Received 28 November 2022, accepted 9 January 2023, date of publication 23 January 2023, date of current version 26 January 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3238899

RESEARCH ARTICLE

Alphabet Size Matching Techniques Based on Non-Binary Gilbert-Varshamov Bounded Limits for Synchronization Finite State Markov Channel

SHAMIN ACHARI AND LING CHENG, (Senior Member, IEEE)
School of Electrical and Information Engineering, University of the Witwatersrand, Johannesburg 2050, South Africa

Corresponding author: Ling Cheng (Ling.Cheng@wits.ac.za)

This work was supported in part by the National Research Foundation of South Africa under Grant 148765, Grant 132651, and Grant 129311.

ABSTRACT The Gilbert-Varshamov (GV) lower bound is used to provide indications and prescriptions for the outer code coding parameters for a memory synchronisation model that focuses solely on the internal resynchronisation process. The binary and q-ary GV bounds are utilised in this analysis to indicate parameters to remove the remaining substitution errors and provide a complete framework. Procedures and examples are provided to determine optimal outer code parameters for given inner-entropies and residual substitution errors produced during resynchronisation. In particular, using the non-binary GV bounds allows us to match the best alphabet size for given parameters. For the cases explored, a 16-ary GV bound provides the best results, with an (n, k, d) code of $(120, 57, 37)$ being a possible outer code when the inner entropy is 0.1. Using GV bounds for outer code parameter considerations frees the system from using stringent codes and instead allows any outer code to be utilised to meet the required error correction needs.

INDEX TERMS Alphabet size matching, channel bounds, Gilbert-Varshamov bounds, synchronisation finite-state Markov channel.

I. INTRODUCTION

Reliable communication is a necessity in today's world with its ever-increasing data transmission demands. Communication channels should be able to recover from errors produced during the transmission process easily. The fundamental principle in allowing communication systems to recover from errors is redundancy [1]. Adding more bits or symbols to a data stream based on predefined rules provides the power to detect and correct errors in the received sequence. It is easy to see that the more additional data is truncated onto the information, the more chance there is to recover from the erroneous transmission. These error correction capabilities, however, come with the cost of wasted transmission bandwidth and, consequently, inefficient use of the channel. This naturally leads to a fundamental problem in information theory where trade-offs between redundancy (and consequently error-correcting capabilities) are contrasted against the efficient use of the channel [1].

In Achari and Cheng [2], consider a channel model and decoding scheme to correct insertion and deletion errors in a system based on a Synchronisation Finite-State Markov Channel (S-FSMC) where Insertion-Deletion-Substitution (IDS) errors are present. The paper presented restricts its focus to the inner decoding of the communication system, where the purpose of the inner decoder is to regain synchronisation by removing the insertion and deletion errors detected. However, this resynchronisation process does not account for substitution errors caused by the channel or the incorrect decoding of the inner decoder. This paper aims to provide considerations and suggestions for the outer code coding parameters of the S-FSMC presented in [2] to ensure reliable communication and provide a framework for a complete system.

There are three main contributions to this paper. Firstly the binary GV lower bound is used to give indications on the parameters of the outer code to ensure reliable

The associate editor coordinating the review of this manuscript and approving it for publication was Abdullah Iliyasu.

communication for the S-FSMC. Secondly, these bounds are extended to q-ary symbols, which seek to reduce bursty errors' effects and provide a more efficient coding scheme. Lastly, using GV bounds to indicate parameters for the outer code frees the synchronisation decoder from stringent outer substitution codes such as LDPC. This allows any code construction and substitution Error Correction Codes (ECC) to be utilised as the outer code, so long as it satisfies the necessary communication requirements.

The rest of the paper is structured as follows. The black box inner process and basics of the GV bounds are outlined in Section II. The methodology and general case studies for both the binary and q-ary cases are described and discussed in Section III. Section IV then uses simulations of the inner decoder to give practical results and analysis of the described procedure. Conclusions are finally drawn in Section V.

II. BACKGROUND AND LITERATURE REVIEW

A. SYNCHRONISATION-FSMC (S-FSMC)

In general, research has either looked at synchronisation errors or memory effects of a channel in isolation. The modelling and analysis of synchronisation errors can be found in references such as [3], [4], [5], and [6] and more recently in [7] and [8]. Likewise, the modelling of memory within channels can be found in literature such as [9], [10], and [11] with an extensive review of these memory channels with error control techniques presented in [12].

In [2], Achari and Cheng have incorporated both of these channel effects and properties and describe a synchronisation channel based on a Finite-State Markov Channel (FSMC). FSMC - also referred to as discrete finite-state channels, are primarily based on Markov processes and was presented by Claude Shannon in his landmark paper in 1948 [13].

In addition to the channel model in [2], the paper provides an algorithm to regain synchronisation by removing the effects of insertions and deletions encountered during data transmission. However, this leaves substitution errors from the decoding process and inherent substitution errors from the channel itself. The work in [2] is based on the memoryless synchronisation channel, and watermark decoding scheme presented by Davey and MacKay in [5], which has been used in various fields such as DNA barcoding in medicine [14], [15] to data hiding in watermarking applications [16].

