# Collisions for 70-step SHA-1: On the Full Cost of Collision Search

Christophe De Cannière[1], Florian Mendel[2*], and Christian Rechberger[2]

[1] Katholieke Universiteit Leuven, Dept. ESAT/SCD-COSIC, Kasteelpark Arenberg 10, B–3001 Heverlee, Belgium

[2] Graz University of Technology Institute for Applied Information Processing and Communications Inffeldgasse 16a, A–8010 Graz, Austria

**Abstract.** The diversity of methods for fast collision search in SHA-1 and similar hash functions makes a comparison of them difficult. The literature is at times very vague on this issue, which makes comparison even harder. In situations where differences in estimates of attack complexity of a small factor might influence short-term recommendations of standardization bodies, uncertainties and ambiguities in the literature amounting to a similar order of magnitude are unhelpful. We survey different techniques and propose a simple but effective method to facilitate comparison. In a case study, we consider a newly developed attack on 70-step SHA-1, and give complexity estimates and performance measurements of this new and improved collision search method.

## 1 Introduction

Recently, claims for small improvements of collision search attacks attract the attention of the cryptographic community. Examples are a $2^3$-fold speed-up for collision search in SHA-1 reduced to 58 steps [14] and full SHA [10] (the predecessor of SHA-1). Apart from the interest in new techniques, reports on new improvements (especially in the case of SHA-1) might also influence short-term recommendations of standardization bodies.

Motivated by the growing importance of estimating the complexity of newly developed or improved collision search attacks on members of the SHA family, we point out a number of technical issues which are, if at all, only very vaguely addressed in the literature.

– Computational cost of message modification (and similar methods)
– Influence of early-stop technique
– Impact of the last conditions of both blocks in a 2-block attack

All these issues contribute to the total cost of a differential collision search. Once devised, these methods require very little memory, are trivially parallelizable with negligible communication cost. Note that this contrasts the situation

* This author is supported by the Austrian Science Fund (FWF), project P18138.

© Springer-Verlag Berlin Heidelberg 2007

in many other types of cryptanalytic attacks, where *e. g.* the need for memory access significantly contributes to the full cost of an attack [21].

As an example of our findings we show that a very new and promising speed-up method named Boomerang-method is less efficient than expected in an collision search. Additionally, we are describing the technical details of a fast collision search method for SHA-1 reduced to 70 steps and for the first time give an example of a colliding message pair.

## 2 Short description of SHA-1

SHA-1 is an iterative hash function that processes 512-bit input message blocks and produces a 160-bit hash value. Like all dedicated hash functions used today, it is based on the design principle of MD4, pioneered by Rivest. In the following we briefly describe the SHA-1 hash function. It basically consists of two parts: the message expansion and the state update transformation. A detailed description of the hash function is given in [11]. For the remainder of this article we follow the notation of [3] and restate it whenever needed.

**Table 1.** Notation

| notation | description |
|---|---|
| $X \oplus Y$ | bit-wise XOR of X and Y |
| $X + Y$ | addition of X and Y modulo $2^{32}$ |
| $X$ | arbitrary 32-bit word |
| $X^2$ | pair of words, shortcut for $(X, X^*)$ |
| $M_i$ | input message word $i$ (32 bits) |
| $W_i$ | expanded input message word $t$ (32 bits) |
| $X \lll n$ | bit-rotation of $X$ by $n$ positions to the left, $0 \leq n \leq 31$ |
| $X \ggg n$ | bit-rotation of $X$ by $n$ positions to the right, $0 \leq n \leq 31$ |
| $N$ | number of steps of the compression function |

### 2.1 Message expansion

The message expansion of SHA-1 is a linear expansion of the 16 message words (denoted by $M_i$) to 80 expanded message words $W_i$.

$$W_i = \begin{cases} M_i, & \text{for } \ 0 \leq i \leq 15, \\ (W_{i-3} \oplus W_{i-8} \oplus W_{i-14} \oplus W_{i-16}) \lll 1 & \text{for } 16 \leq i \leq 79\ . \end{cases} \quad (1)$$

### 2.2 State update transformation

The state update transformation of SHA-1 consists of 4 rounds of 20 steps each. In each step the expanded message word $W_i$ is used to update the 5 chaining variables $A_i, B_i, C_i, D_i, E_i$ as follows:

$$\begin{aligned}
A_{i+1} &= E_i + A_i \lll 5 + f(B_i, C_i, D_i) + K_j + W_i \\
B_{i+1} &= A_i \\
C_{i+1} &= B_i \ggg 2 \\
D_{i+1} &= C_i \\
E_{i+1} &= D_i
\end{aligned}$$

Note that the function $f$ depends on the actual round: round 1 (steps 0 to 19) use $f_{IF}$ and round 3 (steps 40 to 59) use $f_{MAJ}$. The function $f_{XOR}$ is applied in round 2 (steps 20 to 39) and round 4 (steps 60 to 79). The functions are defined as follows:

$$f_{IF}(B, C, D) = B \wedge C \oplus \overline{B} \wedge D \tag{2}$$

$$f_{MAJ}(B, C, D) = B \wedge C \oplus B \wedge D \oplus C \wedge D \tag{3}$$

$$f_{XOR}(B, C, D) = B \oplus C \oplus D \ . \tag{4}$$

Note that $B_i = A_{i-1}$, $C_i = A_{i-2} \ggg 2$, $D_i = A_{i-3} \ggg 2$, $E_i = A_{i-4} \ggg 2$. This also implies that the chaining inputs fill all $A_j$ for $-4 \leq j \leq 0$. Thus it suffices to consider the state variable $A$, which we will for the remainder of this paper.

After the last step of the state update transformation, the chaining variables $A_0, B_0, C_0, D_0, E_0$ and the output values of the last step $A_{80}, B_{80}, C_{80}, D_{80}, E_{80}$ are combined using word-wise modular addition, resulting in the final value of one iteration (feed forward). The result is the final hash value or the initial value for the next message block.

## 3 Collision search strategies

In order to construct efficient attacks, differentials with high probability are used. Since no secret key is involved, in addition to the message difference, also the actual values of bits in certain positions in the message influence the probability of such a differential. Exploitation of this additional degree of freedom led to remarkable progress in the cryptanalysis of hash function in recent years. For hash functions like SHA-1, most (complex) differentials through the earlier parts of the compression function can have for various reasons a very low probability. More recently, the impact of this fact was systematically studied in detail in [3]. It is shown that the degrees of freedom from the message largely neutralize the disadvantages of this low probability. This shifts the goal to optimizing the probability of a differential through the later part of the compression function. Additionally, this allows to remove all restrictions on the input differences of such high probability differentials. By using a second message block as an additional degree of freedom also all restrictions on the output differences of such a high probability differential can be removed.

Methods for searching high probability characteristics through (parts of) the compression function suitable for such an optimization were already discussed in [5,7,12,13,18]. In Section 5.1, we describe an improved variant of such an optimization we used for our case study of 70-step SHA-1.

**Optimality of 2-block approach.** It turns out that by removing the constraint to have a collision already after a single message block, differentials with significantly better probabilities can be found. On the other hand, more than two blocks do not give any additional exploitable degrees of freedom anymore. Hence, aiming for a differential spanning two message blocks is preferable, since the workloads to find the right message pairs for each block add up. Note that due to less effective methods, the first collision for SHA (the predecessor of SHA-1) was built using a differential spanning four message blocks [2].

## 4 Computational cost of differential collision search

By fixing a difference and having random trials we expect to have to try around $2^n$ times. With appropriate choices for differences and part of the actual messages, the aim is to reduce the work to find a colliding pair below the work of a birthday search of order $2^{n/2}$ trials.

We divide the involved computational costs into three categories.

– Determining a suitable message difference
– Determining a suitable characteristic
– Searching for a message pair that roughly follows this characteristic

For complexity estimates in the literature, usually only the last step is considered. We note that the first two steps used to have manual steps. With the possibility to fully automate also these parts (as shown in [3]) it becomes possible to also estimate this computational effort and consider trade-offs with other parts of an collision search attack.

### 4.1 General method to estimate work factor of a chosen characteristic

We briefly recall some methods and definitions given in [3] needed for the subsequent discussion.

**Generalized conditions and generalized characteristics.** Generalized conditions for hash functions were first defined in [3]. The generalized conditions on a particular pair of words $X^2$ will be denoted by $\nabla X$. $\nabla X$ represents as a set, containing the values for which the conditions are satisfied. In order to write this in a more compact way, we will use the notation listed in Table 2.

**Table 2.** Notation for generalized conditions, possible conditions on a pair of bits

| $(x_i, x_i{}^*)$ | $(0,0)$ | $(1,0)$ | $(0,1)$ | $(1,1)$ | $(x_i, x_i^*)$ | $(0,0)$ | $(1,0)$ | $(0,1)$ | $(1,1)$ |
|---|---|---|---|---|---|---|---|---|---|
| ? | ✓ | ✓ | ✓ | ✓ | 3 | ✓ | ✓ | - | - |
| - | ✓ | - | - | ✓ | 5 | ✓ | - | ✓ | - |
| x | - | ✓ | ✓ | - | 7 | ✓ | ✓ | ✓ | - |
| 0 | ✓ | - | - | - | A | - | ✓ | - | ✓ |
| u | - | ✓ | - | - | B | ✓ | ✓ | - | ✓ |
| n | - | - | ✓ | - | C | - | - | ✓ | ✓ |
| 1 | - | - | - | ✓ | D | ✓ | - | ✓ | ✓ |
| # | - | - | - | - | E | - | ✓ | ✓ | ✓ |

**Total work factor for generalized characteristic.** Let us assume that we have given a complete generalized characteristic for SHA-1, specified by $\nabla A_{-4}, \ldots, \nabla A_N$ and $\nabla W_0, \ldots, \nabla W_{N-1}$. Our goal is to estimate how much effort it would take to find a pair of messages which follows this characteristic, assuming a simple depth-first search algorithm which tries to determine the pairs of message words $M_i^2$ one by one starting from $M_0^2$.

