CHAPTER 5

DIGITAL QURAN MODEL IMPLEMENTATION AND EVALUATION

5.1 Introduction

This chapter focuses on developing the DQM prototype as a proof of concept to demonstrate the concept of a lightweight digital Quran.

5.2 DQM Design and Development

This section discusses the implementation process from the first step until evaluation; developing a DQM prototype involves combining and chaining many components, and not all components will need to be executed on every system run. Instances containing the Quranic letters and words must be represented in Hexadecimal, and fined the weight of each word; basically, the implementation process of the proposed system consists of four major processes, which are:

- Letters (Harf) Conversion based on its Hexadecimal
- Words and Verses Conversion
- Sparse Matrix with Look Up Table for unique Quran words
- Compressed Sparse Matric with Double Off-Set Indexing.

UNIVERSITI SAINS ISLAM MALAYSIA
ISLAMIC SCIENCE UNIVERSITY OF MALAYSIA

The process of achieving a feasible solution for Quranic word representation in hexadecimal format using UTF-8 Arabic Presentation Forms A and B Unicode Standard 7.0 for Arabic characters encoding for phase 1 and 2 of the methodology schemas for Surah Al-Baqarah, and surah Al-Fatihah were discussed in chapter 3 and 4 in detail. Figure 5.1 shows the DQM general process flow.

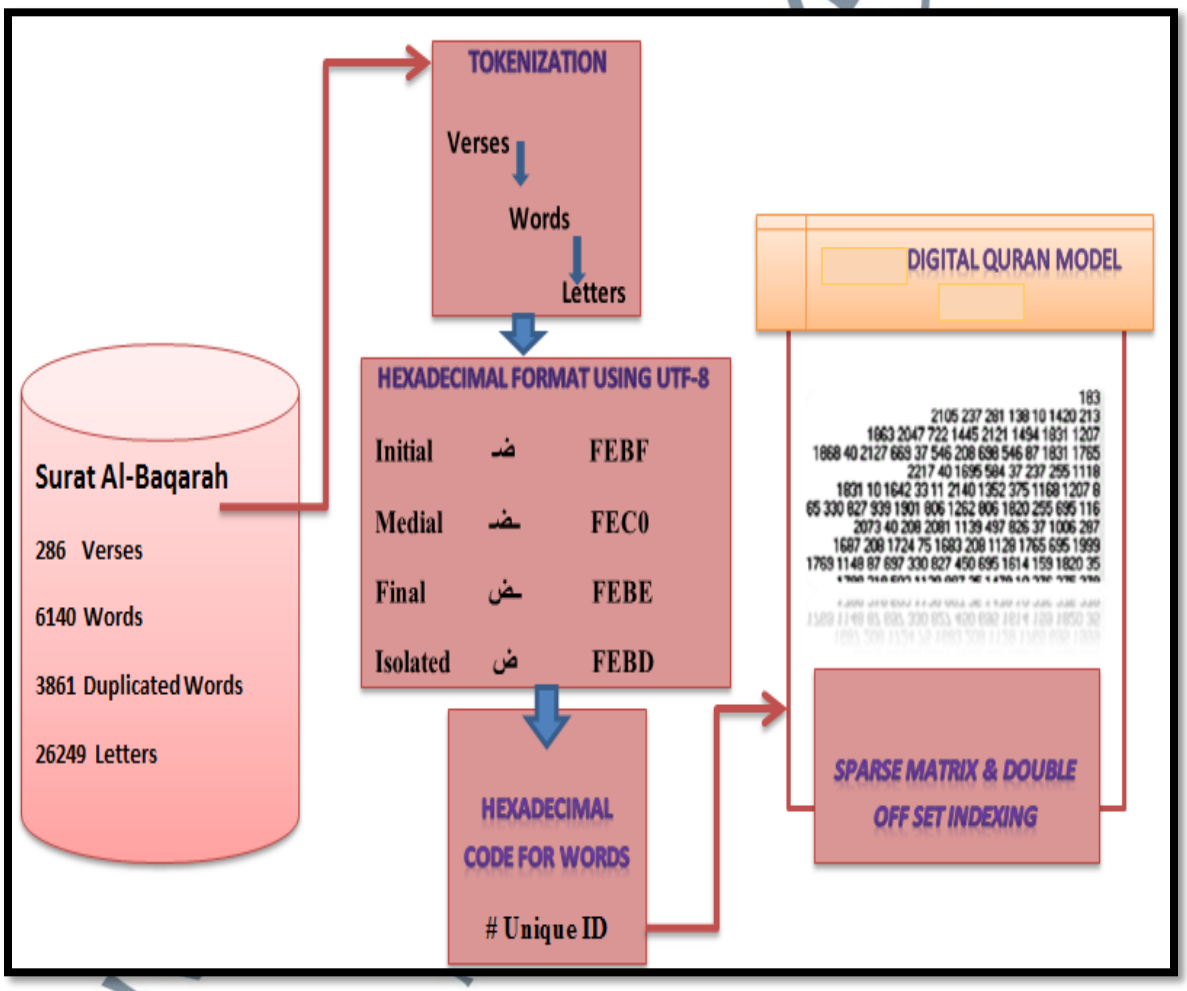


Figure 5.1: DQM General Process Flow

The programming language used in the implementation is Java and Visual Basic 6.0; Java language has been chosen because it is a powerful programming language for implementing the required application.

5.2.1 DQM Prototype Development

These sections discuss some key principles for constructing information structures, such as sparse matrix and double offset indexing, and discuss primitive implementation in Java servers and VB. Such structures provide a foundation for understanding algorithm design considerations that play a central role in computer science, some of which will be presented in later chapters. Matrixes data structure or two-dimension array is one of the critical components of structural computing. The other component is double offset indexing to efficiently represent the matrix with non-zero elements to save empty spaces.

In Java, the class concept is used as an extension of this notion, in the sense that a class provides methods for accessing the data and representing the data internally. Classes and matrixes are the critical building blocks for constructing various data structures. Further discussion of the object concept and its uses appears in the following section. Figure 5.2 below shows the procedure flow in the implemented DQM.