This paper treats the synchronisation channel and inner decoder workings as a black box illustrated in Figure 1. The inputs to this system will include a block-coded frame of length n. The outputs of this black box will be the Bit Error Rate (BER) or Symbol Error Rate (SER) for the q-ary case, which are the substitution errors that remain after the synchronisation process. Additionally, the corresponding entropy of the synchronisation channel, or inner channel entropy, denoted by H_{I}, is also an output of this black box system. The sole focus of this paper remains on the outer decoding process to ensure reliable communication for the given system.

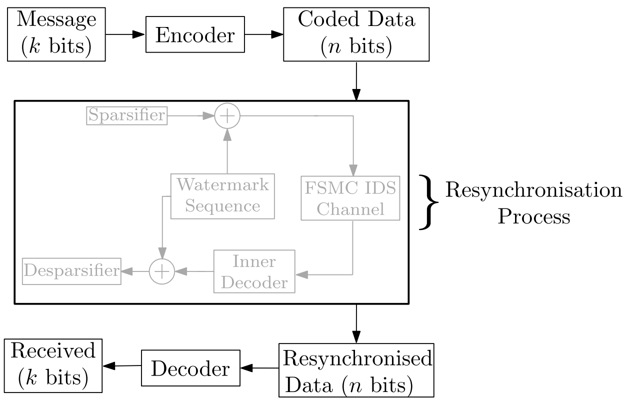

FIGURE 1. Black box approach for resynchronisation process.

B. GILBERT-VARSHAMOV (GV) BOUNDS

Numerous bounds are used to analyse channel models to determine the system performance, where the main trade-off of code rate and error-correction capabilities are explored. The best known and most notable lower bound is the GV bound described independently by Edgar Gilbert [17] and later by Rom Varshamov [18] where the bound was slightly improved [19]. As this is a lower bound, it is a positive result and, as such, provides a definite achievable bound [1], [20]. The GV bound states that for a q-ary code of length n and minimum Hamming distance d, the maximum number of codewords, $A_q(n, d)$ must satisfy Equation (1) [1], [21], [22].

$$A_q(n, d) \geq \frac{q^n}{V_q(n, d-1)} \tag{1}$$

Here $V_q(n, l)$ is the volume or number of strings in a Hamming ball of radius l and is equal to $\sum_{j=0}^{l} \binom{n}{j}(q-1)^j$ which gives rise to Equation (2) [1], [20], [21], [22].

$$A_q(n, d) \geq \frac{q^n}{\sum_{j=0}^{d-1} \binom{n}{j}(q-1)^j} \tag{2}$$

Using Stirling's approximation and simplifying, the GV bound can be written in its asymptotic form shown in Equation (3) where $R(\delta)$ is the corresponding code rate as a function of δ, H_q is the q-ary entropy function defined in Equation (4) [22] and, δ is the relative distance, or fractional minimum distance, which is equal to $\frac{d}{n}$ and is restricted to $0 \leq \delta \leq 1 - \frac{1}{q}$ [1], [20], [21], [22]. It is worth noting that Equation (4) is derived by making use of the change of base formula, and as such, any logarithm base may be used so long as it is consistent throughout.

$$R(\delta) \geq 1 - H_q(\delta) \tag{3}$$

$$H_q(x) = x\frac{\log(q-1)}{\log(q)} - x\frac{\log(x)}{\log(q)} - (1-x)\frac{\log(1-x)}{\log(q)} \tag{4}$$

Formally, the GV Bound is defined in Theorem 1 [1], [21], [22].

Theorem 1 (GV Bound): Let $q \geq 2$. Then for any $0 \leq \delta \leq 1 - \frac{1}{q}$, and any $0 < \epsilon \leq 1 - H_q(\delta)$, there exists a q-ary family

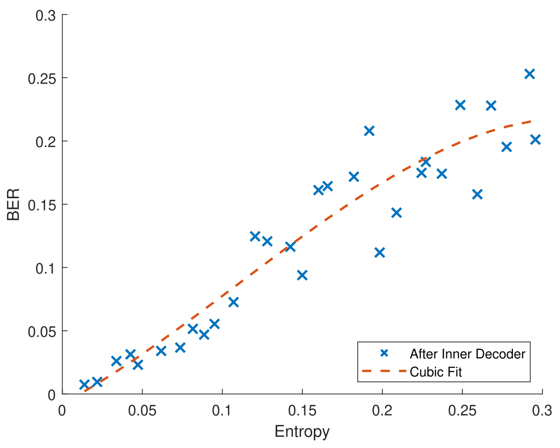

FIGURE 2. BER after resynchronisation process for various H_I values.