In order to estimate the work factor of this algorithm, we will compute the expected number of visited nodes in the search tree. But first some more definitions, which are all needed to estimate the work factor.

**Definition 1.** *The* message freedom $F_W(i)$ *of a characteristic at step $i$ is the number of ways to choose $W_i^2$ without violating any (linear) condition imposed on the expanded message, given fixed values $W_j^2$ for $0 \leq j < i$.*

We note that since the expanded message in SHA-1 is completely determined by the first 16 words, we always have $F_W(i) = 1$ for $i \geq 16$.

**Definition 2.** *The* uncontrolled probability $P_u(i)$ *of a characteristic at step $i$ is the probability that the output $A_{i+1}^2$ of step $i$ follows the characteristic, given that all input pairs do as well, i.e.,*

$$P_u(i) = P\left(A_{i+1}^2 \in \nabla A_{i+1} \mid A_{i-j}^2 \in \nabla A_{i-j} \text{ for } 0 \leq j < 5, \text{ and } W_i^2 \in \nabla W_i\right) .$$

**Definition 3.** *The* controlled probability $P_c(i)$ *of a characteristic at step $i$ is the probability that there exists at least one pair of message words $W_i^2$ following the characteristic, such that the output $A_{i+1}^2$ of step $i$ follows the characteristic, given that all other input pairs do as well, i.e.,*

$$P_c(i) = P\left(\exists W_i^2 \in \nabla W_i : A_{i+1}^2 \in \nabla A_{i+1} \mid A_{i-j}^2 \in \nabla A_{i-j} \text{ for } 0 \leq j < 5\right) .$$

With the definitions above, we can now easily express the number of nodes $N_s(i)$ visited at each step of the compression function during the collision search. Taking into account that the average number of children of a node at step $i$ is $F_W(i) \cdot P_u(i)$, that only a fraction $P_c(i)$ of the nodes at step $i$ have any children at all, and that the search stops as soon as step $N$ is reached, we can derive the following recursive relation:

$$N_s(i) = \begin{cases} 1 & \text{if } i = N\,, \\ \max\left\{N_s(i+1) \cdot F_W(i)^{-1} \cdot P_u^{-1}(i),\, P_c^{-1}(i)\right\} & \text{if } i < N\,. \end{cases}$$

The total work factor is then given by

$$N_w = \sum_{i=1}^{N} N_s(i)\,.$$

Once a characteristic has been fixed, we have to find a message pair that follows the characteristic. By using a simple greedy approach or techniques such as message modification or neutral bits the probability of the characteristic after step 16 can be improved. In the following we will describe these techniques in more detail.

## 4.2 Corrective factors for speed-up methods

In order to include methods that speed-up collision search into the very useful general method to estimate the work factor of a chosen characteristic as described above, we introduce corrective factors. It is easy to see that if a method aims for higher speed, less steps need to be computed and hence less nodes in the search tree are visited.

We now briefly describe how corrective factors can be derived for the different methods that can be found in the literature. Because of its actuality, we chose the so-called Boomerang-method as an example for our model in Section 4.8. The adaption of our model to other methods works similarly.

Before that, we discuss how to incorporate also less probable characteristics and the impact of conditions at the end of a block into this general model by the use of corrective factors $C_{1\ldots n}(i)$ where $i < N$ and $n$ enumerates all considered corrective factors. The corrected probability for each step is hence

$$P_{corr}(i) = P_u(i) \prod C_n(i)\ .$$

The corrected number of nodes $N_{corr}$ and total work factor is then based on $P_{corr}$ instead of $P_u$.

## 4.3 Effect of additionally considering related characteristics

We give here two methods to consider additional characteristics that are related to the originally chosen main characteristic.

**Less probable characteristics.** Even if all message conditions for the main characteristic are already in place, there exist a number of less probable characteristics. For the case of high probability characteristics through the compression function of SHA/SHA-1, these have been systematically studied in [9]. We propose to model the impact of them by setting a $C(i) > 1$ for each disturbance in step $i$ where there exist also less probable characteristics. Examples will be given in Section 5.4.

**Conditions at the end of each block.** By using a 2-block approach, characteristics with a better probability can be found. Furthermore, the conditions at the end of each block can be partially ignored (without further explanation already observed in [18]). This improves the probability of the characteristic significantly. For the first message block all the conditions in the last 2 steps can be ignored. For the second block this is not the case, since for every difference in the initial value a correcting difference is needed to cancel it out. However, if we can guarantee that the sign of the disturbances in the last 2 steps is opposite to the sign of the according differences in the initial value, then we can ignore the carry conditions for these disturbances. This also improves the probability of the characteristic in the second block. In general the attack complexity is dominated by the complexity of the second block, since only the carry conditions in the last 2 steps can be ignored. We propose to model the impact of them by setting a $C(i) > 1$ for each case. An example will be given in Section 5.4.

### 4.4 Greedy approach

The simple greedy approach was introduced in [3]. The idea is to run through all bit positions of every state variable and every expanded message word, check which conditions can be added to improve the total work factor, and finally pick the position and corresponding condition which yields the largest gain. By repeating this many times, the work factor can be gradually improved. No corrective factor is needed.

### 4.5 Message Modification

Message Modification was introduced by Wang *et al.* in the cryptanalysis of MD4 [15], MD5 [19] and the SHA-family [18,20]. The main idea of message modification is to use the degrees of freedom one has in the choice of the message words to fulfill conditions on the state variables. Since every message word is only used once in the first 16 steps, all the conditions on the state variables can be easily fulfilled for these steps. This method is referred to as simple message modification. After step 16 each message word depends on at least 4 previous message words. Hence, a more sophisticated method (referred to as advanced message modification) is needed to fulfill conditions after step 16. It can be described as follows:

1. Check if one of the conditions on the state variables is not satisfied. (starting at the LSB)
2. If one condition does not hold then flip the according bit in message word $W_i$. This causes a change in a previous $W_t$ for some $t < 16$ due to the message expansion. Hence, a change in $A_{t+1}$. This can be compared to introducing a new difference (disturbance) in step $t$.
3. Correct the differences in $A_{t+1}, \ldots, A_{t+6}$ by adjusting the according message words $W_{t+1}, \ldots, W_{t+5}$.

In detail this correction is equal to constructing a *new* local collision with a disturbance in step $t$. Note that this method does not work if the correction in the message words affects one of the conditions on the state variables or message words themselves. Thus, the degree of freedom for advanced message modification is determined by the characteristic in the first 16 steps.

As shown in unpublished but informally presented results by Wang [16,17] the attack complexity of $2^{69}$ can be improved to $2^{63}$ by doing message modification up to step 25. Wang *et al.* estimated the cost for message modification of about $2^2$ SHA-1 compression function evaluations. Note that a new characteristic for the first 16 steps was needed to do message modification up to step 25.

## 4.6 Equation solving

At FSE 2007, Sugita *et al.* presented a new method for message modification in SHA-1 using symbolic computation [14]. Their method reduces the number of trials (needed message pairs) significantly at the cost of increased message modification costs. With their method a collision in 58-step SHA-1 can be constructed with complexity close to $2^8$ message modification (symbolic computation) steps which they claim is approximately $2^{31}$ SHA-1 computations (experimentally). Note that the complexity of Wang's attack on 58-step SHA-1 is about $2^{34}$ hash computations. Unfortunately, Sugita *et al.* do not give any information how this comparison to SHA-1 was done. This makes it very difficult to compare their approach to others. Furthermore, the description of their method is vague and they do not give any estimations for the attack complexity on the full SHA-1 hash function. At the current state it is not clear if this method can lead to any improvements in the attack complexity of SHA-1.

## 4.7 Neutral Bits

This technique was invented by Biham and Chen in the analysis of SHA [1]. The main idea of this approach is to start the collision search from some intermediate step $r$ and hence improving the complexity of the attack. Therefore, Biham and Chen invented the notion of neutral bits. For a given message pair $(m, m^*)$ that follows the characteristic up to step $r$ the $j^{th}$-bit is called neutral if the message pair $(m \oplus 2^j, m^* \oplus 2^j)$ also follows the characteristic up to step $r$. Every set of neutral bits can be used to generate $2^t$ new message pairs, where $t$ denote the number of neutral bits. The attack is based on the observation that a fraction

(1/8) of these message pairs again follow the characteristic up to step $r$. Hence, one get $2^{t-3}$ message pairs following the characteristic up to step $r$. It is easy to see that this reduces the complexity of the collision search in SHA.

## 4.8 Boomerangs/Tunnels

At CRYPTO 2007, Joux and Peyrin presented a new idea on how to improve the attack complexity of SHA-1 [6]. It uses a variant of the boomerang attack, known from analysis of block ciphers. The method is similar to the idea of tunneling as introduced by Klima [8]. Each message pair that follows the characteristic in the first steps is related to another message pair by a high probability auxiliary differential. This auxiliary differential ensure that the characteristic also holds in the first steps for the other message pair. Hence, each auxiliary differential doubles the number of message pairs that follow the characteristic in the first steps, which improves the complexity of the attack. An easy method to construct these auxiliary differentials is to combine several local collisions. With this method auxiliary differentials can be constructed up to step 29. However, to guarantee that the auxiliary differential holds a set of additional conditions have to be fulfilled in the steps before. This on the one hand reduces the degrees of freedom needed in the final collision search and on the other hand a characteristic is needed for the first 16 steps that is compatible with the auxiliary differential. It is an interesting research problem to maximize the number of auxiliary differentials that fit into a suitable characteristic for collision search. However, issues like available message freedom and implementation aspects can *hugely* influence the resulting work factor for collision search.

