hand, in the first computation, we do get the proof that $\mathcal{R}_{out} = \mathcal{L}(0, a)$ is the only maximal impossible relation.

Also note that the choice of the splitting is rather essential in the case of the miss-in-the-middle strategy: $4\,|1$ instead of $3\,|2$ would not have been as easy.

How Large is the Key. The case of dependent halves would be, as usual, troublesome. We could still of course have the condition $E[\mathcal{R}_{in}] \cap E'^{-1}[\mathcal{R}_{out}] = \emptyset$, for all $(E' \circ E) \in \mathcal{C}$, but this would hardly be useful in general save for very simplistic dependency patterns. Nothing general can be said here, except of course that if one denotes $\mathcal{H} = \{E,\, E' \circ E \in \mathcal{C}\}$ and $\mathcal{H}' = \{E',\, E' \circ E \in \mathcal{C}\}$ then $\mathcal{C} \subseteq \mathcal{H}' \circ \mathcal{H}$ and therefore, $\mathcal{C}$ always at least has the impossible relations of $\mathcal{H}' \circ \mathcal{H}$. In the sequel, we always assume independent layers whenever we encounter composition.

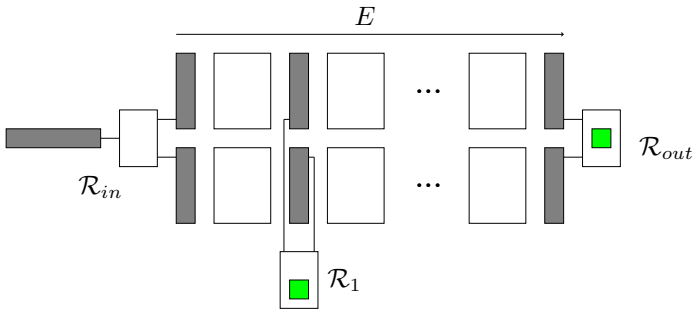
3 Impossible Differentials in the Real World

The implementation model that we describe below is intended as a mental image. It lets us have a generic view of a block cipher as well as a useful platform to relate the different approaches, with their practical significance in mind.

Down to the Mechanics. We shall use a circuit-oriented view of a block cipher. A block cipher firstly has memory slots for data, among which input slots and output slots. Then, it has a network of operations connected with such data slots. Among these operations, some are key-dependent, that is, they have an internal configuration which is loaded prior to any execution. Once loaded with input data, a scheduler lets the data flow through the operations until the output slot is reached. Since the global operation is a permutation, there is an inverse scheduler that from an output value computes back to the original input.

On the other hand, relations must also, indeed, be mechanical. At some point, we need that checking against a relation be implementable by a practical circuit.

Therefore, we are somewhat limited in our capacity of accounting for coverages. A different constraint bears on $\mathcal{R}_{in}$, from which we pick up pairs. We say that $\mathcal{R}_{in}$ is random-accessible when it admits a circuit that taking an index on its input slot outputs a member pair. The whole picture is shown below, where we have also let intermediate slots satisfy some relations. Intermediate slots are indeed well-ordered by the scheduler.



We call the sequence $\mathcal{R}_{in}, \mathcal{R}_1, \ldots, \mathcal{R}_r$ a **propagation trail** of $\mathcal{R}_{in}$ through $\mathcal{C}$ if any execution with any configuration E of $\mathcal{C}$ lets $\mathcal{R}_1, \ldots, \mathcal{R}_r$ output true. When it is so, each $\mathcal{R}_i$ is a superset of the actual coverage after the first i layers.

Propagation trails are usually found by hand, using some understanding of the generic behaviour of the sequential layers, proven as a mathematical property.

A TEA Sample. TEA is Feistel network over twice 32-bit words. It uses addition mod 2^{32} denoted $\boxplus$ as the halves-combiner. The internal function is

$$F(R) = ((R \ll 4) \boxplus C_{high}) \oplus (R \boxplus C_{mid}) \oplus ((R \gg 5) \boxplus C_{low})$$

where $\ll$ (resp. $\gg$) shifts the operand's bits to higher (resp. lower) positions, and the $C_.$'s are round-dependent constants including key material. TEA iterates 64 such rounds [14]. Some impossible differentials of TEA can be found in [13,5]. In [5] the following property of the TEA round function is shown. Let $D[n]$ denote the 32-bit relation where the n-th bits are different and all bits below are equal. Then it is observed that the relation $\mathcal{R} = (0, D[n])$ propagates after one round to $\mathcal{R}' = (D[n], D[n-5])$. Thanks to this property, it can be seen that TEA has a 7-round trail from $(D[26], 0)$ to $(D[1], *)$ (where $*$ means "unspecified"). However it is easy to see that this trail does not account for the coverage very well. The simple proof is that there is also a similar trail in the backward direction, which after 6 rounds contradicts with the first. Therefore while our trail covers everything after 8 rounds, the coverage is in fact incomplete at least 6 rounds further. The miss-in-the-middle strategy demonstrates us that fact.