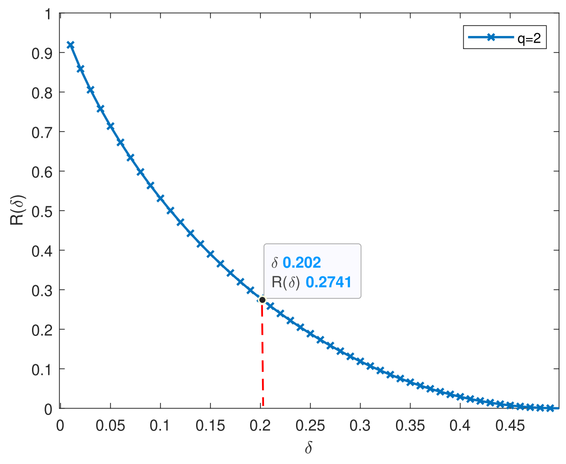

FIGURE 3. Binary GV bound with parameters for given case study.

of codes C where:

$$\delta(C) \geq \delta$$

and

$$R(C) \geq 1 - H_q(\delta) - \epsilon$$

III. OUTER CODE PROCEDURE

A. METHOD

As this paper employs a black box approach for the inner decoder, the only information needed to determine the outer decoder parameters is the residual SER and the corresponding inner entropy (entropy of the S-FSMC).

Figure 2 recreates the results from the inner synchronisation process in [2], and in particular, the corresponding BER for various inner entropy values are plotted. From Figure 2, the residual BER and, consequently, the number of substitution errors remaining after resynchronisation can easily be identified. Thus a corresponding outer code can be prescribed.

It is well known in coding theory that the maximum number of errors a block code can correct, denoted by t, is dependent on d, and this inequality is described in Equation (5). From this, it is easily shown that the required minimum distance of our prescribed code should be at least $2t + 1$.

$$t \leq \lfloor \frac{d-1}{2} \rfloor \tag{5}$$

Consequently, Equation (6) shows the corresponding relative distance that should be chosen given the number of errors we wish to correct.

$$\delta \geq \frac{2t+1}{n} \tag{6}$$

B. BINARY CASE STUDY

We use an example that utilises the data reproduced in Figure 2 to solidify the procedure outlined. For example, if the entropy of the S-FSMC system, H_I, is 0.1, then the average residual BER is at worst 0.1. Using a frame length or coded data size of 480 bits (as in the case of the parameters used in [2]) implies there will be an average of roughly 48 substitution errors that the outer code should try to correct. Using Equation 6 implies that our outer code should have a relative distance of at least 0.2021. Assuming a normal binary data sequence allows the use of the binary entropy function and produces Figure 3, which shows the relationship between code rate, $R(\delta)$, and relative distance for the binary GV bound. As is indicated by the highlighted data point and red line in Figure 3, a maximum outer code rate, R_o, of 0.2741 is achievable for the given relative distance. Additionally, any point below the curve and to the right of the red dotted line would have the required error correction capabilities for the given parameters. However, to ensure the best code rate for the given scenario, it is recommended to stay as close to the curve and the required δ line as possible. Using the maximum R_o of 0.2741, the number of message bits can be calculated by $k = \lfloor R_o \times n \rfloor$. This gives a value of 131 message bits. As this number is rounded down to ensure whole bits are used, the actual R_o is recalculated to be 0.2729. Thus providing possible parameters in terms of an (n, k, d) code, we have $(480, 131, 97)$. Lastly, the overall code rate, R_T, is determined by multiplying the inner code rate and outer code rate together. In this case, the R_i is solely dependent on the sparsifier used during the resynchronisation process, and in the simulations in [2], a 4 to 5 sparsifier is used, making $R_i = 0.8$. Therefore the corresponding R_T in this example is approximately 0.2183. Again, this is not the only possible set of parameters that can be used to correct the given errors for this example.

C. EXTENSION TO q-ARY

In the previous example, the scenario was limited to that of the binary case. However, the remaining errors produced during the inner resynchronisation process may likely occur in bursts. This is expected as the channel model used during the inner portion is a FSMC which has correlated errors. Since

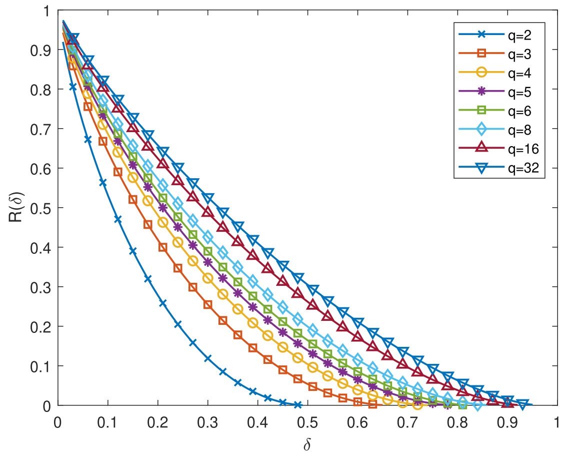

FIGURE 4. GV bound for various *q*-ary values.