We propose to model the impact of auxiliary differentials as follows. Each auxiliary differentials allows to increase a single $C(i)$ by $2 \cdot p_{aux}$, where $i$ is the first step where $P_{corr}(i) < 1$, and $p_{aux}$ is the probability for the auxiliary differential to hold up to step $i$ which is often 1 or close to 1. The details depend on the characteristic being used and the auxiliary differentials. The consequences are interesting: even in favorable settings, the resulting corrected work factor can not be improved by a factor of 2 per auxiliary differential, but noticeably less. As an example, consider a setting where 6 auxiliary differentials are used. Instead of a 64-fold improvement, the improvement is less than 45-fold. This has several reasons, *i. e.* the precise way our model takes the early-stop strategy into account.

## 4.9 Comparison of methods

Comparison of different approaches to speed-up collision search for SHA and SHA-1 is difficult because usually not enough information is provided in the respective descriptions.

Chabaud and Joux count in their attack on SHA [4] the number of needed message pairs for constructing a collision to estimate the attack complexity. Furthermore, they provide some measurements to confirm their estimates. This makes it easy to compare their results with other implementations. Unfortunately, this is not the case for most of the recent published attacks on SHA and SHA-1.

In their recent attacks on SHA and SHA-1 Wang *et al.* count the number of conditions that have to be fulfilled such that the message follows the characteristic to estimate the complexity of attack. Furthermore, they consider improvements achieved by message modification techniques as well as early stop. This lead to an estimated attack complexity for SHA and SHA-1 of about $2^{39}$ and $2^{63}$ hash computations, respectively. However, since the description message modification is vague, it is difficult to compare it to other methods.

In [14], Sugita *et al.* count the number of symbolic computations (message modification steps) needed for their message modification technique. Unfortunately, they do not give any timing information for the algorithm, which makes it difficult to compare the method to others.

In [3], De Cannière and Rechberger count the number of nodes in a search tree that have to be visited to find a collision in the hash function to estimate the complexity of the attack. This estimation already includes improvement of message modification and early stop. With their approach a collision for 64-step SHA-1 can be found with a complexity of about $2^{35}$ hash computations.

An other notable example is the SHA collision by Naito *et al.* given in [10]. They improved the collision attack of Wang *et al.* on SHA by a factor of $2^3$. To estimate the complexity of the attack, they build upon the work of Wang *et al.* in the original attack. In addition to counting conditions for the characteristic and considering the improvement of early stop and the cost for message modification, they also provide measurements. Their finding is that a collision in SHA has a complexity of about $2^{36}$ hash computations and takes on average about 100 hours (on a Pentium4 3.4GHZ CPU). This shows an interesting gap between claimed complexity and measurement. In fact one can expect more than $2^{20}$ SHA compression function calls per second on such a machine and hence would expect a runtime of less than 20 hours.

### 4.10 Proposal

In order to avoid misinterpretation and allow fair comparison of different methods, we propose to directly compare every fast collision search method with a standard implementation of SHA-1 (*e. g.* OpenSSL) on the same platform. This would make comparison of different approaches easier in the future. In cases where collision search can not be implemented it is still possible to give measurement results for parts of the characteristic. We refer to our case study for an example.

## 5 Case study: Collision search for 70-step SHA-1

### 5.1 Message Difference

We developed efficient search algorithms to find suitable message differences. They are based on methods developed in [12], with the improvement that exact probabilities as described in [3,9] instead of Hamming weights are used to prune and rank them.

As described in Section 3, the effort to find colliding message pairs for SHA-1 mainly depends on the number of conditions between state and/or message bits where no method to fulfill them better than random trials is known. For evaluating and comparing candidate message differences, it will be useful to have the following definition:

**Definition 4.** *The* truncated total work from step *i* $N_t(i)$ *of a characteristic is the product of all corrected probabilities down to step i, i.e.,*

$$N_t(i) = \prod P_{corr(j)},$$

where j runs from R downto i.

Assuming that the controlled probability $P_c(i)$ can be ignored (which is perfectly reasonable for $i > 16$), $N_t(i)$ for $i > 16$ can be used as an estimate of the total work without fully specifying the generalized characteristic from step 0 on. The argument $i$ in $N_t(i)$ can be interpreted as the threshold up to which methods more efficient than random trials are known to look for right message pairs.

**Table 3.** Disturbance vectors for 70-step SHA-1

| | steps 18-70 | | steps 20-70 | |
|---|---|---|---|---|
| message | Hamming weight | $N_t(18)$ | Hamming weight | $N_t(20)$ |
| MD 1 | 19 | $2^{51.66}$ | 17 | $2^{47.40}$ |
| MD 2 | 19 | $2^{48.66}$ | 17 | $2^{47.23}$ |
| MD 3 | 19 | $2^{50.36}$ | 16 | $2^{47.37}$ |
| MD 4 | 19 | $2^{50.22}$ | 17 | $2^{49.87}$ |
| MD 5 | 19 | $2^{49.09}$ | 17 | $2^{47.32}$ |
| MD 6 | 19 | $2^{50.14}$ | 18 | $2^{48.50}$ |

For the attack we used the message difference MD 2, since it has the best truncated total work after step 18 and 20. This perfectly matches to the greedy method we use to speed-up collision search.

## 5.2 Detailed Characteristic

Table 4 shows the used message difference, which is the same for both blocks. The characteristics found for the two blocks are given in Tables 5 and 6 respectively. For improved collision search efficiency the probability of them was further improved by fixing the actual values of certain bits in the message and the internal state using available degrees of freedom. Table 7 and 8 show the respective results.

The total expected work for both blocks amounts to about $2^{44}$ compression function equivalents. In contrast to other figures given in the literature this includes the impact of a less than ideal implementation (which is in our case about a factor 10).

We run experiments for parts of the used characteristic to give this estimate. Our experiment which produced an actual 70-step collision confirms this estimates. Note that this includes the impact of a less than optimal implementation of the collision search and compares to a fast implementation of SHA-1 (OpenSSL) on the same platform. For that, we used as a means of comparison the SHA-1 implementation of OpenSSL, which can do about $2^{20}$ compression functions per second on our PC.

A straightforward extension of the method used for the 64-step collision as presented in [3] to 70 steps would have required more than $2^{50}$ compression function equivalents. The gain in speed is partly due to the choice of a different disturbance vector, and partly due to an improvement of the greedy-approach.

## 5.3 The employed improved greedy-approach

In our case study, we employ the greedy approach as described in Section 4.4. We improve upon [3] in the following way. Instead of picking only single bit positions, we pick several of them at once. This results in a larger search space but also in better results. We always pick a set of 7 bits (a local collision) and test for each bit which condition would yield to the largest gain in the work factor. This is an easy way to estimate the improvement of the work factor for one local collision. Note that checking all $2^7$ possibilities to add conditions for a local collision would be inefficient. After testing all local collisions we pick the one with the largest improvement and set the corresponding conditions. By repeating this method the work factor can be gradually improved.

## 5.4 Some corrective factors

**Conditions at the end of each block.** Applying the rules described in Section 4.3 to the characteristic in both blocks (Tables 7 and 8) we can remove all 6 conditions in the last two steps of the first block and 2 conditions in the second block. In terms of corrective factors as introduced in Section 4.2, we arrive at a $C(69) = 2^3$ and $C(70) = 2^3$ for the first block and $C(69) = 2^1$ and $C(70) = 2^1$ for the second block.

**Impact of additional less probable characteristics.** We achieve some more speedup if also less probable characteristics are allowed. Once at bit position 0 in step 34, we would get a speed-up by 25%. The three disturbances in step 62, 65 and 66 would each allow a speedup of about 6.25%. By avoiding strict checking of conditions/differences in the implementation, we can hence expect to visit only about 66% nodes in order to find a suitable message pair. In terms of corrective factors as introduced in Section 4.2, we arrive at a $C(34) = 1.25$, and $C(62) = C(65) = C(66) = 1.0625$. Note that this speed-up applies to both blocks in the same way.