This situation is quite common in the exhibition of impossible differentials as found in the literature. The exhibited trails are stuck after some early point. However they might still be successful, if they can be combined by miss-in-the-middle. Here miss-in-the-middle is used as a tool to mitigate our limited power

in tracking coverages. Of course the question remains as to whether the coverage defect that is successfully exhibited is indeed the longest one.

An Extra Level of Mechanics. Sometimes there are not one but many variants of $\mathcal{R}_{in}$ that we may want to check. This is implemented by $\mathcal{R}_{in}$ having an auxiliary configuration input slot. At the same time, $\mathcal{R}_1, \ldots, \mathcal{R}_r$ are also in several variants, and we need to bind their configuration data to the configuration data of $\mathcal{R}_{in}$. To keep it simple, assume $\mathcal{C}$ iterates a single layer. In that case, one can let $\mathcal{R}_{in}$ define $\mathcal{R}_1$, then $\mathcal{R}_1$ define $\mathcal{R}_2$, and so on, by uniformly using some functional relation $\mathfrak{R}$ that maps the configuration data of one to that of the next. Again the soundness of $\mathfrak{R}$ is guaranteed by a mathematical property.

This is the approach taken in [8,12,15]. It provides indeed some automation and made it possible to recover impossible differentials found by hand and more. However it keeps much in common with the previous manual method. It strictly works in a mathematical model of the block cipher, from which it can only retrieve some "corollary" trail. In fact, since all configurations of $\mathcal{R}_{in}$ are supposed to be tried, it is equivalent to an engine building all the $\#\{\mathcal{R}_{in}\text{'s}\}$ circuits.

4 Working Out Through Approximation

The Core Idea. There would be yet another approach. To simplify, let us focus on the actual coverage after the i-th layer. Call it $\mathcal{R}_i^*$. The intuition behind the previous approaches is that $\mathcal{R}_i^*$ is a structured set. Therefore, despite the fact that it contains an overwhelming number of pairs (indexed over $\mathcal{R}_{in} \times \mathcal{C}$), only a few would suffice to recover it, should its structure be simple enough. It would therefore all boil down to identifying its basis of expression correctly.

Approaching the Coverage. To work out this idea, let us start with trying some approximation of $\mathcal{R}_i^*$. We split the space of symmetric pairs into some partition $(\mathcal{S}_1, \ldots, \mathcal{S}_\ell)$. Of course, membership to each element of the partition is implemented by a circuit. They form of collection of "sensor" relations, consuming ℓ in area. These sensors are special in that their output bit can only change once after setup. We call the collection of these output bits the state bits. Now if we want to locate $\mathcal{R}_i^*$ over this partition, we just put all state bits to zero. Then, we make several executions of the block cipher with random $x, x' \in \mathcal{R}_{in}$ and $E \in \mathcal{C}$. At each execution, the output pair may change some state bits. After some estimated number of trials, we expect the hit classes to be an actual superset of $\mathcal{R}_i^*$. (It will be for later to be done, to make the proof that the upper bound is correct for any $x, x' \in \mathcal{R}_{in}$ and $E \in \mathcal{C}$.)

It is interesting to compare this method with the previous one. The ℓ state bits are the same as an ℓ-bit configuration data for $\mathcal{R}_i$. If $\mathcal{R}_{in}$ also has configuration data, then previously $\mathfrak{R}$ would fill the ℓ-bit configuration data of $\mathcal{R}_i$, while here $\mathcal{R}_i$ is built by some preliminary phase. However in both cases, the real coverage $\mathcal{R}_i^*$ is only approximated with ℓ bits of information. Therefore the previous

approach appears as a subcase of the current approach, where the configuration data of $\mathcal{R}_i$ is hard-coded rather than soft-coded by preliminary sampling.

This preliminary sampling does not come for free though. If all classes in the partition are of equal volume, then a little more than ℓ trials would be enough to ensure that each class has been visited if it had to. However if one class is very small, then we must wait until that class had a chance to be hit. Therefore the sampling time is about $1/p_{min}$ where p_{min} is the smallest class's probability.

Remains to us the choosing of an representation basis. As before, a rough approximation could let the miss-in-the-middle expose a contradiction for a certain number of rounds. However now we wish for the most accurate. In general, the operations of the block cipher suggest a natural family of relations from which $\mathcal{R}_{in}$ is actually chosen. That family would be our primary candidate as an representation basis. In general, $\mathcal{R}_{out}$ would lie in this family too.

In fact, assume that we even have a target $\mathcal{R}_{out}$ and just want to detect it as being avoided after multiple executions. Let $\mathcal{S}$ be the partition's component containing $\mathcal{R}_{out}$. If $\mathcal{S}$ is very large against $\mathcal{R}_{out}$ then the probability that one pair falls into it is very large against the probability that this pair belongs to $\mathcal{R}_{out}$. Therefore, $\mathcal{S}$ would be hit much earlier than $\mathcal{R}_{out}$ would have a chance to. In this case, $\mathcal{R}_{out}$ has no chance to be detected as being unhit. For this to happen, $\mathcal{R}_{out}$ has to be an element of the partition or very close to one.