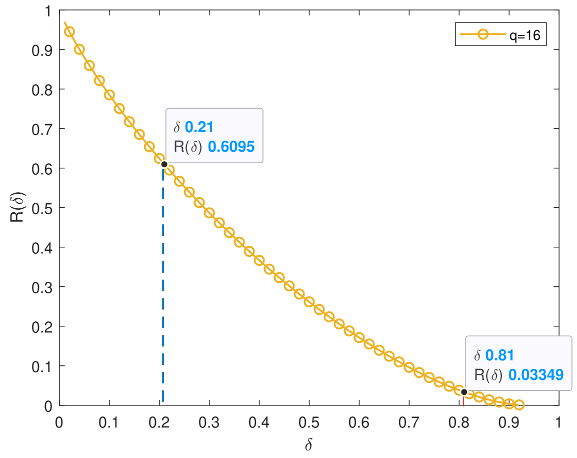

FIGURE 5. 16-ary GV bound with parameters for given case study where red line indicates worst-case and blue indicates best-case scenarios.

the errors may be clumped together in bursts, extending the outer code to a q-ary GV function would be more effective. Figure 4 shows the GV bounds for different q values. Not all q values will be directly beneficial to our application, and only values of $q = 2^b$, where b is a positive integer, will be utilised; however, q values that fall outside of this constraint are shown for completeness. It is easily seen from Figure 4 that higher code rates are achievable for corresponding δ values as larger q values are utilised. In other words, we can obtain similar or better error-correction capabilities by utilising symbols compromised of more bits while having a more efficient code rate. This is common knowledge in the field, as now the focus is on symbol-level data communication, and as such, we can correct up to t' symbols and consequently $t' \times \log_2(q)$ bits.

D. Q-ARY CASE STUDY

Using the values from the binary example, the idea of utilising q-ary codes is further illustrated. We use k' to denote the symbol message length and n' to denote the symbol block length, which is further quantified in Equation (7). This gives rise to $n' = 120$ symbols for a value of $q = 16$ as we still require 480 coded bits at the input of the inner resynchronisation process for the scenario described. Suppose 48 substitution errors remain after resynchronisation; as a worst-case, this will affect 48 separate symbols once the bits are converted to their corresponding symbols. This, in turn, using the symbol equivalent values in Equation (6), suggests that the outer code's relative distance should now be approximately 0.808 as we have $t' = 48$ symbols to correct. This relates to approximately $R_o = 0.0335$ as shown by the red dotted line in Figure 5. At first glance, this may seem like an inferior result when compared with the binary case, as a lower code rate is achieved. However, it is crucial to keep in mind that this is the worst-case scenario, and when looking at the number of bits that can be corrected, the given code can correct up to 192 bits. On the opposite end of the spectrum, a best-case scenario is when all the bit errors occur in one large burst. In this scenario, that would mean 12 symbols overall would be affected, and as such, we would only require a relative distance of approximately 0.208. This corresponds to $R_o = 0.61$ and is indicated by the blue dotted line in Figure 5. The procedure follows as in the binary case study to find the overall code parameters. This gives us a worst-case code of (120, 4, 97) with a corresponding R_o of 0.033 and R_T of 0.027, respectively. The best-case scenario gives a (120, 73, 25) code where the R_o is 0.608, and R_T is 0.487.

$$n' = \frac{n}{log_2(q)} \tag{7}$$

IV. RESULTS AND ANALYSIS

A. RESULTS FROM SIMULATED CHANNEL

This Section runs simulations using the entire process, including inner synchronisation. Consequently, the practicality of the method discussed is illustrated by using different q-ary divisions. Figure 6 shows the SER obtained after the inner resynchronisation process when various q-ary values are employed. As seen in this figure, while all the plots follow a similar trend, at higher H_{I} values, there is generally a higher chance of errors. Additionally, as the q value increases, the higher the SER becomes. This is especially true at inner entropies exceeding 0.1, where the higher the entropy of the S-FSMC, the more evident the difference in SER obtained for corresponding q-ary groupings. This intuitively makes sense as more errors are expected at higher entropies due to more uncertainty within the channel. The various drops in the plot occurring at approximately $H_{\text{I}} = 0.1$ and $H_{\text{I}} = 0.2$ are due to the method used to generate the different inner channels and are again further detailed in [2]. It is evident from these plots that the inner system does cause more sporadic errors as the SER increases with an increase in q size. If the results obtained saw an equal SER with an increasing size of q, this would suggest the errors affect consecutive bits and thus be more situated towards the best-case scenario described in the method.

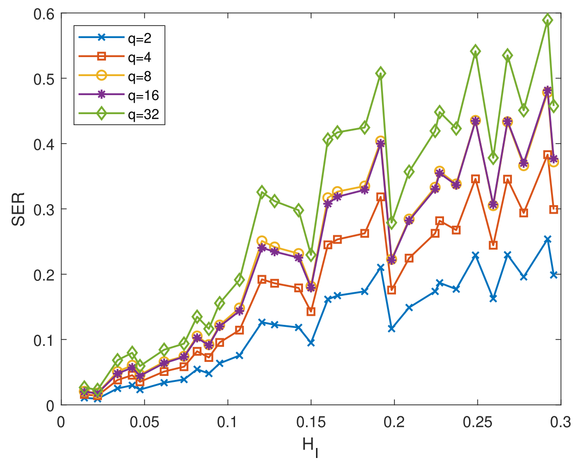

FIGURE 6. SER at the output of inner decoder for various q-ary values across various H_I.

