This is a postprint version of the following published document:

Reviriego, P., Martinez, J., Ottavi, M. (2021). Soft Error Tolerant Count Min Sketches. *IEEE Transactions on Computers*, 70(2), 284–290.

DOI: <https://doi.org/10.1109/TC.2020.2987890>

©2021 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

Soft Error Tolerant Count Min Sketches

Pedro Reviriego¹, Jorge Martínez² and Marco Ottavi³

Abstract—The estimation of the frequency of the elements on a set is needed in a wide range of computing applications. For example, to estimate the number of hits that a video gets or the number of packets in a network flow. In some cases, the number of elements in the set is very large and it is not practical to maintain a table with the exact count for each of them. Instead, simpler and more efficient data structures, commonly referred to as sketches, that provide an estimate are used. Among those structures the Count Min Sketch (CMS) is one of the most popular sketches. The CMS provides estimates that have one sided errors. In more detail, the CMS returns an estimate that is equal to or larger than the actual value. An update or check requires a small and constant number of memory accesses and the memory footprint is fixed and does not depend on the number of elements. The CMS relies on several arrays of counters that are stored in memory. Memories are prone to suffer soft errors that flip the contents of memory cells due for example to ionizing radiation. Therefore, it is of interest to study the impact that soft errors can have on the CMS estimates and to propose protection techniques that minimize their effect while requiring low overhead in terms of additional memory and circuitry. To the best of our knowledge this has not been done before. In this paper, first the effect of soft errors on the CMS is evaluated by injecting errors. Then, in the second part a protection technique that does not require additional memory bits is presented and compared with the protection using a parity bit. In the last part of the paper, the technique is extended to protect also against double adjacent bit errors.

Index Terms—Soft errors, frequency estimation, count min sketch.

1 INTRODUCTION

Estimating the frequency of elements in a data set is an important problem in computing [1]. An exact computation of the frequency can be done by adding the elements with an associated counter to a table and increasing the corresponding counter each time an element is found in the set. However, as the size and number of data sets to analyze grows, an exact computation of the frequency becomes unpractical and it is common to use data structures that provide an approximate estimate [2]. These structures commonly referred to as Sketches do not assign a counter to each element and instead rely on all the elements sharing a pool of counters. This is typically done using several hash functions to map elements to counters [3]. This means that due to hash collisions, several elements can map to the same counter so that it does not store the value that corresponds

to any of them. This can be mitigated by using positive and negative increments for the counters as done in the Count Sketch (CS) [3] or by using only positive increments but selecting the minimum value of the counters that an element maps to. This is done in the Count Min Sketch (CMS) which is one of the most widely used [4]. The CMS can be improved by using conservative increments that only update the counters that have the minimum value [5] or by using fingerprints associated with each counter to reduce the increment when consecutive updates have different fingerprints [6].

In many applications, the frequency of the elements is highly skewed so that a minority of elements appears many times and most elements appear only a few times. This is for example the case in network traffic [7]. In those cases, the values of the counters in a CMS are also skewed with a few counters having large values while most having small ones. This can be exploited to design sketches that adapt to different counting ranges or that try to compress the counters that store small values [8]. Typically, sketches would be able to provide more accurate estimates for the elements that appear many times as they are less affected by hash collisions. Instead, elements with low frequencies can have significant deviations from the sketch estimate to the actual value due to collisions.

Sketches in general and the CMS in particular rely on one or several pools of counters that are stored in memory. A potential issue is that memories are prone to suffer errors that can modify the value of the bits stored in the memory cells [9]. For example, radiation induced soft errors are an important issue in advanced memories [10]. To protect memories and avoid data corruption, Error Correction Codes (ECCs) are widely used in memories [11] and new codes are being developed to target the error patterns relevant for new memory technologies [12], [13]. For example, a parity bit can be added to each memory word to detect single bit errors or a Single Error Correction (SEC) code to correct them. The use of ECCs implies additional costs as memory cells have to be added to store the parity bits and circuitry is needed to encode the data when writing to the memory and decode it to detect or correct errors when reading from it. An alternative, when the memory is used for a particular data structure or application, is to design protection schemes that exploit the application or data structure properties to provide protection against errors. Such algorithmic based error tolerant schemes have for example been developed for Bloom filters [14].