UNIVERSITI SAINS ISLAM MALAYSIA
ISLAMIC SCIENCE UNIVERSITY OF MALAYSIA

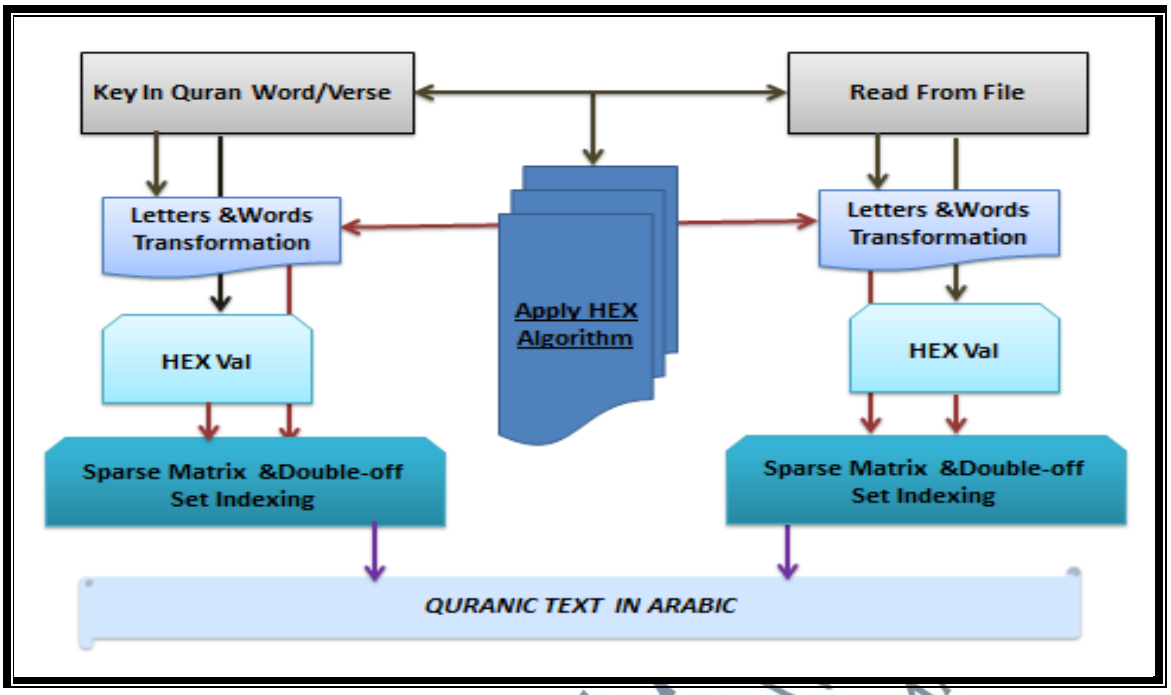


Figure 5.2: DQM Implementation from Arabic Text Format to Hexadecimal Representation

5.2.2 Implementation Tools

In this research, a Digital Quran Model has been described, which aims at providing a conceptual implementation for building Digital Quran-related applications. The proposed approach includes designing and developing the application programming interface (API) for all necessary components for searching, reading, annotating, and building mobile applications and social networks around the central theme of the Digital Quran. A salient feature of the proposed API is the ability to represent and retrieve Quranic verses and words in Arabic with word indexing and storage optimization that also maintains content integrity.

Four key algorithms were developed and used in the new DQM, which are:

a. Quranic words Conversion from Characters (Harf) Conversion based on UTF-8 Arabic Presentation Forms A and B Unicode Standard 7.0 for Arabic characters encoding standers.

b. Quranic verses conversion into series of corresponding words in Hexadecimal.

c. Construction of sparse matrix based on impressive Quran words indexing list with a lookup table of a unique ID.

d. Construction of a compressed sparse matrix with a double offset indexing compression algorithm.

Thus, this study allows flexibility to extend the implementation using any other Arabic words and the whole Holy Quran, as shown in Figure 5.3, which illustrates an architecture for the DQM functionalities with the fourfundamentaly mechanisms to optimize space.

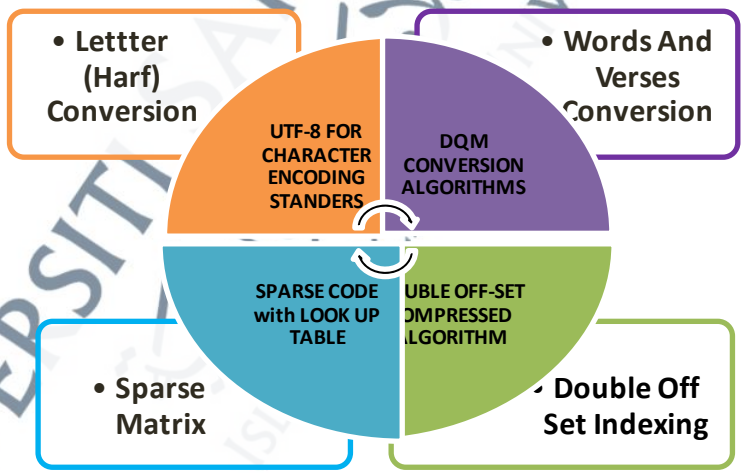


Figure 5.3: DQM Implementation Key Mechanism

5.2.3 Coding DQM

For many years, computers used only the 26 Latin alphabet letters in their English version. The ASCII coding system, for example, did not recognise even the accents in French. After that, the ASCII coding system was extended to cope with Arabic and all European languages. Incompatibilities among keyboards followed: incompatibilities among page code,; problems of electronic transmission of texts, etc. Until now, pure ASCII is still the unique universal standard for coding texts. The lack of a normalized and complete coding system that includes all known alphabets is one of the main problems that face computer software in general. The norm Unicode has been built to answer this question. Unicode is a system on 16-bit; it allows the coding of 65536 different characters (Wesley, 1991). Therefore, it allows typesetting in several languages and multi-language modes.

An 8-bit code table consists of 256 positions arranged in 16 columns and 16 rows. The columns and the rows are numbered 00 to 15. The columns and rows are numbered 0 to F in hexadecimal notation. Notations of the form identify the code table positions xx/yy where xx is the column number and yy is the row number. The value of each byte indicates its UTF-8 function as follows:

- 00 to 7F hex (0 to 127): first and only byte of a sequence.
- 80 to BF hex (128 to 191): continuing byte in a multi-byte sequence.
- C2 to DF hex (194 to 223): first byte of a two-byte sequence.
- E0 to EF hex (224 to 239): first byte of a three-byte sequence.

- F0 to FF hex (240 to 255): first byte of a four-byte sequence.

UTF-8 remains a simple, single-byte, ASCII-compatible encoding method if no characters more significant than 127 are directly present. This means that an HTML document technically declared to be encoded as UTF-8 can remain a regular single-byte ASCII file. The document can remain even though it may contain Unicode characters above 127 if all characters above 127 are referred to indirectly by ampersand entities.

Having 28 alphabets, Arabic is semantic. The orientation in writing is done from the right side towards the left. Moreover, Arabic is the mixed language used during official occasions by the United Nations. The DQM provides the foundation for creating a representation of any word and verse in Surah Al-Baqarah and Surah Al-Fatihah using UTF-8. It is commonly used in HTML and similar protocols to convert all Unicode characters to variable data lengths. Unicode characters corresponding to the famous ASCII group have the same values, and Unicode characters converted to UTF-8 can be used with most software.

5.2.4 Word and Verses Conversion

The implementation of the words and verse conversion algorithm for DQM as a proof of concept has been successfully employed to reduce memory space of the Digital Holy Quran with the intelligent model through handling word duplication mechanism whilst maintaining content integrity.

In addition to the formulation mentioned in Eq 3.1 – 3.3 previously, the letter position on each word must consider either of the four locations: initial, medial, final and isolated. The following example shows the word (قدير) QADEER in hexadecimal. According to formulas 3.1 and 3.2, it will be calculated as the following W (قدير) = L (ـقـ + ـدـ + ـيـ + ـر)

$$= L\ (FED7 + FEAA + FEF3 + FEAE)$$

$$= 3\ FB23$$

This model is helpful in optimizing memory space an,d thus, increasing the speed for searching such as in-system programming, text mining, Quran memorizing and interpretation (Mazlan et al., 2018). The findings of this research are in consensus with existing literature and scholarly findings. Figure 5.4 illustrates the anatomy for (قدير) QADEER word.