Unfortunately this is where the limit of the approach lies. If $\mathcal{R}_{out}$ is the last expected coverage defect, it would cover a small fraction of all pairs. It being a part of the partition would make the sampling phase take forever. There is no working around of this problem. On the other hand, this is no surprise: while this method can let us automatically recover an approximation of the coverage, there is indeed a limit in its precision.

Learn from the Simple. The hope of course is in the ability to build relevant simplified models of a block cipher. These would be in the scope of an accurate approximation. If the models are relevant, results obtained from them may give information about the real block cipher such as what the structure of the coverage looks like or what is the expected length of the longest impossible differential.

Such methods have already been exploited. In 1999, Biham, Biryukov and Shamir showed an impossible differential for 24 rounds of Skipjack [2]. This impossible differential was used to attack 31 out of the 32 rounds of Skipjack. It is in fact an impossible differential of the global structure of Skipjack, which is a type of generalized Feistel Network. This global structure defines a more general block cipher family and its impossible differentials therefore port to the underlying block cipher. The impossible differential was demonstrated by a miss-in-the-middle trail. However it was checked that there were no longer impossible differential of the global structure (call it $\mathcal{C}$) by using instances of $\mathcal{C}$ where the non-linear permutations are replaced with 3-bit random sboxes. This could be done because the syntactic definition of $\mathcal{C}$ (just as 5-round Feistel in our previous example) is independent on the word size. Therefore any word size could be used to analyze it. Biham, Biryukov and Shamir called this technique *shrinking* [2].

They also used it in [3] to find impossible differentials of Khufu (which would have been a much harder case to be treated by hand).

In the remainder of this paper, we develop two extensions of this methodology. We first show how the numerical output of the shrinking technique can be turned into macroscopical information. This is achieved by clustering and is demonstrated on Feistel (and generalized Feistel) networks with permutations. The second extension applies to the TEA block cipher. It first analyses scaled down approximations and then use this information to aim at particular points of the coverage where again complete accuracy is practical.

5 The Shrinkable Cipher

Our first case will be an easy one: we shall quickly revisit our previous example, the classical Feistel network with permutations. Generalized Feistel networks work just the same and the results are given at the end of the section.

As already noticed the syntactic definition of the classical Feistel with permutations is independent on the word size. This means that it is not one block cipher but a family of block ciphers indexed by the word size. This property provides scaled down models natively: one just pick the block cipher defined for the smallest word size. When using permutative components, the smallest word size is at least 3 bits because a w-bit permutation has degree at most $w-1$ and any 2-bit permutation would just be affine. Consider, therefore, the r-round Feistel block cipher over 3-bit words. For any value of r we shall compute coverages over the basis of lines $\mathcal{L}(a,b) = \{(u, u \oplus (a,b))\}$. The lines form a partition with 2^6 components. Each one is random-accessible by $u \mapsto (u, u \oplus (a,b))$ and checkable by $(u,v) \mapsto u \oplus v =?(a,b)$. The expected sampling time is 64 executions.

5.1 Basic Shrinking

To warm up, we look for the longest impossible differentials (with respect to the basis of lines). For that, we try each of the 63 non-zero lines on the input and see if there are unhit (non-zero) lines on the output. To avoid changing the key each time, we have used the same 64 random instances of the block cipher for each input line. For such tiny parameters, it is of course an instantaneous matter to get the impossible differentials for any number of rounds of interest.

For 6 rounds, we find no input line yielding an unhit output line. In this case, the particular instances of the block cipher that were picked could serve as a certificate for this property. For 5 rounds, we find the impossible differentials

```
001 000 <>  000 001
010 000 <>  000 010
011 000 <>  000 011
100 000 <>  000 100
101 000 <>  000 101
110 000 <>  000 110
111 000 <>  000 111
```

Here their common shape $(a,0)(0,a)$ where $a \neq 0$ can be recognized at a glance.

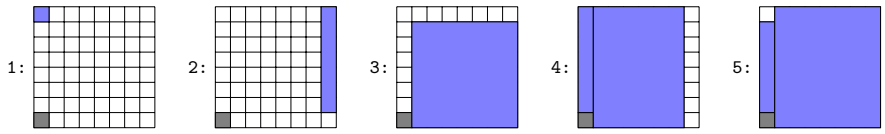
Now, for instance starting from $\mathcal{L}(111, 000)$, we want to compute the coverage after the i-th round, for $i = 1, \ldots, 5$. This gives

```
1: 000 111
2: 111 001, 111 010, 111 011, 111 100, 111 101, 111 110, 111 111
```

whose patterns are still easily recognizable. But after round 3, we get 49 elements. So we are not out of the woods yet. We need a *clustering* facility.

As a matter of fact, since this very simple case has only two dimensions, it could be further worked out at a glance. See Figure 1 below.

Fig. 1. Graphical representation of the coverages 1 to 5 starting from the difference `111 000` at 0 (the difference `000 000` is in gray since it can never be reached).