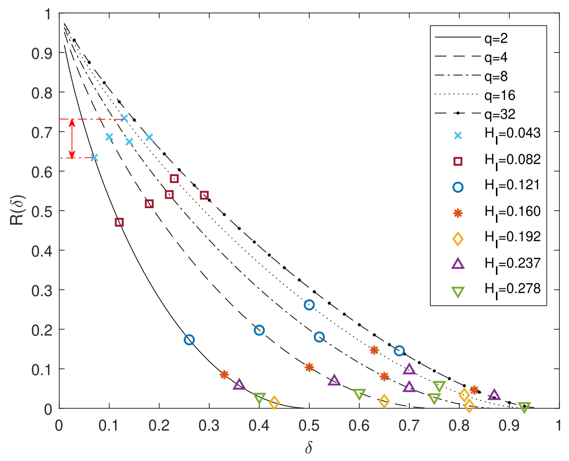

FIGURE 8. Code rate and gain obtained for various q-values.

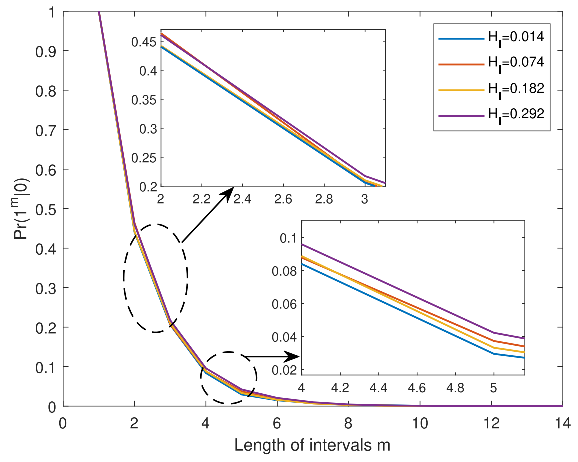

FIGURE 7. Error-run plots for various H_I values after resynchronisation.

It is also worth noting that the plots for $q = 8$ and $q = 16$ in Figure 6 are reasonably similar in SER performance which further suggests that it is unlikely to have groupings of more than three or four consecutive error bits. This is explored further by looking at the error-run plots shown in Figure 7. Here $Pr(1^m|0)$ is the probability of having at least m errors after an error-free communication state. It is noticed that in the case of $H_I = 0.014$, the probability of at least four consecutive errors is 0.08, which drops to approximately 0.03 for at least five consecutive errors, reiterating the notion of having a low chance of getting more than four consecutive error bits. This is confirmed at higher entropy values too. In the case of $H_I = 0.292$, the probability of at least four consecutive errors is 0.096, the probability of at least five consecutive errors is approximately 0.04, and the probability of six consecutive errors drops to 0.02, again reemphasising this idea. It is worth remarking that the error-run plots are all based on a binary case ($q = 2$), as this provides the consecutive raw errors before any symbol segmentation.

Table 1 shows the coding gain achieved at various inner entropy values for corresponding q-ary values. The values of $R(\delta)$ are chosen by looking at the corresponding minimum δ value rounded to the nearest hundredth. The values obtained show that using a q value of 16 provides the best results in terms of code rate (outer and total) for the given channel simulations. This appears optimal for the given channel as increasing to $q = 32$ starts to incur a decreasing performance in code gain. The potential reason a 16−ary code performs the best in these tests could be attributed to using a 4-5 sparsifier during the resynchronisation process. Intuitively, the number of bits used at the input of the sparsifier can be seen as the bits that constitute a single symbol at the outer q-ary code. As such, matching these values inherently makes sense as errors are contained and thus decoded within each individual symbol. Choosing an outer code such that the makeup of each symbol exceeds the number of bits at the sparsifier input potentially allows errors to affect more symbols during resynchronization, thus producing a poorer error rate performance. Definite indications of the effect of the sparsifier on the outer code require further research. Figure 8 illustrates a graphical representation of Table 1 where the red dotted lines and arrows show the code gain between $q = 2$ and $q = 16$ for an H_I of 0.043. In this figure, the corresponding symbols show the code gain for a given H_I using the various q-ary GV bounds. Again, it is fairly evident for respective values of H_I that using $q = 16$ provides the best code rate for the given error correction requirements.