**Table 4.** Example of a 70-step SHA-1 collision using the standard IV

| $i$ | Message 1 ($m_0$), first block | | | | Message 1 ($m_1$), second block | | | |
|---|---|---|---|---|---|---|---|---|
| 1–4 | 3BB33AAE | 85AECBBB | 57A88417 | 8137CB9C | ABDDBEE2 | 42A20AC7 | A915E04D | 5063B027 |
| 5–8 | 4DE99220 | 5B6F12C7 | 726BD948 | E3F6E9B8 | 4DDF989A | E0020CF7 | 7FFDC0F4 | EFEFE0A7 |
| 9–12 | 23607799 | 239B2F1D | AAC76B94 | E8009A1E | 0FFBC2F0 | C8DE16BF | 81BBE675 | 254429CB |
| 13–16 | C24DE871 | 5B7C30D8 | 000359F5 | 90F9ED31 | 5F37A2C6 | CD1963D3 | FFCA1CB9 | 9642CB56 |

| $i$ | Message 2 ($m_0^*$), first block | | | | Message 2 ($m_1^*$), second block | | | |
|---|---|---|---|---|---|---|---|---|
| 1–4 | ABB33ADE | 35AECBE8 | 67A8841F | 8137CBDF | 3BDDBE92 | F2A20A94 | 9915E045 | 5063B064 |
| 5–8 | 9DE99252 | EB6F12D7 | 826BD92A | 23F6E9FA | 9DDF98E8 | 50020CE7 | 8FFDC096 | 2FEFE0E5 |
| 9–12 | 236077A9 | C39B2F5F | 8AC76BF4 | 08009A5F | 0FFBC2C0 | 28DE16FD | A1BBE615 | C544298A |
| 13–16 | E24DE821 | 9B7C3099 | E0035987 | 30F9ED32 | 7F37A296 | 0D196392 | 1FCA1CCB | 3642CB55 |

| $i$ | XOR-difference are the same for both blocks | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1–4 | 90000070 | B0000053 | 30000008 | 00000043 | 90000070 | B0000053 | 30000008 | 00000043 |
| 5–8 | D0000072 | B0000010 | F0000062 | C0000042 | D0000072 | B0000010 | F0000062 | C0000042 |
| 9–12 | 00000030 | E0000042 | 20000060 | E0000041 | 00000030 | E0000042 | 20000060 | E0000041 |
| 13–16 | 20000050 | C0000041 | E0000072 | A0000003 | 20000050 | C0000041 | E0000072 | A0000003 |

| $i$ | The colliding hash values | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1–5 | 151866D5 | F7940D84 | 28E73685 | C4D97E18 | 97DA712B | | | |

## 6 Conclusions

Currently known differential collision search attacks on hash functions like SHA-1 need little memory and are trivially parallizable. Still, theoretical analysis (counting conditions, calculate probabilities for successful message modification) often leads to optimistic conclusions about actual collision search implementation costs.

Measurement results and comparison with standard hash implementations on the same platform are needed to compare different collision search strategies. As a case study, a collision search method and an example of a colliding message pair for 70-step SHA-1 was used. The highest number of steps for which a SHA-1 collision was published so far was 64.

## References

1. Eli Biham and Rafi Chen. Near-Collisions of SHA-0. In Matthew K. Franklin, editor, *CRYPTO*, volume 3152 of *LNCS*, pages 290–305. Springer, 2004.
2. Eli Biham, Rafi Chen, Antoine Joux, Patrick Carribault, Christophe Lemuet, and William Jalby. Collisions of SHA-0 and Reduced SHA-1. In Ronald Cramer, editor, *EUROCRYPT*, volume 3494 of *LNCS*, pages 36–57. Springer, 2005.
3. Christophe De Cannière and Christian Rechberger. Finding SHA-1 Characteristics: General Results and Applications. In Xuejia Lai and Kefei Chen, editors, *ASIACRYPT*, volume 4284 of *LNCS*, pages 1–20. Springer, 2006.
4. Florent Chabaud and Antoine Joux. Differential Collisions in SHA-0. In Hugo Krawczyk, editor, *CRYPTO*, volume 1462 of *LNCS*, pages 56–71. Springer, 1998.
5. Terutoshi Iwasaki, Jun Yajima, Yu Sasaki, Yusuke Naito, Takeshi Shimoyama, Noboru Kunihiro, and Kazuo Ohta. On the complexity of collision attack against SHA-1 and new disturbance vectors, August 2006. Presented at rump session of CRYPTO 2006.
6. Antoine Joux and Thomas Peyrin. Hash Functions and the (Amplified) Boomerang Attack. In Alfred Menezes, editor, *CRYPTO*, volume 4622 of *LNCS*, pages 244–263. Springer, 2007.

7. Charanjit S. Jutla and Anindya C. Patthak. Provably Good Codes for Hash Function Design. In Ronald Cramer, editor, *Proceedings of SAC 2006*, LNCS. Springer, 2006. To appear.
8. Vlastimil Klima. Tunnels in Hash Functions: MD5 Collisions Within a Minute. Cryptology ePrint Archive, Report 2006/105, 2006. `http://eprint.iacr.org/`.
9. Florian Mendel, Norbert Pramstaller, Christian Rechberger, and Vincent Rijmen. The Impact of Carries on the Complexity of Collision Attacks on SHA-1. In Matthew J. B. Robshaw, editor, *FSE*, volume 4047 of *LNCS*, pages 278–292. Springer, 2006.
10. Yusuke Naito, Yu Sasaki, Takeshi Shimoyama, Jun Yajima, Noboru Kunihiro, and Kazuo Ohta. Improved Collision Search for SHA-0. In Xuejia Lai and Kefei Chen, editors, *ASIACRYPT*, volume 4284 of *LNCS*, pages 21–36. Springer, 2006.
11. National Institute of Standards and Technology (NIST). FIPS-180-2: Secure Hash Standard, August 2002. Available online at `http://www.itl.nist.gov/fipspubs/`.
12. Norbert Pramstaller, Christian Rechberger, and Vincent Rijmen. Exploiting Coding Theory for Collision Attacks on SHA-1. In Nigel P. Smart, editor, *IMA Int. Conf.*, volume 3796 of *LNCS*, pages 78–95. Springer, 2005.
13. Vincent Rijmen and Elisabeth Oswald. Update on sha-1. In Alfred Menezes, editor, *CT-RSA*, volume 3376 of *LNCS*, pages 58–71. Springer, 2005.
14. Makoto Sugita, Mitsuru Kawazoe, Ludovic Perret, and Hideki Imai. Algebraic cryptanalysis of 58-round sha-1. In Alex Biryukov, editor, *FSE*, volume 4593 of *LNCS*, pages 349–365. Springer, 2007.
15. Xiaoyun Wang, Xuejia Lai, Dengguo Feng, Hui Chen, and Xiuyuan Yu. Cryptanalysis of the Hash Functions MD4 and RIPEMD. In Ronald Cramer, editor, *EUROCRYPT*, volume 3494 of *LNCS*, pages 1–18. Springer, 2005.
16. Xiaoyun Wang, Andrew Yao, and Frances Yao. Cryptanalysis of SHA-1. Presented at the Cryptographic Hash Workshop hosted by NIST, October 2005.
17. Xiaoyun Wang, Andrew Yao, and Frances Yao. New Collision Search for SHA-1, August 2005. Presented at rump session of CRYPTO 2005.
18. Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu. Finding Collisions in the Full SHA-1. In Victor Shoup, editor, *CRYPTO*, volume 3621 of *LNCS*, pages 17–36. Springer, 2005.
19. Xiaoyun Wang and Hongbo Yu. How to Break MD5 and Other Hash Functions. In Ronald Cramer, editor, *EUROCRYPT*, volume 3494 of *LNCS*, pages 19–35. Springer, 2005.
20. Xiaoyun Wang, Hongbo Yu, and Yiqun Lisa Yin. Efficient Collision Search Attacks on SHA-0. In Victor Shoup, editor, *CRYPTO*, volume 3621 of *LNCS*, pages 1–16. Springer, 2005.
21. Michael J. Wiener. The Full Cost of Cryptanalytic Attacks. *J. Cryptology*, 17(2):105–124, 2004.