5.2 Rectangle Partitioning to the Rescue

More generally, let the parameterization space of our basis have ℓ dimensions. Furthermore, each coordinate can take values within a set, which for convenience we assume to be the same for each coordinate. Call it W. The representation of a coverage is a set of elements in W^ℓ. We call a **rectangle** a subset of elements of W^ℓ of the shape $C_1 \times \ldots \times C_\ell$ where C_i describes values in the i-th coordinate. Each C_i is termed a side of the rectangle. The Rectangle Partitioning procedure takes a set of elements of W^ℓ and outputs a set of rectangles over these elements. Here is how it works.

Let S be the input set. Each element indeed makes for a rectangle by itself. So we can trivially convert S to an initial set of rectangles.

Now let i be one of the ℓ coordinates. For any rectangle R, let $H_i(R)$ be the projection of R to all but the i-th coordinate (this is Cartesian product of the rectangle's sides save the i-th one). We may just group rectangles with the same value through H_i. But since these rectangles coincide at all but one coordinate, they can actually be **joined**, that is, their union as sets forms a valid new rectangle where the i-th side is just the union of the i-th sides of all rectangles in the group.

For some chosen order of the ℓ coordinates, we just repeat the above hash-join procedure. The algorithm is clearly linear (up to maybe a logarithmic factor) in the size of the input set.

It can be seen, even on the graphical example above, that the formed rectangles depend on the order we choose for the successive coordinates. Unless we have a criterium to select one order over another, we may choose one arbitrarily.

By using this algorithm the previous outputs rewrite

```
1: { {000}x{111}}
2: { {111}x{001,010,011,100,101,110,111} }
```

where the `x` symbol separates sides. We can improve this output by letting a coordinate-set be given from its complement when shorter, and by giving a specific symbol to the all-values set. And *voilà*!

```
1: { {000}x{111} }
2: { {111}xNOT{000} }
3: { NOT{000}xNOT{111} }
4: { {000}xNOT{000} NOT{000,111}xALL }
5: { {000}xNOT{000,111} NOT{000}xALL }
```

From such an output we indeed get the global picture which would let us make an actual proof. However, for that, we need to understand the global transitions, and this trail may not be the easiest to follow. Alternatively, we have access to the complement of the 5-th coverage, and we could work our way up. For any intermediate point, we have a trail from outside in, and some of them may be easier to follow. "Easy to follow" needs now to be defined. Several measures of complexity of a trail are possible. One is the accumulated number of rectangles, as it could be expected that one would understand the propagation rectangle by rectangle. It could also be the accumulated size of coverages. For any measure of complexity, we can compute a least complicated trail iteratively, by choosing the minimal increase round by round. There could be situations of equal increase. When that is, either one branch completes the trail and therefore all the others would just do the same. Or one branch does not complete the trail and parts of other branches contribute the close it, until perhaps they are all completely consumed. This may create a combinatorial number of equivalent choices around the completion point. In general, one would break the tie by using an enhanced measure of complexity or arbitrarily.

We have implemented the above two measures of complexity of a trail. Depending on the intermediate point that we choose we find the following values of the accumulated size of coverages and the accumulated number of rectangles.

Intermediate Point	0	1	2	3	4	5
Accumulated Size of Coverages	176	115	67	67	115	176
Accumulated Number of Rectangles	9	8	7	7	8	9

It turns out the two measures give equivalent results on this example. An easiest-to-follow trail is therefore the miss-in-the-middle trail (there are indeed two of them, by symmetry of the Feistel scheme)

```
0: { {111}x{000} }
1: { {000}x{111} }
2: { {111}xNOT{000} }
3: { NOT{000}xNOT{111} }
----
3: { NOT{000}x{111} }
4: { {111}x{000} }
5: { {000}x{111} }
```

Upon checking that the same trail holds for any other non-zero value that `111`, the trail is indeed valid for any non-zero a. This step of the automation has not been implemented.

Note that the miss-in-the-middle technique is here diverted from its classical use. Usually, it is used to search for a contradiction. Here, we know the contradiction exists at every intermediate round because the coverage is tracked with respect to the same basis all the way down to the end where impossible differences were found. Therefore, we search among these all contradictory miss-in-the-middle trails for ones that have the minimal complexity of description. This indeed is what would let us the most easily check the validity of the exhibited transitions and therefore prove the impossibility.

5.3 Generalized Feistel networks with Permutations

can be treated similarly provided the number of threads is not too large. Table 1 below gives the patterns of the longest impossible differentials as found experimentally. All these longest impossible differentials were already known in [2,3] or later in [12]. As before, we can output a best trail to account for each pair of impossible differentials.

Table 1. Last Impossible Differentials of GFNs with Permutations before complete coverage (obtained by sampling with 3-bit sboxes on a single PC in about 5 minutes).