This paper considers the effects of soft errors on the

¹P. Reviriego is with Universidad Carlos III de Madrid, Leganés 28911, Madrid, Spain. email: revirieg@it.uc3m.es

²J. Martínez is with Universidad Antonio de Nebrija, 28040, Madrid, Spain. email: jmartine@nebrija.es

³M. Ottavi is University of Rome Tor Vergata, Department of Electronic Engineering, Rome, Italy. email: marco.ottavi@uniroma2.it

CMS and presents an efficient protection technique that does not require additional memory bits. In the first part of the paper, the effects of soft errors on a CMS are evaluated using as case study a network monitoring application that counts the number of packets in network flows. The results show that the CMS is able to mask some errors so that they do not affect the frequency estimates. This occurs for example, when a counter that stores a large value and thus is not used as estimate for any flow suffers an error that increases the value. However, some errors can introduce significant deviations on the estimates for some flows. The second part of the paper considers the protection of the CMS. To protect the CMS two alternatives are considered, the use of a parity bit per counter and a new technique that exploits the features of the CMS to provide protection without adding parity bits. Both schemes have been evaluated and the results show that the proposed scheme provides almost the same level of protection at a lower cost. The main contributions of the paper are to analyze the effects of soft errors on the Count Min Sketch (CMS) and to show that efficient algorithmic error tolerant protection schemes can be designed for counting sketches. This paves the way for further research on designing efficient error protection schemes for other sketches given their wide adoption in computing and networking applications.

The rest of the paper is organized as follows. Section 2 provides a brief overview of the Count Min Sketch (CMS). The effect of soft errors on the CMS is evaluated on section 3 that summarizes the results of error injection simulations. The proposed protection scheme is presented in section 4 and evaluated in section 5. In section 6, the extension of the proposed protection scheme to protect against multiple bit errors is briefly discussed. The paper ends with the conclusions in section 7.

2 THE COUNT MIN SKETCH

To estimate the frequency of elements, the Count Min Sketch (CMS) uses a set of r arrays of k counters to which elements are mapped using hash functions. The elements are mapped to counter $h_i(x)$ on the i^{th} array such that each element is mapped to r counters, one per array [4]. The CMS supports two operations $Update(x)$ and $Estimate(x)$. The update operation, increases the counters associated with element x . To do so, the counters on positions $h_i(x)$ are read and incremented on the i^{th} array. To estimate the frequency of an element x , the CMS reads the counters on positions $h_i(x)$ on the i^{th} array and returns the minimum value. The CMS is illustrated in Figure 1 that shows the estimate operation for an element x . The algorithms to update and estimate the frequency of an element are given in Algorithms 1,2.

Algorithm 1 Update(x)

- 1: **for** $i \leftarrow 1$ to r **do**
 - 2: read *counter* in position $h_i(x)$ of the i^{th} array.
 - 3: set *counter* = *counter* + 1
 - 4: write *counter* in position $h_i(x)$ of the i^{th} array
 - 5: **end for**
-

The estimates of a CMS can only be equal to or larger than the actual value. Therefore estimation errors are one

Fig. 1. Illustration of a CMS with $r = 4$ and $k = 10$.

Algorithm 2 Estimate(x)

- 1: set c_{min} to a large value
 - 2: **for** $i \leftarrow 1$ to r **do**
 - 3: read *counter* in position $h_i(x)$ of the i^{th} array.
 - 4: **if** *counter* < c_{min} **then**
 - 5: set $c_{min} = \text{counter}$
 - 6: **end if**
 - 7: **end for**
 - 8: **return** c_{min}
-

sided towards overestimation. The actual value is obtained when no other elements are mapped to at least one of the r counters associated with element x . Therefore, better estimates are obtained when either r or k are large. This however increases the memory footprint needed to implement the CMS. The accuracy of the CMS can be improved by using conservative updates so that when updating the counters associated with an element, only those with the minimum value are increased [5]. Fingerprints can also be used to detect when different elements update the same counter and reduce the increments [6].