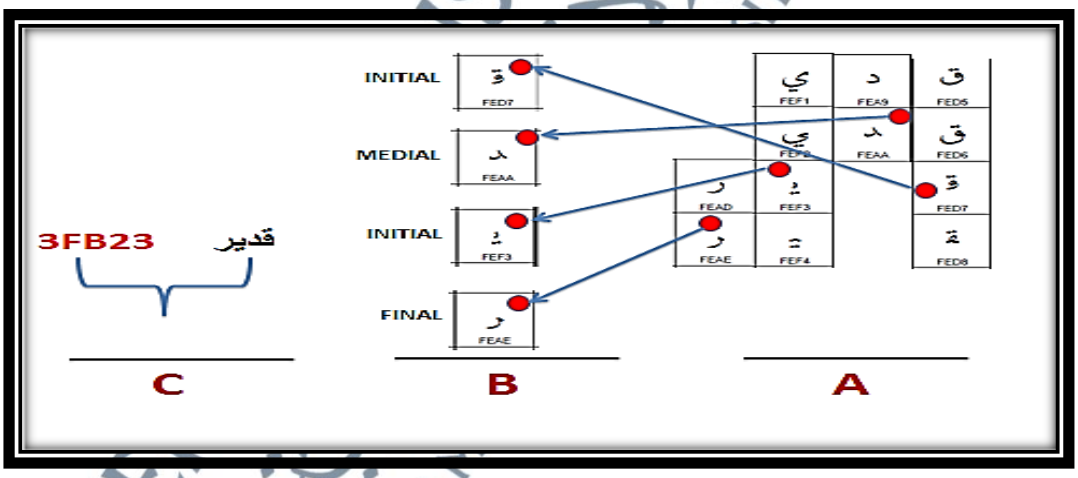


Figure 5.4: Anatomy Design for (قدير) QADEER Word

Figure 5.5 is the screenshot from the proof-of-concept implementation of the same word. Hence the result of the new calculated hexadecimal representation is 3FB23 which yields

about 37.33% of reduction that has 45 occurrences in the Quran as Equation 4.1 page 87. This is an important matter for the current research as it develops on the recent findings from similar studies (e.g. Mazlan et al., 2018; Hakak et al., 2019; Almazrooie et al., 2018, 2020).

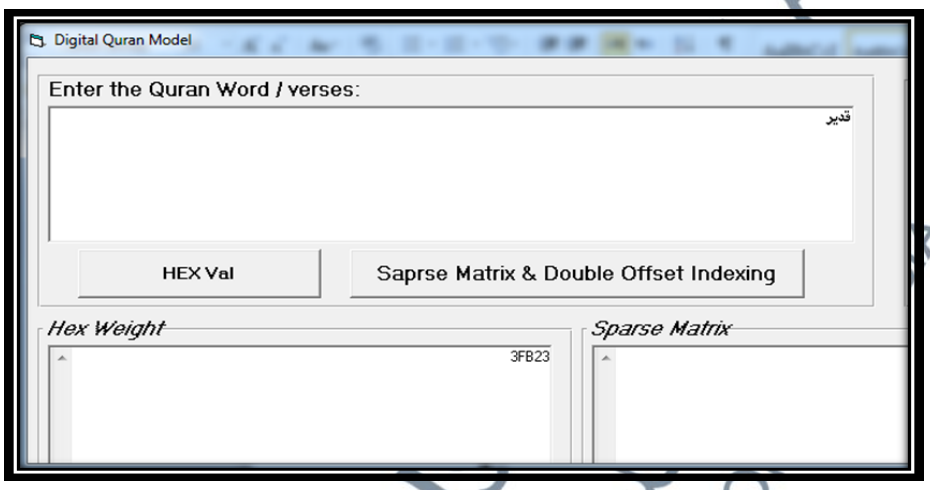


Figure 5.5: Implementation for (قدير) QADEER Word

Figure 5.6 shows a screenshot of Java program code for surah Al-Fatihah.

```
public class DataSetCode {
    private HashMap <String, String> DataSet;

    public HashMap <String, String> insertData ()
    {
        DataSet = new HashMap<>();

        DataSet.put("3", "يمج الر.ثمج الر.ه الل.مهب");
        DataSet.put("4", "دمج ال");
        DataSet.put("2", "دلل.");
        DataSet.put("5", "بر");
        DataSet.put("6", "يثم الع ال");

        DataSet.put("7", "ثمج الر");
        DataSet.put("8", "يمج الر.");
```

Figure 5.6: Program Code Represent Surah Al-Fatihah (الفاتحة) in Hexadecimal

Figure 5.7 depict the flow chart that represents hexadecimal code of input file of the hash map, using data set and displays of the output file for Surah Al-Fatihah (الفاتحة) which contains 7 verses.

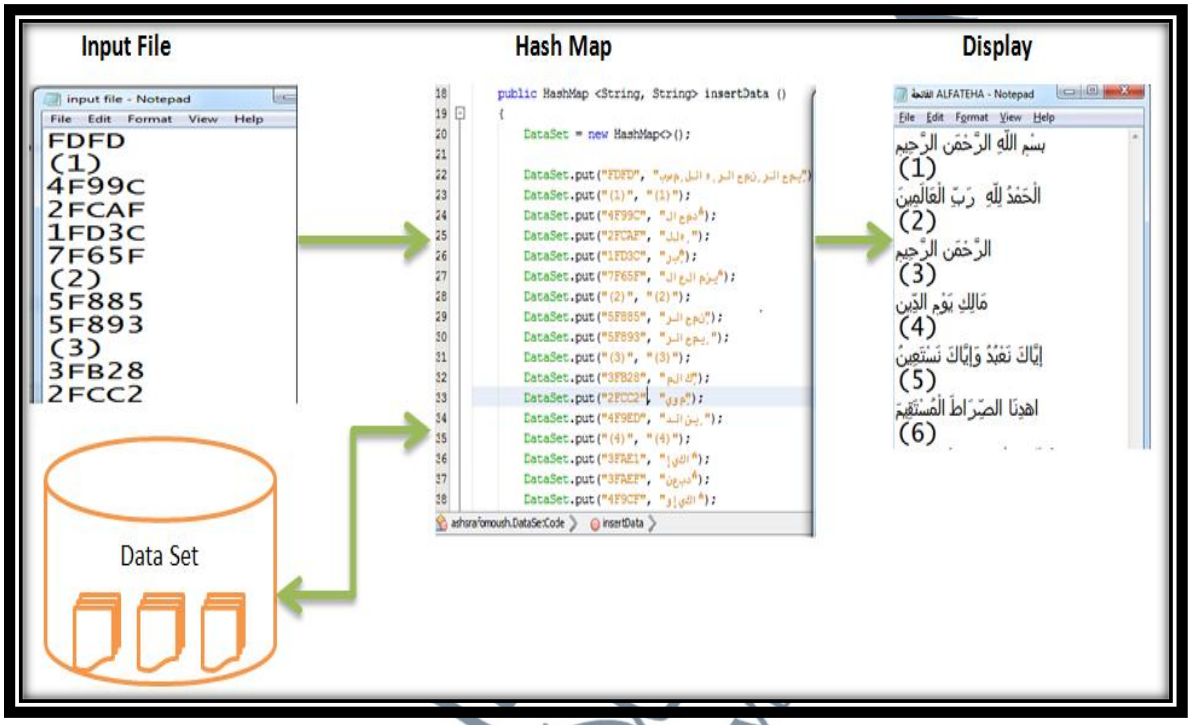


Figure 5.7: Flow Representing the Code, Input File, Hash Map, Display and Data Set

Each code in the input file represents one word for example. For verse 1 (بسم الله الرحمن الرحيم) has a standard code in the UTF 8 equal to FDFD in hexadecimal representation. In Figure 5.7, surah Al-Fatihah is displayed in hexadecimal of its equivalent of the Arabic text.