B. SIMULATED CHANNEL CASE STUDY

To further illustrate the method described, the use of the simulation results is explored in order to give a more real-world case study. Again, using $H_I = 0.1$ in this example, the SER, no matter the number of bits composing a symbol, will always be less than 0.2 according to Figure 6. As previously mentioned, using a q value of 16 would be the best option for the described channel, and thus symbols consisting of 4 bits are opted for in this example. Not forgetting the

TABLE 1. Comparison of code rates and code gain using different *q* values for a given inner entropy.

	q	**SER**	**Minimum** t	**Minimum** δ	$R(\delta)$	**Outer Percentage Code Rate Gain Over** $q=2$	R_T	**Overall Percentage Code Rate Gain Over** $q=2$
H_I = **0.043**	2	0.03	15	0.0646	0.6341	0	0.50728	0
	4	0.045	11	0.0959	0.6863	0.0522	0.54904	0.04176
	8	0.06	10	0.1313	0.6742	0.0401	0.53936	0.03208
	16	0.057	7	0.125	**0.7337**	**0.0996**	**0.58696**	**0.07968**
	32	0.08	8	0.1771	0.6856	0.0515	0.54848	0.0412
H_I = **0.082**	2	0.054	26	0.1105	0.4706	0	0.37648	0
	4	0.082	20	0.1709	0.5173	0.0467	0.41384	0.03736
	8	0.105	17	0.2188	0.5407	0.0701	0.43256	0.05608
	16	0.103	13	0.225	**0.5809**	**0.1103**	**0.46472**	**0.08824**
	32	0.135	13	0.2813	0.5389	0.0683	0.43112	0.05464
H_I = **0.121**	2	0.126	61	0.2563	0.1733	0	0.13864	0
	4	0.192	47	0.3959	0.1975	0.0242	0.158	0.01936
	8	0.251	41	0.5188	0.1804	0.0071	0.14432	0.00568
	16	0.24	29	0.4917	**0.2616**	**0.0883**	**0.20928**	**0.07064**
	32	0.326	32	0.6771	0.1454	-0.0279	0.11632	-0.02232
H_I = **0.160**	2	0.162	78	0.3271	0.08507	0	0.068056	0
	4	0.245	59	0.4959	0.1038	0.01873	0.08304	0.014984
	8	0.317	51	0.6438	0.08038	-0.00469	0.064304	-0.003752
	16	0.308	37	0.625	**0.147**	**0.06193**	**0.1176**	**0.049544**
	32	0.406	39	0.823	0.04606	-0.03901	0.036848	-0.031208
H_I = **0.192**	2	0.21	101	0.423	0.01418	0	0.011344	0
	4	0.318	77	0.6459	0.01785	0.00367	0.01428	0.002936
	8	0.404	65	0.8188	0.005964	-0.008216	0.0047712	-0.0065728
	16	0.4	48	0.8084	**0.03349**	**0.01931**	**0.026792**	**0.015448**
	32	0.508	49	N/A	N/A	N/A	N/A	N/A
H_I = **0.237**	2	0.177	85	0.3563	0.05732	0	0.045856	0
	4	0.268	65	0.5459	0.06775	0.01043	0.0542	0.008344
	8	0.338	55	0.6938	0.05119	-0.00613	0.040952	-0.004904
	16	0.336	41	0.6917	**0.09597**	**0.03865**	**0.076776**	**0.03092**
	32	0.423	41	0.8646	0.03103	-0.02629	0.024824	-0.021032
H_I = **0.278**	2	0.196	95	0.398	0.02905	0	0.02324	0
	4	0.294	71	0.5959	0.03904	0.00999	0.031232	0.007992
	8	0.366	59	0.7438	0.02774	-0.00131	0.022192	-0.001048
	16	0.37	45	0.7584	**0.05893**	**0.02988**	**0.047144**	**0.023904**
	32	0.451	44	0.9271	0.005335	-0.023715	0.004268	-0.018972

multitude of 16-ary code constructions available in practice makes this option the easy choice. Again, from Figure 6, using a 16-ary value/code at an entropy of 0.1 for the scenario gives a SER of approximately 0.1437. Overcompensating, an SER of 0.15 can be coded for, which means 18 (four-bit) symbol errors can be corrected, giving rise to $\delta = 0.308$ and consequently gives $R_o = 0.4739$ and $k' = 57$. This gives an overall rate of approximately 0.38 and a code of (120, 57, 37). While not the absolute best case for the 16-ary instance, these parameters tend more towards the blue dotted line in Figure 5 and again shows that the simulated channel benefits from using the q-ary case over standard binary.

C. SIMULATED CHANNEL WITH REED-SOLOMON OUTER CODING

Finally, the complete framework, including a choice of outer code, is simulated to show the application of the methods discussed. In this case, the Reed–Solomon (RS) code is chosen due to its multiple use cases in practical systems, especially in applications where burst errors are prevalent. Again various q-ary segmentations are tested, corresponding to the various RS parameters outlined in Table 2. As RS codes are non-binary, here N corresponds to the symbol codeword length or block length, and K is the length of the symbol-wise message. A RS coding scheme can correct up to $t' = \frac{N-K}{2}$ symbols.