3 IMPACT OF SOFT ERRORS ON THE CMS

As discussed in the introduction, a soft error can flip a bit stored in a memory. For a CMS such a bit flip can either increment or decrement a counter. If the counter value increases as a result of the bit flip, then the CMS may introduce a larger error in the estimation for some elements but the estimation error remains one sided. Instead, if the bit flip decreases the value of the counter, the CMS may return an estimate that is smaller than the actual value. Therefore, estimation errors are no longer one sided. In more detail, a bit flip that increments a counter would only affect the estimates of the elements that had that counter as their minimum. Instead, errors that decrement a counter can potentially affect all the elements that map to that counter. That would be the case when for example the counter stores a value '10000000' and the bit flip changes the counter to '00000000'. Therefore, it seems that errors that reduce the counter value have potentially a larger effect on the CMS.

To evaluate the impact of soft errors on the CMS on a realistic configuration, bit flips have been injected on a CMS used to estimate the number of packets in network flows. This is done using publicly available packet traces from the CAIDA network monitors in Chicago (CHI15) [15] and San Jose (SJ12) [16] and from the MAWI project [17]. The

traces correspond to one minute of traffic and the number of packets and flows for each trace are shown in Table 1.

TABLE 1
Traces used to evaluate the CMS

Trace	Number of packets	Number of flows
CHI15	36.8M	1.6M
SJ12	32.8M	13.5M
MAWI	12.9M	0.3M

The parameter used to measure the impact of a soft error on a counter of the CMS is the deviation that it introduces on the estimates of the elements that map to that counter. More precisely, let us consider that an error is introduced on a counter c and that the error free CMS estimate of element x that maps to that counter is $e_{noerror}(x)$ and the CMS estimate once the error is inserted is $e_{error}(x)$. Then the deviation suffered by element x is the difference of the CMS estimates $d(x) = e_{error}(x) - e_{noerror}(x)$. When the deviation is negative, the error reduces the estimate and can lead to underestimation and conversely when it is positive, it increases the estimate and can lead to overestimation.

In line with other works on radiation induced soft errors, the error model considered is single bit errors on a counter [9]. This means that at a given time, there is only a counter in error which is reasonable as soft errors are rare events and it is inline with the assumptions made when designing most error correction codes for memories [11].

To evaluate the impact of soft errors, bit flips have been injected on each of the counter bits and counter and then the estimates for the elements that map to that counter have been computed and compared to those of an error free CMS. To speed up the error injection simulations, a bit position b is selected and then that bit is flipped in all the counters of one of the r arrays and the estimates for all flows are recomputed. Note that this ensures that exactly one counter has suffered a flip on bit b for each element x as each element maps to one and only one counter per array. Therefore, this accelerated simulation effectively tests the error model on which any element is affected by a single error. The difference between the estimate and the error free estimate is logged for all the flows. The process is repeated for all the r arrays.

As an example, the results for the SJ12 trace with $r = 4$, $k = 524288$ and counters of 16 bits are discussed next. Figure 2 shows the deviations from the error free estimation when injecting errors on the fourth bit (starting from the least significant one). In this case, the counter value is increased (reduced) by eight when the bit is flip from 0 to 1 (1 to 0). The Figure shows both the cases on which the bit flip leads to overestimation (top) and underestimation (bottom). The x-axis has the magnitude of the deviation from the error free estimate and the y-axis the percentage of times that this value occurs. The plots are on logarithmic scale on both axis so that small values of the deviation and frequency can be seen. A first observation is that the maximum deviation is eight as expected. The bit flip can lead to both overestimation and underestimation. The first case occurs when a counter that is the minimum for a flow increases due to the bit flip. The second when the counter that was the minimum decreases or when another counter

decreases and becomes the minimum. Since the counter values are much larger than eight, the deviations take all possible values.