The second verse which contains 4 words (الحمد لله رب العالمين) and each word respectively represented with the formulation in Eq 3.1. Thus, in hexadecimal according

to the Unicode Arabic presentation standard are 4F99C, 2FCAF, 1FD3C, 7F65F respectively. Thus, the word (الحمد) AHAMDU according to formulation calculated as the following:

$$W(\text{الحمد}) = L(\text{ا} + \text{ـلـ} + \text{ـحـ} + \text{ـمـ} + \text{د})$$
$$= L\,(FE8D + FEDD + FEA4 + FEF4 + FEAA)$$
$$= 4F99C$$

This representation reduces the memory space up to 48.7% for **الحمد لله رب العالمين** in surah Al-Fatihah after applying the hexadecimal representation algorithm as shown in Table 5.1.

Table 5.1: Comparison of Memory Size Arabic Verse and Its Hexadecimal Representation

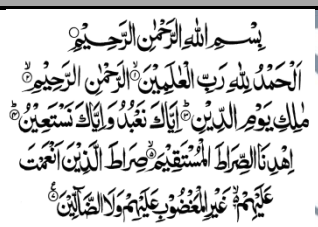
	No. of Characters	**Size in Bytes**	**Word in Hexadecimal**	**No. of Words**	**Size in Bytes**
الحمد	5	10	4F99C	1	5
لله	3	6	2FCAF	1	5
رب	2	4	1FD3C	1	5
العالمين	8	16	7F65F	1	5
الحمد لله رب العالمين	21	**39**	*Total*	4	**20**
REDUCTION	(39-20)/ 39 % = 48.7%				

***1 character = 2 Bytes**

Various types of digital Quran expounded such as image-based, audio, video and text-based format. Table 5.2 shows the display of Arab characters of surah Al-Fatihah in

various formats, taking advantage of hexadecimal representation to reduce the storage space after text transformation into hexadecimal representation.

Table 5.2: Comparison of Various Digital Representations for Surah Al-Fatihah

Measurements	Al-FatihahFormat	Memory Size
Image		37.4 KB
Video	https://www.youtube.com/watch?v=WsT1tL1fogk	1.63 MB
Audio	https://www.youtube.com/watch?v=0DrQPfsaaYg	725 KB
Text	بسم الله الرحمن الرحيم الحمد لله رب العالمين الرحمن الرحيم مالك يوم الدين إياك نعبد وإياك نستعين اهدنا الصراط المستقيم صراط الذين أنعمت عليهم غير المغضوب عليهم ولا الضالين	159 Bytes
Hex	FDFD F99F FCA9 FD3C F667 F889 F897 FB2B FCC2 F9F3 FAE2 FAEF F9D0 F8D9 F999 F826 F64D FAB8 F9F3 F9B1 FA6D FC71 F75E FA6D FDE9 F775	81 Bytes

5.3 Storage Optimization with Compressed Sparse Matrix and Handling Duplications

A sparse matrix is used to represent all words and verses in hexadecimal form with a unique ID that represents all words and unique words of the two surah's. Figure 5.8 shows the representation of words in Hexadecimal and verses in sparse matrix. In this study, Surah Al-Fatihah and Al-Baqarah were used as test cases.

Duplications were handled by creating an index listing of all unique words in the Quran. A word representation lookup table was created with three columns which are the Hexadecimal code of the word, the Arabic text and the unique ID. Verses will be rewrite and represented using the unique ID in the form of a matrix as shown in Fig. 5.8.

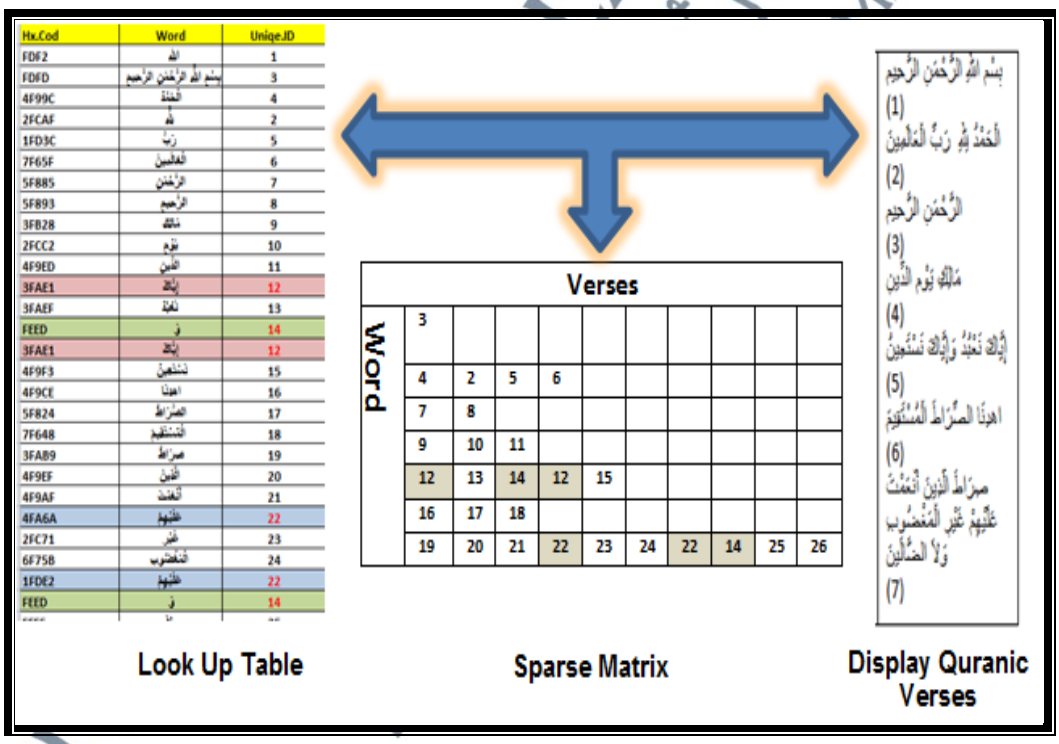
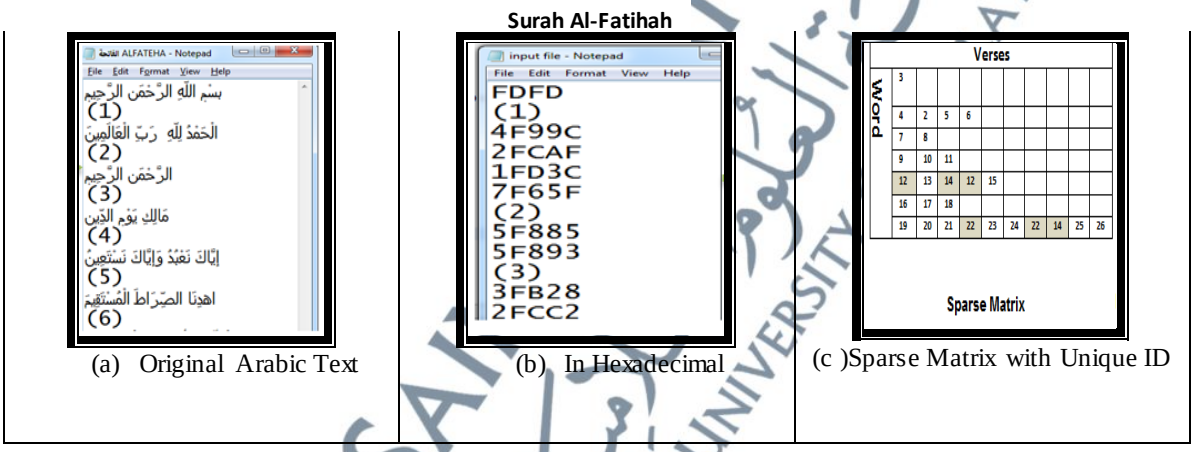