**Table 5.** Characteristic for the first block of the 70-step collision before optimization.

| $i$ | $\nabla A_i$ | $\nabla W_i$ | $F_W$ | $P_u(i)$ | $P_c(i)$ | $N_s(i)$ |
|---:|---|---|---:|---:|---:|---:|
| -4: | 00001111010010111000011111000011 | | | | | |
| -3: | 01000000110010010101000111011000 | | | | | |
| -2: | 01100010111010110111001111111010 | | | | | |
| -1: | 11101111110011011010101110001001 | | | | | |
| 0: | 01100111010001010010001100000001 | n--u1011--1-------111---1nun1110 | 14 | -4.42 | 0.00 | 0.00 |
| 1: | u1-u1011--10---1--010-1-nuun0001 | u0nn----------1----01----n1u--uu | 20 | -13.00 | -0.79 | 0.79 |
| 2: | 0u0u10--0-01--100--0nun1-00nnn1u | --nu-----------0----0-----01n111 | 22 | -17.09 | -1.00 | 1.00 |
| 3: | 111n111---1-nuuu1u-00n11u-0110u1 | 100------------11----01-1n0111nn | 17 | -16.00 | -1.00 | 1.83 |
| 4: | 101001111--nn011n1u11n000101n10n | nu0n--------10------0010-nun00n0 | 15 | -14.83 | -2.83 | 2.83 |
| 5: | 1100100u0--010101u110unn-100u10u | n1nu--11-------------------n011- | 21 | -19.09 | -1.00 | 1.00 |
| 6: | n10--0unnnnnnnnnnnnn0--0n10-0un- | nuuu00-----------------1-un---n- | 22 | -18.27 | -2.00 | 2.00 |
| 7: | 000111----11101u001u10-1100nn-00 | uu------------1---1------n--10n0 | 23 | -18.16 | -2.75 | 2.75 |
| 8: | u-0-----10111111--1010u-0-0uu0u1 | 0-1-----------------------nu---1 | 26 | -11.00 | -1.42 | 1.42 |
| 9: | 1-0--------------0-111u-1----01- | nnu----------------------n0---n- | 25 | -11.00 | -3.42 | 3.42 |
| 10: | 0-0-----------------1nu---nn001- | 1-u----------------------nn----0 | 23 | -6.00 | -1.00 | 1.00 |
| 11: | ----------------------n1----100n | uuu----------------------n-1---n | 25 | -7.00 | -3.42 | 3.42 |
| 12: | u1--------------------------00nn | -1n----------------------u-u---- | 22 | -7.00 | -0.61 | 13.00 |
| 13: | u-1-0-------------------1--0-1n- | nu-----------------------u-----n | 13 | -3.00 | -1.00 | 28.00 |
| 14: | -0-------------------------1---u | nnn----------------------uuu--n- | 11 | -6.00 | -2.19 | 38.00 |
| 15: | 110-0-----------------------0u-- | u0n---------------------------nu | 14 | -4.00 | -2.42 | 43.00 |
| 16: | n0--------------------------1-01 | nn-------------------------u--n- | 0 | -3.00 | -1.00 | 53.00 |
| 17: | --0---------------------------u1 | 0un---------------------1n---1-n | 0 | -3.00 | -2.00 | 50.00 |
| 18: | n-0----------------------------- | nu------------------------un--n- | 0 | 0.00 | -0.00 | 47.00 |
| 19: | -------------------------------- | 0-u-----------------------1---1n | 0 | -0.00 | -0.00 | 47.00 |
| 20: | -------------------------------- | nu----------------------0-----u- | 0 | -2.00 | -1.42 | 47.00 |
| 21: | ------------------------------u- | nun----------------------n----u1 | 0 | -2.00 | -2.00 | 45.00 |
| 22: | ------------------------------u- | 1un----------------------n----n0 | 0 | -1.00 | -1.00 | 43.00 |
| 23: | -------------------------------- | n----------------------------1u0 | 0 | -1.00 | -1.00 | 42.00 |
| 24: | -------------------------------- | -------------------------------1 | 0 | -0.00 | 0.00 | 41.00 |
| 25: | -------------------------------- | 10----------------------0------0 | 0 | 0.00 | -0.00 | 41.00 |
| 26: | -------------------------------- | u0-------------------------0---- | 0 | 0.00 | 0.00 | 41.00 |
| 27: | -------------------------------- | 100---------------------------u- | 0 | -1.00 | -0.42 | 41.00 |
| 28: | ------------------------------u- | 0------------------------n------ | 0 | 0.00 | 0.00 | 40.00 |
| 29: | -------------------------------- | 0------------------------------0 | 0 | -2.00 | -1.00 | 40.00 |
| 30: | ------------------------------n- | u------------------------u---1-0 | 0 | -0.00 | -0.00 | 38.00 |
| 31: | -------------------------------- | u-0----------------------------1 | 0 | -2.00 | -1.00 | 38.00 |
| 32: | ------------------------------n- | 0-0----------------------u----1- | 0 | -0.00 | -0.00 | 36.00 |
| 33: | -------------------------------- | n0----------------------------0n | 0 | -2.00 | -2.00 | 36.00 |
| 34: | -------------------------------u | -------------------------nu----0 | 0 | 0.00 | 0.00 | 34.00 |
| 35: | -------------------------------- | n1---------------------------0nu | 0 | -1.00 | -1.00 | 34.00 |
| 36: | -------------------------------- | -n----------------------------u1 | 0 | -2.00 | -2.00 | 33.00 |
| 37: | ------------------------------u- | uu-----------------------n----1- | 0 | -1.00 | -1.00 | 31.00 |
| 38: | -------------------------------- | nu---------------------0------n- | 0 | -1.00 | -1.00 | 30.00 |
| 39: | -------------------------------- | u----------------------------00- | 0 | 0.00 | -0.00 | 29.00 |
| 40: | -------------------------------- | u------------------------------- | 0 | -1.00 | -1.00 | 29.00 |
| 41: | -------------------------------- | u-0---------------------------u0 | 0 | -1.00 | -1.00 | 28.00 |
| 42: | ------------------------------u- | 0------------------------n------ | 0 | 0.00 | 0.00 | 27.00 |
| 43: | -------------------------------- | -1----------------------------n0 | 0 | -1.00 | -1.00 | 27.00 |
| 44: | -------------------------------- | u------------------------------- | 0 | -1.00 | -1.00 | 26.00 |
| 45: | -------------------------------- | n-----------------------------10 | 0 | -1.00 | -1.00 | 25.00 |
| 46: | -------------------------------- | n-----------------------------0- | 0 | -0.00 | 0.00 | 24.00 |
| 47: | -------------------------------- | 11---------------------------0u1 | 0 | -1.00 | -1.00 | 24.00 |
| 48: | ------------------------------u- | -------------------------n----0- | 0 | 0.00 | 0.00 | 23.00 |
| 49: | -------------------------------- | -----------------------------1-1 | 0 | -2.00 | -1.00 | 23.00 |
| 50: | ------------------------------u- | u------------------------n-----0 | 0 | -1.00 | -1.00 | 21.00 |
| 51: | -------------------------------- | x---------------------------11n- | 0 | -2.00 | -2.00 | 20.00 |
| 52: | -------------------------------- | 0------------------------------- | 0 | -1.00 | -1.00 | 18.00 |
| 53: | -------------------------------- | x0----------------------------11 | 0 | -1.00 | -1.00 | 17.00 |
| 54: | -------------------------------- | x------------------------------- | 0 | -0.00 | 0.00 | 16.00 |
| 55: | -------------------------------- | -------------------------------1 | 0 | 0.00 | 0.00 | 16.00 |
| 56: | -------------------------------- | -------------------------------- | 0 | -0.00 | 0.00 | 16.00 |
| 57: | -------------------------------- | ----------------------------0--- | 0 | 0.00 | 0.00 | 16.00 |
| 58: | -------------------------------- | ----------------------------1-0- | 0 | 0.00 | 0.00 | 16.00 |
| 59: | -------------------------------- | -------------------------------- | 0 | -0.00 | 0.00 | 16.00 |
| 60: | -------------------------------- | -------------------------------- | 0 | 0.00 | -0.00 | 16.00 |
| 61: | -------------------------------- | -----------------------------u-1 | 0 | -1.00 | -0.42 | 16.00 |
| 62: | -----------------------------u-- | ------------------------n--0---- | 0 | 0.00 | 0.00 | 15.00 |
| 63: | -------------------------------- | -----------------------------x-- | 0 | -1.00 | 0.00 | 15.00 |
| 64: | -------------------------------- | ----------------------------n--x | 0 | -2.00 | -0.19 | 14.00 |
| 65: | ----------------------------n--- | -----------------------u--0--n-x | 0 | -2.00 | -0.42 | 12.00 |
| 66: | -----------------------------n-- | ------------------------u---x--n | 0 | -1.00 | 0.00 | 10.00 |
| 67: | -------------------------------- | ---------------------------n-xx- | 0 | -3.00 | -0.36 | 9.00 |
| 68: | ---------------------------n---- | ----------------------u-----u-xx | 0 | -3.00 | -0.42 | 6.00 |
| 69: | ----------------------------u--- | -----------------------n---xx-ux | 0 | -3.00 | -0.42 | 3.00 |
| 70: | ----------------------------x--- | | | | | |