Fig. 2. Percentage of flips on bit four that introduce a given deviation on the estimate for the SJ12 trace with $k = 512k$, $r = 4$

Figure 3 shows the deviations when the bit flip affects bit eight. In this case, the deviations are limited to 128. For overestimation, larger values are significantly less frequent. This can be explained as overestimation only occurs when the minimum counter for a flow increases. In that case, if the increment is large, it is likely that another counter becomes the minimum and the overestimation is smaller than 128. For underestimations, the probabilities are in general smaller. This is because many counters are smaller than 128 and thus a bit flip on bit 8 cannot reduce their value as that bit is zero. There is a peak for deviations that correspond to the bit value. Those occur when the minimum counter has the bit set to 1 and the bit flip resets it to 0. The same trends are seen for bit 12 on Figure 4.

Fig. 3. Percentage of flips on bit eight that introduce a given deviation on the estimate for the SJ12 trace with $k = 512k$, $r = 4$

The results for the most significant bit are shown in Figure 5. In this case, there are very few underestimations. This is because only a very small fraction of the counters take values larger than 32768 and are thus susceptible to

Fig. 4. Percentage of flips on bit twelve that introduce a given deviation on the estimate for the SJ12 trace with $k = 512k$, $r = 4$

underestimation. A similar behaviour occurs for overestimation which does not occur for large values.

Fig. 5. Percentage of flips on bit sixteen that introduce a given deviation on the estimate for the SJ12 trace with $k = 512k$, $r = 4$

After discussing the impact on bit flips on different bits, let us consider the overall impact for all bits. The results when considering errors on all bits are summarized on Figure 6 where the contributions of each bit to the underestimation are clearly observed. Looking at the overall results, the CMS shows some robustness as in many cases, there are no deviations and when the estimate suffers a deviation it tends to be small. For an application perspective, underestimations are potentially more dangerous as they can modify the CMS behaviour qualitatively (recall that the CMS has one sided deviations with no underestimation). The results for the other two traces are shown in Figures 7,8 and show a similar behaviour.

4 PROTECTING THE CMS

To protect the CMS a direct solution would be to use a memory protected by an error correction code to store the counters [11]. For example, if a Single Error Correction (SEC) code is used, then single bit flips would be corrected and therefore would not affect the CMS estimates. This however has a number of issues. The first one are the costs associated with an error correction code: additional memory bits per

Fig. 6. Percentage of bit flips that introduce a given deviation on the estimate for the SJ12 trace with $k = 512k$, $r = 4$

Fig. 7. Percentage of bit flips that introduce a given deviation on the estimate for the MAWI trace with $k = 32k$, $r = 4$

word and circuitry to encode and decode when writing and reading from the memory. For the 16 bit counters considered, the number of parity check bits needed by a SEC code would be 5 thus introducing a 5/16 overhead on the memory size. This is above 30% and thus can have a significant impact on the total cost. Instead, the encoding and decoding circuitry for a SEC code is not complex and could in many cases be simpler than that needed for the hash functions needed in the CMS. The second one is that in some cases, the hardware platform is given and if the memory is not protected, the designer cannot change it. Therefore, in general it would be good to find protection techniques that require a lower cost and that do not require changes on the underlying memory, even if the protection is not as good as that of using an error correction code.

An alternative to lower the cost would be to use a single parity bit per counter (as in this case the memory overhead is only 1/16) to detect errors and then discard the counters that are in error. This could introduce some deviations on the CMS estimates but would preserve the one sided errors. Intuitively, the impact on the estimations should be small in most cases as the remaining $r - 1$ counters can be used for the estimation. This is the first protection technique considered in the rest of the paper. However, when the memory used is not parity protected, this is not an option.