Figure 5.8: Quran Representation with Unique ID In A Sparse Matrix for Surah Al-Fatihah

The storage of words is being optimized due to the use of one memory space for that word instead of one memory space for each Arabic character in the particular word. For example, الحمد has 5 letters meaning 2 Bytes x 5 letters = 10 B memory space where each letter is stored in 2 bytes (16 bits). Its conversion to hexadecimal $4F99C$ requires 5 bytes. In the case of surah Al-Fatihah, after applying hexadecimal representation was 49.05%. However, with a sparse matrix, the reduction in the storage size is 54.08%. Figure 5.9 shows the comparison between the different representations of Surah Al-Fatihah.



(a) Original Arabic Text (b) In Hexadecimal (c)Sparse Matrix with Unique ID

Figure 5.9: Comparison of Al-Fatihah Representation

In this stage, the implementation of the algorithm achieved the expected results which were evaluated based on Holy Quran words and verses of Surah Al-Fatihah (Adhoni & Siddiqi, 2013) and Surah Al-Baqarah as a real test case.

Due to the vast number of words and repeated words, the proposed technique represents words of the Quran using Unicode by calculating the hexadecimal

representation of each word. Word conversion ID is based on letters conversion Unicode standards. This hexadecimal representation will reduce the searching speed since it is in the presentation layer similar to the operating system and able to handle repeated words and verses by manipulating indexes of unique Quran words. This is in contrast with AlMazrooei et. al (2020) and Hakak et al. (2017) that concatenation the hexadecimal of each letter thus form a string of concatenated hexadecimal characters. It does not optimize space, nor does it handle repeating words of the Quran.

The solution for any repeated word is through the mechanism where word is given a unique ID; as a result, there is no way for duplication. This conversion is represented on a Lookup Table after calculating the word representation that hold three columns; the first column contains the word in Arabic or Kalimah, the second column contains a counter that counts the number of each repeated word, the third column contains hexadecimal conversion for each word and the last column contains an ID which refers to the position of the word to eliminate a repeated word which saves memory. Table 5.3 represents the most frequent words in surah Al-Baqarah with their IDs,

Table 5.3: The Most Frequently Words in Surah Al-Baqarah with Their ID's.

Kalimah	Count	New Hex	ID
يقاتلونكم	9	8F507	2260
فللوالدين	9	8F570	2261
وللكافرين	9	8F571	2262
ولنبلونكم	9	F5BA8	2263
فسيكفيكهم	9	8 F5C9	2264
فليستجيبوا	10	9 F394	2265
وبالوالدين	10	9 F3C9	2266

Sparse Matrix will be constructed containing the encoded hexadecimal form of Arabic characters for surah Al-Baqarah and Surah Al-Fatiha. Table 5.4 below displays the longest ayat (129 words) in the Holy Quran which is Al Baqarah verse 282 with the sparse metric code for each word and verse.

Table 5.4: Sparse Code for the Longest Ayah in the Holy Quran (Al-Baqarah, 282)

Longest Ayah on the Holy Quran (Al-Baqarah, 282)	**Sparse Code using Unique ID of Quran Words**
يا أيها الذين آمنوا إذا تداينتم بدين إلى أجل مسمى فاكتبوه	18 1562 1207 1128 45 1963 595 203 82 779 2009
وليكتب بينكم كاتب بالعدل ولا يأب كاتب أن يكتب كما علمه الله	1740 1311 353 1493 202 92 353 549 182 788 695
فليكتب وليملل الذي عليه الحق وليتق الله ربه ولا يبخس منه	1709 1865 581 808 515 1316 695 133 202 446 282
شيئا فإن كان الذي عليه الحق سفيها أو ضعيفا أو لا يستطيع أن	403 171 187 581 808 515 1216 9 1106 9 10 1536
يمل هو فليملل وليه بالعدل واستشهدوا شهيدين من رجالكم فإن	289 43 1862 832 1493 2239 1806 37 1591 171 33
لم يكونا رجلين فرجل وامرأتان ممن ترضون من الشهداء أن	1321 1238 567 2139 277 1098 37 1893 139 1887
تضل إحداهما فتذكر إحداهما الأخرى ولا يأب الشهداء إذا ما	1029 1887 1451 202 92 1893 45 13 467 202 1444
دعوا ولا تسأموا أن تكتبوه صغيرا أو كبيرا إلى أجله ذلكم أقسط	1580 945 9 1021 203 513738 418 162 695 1279
عند الله وأقوم للشهادة وأدنى ألا ترتابوا إلا أن تكون تجارة	1974 1197 2268 1876 75 734 838 845 2171 1311
حاضرة تديرونها بينكم فليس عليكم جناح ألا تكتبوها وأشهدوا	772 1345 385 2268 1943 1976 45 1913 202 471 353
إذا تبايعتم ولا يضار كاتب ولا شهيد وإن تفعلوا فإنه فسوق بكم	202 646 201 1621 674 751 186 1558 695 2113 695
واتقوا الله ويعلمكم الله والله بكل شيء عليم	1296 181 128 804

UNIVERSITI SAINS ISLAM MALAYSIA
ISLAMIC SCIENCE UNIVERSITY OF MALAYSIA

5.4 Letters and Words Conversion for Surah Al-Baqarah

Letters, words, and verses conversion for surah Al-Baqarah using hexadecimal representation is computed according to Eq 3.1and 3.2. For example, the verse in Surah al Baqarah (**وبالوالدين, فليستجيبوا**) which are repeated ten times and the size of these words before applying the words transformation algorithm is 20 bytes for both. Next, after applying the transformation of the word into hexadecimal, the size is 7 bytes as in Figure 5.10. The sizes in terms of occurrence in Surah al Baqarah which is ten times imply 200 bytes of the original Arabic text and 70 bytes in hexadecimal respectively which can save up to 65% of memory space.