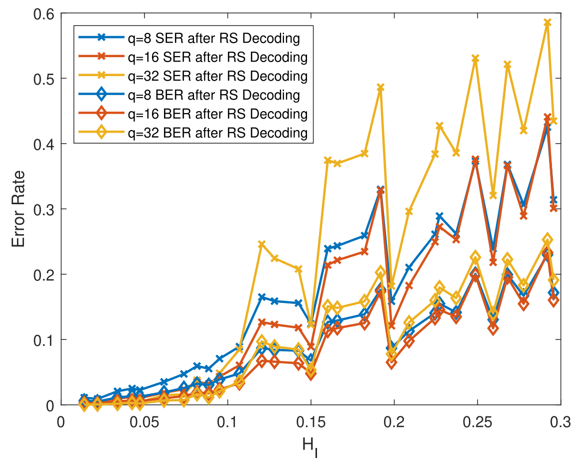

FIGURE 9. SER and corresponding BER after RS decoding.

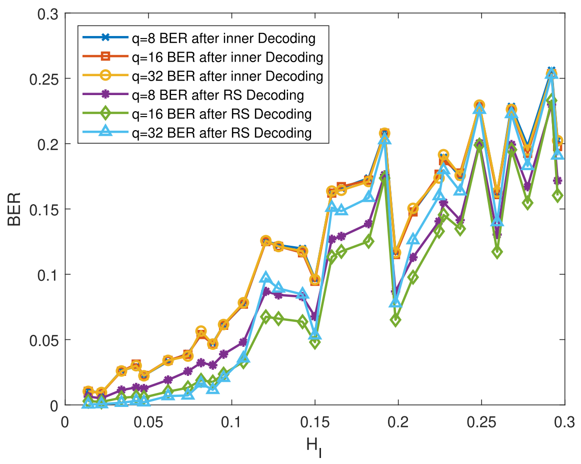

FIGURE 10. BER after resynchronisation process and corresponding BER after RS decoding.

To keep a fair comparison, we resume from the previous section and choose the RS parameters in such a way as to keep R_o as close to 0.4739. The number of blocks sent, n_b, is chosen to ensure that the inner resynchronisation process receives 480 bits (padding may be required to get exactly 480 bits). For the given case, $n_b = \lfloor \frac{480}{q \times N} \rfloor$. Figure 9 shows the SER and corresponding BER after the RS decoding. The BER is calculated by converting its symbol data stream into its binary counterpart based on the number of bits for that specific q-ary code. As can be seen from the figure, the $q = 16$ and $q = 32$ have relatively similar performances in lower H_I values. The $q = 16$ coding scheme shows substantial coding gains after an inner entropy of around 0.1. Figure 10 highlights these effects further by plotting both the BER at the output of the inner decoder (after the resynchronisation process) and finally the BER after the RS decoding. It is clearly seen that the BER after resynchronisation is relatively the same no matter the segmentation of bits that compromise a symbol, again reiterating the usefulness of the black box approach. Again at lower inner entropy values, the capabilities of the $q = 16$ and $q = 32$ cases have very similar results, with the $q = 32$ case having slightly better performance. After an inner entropy of around 0.1, which corresponds to BER values of above 0.1 after resynchronisation, we see substantial performance improvements for the 16-ary coding scheme. Again, this agrees with intuition as the parameters in this example application were chosen for these corresponding regions of interest.

TABLE 2. Parameters used for RS outer code.

q	N	K	t'	d	n_b	R_o	R_T
8	7	3	2	5	22	0.4286	0.3429
16	15	7	4	9	8	0.4667	0.3733
32	31	15	8	17	3	0.4839	0.3871

V. CONCLUSION

The S-FSMC is presented as a black box implementation along with the theory of GV bounds. From this, a method is presented to deduce the relevant substitution outer code parameters that will, in essence, correct the remaining substitution errors produced from the synchronisation channel and the resynchronisation process. Firstly, indications are given based on a binary setting, but it is shown that depending on the distribution of errors, the system may benefit from using the q-ary GV bounds. This, however, depends on the structure of the errors and in fact, the system may indeed suffer from utilising the q-ary counterpart if the errors produced are spread out in the data frame. From simulations of the synchronisation channel, it is shown that using a q-ary value of 16 will see the best results in terms of coding gain. Increasing higher than $q = 16$ starts to provide diminishing results for the given channel. The simulated channel case study shows that the system benefits from using the q-ary GV bounds, and a good indication of the outer code parameters for an inner entropy of 0.1 would be (120, 57, 37) for a 16-ary partition. The proposed use of GV bounds for outer code parameter considerations liberates the coding construction and decoder from specialised outer error correction codes. It thus allows any code to be implemented as long as the coding requirements and parameters are met.

REFERENCES

[1] V. Guruswami, A. Rudra, and M. Sudan. (2012). *Essential Coding Theory*. [Online]. Available: https://cse.buffalo.edu/faculty/atri/courses/coding-theory/book/

[2] S. Achari and L. Cheng, "Watermark-based code construction for finite-state Markov channel with synchronisation errors," 2022, *arXiv:2207.10204*.

[3] R. G. Gallager, "Sequential decoding for binary channels with noise and synchronization errors," Massachusetts Inst. Tech. Lexington Lincoln Lab., Lexington, MA, USA, Tech. Rep., Oct. 1961.

[4] K. Zigangirov, "Sequential decoding for a binary channel with drop-outs and insertions," *Probl. Peredachi Inf.*, vol. 5, no. 2, pp. 23–30, 1969.