Fig. 8. Percentage of bit flips that introduce a given deviation on the estimate for the CHI15 trace with $k = 128k$, $r = 4$

To protect the CMS without adding additional parity bits, it is interesting to recall that in many applications the frequency of elements is highly skewed and only a few elements have large counter values. Therefore, the upper counter bits would be zero in most cases. For example, for the CMS evaluated in the first experiment of the previous section, the MSB of the counter was zero in most of the counters for the CHI15, SJ12 and MAWI traces. This observation can be exploited to propagate errors to the MSB as proposed in [18]. The idea is to replace the MSB with the parity of the counter when it is written. Then when reading the counter, the original MSB is recovered by doing an *xor* of the stored MSB and the rest of the bits. Therefore, in an error free scenario, all counter bits can be used to store the values normally. However, when there is any single bit error on the stored bits the value of the MSB would also be flipped. This is better seen with some examples.

The overall scheme is shown in Figure 9 that illustrates how instead of the original MSB, the parity of all counter bits is written to the memory in the MSB. In the example shown, a value of '10010101' is stored instead of the original value of '00010101'. Then when reading, the same operation is done to recover the original MSB. Therefore, the MSB can be used normally and the counter can store the same range of values as in an unprotected implementation. Let us consider now that a bit stored in memory is flipped as shown in Figure 10 for the bit underlined in red. Then, the recovered MSB is also flipped as shown in the Figure. Therefore, the value read would be '10000101' instead of '00010101' so that two bits are different. In particular, the MSB has been changed from 0 to 1 which for a CMS would mean that now the counter has a large value. This would be in most cases equivalent to removing that counter from consideration as now it will not be the minimum for any element. This is an interesting observation as it means that since most counters will have small values, the scheme would provide a protection similar to that of parity. Let us now consider that the counter takes a value '10000010', then the value '00000010' would be stored in the counter. If later the counter is read, the logic would decode the MSB and recover the original value '10000010'. This second example illustrates how in the error free case, all the bits in the counter can be used to store values and

thus the counters can store the same range of values as in an unprotected implementation.

Fig. 9. Error propagation scheme to the MSB [18].

Fig. 10. Example of a bit flip propagating to the MSB.

Let us describe the proposed protection scheme more formally. The write and read operations to the memory are modified such that to write a value c formed by bits c_i with $i = 0, 1, \dots, MSB$, we compute w_i as follows: $w_i = c_i$ for all bits except the MSB. For the MSB we do $w_{MSB} = c_0 \text{ xor } c_1 \text{ xor } \dots \text{ xor } c_{MSB}$. Conversely, when reading from the memory the counter bits are recovered by doing $c_i = w_i$ for all bits except the MSB. For the MSB we do $c_{MSB} = w_0 \text{ xor } w_1 \text{ xor } \dots \text{ xor } w_{MSB}$. This is the only change needed. The update and estimate operations of the CMS remain the same. The main advantage compared to a parity protection is that we do not need to add a parity bit to the memory. Therefore, the protection scheme can be used for systems on which the memory is unprotected.

The effectiveness of the proposed scheme will depend on the distribution of the counter values and the CMS parameters. In the next section, this is evaluated using the same packet traces used to assess the impact of soft errors on the unprotected CMS.

Finally, an interesting observation that applies both to parity protection and the proposed scheme is that it may be counterproductive to protect the lower bits. For example, an error on the least significant bit can only introduce a deviation of one in the estimate. Instead if the bit is protected with parity, we would not use that counter and the estimate would be done based on the remaining $r - 1$ counters.

This would in most cases introduce a larger overestimation. However, a drawback of not protecting the lower bits is that there can be underestimation due to bit flips. In the following we focus on the case where we want to avoid underestimation and thus all bits are protected.