**Table 6.** Characteristic for the second block of the 70-step collision before optimization.

| $i$ | $\nabla A_i$ | $\nabla W_i$ | $F_W$ | $P_u(i)$ | $P_c(i)$ | $N_s(i)$ |
|---|---|---|---|---|---|---|
| -4: | 01010011010101010100101101110n00 | | | | | |
| -3: | 01100110101100010100011111001010 | | | | | |
| -2: | 00000011110101011011000000nu1011 | | | | | |
| -1: | 0100010101011010000011001101u010 | | | | | |
| 0: | 0101001110100011001110111000n101 | u01n1011-----------111101uun0010 | 11 | -2.52 | 0.00 | 0.00 |
| 1: | n11n1000-------1-1-11001nnun0nuu | n1nn-----------------0101u0n01uu | 17 | -8.00 | -0.96 | 0.96 |
| 2: | 1u0u10100------0---n-nn11100010u | 10un-----------------0000100u101 | 17 | -9.38 | -0.35 | 0.35 |
| 3: | u1un00001-----nn-u-0-00nu1011u0n | 010----------------1-0000n1001uu | 17 | -14.00 | 0.00 | 3.79 |
| 4: | n0101110n1u-u-101-n1011011uu0010 | nu0n11-------------1-0001nnu10u0 | 14 | -14.62 | -2.61 | 6.79 |
| 5: | 010n011u11010n1-nu-10u01u0101nnu | u1un000000-------------0111u0111 | 13 | -16.19 | -5.17 | 6.18 |
| 6: | 1n000111100uu001-0100-n00011001u | nuuu1111111------------01uu--1n0 | 14 | -12.44 | -2.99 | 2.99 |
| 7: | 1nnnnnnnnnnnnnnn-111--01nn1--u00 | uu1-111111--------------1n10-1u1 | 16 | -6.00 | 0.00 | 0.00 |
| 8: | 100-11000-----1100-u---1un11-101 | 000----------------------1uu---0 | 25 | -19.68 | 0.00 | 0.00 |
| 9: | 010011111111100111--01-0-100--0n | uun----------------------n111-u1 | 23 | -10.00 | -4.00 | 4.00 |
| 10: | u1--01--------------11--nu0u0un0 | 10n----------------------uu10--1 | 22 | -6.42 | -1.61 | 1.61 |
| 11: | 111-1-----------------n--0000011 | nnu----------------------x00---u | 25 | -7.83 | -2.61 | 2.61 |
| 12: | 0-1-0---------------------1n-1n1 | 01n----------------------u-n---0 | 22 | -2.42 | -0.68 | 9.51 |
| 13: | u--------------------------0-0-u | uu0----------------------x0----u | 14 | -7.00 | -3.00 | 29.09 |
| 14: | 101------------------------1-1-u | uuu----------------------xuu--n1 | 13 | -5.09 | -0.29 | 36.09 |
| 15: | --1-0----------------------11u-- | u0n------------------------10-un | 13 | -4.00 | -2.83 | 44.00 |
| 16: | n1--------------------------0-11 | un0----------------------0-u--n1 | 0 | -3.00 | -2.00 | 53.00 |
| 17: | --0---------------------------n1 | 0un----------------------u0--10n | 0 | -3.00 | -2.00 | 50.00 |
| 18: | n-0----------------------------- | uu1-----------------------un--n1 | 0 | 0.00 | 0.00 | 47.00 |
| 19: | -------------------------------- | 00u-----------------------0---0n | 0 | -0.00 | 0.00 | 47.00 |
| 20: | -------------------------------- | nu0---------------------1-----n0 | 0 | -2.00 | -2.00 | 47.00 |
| 21: | ------------------------------n- | nun----------------------u----n0 | 0 | -2.00 | -2.00 | 45.00 |
| 22: | ------------------------------n- | 0un----------------------u----u1 | 0 | -1.00 | -1.00 | 43.00 |
| 23: | -------------------------------- | u11--------------------------1n1 | 0 | -1.00 | -1.00 | 42.00 |
| 24: | -------------------------------- | 100---------------------10----00 | 0 | 0.00 | 0.00 | 41.00 |
| 25: | -------------------------------- | 110---------------------------11 | 0 | -0.00 | -0.00 | 41.00 |
| 26: | -------------------------------- | u00------------------------0--11 | 0 | 0.00 | -0.00 | 41.00 |
| 27: | -------------------------------- | 100---------------------------n0 | 0 | -1.00 | -1.00 | 41.00 |
| 28: | ------------------------------n- | 011----------------------u----00 | 0 | 0.00 | -0.00 | 40.00 |
| 29: | -------------------------------- | 001---------------------------01 | 0 | -2.00 | -2.00 | 40.00 |
| 30: | ------------------------------u- | n10----------------------n---111 | 0 | 0.00 | -0.00 | 38.00 |
| 31: | -------------------------------- | u00--------------------------110 | 0 | -2.00 | -2.00 | 38.00 |
| 32: | ------------------------------u- | 100----------------------n--1011 | 0 | 0.00 | 0.00 | 36.00 |
| 33: | -------------------------------- | u10-------------------------000u | 0 | -2.00 | -2.00 | 36.00 |
| 34: | -------------------------------n | 011----------------------un-0001 | 0 | 0.00 | -0.00 | 34.00 |
| 35: | -------------------------------- | u10-------------------------01un | 0 | -1.00 | -1.00 | 34.00 |
| 36: | -------------------------------- | 0u0------------------------1-1n1 | 0 | -2.00 | -2.00 | 33.00 |
| 37: | ------------------------------n- | uu-----------------------u--1011 | 0 | -1.00 | -1.00 | 31.00 |
| 38: | -------------------------------- | un---------------------1-----1n0 | 0 | -1.00 | -1.00 | 30.00 |
| 39: | -------------------------------- | u10--------------------------011 | 0 | -0.00 | -0.00 | 29.00 |
| 40: | -------------------------------- | u-0--------------------1-----110 | 0 | -1.00 | -1.00 | 29.00 |
| 41: | -------------------------------- | n-0--------------------------1n0 | 0 | -1.00 | -1.00 | 28.00 |
| 42: | ------------------------------n- | 10-----------------------u---010 | 0 | 0.00 | -0.00 | 27.00 |
| 43: | -------------------------------- | -1---------------------------0u1 | 0 | -1.00 | -1.00 | 27.00 |
| 44: | -------------------------------- | x11--------------------------100 | 0 | -1.00 | -1.00 | 26.00 |
| 45: | -------------------------------- | u----------------------------101 | 0 | -1.00 | -1.00 | 25.00 |
| 46: | -------------------------------- | n-1-------------------------011- | 0 | 0.00 | 0.00 | 24.00 |
| 47: | -------------------------------- | 11--------------------------00n- | 0 | -1.00 | -0.42 | 24.00 |
| 48: | ------------------------------n- | -------------------------u--0011 | 0 | 0.00 | -0.00 | 23.00 |
| 49: | -------------------------------- | -1----------------------1---10-0 | 0 | -2.00 | -1.00 | 23.00 |
| 50: | ------------------------------n- | u-1----------------------u--11-0 | 0 | -1.00 | -1.00 | 21.00 |
| 51: | -------------------------------- | x-1-------------------------10u- | 0 | -2.00 | -2.00 | 20.00 |
| 52: | -------------------------------- | 1---------------------------1-1- | 0 | -1.00 | -1.00 | 18.00 |
| 53: | -------------------------------- | x0--------------------------0-10 | 0 | -1.00 | -1.00 | 17.00 |
| 54: | -------------------------------- | x---------------------------11-- | 0 | 0.00 | 0.00 | 16.00 |
| 55: | -------------------------------- | -----------------------------0-1 | 0 | 0.00 | 0.00 | 16.00 |
| 56: | -------------------------------- | 0----------------------------01- | 0 | 0.00 | 0.00 | 16.00 |
| 57: | -------------------------------- | 0---------------------------0--- | 0 | 0.00 | 0.00 | 16.00 |
| 58: | -------------------------------- | ----------------------------0-1- | 0 | 0.00 | 0.00 | 16.00 |
| 59: | -------------------------------- | ----------------------------11-1 | 0 | -0.00 | 0.00 | 16.00 |
| 60: | -------------------------------- | -------------------------------- | 0 | 0.00 | -0.00 | 16.00 |
| 61: | -------------------------------- | -----------------------------n-0 | 0 | -1.00 | -0.42 | 16.00 |
| 62: | -----------------------------n-- | ------------------------u--01-0- | 0 | 0.00 | 0.00 | 15.00 |
| 63: | -------------------------------- | -----------------------------x1- | 0 | -1.00 | 0.00 | 15.00 |
| 64: | -------------------------------- | ----------------------------u--x | 0 | -2.00 | -0.19 | 14.00 |
| 65: | ----------------------------u--- | -----------------------n1-11-u-x | 0 | -2.00 | -0.42 | 12.00 |
| 66: | -----------------------------u-- | ------------------------n---x--u | 0 | -1.00 | 0.00 | 10.00 |
| 67: | -------------------------------- | -----------------------1---u-xx- | 0 | -3.00 | -0.36 | 9.00 |
| 68: | ---------------------------u---- | -----------------------n----n-xx | 0 | -3.00 | -0.42 | 6.00 |
| 69: | ----------------------------n--- | -----------------------u---xu-nx | 0 | -3.00 | -0.42 | 3.00 |
| 70: | ----------------------------u--- | | | | | |

**Table 7.** Characteristic for the first block of the 70-step collision after applying the greedy approach. Bold numbers in column $P_{corr}(i)$ highlight impact of new corrective factors.