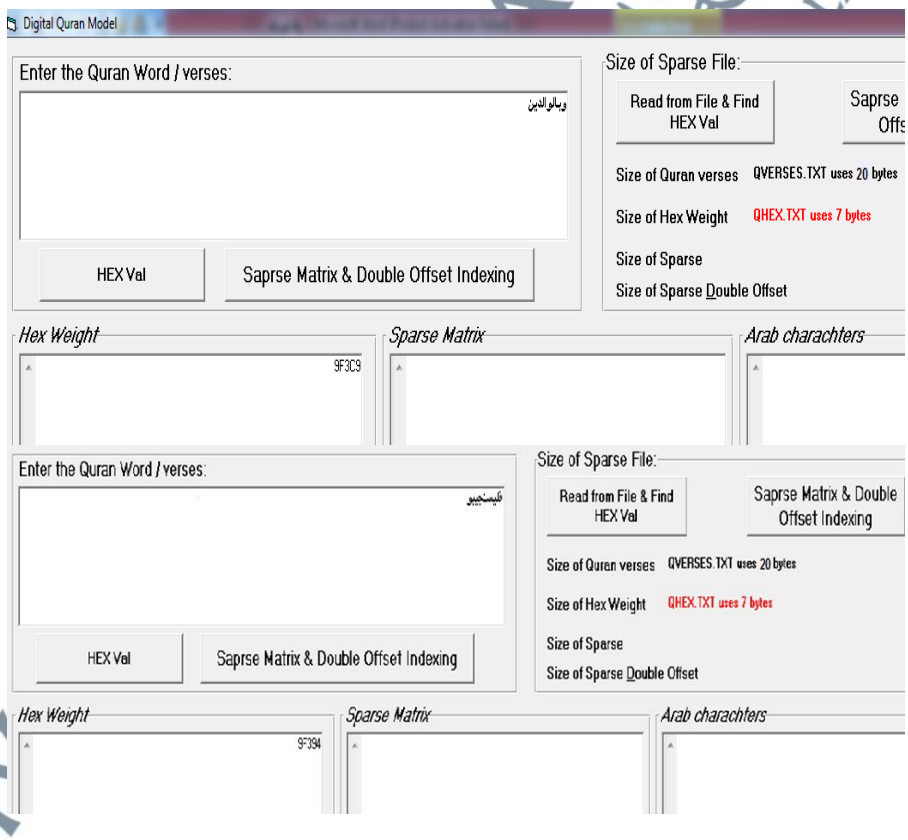


Figure 5.10: Screen Shot of Words Conversion (**وبالوالدين, فليستجيبوا**)

5.5 Surah Al-Fatihah Implementation and Performance Evaluation

Word conversion algorithm applied to the Quranic word in Arabic text represented in hexadecimal. The Quran Parsing module as in Figure 5.11 represents the use of a scientific calculator in hexadecimal word format.

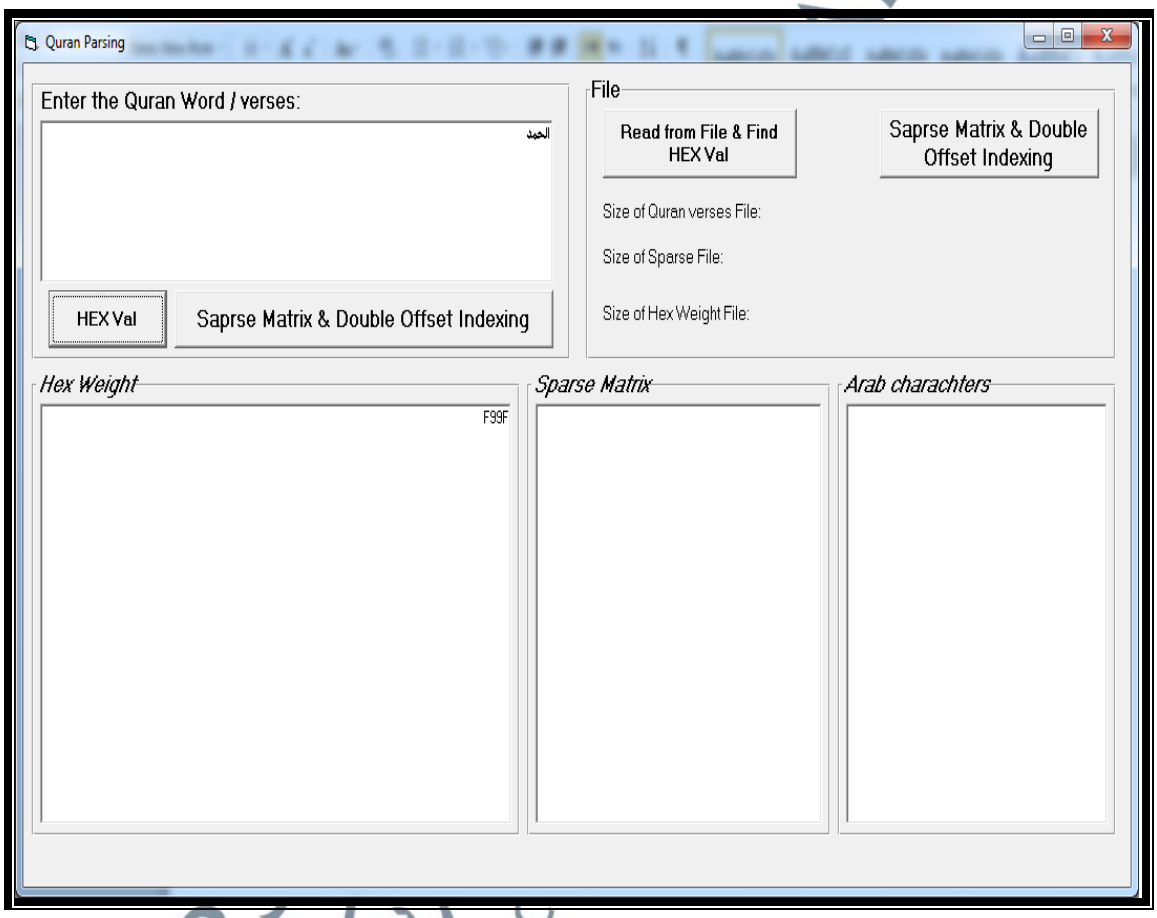


Figure 5.11: Quran Parsing Program to Calculate Words in Hexadecimal Space Size

The aim of the evaluation process is the solution quality, which compares the size of the file before applying the Digital Quran algorithm.

The first verse in the Al-Quran بسم الله الرحمن الرحيم repeated 114. So, if we compare the size of the verse (بسم الله الرحمن الرحيم) will be 23 bytes instead of 4 bytes hexadecimal representation. Imagine the reduction when repeated 114, the reduction will be 2622 bytes compared to 456 bytes after applying the conversion algorithm which equals 82.60%. Table 5.5 presents the results of the comparatives sizes for the mentioned words and verses

Table 5.5: Comparatives Results for Words Before and After Applying Algorithm

Word/Verse	Number of Letters	Size of Arabic Text (Bytes)	Size of Hex Text (Bytes)	Occurrence in Quran (Freq)	Total Size Reduction (%)
الـرحـيـم	6	12	5	148	58.33%
قدير	4	7	5	45	37.50%
بسم الله الرحمن الرحيم	19	23	4	114	82.60%
فليستجيبوا	10	20	7	10	66.0%
وبالوالدين	10	20	7	10	66.0%

As for Surah Al-Fatiha, the first chapter of the Holy Quran consists of 143 letters, 31 words, 7 verses and 17 unique words. Table 5.6 presents the results after the conversion to hexadecimal representation with a 49.05% storage reduction.