GFN type	Threads	Rounds	Impossible Differentials
Type-1 (Gen-CAST256) [6]	4	19	$(a,0,0,0) \nrightarrow (0,0,0,a)$
Type-2 (Gen-CLEFIA) [6]	4	9	$(0,0,a,0) \nrightarrow (0,0,0,a)$ $(a,0,0,0) \nrightarrow (0,a,0,0)$
Gen-RC6 [12]	4	9	$(0,0,0,a) \nrightarrow (a,0,0,0)$ $(0,a,0,0) \nrightarrow (0,0,a,0)$
Gen-MARS [12]	4	11	$(a,0,0,0) \nrightarrow (0,0,0,a)$
Gen-FourCell [12]	4	18	$(0,0,0,a) \nrightarrow (0,0,b,b)$

5.4 Testing Design Variants

Assume we want to test the impact of shifting from independent random internal permutations to one random internal permutation input-xored by independent random keys. This is of course a practical setting and we want to see if something is changed from the impossible differential point of view.

The impact of this shifting on the impossible differentials of GFNs was addressed in [4]. They showed a 1-round improvement over the impossible differentials for the Gen-CAST256 and the Gen-MARS with 4 threads shown in [8,7]. But these impossible differentials were not the longest as known since Biham *et al.* and even with one extra round they remain shorter than the ones of Table 1.

Let us therefore shortly re-address the xor-key variant. We find experimentally that, for a randomly selected internal permutation, the xor-key variant does

not yield longer impossible differentials, neither on the classical Feistel nor the three GFN schemes from Table 1. At the maximal length, there are, however, a few more impossible differentials. The optimal splice of trail remains the same and would let us figure the structure of the impossible pairs.

Note that obtaining this kind of information comes almost at no cost. The algorithmic (and the code) remains exactly the same, only the definition of the cipher's round has to be changed. This is definitely superior to methods that build a model of the differential propagation [8,12,15] since such a model does not have to be built; the obtained results can't be challenged or improved since they are experimental; even a best trail can be produced to start off a proof by using the `RectanglePartitioning` technique described above.

6 The Shrunken Cipher

For our second case, we return to our second favourite example: TEA. Recall that TEA is a Feistel cipher (using the $\boxplus$ operator) with internal function

$$F(R) = ((R \ll l) \boxplus C_{high}) \oplus (R \boxplus C_{mid}) \oplus ((R \gg r) \boxplus C_{low})$$

where $\boxplus$ is integer addition, $\ll$ and $\gg$ are respectively upper/lower shifting, and the $C_.$'s are round-dependent constants including key material. In the actual TEA, the two halves are $w = 32$ bits, $l = 4$, $r = 5$, C_{high} and C_{low} are fixed chunks of the key and C_{mid} is a round counter derived from the golden ratio. Here we shall consider TEA-like schemes with modified size w, modified amounts of shifting l and r, and arbitrary values for the $C_.$'s at each round. Let `gen-TEA(w,l,r)` denote this family of variants. Also, whereas the actual TEA is prescribed to 64 rounds, we shall consider any number of rounds.

6.1 Cryptanalytic Dwarf Tossing: the Learning Phase

In this phase, we collect properties of the longest impossible differentials in function of the parameters.

Length of the longest. As a first step, we restrict our focus to the length of the longest impossible differentials (for the basis of all $\oplus$-lines). To this aim, we consider small variants in `gen-TEA(w,l,r)` and see how far they go. All $2^{2w} - 1$ possible input lines are considered. For now, we only record the length of the longest impossible differentials.

To warm up, we consider variants with $w = 5$ and all possible values of l and r from 0 to 4. We summarize the results in Table 2 below.

As one can see, the length of the longest impossible differentials does not depend on l. Furthermore, we reviewed the found patterns and observed that their numbers and shape do not depend on l either. Therefore we simply set this value to $r - 1$ as in TEA and do not consider it any further.

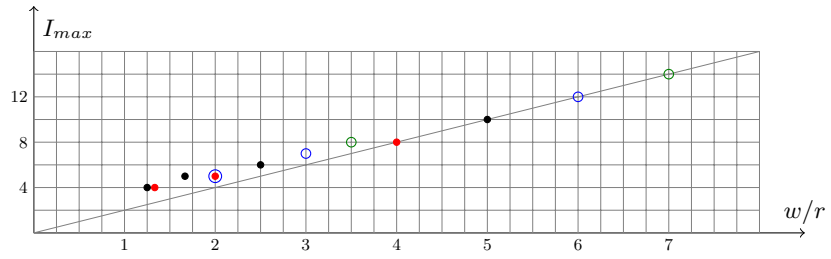
Table 2. Length of the Longest Impossible Differentials of `gen-TEA(w,l,r)` for $w = 5$.

$r \backslash l$	0	1	2	3	4
0	∞	∞	∞	∞	∞
1	10	10	10	10	10
2	6	6	6	6	6
3	5	5	5	5	5
4	4	4	4	4	4

Two side notes may be in order. First, for $r = 0$, we observed no coverage defect at any round. This can be understood by observing that for $r = 0$, F is linear in the most significant bit and and differences in the most significant bits of the two halves form of the stable subspace of the Feistel round. Also, for $r = 4$, we observed complete coverage as early as round 5, which shows that F is not a permutation as otherwise it would not have happened until round 6.