[5] M. C. Davey and D. J. C. MacKay, "Reliable communication over channels with insertions, deletions, and substitutions," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 687–698, Feb. 2001.

[6] M. Mitzenmacher, ''A survey of results for deletion channels and related synchronization channels,'' *Probab. Surv.*, vol. 6, pp. 1–33, Jan. 2009.
[7] M. Cheraghchi and J. Ribeiro, ''An overview of capacity results for synchronization channels,'' *IEEE Trans. Inf. Theory*, vol. 67, no. 6, pp. 3207–3232, Jun. 2021.
[8] B. Haeupler and A. Shahrasbi, ''Synchronization strings and codes for insertions and deletions—A survey,'' *IEEE Trans. Inf. Theory*, vol. 67, no. 6, pp. 3190–3206, Jun. 2021.
[9] E. N. Gilbert, ''Capacity of a burst-noise channel,'' *Bell Syst. Tech. J.*, vol. 39, no. 5, pp. 1253–1265, 1960.
[10] E. O. Elliott, ''Estimates of error rates for codes on burst-noise channels,'' *Bell Syst. Tech. J.*, vol. 42, no. 5, pp. 1977–1997, Sep. 1963.
[11] B. Fritchman, ''A binary channel characterization using partitioned Markov chains,'' *IEEE Trans. Inf. Theory*, vol. IT-13, no. 2, pp. 221–227, Apr. 1967.
[12] L. N. Kanal and A. R. K. Sastry, ''Models for channels with memory and their applications to error control,'' *Proc. IEEE*, vol. 66, no. 7, pp. 724–744, Jul. 1978.
[13] C. E. Shannon, ''A mathematical theory of communication,'' *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, Jul./Oct. 1948.
[14] D. Kracht and S. Schober, ''Using the Davey-MacKay code construction for barcodes in DNA sequencing,'' in *Proc. 8th Int. Symp. Turbo Codes Iterative Inf. Process. (ISTC)*, Aug. 2014, pp. 142–146.
[15] D. Kracht and S. Schober, ''Insertion and deletion correcting DNA barcodes based on watermarks,'' *BMC Bioinf.*, vol. 16, no. 1, pp. 1–14, Dec. 2015.
[16] D. J. Coumou and G. Sharma, ''Insertion, deletion codes with feature-based embedding: A new paradigm for watermark synchronization with applications to speech watermarking,'' *IEEE Trans. Inf. Forensics Security*, vol. 3, no. 2, pp. 153–165, Jun. 2008.
[17] E. N. Gilbert, ''A comparison of signalling alphabets,'' *Bell Syst. Tech. J.*, vol. 31, no. 3, pp. 504–522, May 1952.
[18] R. R. Varshamov, ''Estimate of the number of signals in error correcting codes,'' *Doklady Akademii Nauk SSSR*, vol. 117, no. 5, pp. 739–741, 1957.
[19] P. Gaborit and G. Zemor, ''Asymptotic improvement of the Gilbert–Varshamov bound for linear codes,'' *IEEE Trans. Inf. Theory*, vol. 54, no. 9, pp. 3865–3872, Sep. 2008.
[20] A. Mazumdar. *Coding Theory and Applications*. Accessed: Jan. 20, 2023. [Online]. Available: https://people.cs.umass.edu/~arya/courses/690T/CS690T.html
[21] T. K. Moon, *Error Correction Coding: Mathematical Methods and Algorithms*. Hoboken, NJ, USA: Wiley, 2020.
[22] J. H. Van Lint, *Introduction to Coding Theory*. vol. 86. Cham, Switzerland: Springer, 2012.

SHAMIN ACHARI received the B.Sc. degree in electrical and information engineering from the University of the Witwatersrand (WITS), Johannesburg, South Africa, in 2015, where he is currently pursuing the Ph.D. degree in electrical engineering. His M.Sc. degree in visible light communication with special focus on error correction schemes for visible light systems, which was subsequently upgraded to a Ph.D. which mainly focuses on channel modeling and methods of correcting synchronization errors. His research interests include visible light communications, machine learning, and the Internet of Things (IoT).

LING CHENG (Senior Member, IEEE) received the B.Eng. degree (cum laude) in electronics and information from the Huazhong University of Science and Technology (HUST), in 1995, the M.Ing. degree (cum laude) in electrical and electronics, in 2005, and the D.Ing. degree in electrical and electronics from the University of Johannesburg (UJ), in 2011. In 2010, he joined the University of the Witwatersrand, where he was promoted to a Full Professor, in 2019. He has been a visiting professor at five universities and the principal advisor for over forty full research post-graduate students. He has published more than 100 research papers in journals and conference proceedings. His research interests include telecommunications and artificial intelligence. He was awarded the Chancellor's medals, in 2005 and 2019; and the National Research Foundation rating, in 2014. The IEEE ISPLC 2015 Best Student Paper Award was made to his Ph.D. student at Austin. He is the Vice-Chair of IEEE South African Information Theory Chapter. He serves as an associate editor of three journals.