| $i$ | $\nabla A_i$ | $\nabla W_i$ | $F_W$ | $P_u(i)$ | $P_{corr}(i)$ | $N_s(i)$ | $N_{corr}(i)$ |
|---|---|---|---|---|---|---|---|
| -4: | 01010011010101010100101101110n00 | | | | | | |
| -3: | 01100110101100010100011111001010 | | | | | | |
| -2: | 00000011110101011011000000nu1011 | | | | | | |
| -1: | 01000101010110100000110011 01u010 | | | | | | |
| 0: | 01100111010001010010001100000001 | n01u1011101100110011 10--1nun1110 | 2 | -1.00 | 0.00 | 1.02 | 0.00 |
| 1: | u10u1011011001111101001-nuun0001 | u0nn010110101110 11-010111n1u10uu | 1 | -1.00 | 0.00 | 2.02 | 0.00 |
| 2: | 0u0u1001010110100000nun1000nnn1u | 01nu011110101000100001000001n111 | 0 | 0.00 | 0.00 | 2.02 | 0.00 |
| 3: | 111n11111111nuuu1u100n11u10110u1 | 1000000100110111110010111n0111nn | 0 | 0.00 | 0.00 | 2.02 | 0.00 |
| 4: | 10100111100nn011n1u11n000101n10n | nu0n110111101001100100100nun00n0 | 0 | 0.00 | 0.00 | 2.02 | 0.00 |
| 5: | 1100100u000010101u110unn0100u10u | n1nu1011011011110001001---0n0111 | 3 | -3.00 | 0.00 | 2.02 | 0.00 |
| 6: | n10000unnnnnnnnnnnnn0100n1000un1 | nuuu00100110101111 0110-10un010n0 | 1 | 0.00 | -3.00 | 2.02 | 0.00 |
| 7: | 00011111001110 1u001u10-1100nn000 | uu10001111110110 1-1010--1n1110n0 | 3 | -2.00 | -1.00 | 3.02 | 0.00 |
| 8: | u00011011011111 1011010u-010uu0u1 | 001000110110000001-1011-10nu1001 | 2 | -1.00 | -5.00 | 4.02 | 0.00 |
| 9: | 1100111111110001100111u-11001011 | nnu0001110011011--------0n0111n1 | 8 | -4.00 | -2.00 | 5.02 | 0.00 |
| 10: | 01000100001000101---1nu-10nn0010 | 10u0101011000-110--------nn10100 | 9 | -6.00 | -4.00 | 9.02 | 2.88 |
| 11: | 10011101100011-1000---n1--10100n | uuu010000-00000----------n01111n | 11 | -3.00 | -7.42 | 12.02 | 5.88 |
| 12: | u101111110100001----------0000nn | 11n000100100-1-----------u1u0001 | 12 | -8.00 | -7.00 | 20.02 | 13.88 |
| 13: | u11001111110011111------1-1011n1 | nu011011011111-00--------u01100n | 9 | -4.00 | -2.00 | 24.02 | 17.88 |
| 14: | 1010111000100000----------11111u | nnn000000000001----------uuu01n1 | 10 | -1.00 | -8.00 | 29.02 | 22.88 |
| 15: | 110101110111----------------0u01 | u0n10000111-1---1----------100nu | 14 | -5.00 | -6.00 | 38.02 | 31.88 |
| 16: | n0111011--0011--------------1001 | nn10100000001-11--------111u00n0 | 0 | -0.02 | -0.55 | 47.02 | 40.88 |
| 17: | 000--1111-------------------11u1 | 0un01110000000----------1n01111n | 0 | 0.00 | 0.00 | 47.00 | 40.86 |
| 18: | n00----------------------------- | nu00000011-1---1----------un01n0 | 0 | -1.00 | -1.00 | 47.00 | 40.86 |
| 19: | --0----------------------------- | 00u10100-010-1-----------110101n | 0 | 0.00 | 0.00 | 46.00 | 39.86 |
| 20: | -------------------------------- | nu100111100-1-----------000011u1 | 0 | -1.00 | -1.00 | 46.00 | 39.86 |
| 21: | ------------------------------u- | nun001100-1---0----------n0001u1 | 0 | -2.00 | -2.00 | 45.00 | 38.86 |
| 22: | ------------------------------u- | 1un0101-010-1-----------1n0111n0 | 0 | -1.00 | -1.00 | 43.00 | 36.86 |
| 23: | -------------------------------- | n110111000-1--------------1101u0 | 0 | -1.00 | -1.00 | 42.00 | 35.86 |
| 24: | -------------------------------- | 11001111-0---0----------11110101 | 0 | 0.00 | 0.00 | 41.00 | 34.86 |
| 25: | -------------------------------- | 100111-1-0-0------------00000010 | 0 | -0.00 | 0.00 | 41.00 | 34.86 |
| 26: | -------------------------------- | u00011001-0-------------11101110 | 0 | -0.00 | 0.00 | 41.00 | 34.86 |
| 27: | -------------------------------- | 1001000-----0-----------101100u0 | 0 | -1.00 | -1.00 | 41.00 | 34.86 |
| 28: | ------------------------------u- | 01110-0-1---------------0n010010 | 0 | 0.00 | 0.00 | 40.00 | 33.86 |
| 29: | -------------------------------- | 00000010-1----------------000000 | 0 | -2.00 | -2.00 | 40.00 | 33.86 |
| 30: | ------------------------------n- | u11001-----0------------0u010110 | 0 | -0.00 | 0.00 | 38.00 | 31.86 |
| 31: | -------------------------------- | u000-0-0------------------010111 | 0 | -2.00 | -2.00 | 38.00 | 31.86 |
| 32: | ------------------------------n- | 0100101-0-------------100u000011 | 0 | -0.00 | 0.00 | 36.00 | 29.86 |
| 33: | -------------------------------- | n0000-----0-------------0100000n | 0 | -2.00 | **-1.91** | 36.00 | 29.86 |
| 34: | -------------------------------u | 010-0-1-----------------1nu00100 | 0 | -0.00 | 0.00 | 34.00 | 27.94 |
| 35: | -------------------------------- | n10100-0----------------001110nu | 0 | -1.00 | -1.00 | 34.00 | 27.94 |
| 36: | -------------------------------- | 1n11------------------11100000u1 | 0 | -2.00 | -2.00 | 33.00 | 26.94 |
| 37: | ------------------------------u- | uu-1-0-1----------------0n101010 | 0 | -1.00 | -1.00 | 31.00 | 24.94 |
| 38: | -------------------------------- | nu100-1----------------0000011n1 | 0 | -1.00 | -1.00 | 30.00 | 23.94 |
| 39: | -------------------------------- | u00---------------------01000001 | 0 | 0.00 | 0.00 | 29.00 | 22.94 |
| 40: | -------------------------------- | u-1-0------------------011100101 | 0 | -1.00 | -1.00 | 29.00 | 22.94 |
| 41: | -------------------------------- | u101-1-----------------1111110u0 | 0 | -1.00 | -1.00 | 28.00 | 21.94 |
| 42: | ------------------------------u- | 01-----1---------------00n110010 | 0 | -0.00 | 0.00 | 27.00 | 20.94 |
| 43: | -------------------------------- | -1-0---------------------10111n0 | 0 | -1.00 | -1.00 | 27.00 | 20.94 |
| 44: | -------------------------------- | u11-0------------------001111011 | 0 | -1.00 | -1.00 | 26.00 | 19.94 |
| 45: | -------------------------------- | n-----0-----------------00011110 | 0 | -1.00 | -1.00 | 25.00 | 18.94 |
| 46: | -------------------------------- | n-0---------------------1000100- | 0 | -0.00 | 0.00 | 24.00 | 17.94 |
| 47: | -------------------------------- | 11-0---------------------10110u1 | 0 | -1.00 | -1.00 | 24.00 | 17.94 |
| 48: | ------------------------------u- | ----------------------100n111001 | 0 | 0.00 | 0.00 | 23.00 | 16.94 |
| 49: | -------------------------------- | -0-1-------------------0000101-1 | 0 | -2.00 | -2.00 | 23.00 | 16.94 |
| 50: | ------------------------------u- | u-1---------------------1n011100 | 0 | -1.00 | -1.00 | 21.00 | 14.94 |
| 51: | -------------------------------- | x-----------------------111011n- | 0 | -2.00 | -2.00 | 20.00 | 13.94 |
| 52: | -------------------------------- | 0---------------------1111000-0- | 0 | -1.00 | -1.00 | 18.00 | 11.94 |
| 53: | -------------------------------- | x0---------------------111010011 | 0 | -1.00 | -1.00 | 17.00 | 10.94 |
| 54: | -------------------------------- | x---------------------01000110-- | 0 | -0.00 | 0.00 | 16.00 | 9.94 |
| 55: | -------------------------------- | ------------------------0100-0-1 | 0 | 0.00 | 0.00 | 16.00 | 9.94 |
| 56: | -------------------------------- | 1---------------------100111101- | 0 | -0.00 | 0.00 | 16.00 | 9.94 |
| 57: | -------------------------------- | ------------------------01010--- | 0 | 0.00 | 0.00 | 16.00 | 9.94 |
| 58: | -------------------------------- | ----------------------01001-1-0- | 0 | 0.00 | 0.00 | 16.00 | 9.94 |
| 59: | -------------------------------- | ------------------------101010-1 | 0 | 0.00 | 0.00 | 16.00 | 9.94 |
| 60: | -------------------------------- | ----------------------001100---- | 0 | 0.00 | 0.00 | 16.00 | 9.94 |
| 61: | -------------------------------- | ------------------------01-1-u-1 | 0 | -1.00 | **-0.99** | 16.00 | 9.94 |
| 62: | -----------------------------u-- | ----------------------00n0000-0- | 0 | 0.00 | 0.00 | 15.00 | 8.96 |
| 63: | -------------------------------- | ------------------------100--x-- | 0 | -1.00 | -1.00 | 15.00 | 8.96 |
| 64: | -------------------------------- | --------------------10110-0-n--x | 0 | -2.00 | **-1.98** | 14.00 | 7.96 |
| 65: | ----------------------------n--- | --------------------001u0101-n-x | 0 | -2.00 | **-1.98** | 12.00 | 5.98 |
| 66: | -----------------------------n-- | ---------------------011u1--x--n | 0 | -1.00 | -1.00 | 10.00 | 4.00 |
| 67: | -------------------------------- | ---------------------111-0-n-xx- | 0 | -3.00 | -3.00 | 9.00 | 3.00 |
| 68: | ---------------------------n---- | ---------------------0u0100-u-xx | 0 | -3.00 | **0.00** | 6.00 | 0.00 |
| 69: | ----------------------------u--- | ----------------------1n0--xx-ux | 0 | -3.00 | **0.00** | 3.00 | 0.00 |
| 70: | ----------------------------x--- | | | | | | |

**Table 8.** Characteristic for the second block of the 70-step collision after applying the greedy approach. Bold numbers in column $P_{corr}(i)$ highlight impact of new corrective factors.