Dependency on the other parameters. Our second step is to investigate the relationship between r and w. For $w = 4$ to 7, we considered several values of r. For each of these variants, we considered the length I_{max} of the longest impossible differentials (detected after sampling over many random instances). In the course of the experiments, we observed seemingly monotonic behaviour of I_{max} in term of the ratio w/r. Figure 2 below makes this statement more specific. Each color is for a value of w. As one can see, I_{max} is roughly proportional to

Fig. 2. Lengths of the Longest Impossible Differentials of `gen-TEA(w,l=r-1,r)` for $w = 4, 5, 6, 7$ and r between 1 and $w - 1$.



w/r. Parameters with $r = 1$ are on the line, while the others are above it. The proportionality coefficient is roughly 2.

Following these estimates, the length of the longest impossible differentials of the actual TEA would be lower-bounded by $\lceil I_{max}(32/5) \rceil = \lceil 2 * 6.4 \rceil = 13$. Considering the global behaviour we do not expect it very far from that. Note that the impossible differential from [5] covers 13 rounds. So it might be optimal.

Supersets of the extremal differences. The impossible differentials for $(w, r) = (6, 1)$ and $(7, 1)$ happen to be in the same number. They respectively are $(\delta_{in}, \delta_{out}) =$

```
010000 100000 <> 000000 100000        0100000 1000000 <> 0000000 1000000
010000 100000 <> 100000 010000        0100000 1000000 <> 1000000 0100000
010000 100000 <> 100000 110000        0100000 1000000 <> 1000000 1100000
100000 000000 <> 000000 100000        1000000 0000000 <> 0000000 1000000
100000 000000 <> 100000 010000        1000000 0000000 <> 1000000 0100000
100000 000000 <> 100000 110000        1000000 0000000 <> 1000000 1100000
110000 100000 <> 000000 100000        1100000 1000000 <> 0000000 1000000
110000 100000 <> 100000 010000        1100000 1000000 <> 1000000 0100000
110000 100000 <> 100000 110000        1100000 1000000 <> 1000000 1100000
```

They obviously have very much in common. Similarly, the impossible differentials for $(5, 2)$ and $(7, 2)$ are in the same number 13 and have the same shape, which is bit different from the one above. What all these patterns have in common though is that they are only different in their most significant bits. For some fixed u and v, the input patterns lie in the set

$$(\{0,1\}^u\{0\}^{w-u}, \{0,1\}^v\{0\}^{w-v})$$

and similarly for the output patterns (with, due to the Feistel symmetry, swapped values of u and v). Furthermore one can observe that in the above listing $u = r+1$ and $v = 1$, and the same can be seen in the impossible patterns of the second family of parameters too.

$$\begin{array}{c}\text{Longest-length Impossible}\\ (\delta_{in}, \delta_{out})\end{array} \implies \begin{cases} \delta_{in} \in (\{0,1\}^{r+1}\{0\}^{w-r-1}, \{0,1\}^1\{0\}^{w-1}) \\ \delta_{out} \in (\{0,1\}^1\{0\}^{w-1}, \{0,1\}^{r+1}\{0\}^{w-r-1}) \end{cases}.$$

Tracking the coverage. For impossible pairs of extremal differences $(\delta_{in}, \delta_{out})$, we can compute shortest description trails with respect to their rectangle complexity (for some arbitrary order of the coordinates in the Rectangle Partitioning procedure). We use $(w, r) = (5, 2)$ as an illustrative example. For each impossible pair $(\delta_{in}, \delta_{out})$, we find a single shortest description trail. Here is one example (we have used a nicer display of rectangles here; also the two halves have been merged in one vector):

```
0: { (0,0,1,0,0,1,0,0,0,0) }
1: { (1,0,0,0,0,*,*,0,0,0) }
2: { (0,0,0,0,0,1,0,0,0,0) (*,1,0,0,0,*,*,*,1,0) (1,0,0,0,0,*,*,1,0,0) }
----
2: { (*,*,*,*,0,*,*,1,*,1) (*,*,*,*,1,*,*,0,*,1) }
3: { (*,*,*,*,1,*,*,1,0,0) }
4: { (*,*,1,0,0,1,0,0,0,0) }
5: { (1,0,0,0,0,0,0,0,0,0) }
6: { (0,0,0,0,0,1,0,0,0,0) }
```

Here is a second example:

```
0: { (1,0,0,0,0,0,0,0,0,0) }
1: { (0,0,0,0,0,1,0,0,0,0) }
2: { (1,0,0,0,0,*,*,1,0,0) }
3: { (*,*,1,0,0,*,*,*,*,1) }
----
3: { (*,*,*,*,1,*,*,1,0,0) }
4: { (*,*,1,0,0,1,0,0,0,0) }
5: { (1,0,0,0,0,0,0,0,0,0) }
6: { (0,0,0,0,0,1,0,0,0,0) }
```

While the second trail is simple enough to be detected by hand, the first is much more complicated as the contradictory coverages are composed of several rectangles. And yet, the two impossible differentials are of equal interest since they cover the same number of rounds.