Table 5.6: Summary of the Comparison Words and Verses Conversion

SURAH	Number of Letters	Number of Words	Size of Arabic Text (Bytes)	Size of Hex Text (Bytes)	Total Size Reduction (%)
Surah Al-Fatiha	143	29	159	81	49.05%

Next, evaluation of the sparse matrix technique, Table 5.7 shows how a sparse matrix with word index reduces the files sizes for surah Al-Fatihah. The size of the Quran verse file is 159 bytes, and the size of the sparse file is 73 bytes with a 54.08% reduction in the storage size. The table shows the comparison between the file of the Arabic text, hexadecimal and sparse file representation for Al-Fatihah.

Table 5.7: Sparse Matrix Technique to Reduce Storage Surah Al-Fatihah

SURAH	Number of Letters	Number of Words	Size of Arabic Text (Bytes)	Size of Hex Text (Bytes)	Size of Sparse File (Bytes)	Total Size Reduction (%)
SURAH AL-FATIHA	143	29	159	81	73	54.08%

Next, evaluation of compressed sparse matrix with double offset indexing technique. Table 5.8 shows how the compressed sparse matrix reduces the storage for surah Al-Fatihah.

Table 5.8: Sparse Matrix Technique with **Double**-off Set Indexing Compression Algorithm

SURAH	Number of Words	Size of Arabic Text (Bytes)	Size of Hex Text (Bytes)	Size of Sparse File (Bytes)	Size of Sparse File & Double-off Set Indexing (Bytes)	Total Size Reduction (%)
Surah Al-Fatiha	**29**	**159**	**81**	**73**	**48**	**69.81%**

The size of the Quran verse file is 159 bytes and the size of the compressed sparse matrix with double offset indexing file is 48 bytes with a 69.81% reduction in the storage size. The table shows the comparison between the file size of the Arabic text, hexadecimal representation, sparse matrix representation and sparse matrix with double offset indexing file for Al-Fatihah.

5.6 Surah Al-Baqarah Implementation and Performance Evaluation

Similar process with Surah Al-Baqarah, conversion to hexadecimal form with a unique ID for all unique words in the surah, DQM algorithm with a sparse matrix to

represent each word and verses in Surah Al-Baqarah. Repeated words will be replaced with their unique ID as a reference thus reducing the usage of memory.

Figure 5.12 displays the screenshot of the prototype of DQM which shows the three approaches of surah Al-Baqarah representation in Arabic text format, hexadecimal format, and sparse matrix format.

Figure 5.12: Surah Al-Baqarah Representation

Table 5.9 presents the results of the first level of comparison and the result was obtained using the algorithm to convert Arabic text to hexadecimal representation which was $\frac{(33199-30652)}{33199} = 7.67\%$, reduce storage.

Table 5.9: Conversion of Arabic Word into Hexadecimal Form

SURAH	Number of Letters	Number of Words	Size of Arabic Text (Bytes)	Size of Hex Text (Bytes)	Total Size Reduction (%)
Surah Al-Baqarah	26249	6140	٣٣١٩٩	٣٠٦٥٢	07.67%

Figure 5.13 shows the comparison between the file size of the Arabic text and hexadecimal representation for surah Al-Baqarah, which was 33.199 MB for the text file and 30.625 MB for the hexadecimal file.

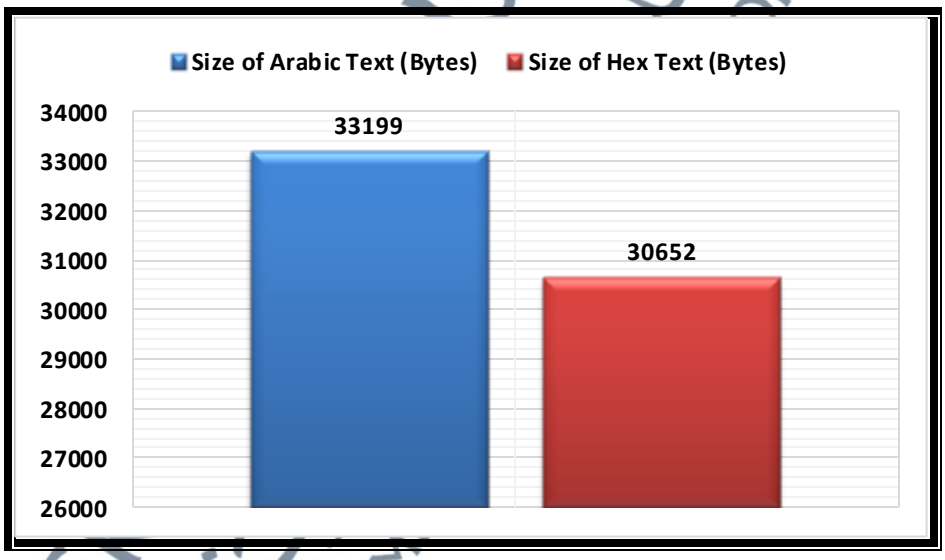


Figure 5.13: Size of Arabic Text File & Hex Text for Surah Al-Baqarah

Table 5.10 shows how sparse matrix reduces the storage with the size of the file of 33.199 MB for Arabic text, 30.625MB for hexadecimal file and 26.149 MB for sparse matrix file with a 24.24% reduction in the storage.

Table 5.10: Sparse Matrix Technique to Reduce Storage

SURAH	Number of Words	Repeated Words	Size of Arabic Text (Bytes)	Size of Hex Text (Bytes)	Size of Sparse File (Bytes)	Total Size Reduction (%)
Surah Al-Baqarah	6140	2226	٣٣١٩٩	٣٠٦٥٢	25149	24.24%

Figure 5.14 shows the second level of the comparison between the file size of the Arabic text, hexadecimal and sparse matrix file representation for surah Al-Baqarah.