| $i$ | $\nabla A_i$ | $\nabla W_i$ | $F_W$ | $P_u(i)$ | $P_{corr}(i)$ | $N_s(i)$ | $N_{corr}(i)$ |
|---|---|---|---|---|---|---|---|
| -4: | 0101001101010101010100101101110n00 | | | | | | |
| -3: | 01100110101100010100011111001010 | | | | | | |
| -2: | 000000111101010110110000000nu1011 | | | | | | |
| -1: | 0100010101011010000011001101u010 | | | | | | |
| 0: | 010100111010001100111011110000n101 | u01n1011110111011011111101uun0010 | 0 | 0.00 | 0.00 | 0.97 | 0.00 |
| 1: | n11n1000100100010101100 1nnun0nuu | n1nn00101010001000001010 1u0n01uu | 0 | 0.00 | 0.00 | 0.97 | 0.00 |
| 2: | 1u0u10100101001 0010n0nn11100010u | 10un100100010101111000000100u101 | 0 | 0.00 | 0.00 | 0.97 | 0.00 |
| 3: | u1un0000100111nn1u00100nu1011u0n | 01010000011000111011 00000n1001uu | 0 | 0.00 | 0.00 | 0.97 | 0.00 |
| 4: | n0101110n1u1u11010n1011011uu0010 | nu0n11011101111110011 0001nnu10u0 | 0 | 0.00 | 0.00 | 0.97 | 0.00 |
| 5: | 010n011u11010n10nu010u01u0101nnu | u1un00000000001000001-00111u0111 | 1 | 0.00 | 0.00 | 0.97 | 0.00 |
| 6: | 1n000111100uu00110100-n00011001u | nuuu1111111111---1000-001uu101n0 | 4 | -3.00 | -3.00 | 1.97 | 0.00 |
| 7: | 1nnnnnnnnnnnnnnn01111-01nn101u00 | uu10111111101111-1100--01n1001u1 | 3 | -1.00 | -1.00 | 2.97 | 0.24 |
| 8: | 1000110000111111001u---1un110101 | 0000111111111011--000---11uu0000 | 5 | -5.00 | -5.00 | 4.97 | 2.24 |
| 9: | 0100111111111001110001-00100010n | uun0100011011110--------1n1111u1 | 8 | -2.00 | -2.00 | 4.97 | 2.24 |
| 10: | u1010110010011------11--nu0u0un0 | 10n0000110111-1-1--------uu10101 | 10 | -4.00 | -4.00 | 10.97 | 8.24 |
| 11: | 111010101111----------n--0000011 | nnu001010-00010----------x00101u | 12 | -7.42 | -7.42 | 16.97 | 14.24 |
| 12: | 0011001000011-------------1n01n1 | 01n111110011-1-----------u0n0110 | 12 | -7.00 | -7.00 | 21.55 | 18.82 |
| 13: | u0010001111000-----------000001u | uu0011010001-0-1---------x01001u | 12 | -2.00 | -2.00 | 26.55 | 23.82 |
| 14: | 10110001010----------------1111u | uuu1111111-0101----------xuu10n1 | 12 | -8.00 | -8.00 | 36.55 | 33.82 |
| 15: | 0010010001111--------------11u10 | u0n10110010-0-1-1----------101un | 13 | -6.00 | -6.00 | 40.55 | 37.82 |
| 16: | n1100111111000-------------10011 | un000000010-0-0----------00u10n1 | 0 | -0.55 | -0.55 | 47.55 | 44.82 |
| 17: | 000001111--------------------1n1 | 0un010111-1010-----------u00110n | 0 | 0.00 | 0.00 | 47.00 | 44.27 |
| 18: | n00--------------------------100 | uu10011001-0-1-0----------un10n1 | 0 | -1.00 | -1.00 | 47.00 | 44.27 |
| 19: | --0----------------------------- | 00u01010-1-0-0-----------000010n | 0 | 0.00 | 0.00 | 46.00 | 43.27 |
| 20: | -------------------------------- | nu001101-11-1-----------110010n0 | 0 | -1.00 | -1.00 | 46.00 | 43.27 |
| 21: | ------------------------------n- | nun010010-1-0-0----------u0101n0 | 0 | -2.00 | -2.00 | 45.00 | 42.27 |
| 22: | ------------------------------n- | 0un0101-0-0-1------------u1100u1 | 0 | -1.00 | -1.00 | 43.00 | 40.27 |
| 23: | -------------------------------- | u111100-00-1--------------0011n1 | 0 | -1.00 | -1.00 | 42.00 | 39.27 |
| 24: | -------------------------------- | 10001110-1-0-0----------10010000 | 0 | 0.00 | 0.00 | 41.00 | 38.27 |
| 25: | -------------------------------- | 110110-0-1-1-------------0010111 | 0 | -0.00 | 0.00 | 41.00 | 38.27 |
| 26: | -------------------------------- | u00000-11-1-------------10101111 | 0 | 0.00 | 0.00 | 41.00 | 38.27 |
| 27: | -------------------------------- | 1001100---1-1------------00110n0 | 0 | -1.00 | -1.00 | 41.00 | 38.27 |
| 28: | ------------------------------n- | 01101-1-1----------------u000000 | 0 | -0.00 | 0.00 | 40.00 | 37.27 |
| 29: | -------------------------------- | 00100-11-0----------------111101 | 0 | -2.00 | -2.00 | 40.00 | 37.27 |
| 30: | ------------------------------u- | n10110-----1-------------n010111 | 0 | 0.00 | 0.00 | 38.00 | 35.27 |
| 31: | -------------------------------- | u001-1-0------------------101110 | 0 | -2.00 | -2.00 | 38.00 | 35.27 |
| 32: | ------------------------------u- | 1001-11-1--------------10n111011 | 0 | 0.00 | 0.00 | 36.00 | 33.27 |
| 33: | -------------------------------- | u1000---1-0------------11001000u | 0 | -2.00 | **-1.91** | 36.00 | 33.27 |
| 34: | -------------------------------n | 011-1-0-----------------0un00001 | 0 | 0.00 | 0.00 | 34.00 | 31.36 |
| 35: | -------------------------------- | u10-10-1----------------111001un | 0 | -1.00 | -1.00 | 34.00 | 31.36 |
| 36: | -------------------------------- | 0u01-------------------0110101n1 | 0 | -2.00 | -2.00 | 33.00 | 30.36 |
| 37: | ------------------------------n- | uu-1-1-1----------------1u001011 | 0 | -1.00 | -1.00 | 31.00 | 28.36 |
| 38: | -------------------------------- | un-01-1----------------1001001n0 | 0 | -1.00 | -1.00 | 30.00 | 27.36 |
| 39: | -------------------------------- | u10---0-----------------11000011 | 0 | 0.00 | 0.00 | 29.00 | 26.36 |
| 40: | -------------------------------- | u-0-1------------------110011110 | 0 | -1.00 | -1.00 | 29.00 | 26.36 |
| 41: | -------------------------------- | n-01-0------------------011101n0 | 0 | -1.00 | -1.00 | 28.00 | 25.36 |
| 42: | ------------------------------n- | 10-----0----------------1u011010 | 0 | 0.00 | 0.00 | 27.00 | 24.36 |
| 43: | -------------------------------- | -1-1---------------------01110u1 | 0 | -1.00 | -1.00 | 27.00 | 24.36 |
| 44: | -------------------------------- | x11-1------------------011101100 | 0 | -1.00 | -1.00 | 26.00 | 23.36 |
| 45: | -------------------------------- | u---0-0-----------------10000101 | 0 | -1.00 | -1.00 | 25.00 | 22.36 |
| 46: | -------------------------------- | n-1---------------------0110011- | 0 | 0.00 | 0.00 | 24.00 | 21.36 |
| 47: | -------------------------------- | 11-0---------------------01000n- | 0 | -1.00 | -1.00 | 24.00 | 21.36 |
| 48: | ------------------------------n- | -----------------------01u000011 | 0 | 0.00 | 0.00 | 23.00 | 20.36 |
| 49: | -------------------------------- | -1-1-------------------0110010-0 | 0 | -2.00 | -2.00 | 23.00 | 20.36 |
| 50: | ------------------------------n- | u-1---------------------1u0111-0 | 0 | -1.00 | -1.00 | 21.00 | 18.36 |
| 51: | -------------------------------- | x-1---------------------101010u- | 0 | -2.00 | -2.00 | 20.00 | 17.36 |
| 52: | -------------------------------- | 1---------------------1110101-1- | 0 | -1.00 | -1.00 | 18.00 | 15.36 |
| 53: | -------------------------------- | x0---------------------010100-10 | 0 | -1.00 | -1.00 | 17.00 | 14.36 |
| 54: | -------------------------------- | x----------------------0111011-- | 0 | 0.00 | 0.00 | 16.00 | 13.36 |
| 55: | -------------------------------- | ------------------------0111-0-1 | 0 | 0.00 | 0.00 | 16.00 | 13.36 |
| 56: | -------------------------------- | 0----------------------00100-01- | 0 | 0.00 | 0.00 | 16.00 | 13.36 |
| 57: | -------------------------------- | 0-----------------------11010--- | 0 | 0.00 | 0.00 | 16.00 | 13.36 |
| 58: | -------------------------------- | -----------------------1001-0-1- | 0 | 0.00 | 0.00 | 16.00 | 13.36 |
| 59: | -------------------------------- | ------------------------101-11-1 | 0 | -0.00 | 0.00 | 16.00 | 13.36 |
| 60: | -------------------------------- | -----------------------11110---- | 0 | -0.00 | 0.00 | 16.00 | 13.36 |
| 61: | -------------------------------- | ------------------------01-0-n-0 | 0 | -1.00 | **-0.99** | 16.00 | 13.36 |
| 62: | -----------------------------n-- | -----------------------1u1-01-0- | 0 | 0.00 | 0.00 | 15.00 | 12.37 |
| 63: | -------------------------------- | ------------------------111--x1- | 0 | -1.00 | -1.00 | 15.00 | 12.37 |
| 64: | -------------------------------- | --------------------10000-1-u--x | 0 | -2.00 | **-1.98** | 14.00 | 11.37 |
| 65: | ----------------------------u--- | --------------------010n1-11-u-x | 0 | -2.00 | **-1.98** | 12.00 | 9.39 |
| 66: | -----------------------------u-- | ---------------------001n1--x--u | 0 | -1.00 | -1.00 | 10.00 | 7.42 |
| 67: | -------------------------------- | ----------------------11-0-u-xx- | 0 | -3.00 | **-2.42** | 9.00 | 6.42 |
| 68: | ---------------------------u---- | ---------------------1n0-01-n-xx | 0 | -3.00 | **-2.00** | 6.00 | 4.00 |
| 69: | ----------------------------n--- | ----------------------1u1--xu-nx | 0 | -3.00 | **-2.00** | 3.00 | 2.00 |
| 70: | ----------------------------u--- | | | | | | |