By reviewing several examples like these, we observe the following

key property*: At the optimal splice point, the contradiction shows at the rightmost values of one or the other half. Therefore, the optimal splice point, chosen for shortness of description purposes, reveals that tracking down these rightmost values only (although this can only be done up to a certain intermediate point) is just enough to recover the same trail, should we rather use the miss-in-the-middle technique. This reveals us at the same time the outline of a propagation model and gives us the fact that the miss-in-the-middle approach, based on this propagation model, is able to unravel the longest impossible differential.*

Supersets of the intermediate differences. So let us go ahead and define this propagation model. By symmetry of the Feistel network, we can focus on the input difference. The right part of an intermediate difference, starting from the rightmost bits has deterministically some zeros, followed by some non-trivial value, followed by a great number (possibly all) values. More precisely, the right half of a difference after round i is lies in the set

$$R_i = \{*\}^{r.(i-1)} \times \{0,1\}^r \times \{0\}^{w-1-r.i}.$$

This superset is non-trivial until $r(i-1) < w$ of course. Obviously, since this is Feistel, the left part is equal to the right part of the previous round. At round i, we only record the values at the right half ignoring the $r(i-1)$ leftmost bits; this guarantees that (starting from the prescribed extremal differences) the number of values to record is bounded by 2^r, for any i. The propagation model in the backward direction is just the same up to swapping the two halves. For any R_i, we call its support the bits that are not ignored; bits that are ignored are just assumed to take any possible value.

At some intermediate round i where we seek for a contradiction (in, say, the right half) we have recorded a subset $\Delta_i^{\rightarrow}$ of the forward superset $R_i^{\rightarrow}$ and a subset $\Delta_{r-i}^{\leftarrow}$ of the backward superset $R_{r-i}^{\leftarrow}$, which are the respective possible intermediate differences. When there is a contradiction it lies in the projection of these values to the intersection of the supports of $R_i^{\rightarrow}$ and $R_{r-i}^{\leftarrow}$ (if these

supports do not partially overlap, no contradiction can be infered). So we just take the projections of $\Delta_i^{\rightarrow}$ and $\Delta_{r-i}^{\leftarrow}$ and see if they are disjoint.

6.2 A Mad TEA-Party

The previous experiments on small instances in the `Gen-TEA` family have permitted us to very well profile both the impossible differences and a coverage approximation to track the propagation of these differences until a point of contradiction. This is material to discover all the impossible differentials of an arbitrary instance in `Gen-TEA` family by using the miss-in-the-middle strategy. In particular all the impossible differentials and trails that exist for `gen-TEA(32,4,5)` apply to the actual TEA block cipher.

For `gen-TEA(32,4,5)`, the length of the longest impossible differential that we can derive from our propagation model is upper-bounded by 15. This is because 15 is the length of the longest combinations of $R_i^{\rightarrow}$ and $R_{r-i}^{\leftarrow}$ with overlapping support. But this would actually completely fulfill our hopes because if we go back to Figure 2 we see that we would hardly expect a length above 14.

After implementing the miss-in-the-middle search for the impossible differentials of `gen-TEA(32,4,5)`, we indeed find 4 15-round impossible differentials. It takes no more than 5 minutes on a single PC. By symmetry, these differentials are equivalently obtained from the splice 7/8 or the splice 8/7. They are

```
(010...,0...) <> (0...,100...)
(100...,0...) <> (0...,010...)
(100...,0...) <> (0...,110...)
(110...,0...) <> (0...,100...)
```

Let us review the first impossible differential. The rectangle partitioning outputs that after 7 rounds `(010...,0...)` has propagated into `(?...,?...*1)` where the `?` means a bit that is not traced by the propagation model. Similarly in the backward direction, after the 8 rounds `(0...,100...)` has propagated into `(?...,?...10)`. $R_7^{\rightarrow}$ and $R_8^{\leftarrow}$ have identical supports, the last two bits of the right halves, and here these last two bits cover `*1` in the forward direction and `10` in the backward direction, and hence are disjoint. It can be easily checked by hand that it is indeed so.

With the same extremal supersets (and the same propagation model), we also find 625 14-round impossible differentials, which are contributed by the splices 6/8, 7/7 and 8/6, and 8881 13-round impossible differentials, which are contributed by the splices 5/8, 6/7, 7/6 and 8/5.

7 Closing Discussion

Many of the impossible differential trails that have been developed in the literature use so-called truncated differentials [10,11]. With our terminology, this concept (in its usual application) is equivalent to a coordinate-oriented basis of representation of a coverage. In this statement a coordinate can be a bit as encountered in TEA or a word of a few bits, and each make for a small number of