5 EVALUATION

In this section, the effectiveness of the proposed scheme is evaluated and compared to that of parity protection. To do so, the same simulations presented in section 3 have been run but on a parity protected CMS and a CMS protected with the proposed technique. The results obtained for the three traces considered are presented in Figures 11,12,13. The first (top) plot shows the overestimation for the parity protected CMS. The second plot shows the overestimation when the CMS is protected using the proposed scheme. Finally, the third (bottom) plot shows the underestimation when the CMS is protected using the proposed scheme. It is important to recall that for parity protection underestimation is not possible and thus no plot is needed. It can be seen that the overestimation is very similar for both schemes. As for underestimation, it can occur in the proposed scheme but with very low probability. In more detail for the SJ12 trace, all underestimation values are below 10^{-5} . For the MAWI trace the probabilities are below $2 \cdot 10^{-4}$ and for CHI15 below 10^{-4} . The overestimation values and probabilities are similar for both the proposed technique and parity protection. These results show that the proposed scheme achieves a similar protection to that of parity for overestimation and eliminates most of the underestimations leaving only some residual cases that occur with low probability. Therefore, the proposed scheme can be an interesting option to protect the CMS on systems in which the underlying memory is unprotected and adding parity is not an option.

Fig. 11. Percentage of bit flips that introduce a given deviation on the estimate of CMS protected with the proposed scheme and parity for the SJ12 trace with $k = 512k$, $r = 4$

Finally, it is of interest to discuss the protection of the lower bits. To that end, the deviations for an unprotected and a parity protected CMS are compared for different bits in Figures 14,15 for the SJ12 trace. It can be seen, that not

Fig. 12. Percentage of bit flips that introduce a given deviation on the estimate of CMS protected with the proposed scheme and parity for the MAWI trace with $k = 32k$, $r = 4$

Fig. 13. Percentage of bit flips that introduce a given deviation on the estimate of CMS protected with the proposed scheme and parity for the CHI15 trace with $k = 128k$, $r = 4$

protecting the lower bits would reduce the overestimation error as discussed in the previous section. For this trace, this occurs up to approximately bit six for which the benefit becomes small as seen in Figure 15. However, leaving those bits unprotected would lead to underestimation as seen in both Figures. Therefore, depending on the importance of over (under) estimations for a given application, it may be better to leave those bits unprotected or not.

6 PROTECTING AGAINST MULTIPLE BIT ERRORS

Although single bit errors are the most frequent soft error pattern, as technology scales, soft errors can affect more than one bit [10], [19]. In particular, adjacent bit errors can occur when radiation affects an area larger than a single memory cell [12]. In that case, both parity and the proposed protection are not effective when the number of bits flipped is even, as is the case in a double adjacent error. To protect

Fig. 14. Percentage of flips on bit four that introduce a given deviation on the estimate for the SJ12 trace with $k = 512k$, $r = 4$

Fig. 15. Percentage of flips on bit six that introduce a given deviation on the estimate for the SJ12 trace with $k = 512k$, $r = 4$

against such errors, interleaved parity bits can be used. For example, if a parity bit is used for the even bits and another for the odd bits, then a double adjacent error would produce a parity mismatch on both parity bits. Obviously, duplicating the parity bits implies an additional cost in terms of memory bits.

For the proposed scheme, the same idea can be used so that errors on even (odd) bits are propagated to the MSB (MSB-1). Then, half of the single bit errors would not flip the MSB (they would flip the MSB-1). This may reduce the effectiveness of the protection as the MSB-1 would be used more frequently than the MSB in most applications. To avoid this issue, the following encoding is proposed:

$$c_{MSB} = d_{MSB} \text{ xor } d_{MSB-2} \text{ xor } d_{MSB-3} \text{ xor } \dots d_0$$

$$c_{MSB-1} = d_{MSB-1} \text{ xor } d_{MSB-3} \text{ xor } d_{MSB-5} \text{ xor } \dots d_0$$

$$c_{rest} = d_{rest}$$

where d is the counter value to store and c are the bits written to the memory. The same equations using c on the right side are used to recover the counter value on the left side when reading from memory.

It can be seen that the MSB now stores the xor of all the bits except the MSB-1. Therefore, any single error in any of those bits would flip the MSB. Only a single bit on the MSB-1 would not flip the MSB. This ensures that protection is optimized for single bit errors that are usually the dominant error pattern. Instead, double adjacent bit errors would flip the MSB-1 that is the xor of the even bits (assuming counters have an even number of bits) providing also some protection.