basis elements. Working on a coordinate-oriented basis is always very practical and can often be handled by hand. As opposed to that, the shrinking technique considers all possible values of differences. We may call this a value-oriented basis of representation of the coverage. Here, all combinations of coordinates also make for basis elements. Because of the large cardinality of such a basis, this approach is only practical at small scale. On the other hand, the shrinking technique makes no assumption on the behaviour of the cryptosystem. Therefore, whenever a block cipher structure is at reach of the shrinking technique, it seems like an enormous waste not the use it. This is the case of generalized Feistel networks, as we have seen, where the only small technical problem was to turn the output of the shrinking technique into the relevant macroscopical information. However a complete block cipher design is not, in general, shrinkable. When facing such a case, the problem remains as to what a nice representation basis would be. In the case of TEA, we have been lucky, because the design is simple enough to yield approximate scaled down variants. Thanks to that, we could work out a relevant simplification of the block cipher. Also, the small scaled down variants revealed to us the relevant input patterns, which were not trivial. Even if the case of TEA seems simple enough to be treated by hand, as it was, we insisted on using a systematic search (at practical scale) and it paid off. Manual ways, on the other hand, always tend to oversimplify and therefore limit the results by making initial restrictions. As we have seen in Section 4, any way of representing a coverage implies a level of approximation. And defining a coverage basis that can be propagated without block cipher sampling means that the precision must be very low. We conclude this paper with the feeling that the shrinking technique is a great technique of systematic investigation. In this paper, we have only slightly extended its application. Extending its range to more complicated block cipher designs will be the challenge of future work.

Acknowledgement

I would like to thank one of the anonymous referees of FSE 2014 for his support.

References

1. Eli Biham. Cryptanalysis of ladder-des. In *Fast Software Encryption, LNCS 1267, Springer-Verlag*, pages 134–138. Springer-Verlag, 1997.
2. Eli Biham, Alex Biryukov, and Adi Shamir. Cryptanalysis of skipjack reduced to 31 rounds using impossible differentials. In *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques*, volume 1592 of *Lecture Notes in Computer Science*, pages 12–23. Springer, 1999.
3. Eli Biham, Alex Biryukov, and Adi Shamir. Miss in the middle attacks on idea and khufu. In Lars R. Knudsen, editor, *FSE*, volume 1636 of *Lecture Notes in Computer Science*, pages 124–138. Springer, 1999.

4. Charles Bouillaguet, Orr Dunkelman, Pierre-Alain Fouque, and Gaëtan Leurent. New insights on impossible differential cryptanalysis. In Ali Miri and Serge Vaudenay, editors, *Selected Areas in Cryptography*, volume 7118 of *Lecture Notes in Computer Science*, pages 243–259. Springer, 2011.
5. Jiazhe Chen, Meiqin Wang, and Bart Preneel. Impossible differential cryptanalysis of the lightweight block ciphers tea, xtea and hight. In Aikaterini Mitrokotsa and Serge Vaudenay, editors, *AFRICACRYPT*, volume 7374 of *Lecture Notes in Computer Science*, pages 117–137. Springer, 2012.
6. Viet Tung Hoang and Phillip Rogaway. On generalized feistel networks. In Tal Rabin, editor, *CRYPTO*, volume 6223 of *Lecture Notes in Computer Science*, pages 613–630. Springer, 2010.
7. Jongsung Kim, Seokhie Hong, and Jongin Lim. Impossible differential cryptanalysis using matrix method. *Discrete Mathematics*, 310(5):988–1002, 2010.
8. Jongsung Kim, Seokhie Hong, Jaechul Sung, Changhoon Lee, and Sangjin Lee. Impossible differential cryptanalysis for block cipher structures. In Thomas Johansson and Subhamoy Maitra, editors, *INDOCRYPT*, volume 2904 of *Lecture Notes in Computer Science*, pages 82–96. Springer, 2003.
9. Lars Knudsen. Deal - a 128-bit block cipher. In *NIST AES Proposal*, 1998.
10. Lars R. Knudsen. Truncated and higher order differentials. In *Fast Software Encryption - Second International Workshop, Leuven, Belgium, LNCS 1008*, pages 196–211. Springer-Verlag, 1995.
11. Lars R. Knudsen and Thomas A. Berson. Truncated differentials of safer. In Dieter Gollmann, editor, *FSE*, volume 1039 of *Lecture Notes in Computer Science*, pages 15–26. Springer, 1996.
12. Yiyuan Luo, Zhongming Wu, Xuejia Lai, and Guang Gong. A unified method for finding impossible differentials of block cipher structures. *IACR Cryptology ePrint Archive*, 2009:627, 2009.
13. Dukjae Moon, Kyungdeok Hwang, Wonil Lee, Sangjin Lee, and Jongin Lim. Impossible differential cryptanalysis of reduced round xtea and tea. In Joan Daemen and Vincent Rijmen, editors, *FSE*, volume 2365 of *Lecture Notes in Computer Science*, pages 49–60. Springer, 2002.
14. David J. Wheeler and Roger M. Needham. Tea, a tiny encryption algorithm. In Bart Preneel, editor, *FSE*, volume 1008 of *Lecture Notes in Computer Science*, pages 363–366. Springer, 1994.
15. Shengbao Wu and Mingsheng Wang. Automatic search of truncated impossible differentials for word-oriented block ciphers. In Steven D. Galbraith and Mridul Nandi, editors, *INDOCRYPT*, volume 7668 of *Lecture Notes in Computer Science*, pages 283–302. Springer, 2012.