The protection described has been implemented and tested on the same packet traces. Firstly, single bit errors were injected to check that the level of protection was similar to that of the original technique for single errors. Then double adjacent errors were inserted and the results compared to an unprotected implementation. Figures 16,17 show the results for the MAWI trace with no protection and the proposed protection respectively. It can be observed that the proposed scheme is able to drastically reduce the underestimation providing an efficient protection. Similar results were obtained for the other two traces.

Fig. 16. Percentage of double adjacent bit flips that introduce a given deviation on the estimate of an unprotected CMS for the MAWI trace with $k = 32k$, $r = 4$

7 CONCLUSIONS AND FUTURE WORK

In this paper, the effects of soft errors on the Count Min Sketch (CMS) have been considered. First their impact on the CMS frequency estimations has been evaluated using fault injection. Then based on the analysis of the results, the protection of the CMS has been considered and a new protection technique has been presented. The proposed scheme exploits the skew in the counter values that is found in many applications to propagate errors to the most significant bit. This increases the erroneous counter value thus removing it in most cases as candidate for estimation. Therefore, it has a similar effect as that of parity protection but does not require additional parity bits in the memory.

The proposed scheme has been tested to evaluate its effectiveness. The results show that it achieves a similar overestimation as parity protection while underestimation

Fig. 17. Percentage of double adjacent bit flips that introduce a given deviation on the estimate of CMS protected with the proposed scheme for adjacent errors for the MAWI trace with $k = 32k$, $r = 4$

is almost eliminated. Therefore, the proposed scheme can be of interest to protect CMS implemented on systems that use an unprotected memory. In the last part of the paper, an extension of the proposed scheme to deal with double adjacent errors was briefly discussed to show that protection against multiple bit errors can also be supported. Finally, as part of the evaluation, it is also shown that protecting the lower bits may be counterproductive. This is because the estimation deviation introduced by removing counters with an error on the lower bits may be larger than that of the error due to the bit flip. However, leaving bits unprotected would lead to underestimation for some errors. The optimization of the protection to cover only a fraction of the bits for a given application is left for future work to further improve the protection.

The analysis of the impact of soft errors and the protection schemes discussed in this paper can be potentially used in other sketches. For example, the Elastic sketch [20] is formed by a heavy part intended to keep track of heavy hitters and a light part that is a Count Min Sketch. Therefore, the ideas presented in the paper may be applicable to the light part of Elastic sketches. Instead, sketches like the Heavy Guardian [21] store all the information related to an element on a single bucket. Therefore, if that information is corrupted there is no alternative bucket from which to try to recover an estimate for that element and the proposed scheme is not applicable. Finally, to optimize the use of space it is possible to organize counters in a hierarchical way so that counters at the higher levels are shared by several lower level counters [22]. In this case, the proposed scheme may be used to protect the upper levels as the counters would tend to have low values while an alternative technique would be needed for the lower levels and control information used to link levels. More generally, studying the effects of soft errors on other sketches and designing efficient protection techniques for them is an interesting area for future work given the growing importance of sketches in computing and networking applications.

ACKNOWLEDGMENT

Pedro Reviriego was partially supported by the TEXEO project TEC2016-80339-R funded by the Spanish Ministry of Economy and Competitiveness and by the Madrid Community research project TAPIR-CM grant no. P2018/TCS-4496.

REFERENCES

- [1] G. Cormode and M. Hadjieleftheriou, "Methods for finding frequent items in data streams," *The VLDB Journal* 19, 1, February 2010.
- [2] G. Cormode, M. Garofalakis, P. J. Haas, and C. Jermaine, "Synopses for massive data: Samples, histograms, wavelets, sketches," *Foundations and Trends in Databases*, vol. 4, no. 1–3, pp. 1–294, 2012.
- [3] M. Charikar, K. Chen, and M. Farach-Colton, "Finding frequent items in data streams," in *Proceedings of the International Colloquium on Automata, Languages and Programming (ICALP)*, 2002.
- [4] G. Cormode and S. Muthukrishnan, "An Improved Data Stream Summary: The Count-Min Sketch and Its Applications," *Journal of Algorithms*, 2005.
- [5] G. Bianchi, K. Duffy, D. Leith and V. Shneer, "Modeling conservative updates in multi-hash approximate count sketches," in *Proceedings of the 24th International Teletraffic Congress (ITC 24)*, Krakow, 2012.
- [6] P. Reviriego, J. Martínez and S. Pontarelli, "Improving Packet Flow Counting with Fingerprint Counting" in *IEEE Communications Letters*, vol. 24, no. 1, pp. 76-80, Jan. 2020.
- [7] C. Estan and G. Varghese, "New directions in traffic measurement and accounting: Focusing on the elephants, ignoring the mice," *ACM Transactions on Computing Systems*, 21(3), 2003.
- [8] T. Yang et al. , "A Generic Technique for Sketches to Adapt to Different Counting Ranges," in *Proceedings of INFOCOM*, Paris, France, 2019, pp. 2017-2025.
- [9] N. Kanekawa, E. H. Ibe, T. Suga Y. and Uematsu, "Dependability in Electronic Systems," Springer, New York, 2011.
- [10] B. Narasimham, S. Gupta, D. Reed, J. K. Wang, N. Hendrickson and H. Taufique, "Scaling trends and bias dependence of the soft error rate of 16 nm and 7 nm FinFET SRAMs," *IEEE International Reliability Physics Symposium (IRPS)*, 2018.
- [11] C. L. Chen and M. Y. Hsiao, "Error-correcting codes for semiconductor memory applications: a state-of-the-art review," *IBM J. Res. Develop.*, vol. 28, no. 2, pp. 124-134, Mar. 1984.
- [12] K. Namba and F. Lombardi, "A Single and Adjacent Error Correction Code for Fast Decoding of Critical Bits," in *IEEE Transactions on Computers*, vol. 67, no. 10, pp. 1525-1531, 1 Oct. 2018.
- [13] K. Namba and F. Lombardi, "Parallel Decodable Multi-Level Unequal Burst Error Correcting Codes for Memories of Approximate Systems," in *IEEE Transactions on Computers*, vol. 65, no. 12, pp. 3794-3801, 1 Dec. 2016.
- [14] P. Reviriego, S. Pontarelli, J. A. Maestro and M. Ottavi, "A method to protect Bloom filters from soft errors," *IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFTS)*, 2015, pp. 80-84.
- [15] The CAIDA UCSD anonymized Internet traces - Chicago 2015-02-19. http://www.caida.org/data/passive/passive_2015_dataset.xml.
- [16] The CAIDA UCSD anonymized Internet traces - San Jose 2012-11-15. http://www.caida.org/data/passive/passive_2012_dataset.xml.
- [17] MAWILab traffic trace - 2015-07-20. <http://www.fukuda-lab.org/mawilab/v1.1/2015/07/20/20150720.html>.
- [18] J. Martínez, J.A. Maestro and P. Reviriego, "A Scheme to Improve the Intrinsic Error Detection of the Instruction Set Architecture," *IEEE Computer Architecture Letters*, vol. 16, no. 2, July-December 2017, pp. 103-106.
- [19] P. Reviriego and J.A. Maestro, "Study of the Effects of Multibit Error Correction Codes on the Reliability of Memories in the Presence of MBUs," *IEEE Transactions on Device and Materials Reliability*, vol. 9, no. 1, pp. 31-39, March 2009.
- [20] T. Yang et al., "Elastic sketch: Adaptive and fast network-wide measurements," in *Proceedings of SIGCOMM*, pp. 561-575, Aug. 2018.
- [21] T. Yang et al. "HeavyGuardian: Separate and Guard Hot Items in Data Streams," in *Proceedings of KDD*, 2018.
- [22] T. Yang et al. "Pyramid Sketch: a Sketch Framework for Frequency Estimation of Data Streams," *PVLDB*, pp. 1442-1453, 2017.