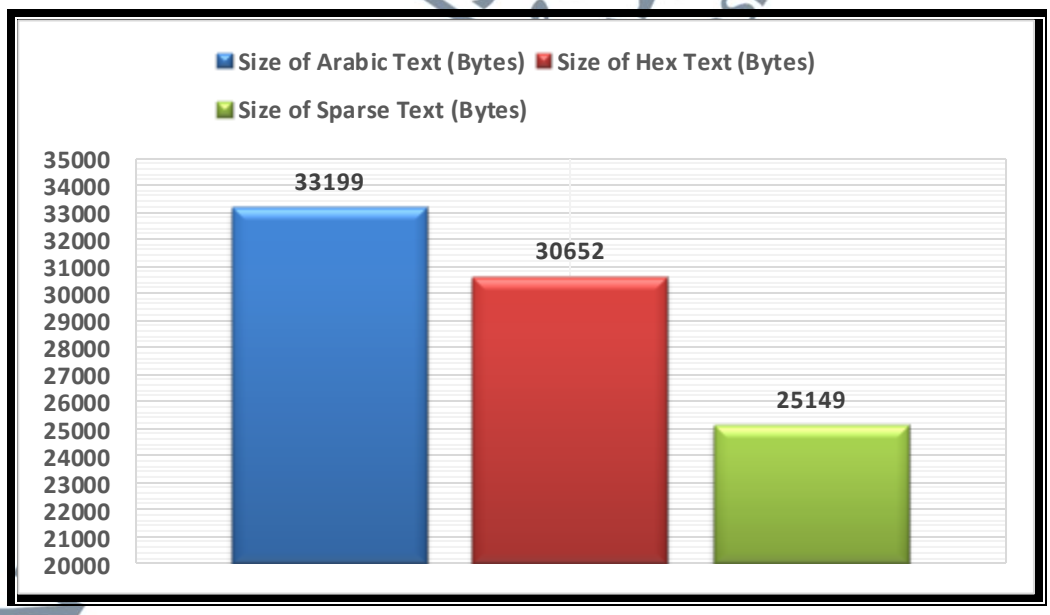


Figure 5.14: Size of Arabic, Hex, Sparse Text for Surah Al-Baqarah

From Figure 5.14 and Table 5.10, the size of the sparse matrix file was found to be smaller than hexadecimal text and Arabic text for surah Al-Baqarah with 24.24 % reduction of file size, which is a similar result with Surah al-Fatihah. This implies that the processing speed could also be improved as the space and data needed for data transfer are much smaller. Also, words duplication has been solved using a word indexing list with the unique ID stored into the sparse matrix. All these results prove that the DQM model able to reduce storage thus optimizing memory usage which could lead to faster processing due its lightweight feature.

The next evaluation is the sparse matrix and doubles offset indexing technique as in Table 5.11 to show how compressed sparse matrix with double offset indexing is able to reduce the storage by comparing the files sizes which were 16.347MB for a compressed sparse matrix with double offset indexing file with 50.67% reduction.

Table 5.11: Sparse Matrix Technique & Double-off Set Indexing Compression Algorithm

SURAH	Number of Words	Repeated Words	Size of Arabic Text (Bytes)	Size of Hex Text (Bytes)	Size of Sparse File (Bytes)	Size of Sparse File & Double-off Set Indexing (Bytes)	Total Size Reduction (%)
Surah Al-Baqarah	**6140**	**2226**	٣٣١٩٩	٣٠٦٥٢	**25149**	**16347**	**50.76%**

Table 5.11 shows the sparse matrix technique with double-off set indexing compression algorithm reduced the storage together with handling word duplications through word indexing which reduce 50.76% of file size.

Based on Figure 5.15, it can be concluded that the size of the compressed sparse metric text file with double-off set indexing is smaller due to the removal of the empty item in the matrix. The compressed sparse matrix is much smaller than the sparse text file, the hexadecimal representation text, and the original Arabic text surah Al-Baqarah. This also again implies that the processing speed could also be improved as the space and data needed for data transfer are much smaller. These results prove the DQM technique able to optimize memory space as stated in the research objective.

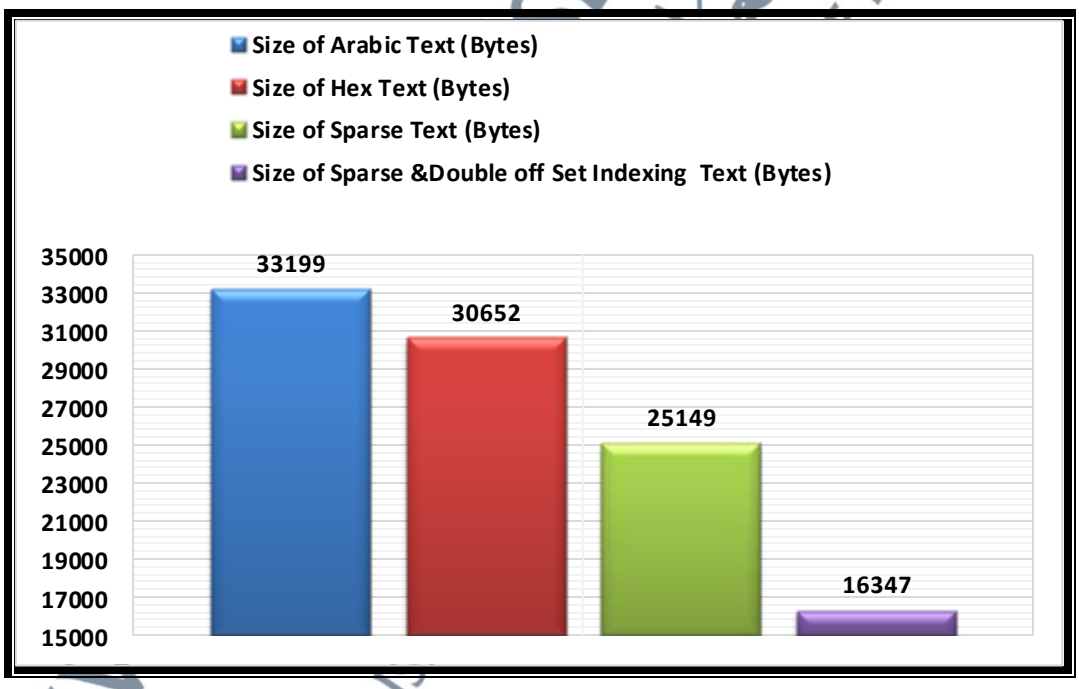


Figure 5.15: Arabic Text, Hex, Sparse and Double Off-Set Indexing for Surah Al-Baqarah

In summary, Table 5.12 presents the reduction for surah Al-Fatihah after applying hexadecimal representation is 49.05% and the reduction in the storage size is 54.08% after applying sparse matrix, the third level of comparison was using double off-set indexing compression algorithm for the same Surah which is 69.81%.

Table 5.12: Comparison of Storage Size According to Representation: Al-Fatihah& Al-Baqarah

Space Reduction	Hexadecimal Representation	Sparse Matrix	Double Offset Indexing
Al-Fatiha	49.05%	54.08%	69.81%
Al-Baqarah	7.67%	24.24%	50.76%

The space reduction for Al-Baqarah after applying hexadecimal representation is 07.67% and the storage size is 24.24% after applying sparse matrix. The composition of the Quran is 77,797 words with 14,870 words (qurananalysis.com) being unique which reflects 75% repetition of words and verses.

5.7 DQM Evaluation Methods

The above-mentioned method shows the evaluation overhead hierarchy in-memory optimization starting from letters conversion to hexadecimal representation followed by structuring the Quran content using sparse matrix together with handling words duplication by a word index lookup table, compressing it with double offset indexing algorithm. The performance overhead of DQM is also compared to existing digital Quran storage capacity with domain experts for content validity in the following section.

5.7.1 Comparison Results of DQM with Existing Applications

Evaluation of the DQM performance with digital Quran representation focuses on the integrity check and authentication of Quranic verses. In addition, a survey of digital Quran applications has been done as the second stage of comparison with more than 30 techniques and applications. Table 2.3 in Chapter 2 summarizes around 15 techniques. However, none focuses on memory optimizations through digital Quran content structure especially handling words duplications. There is related work on Quranic Code for Representing the Quran and Quran Recitation Software which provides features like audio files compressed in AMR format for reduced storage requirements. This technique provides audio files in AMR format, compressed and thus they occupy less storage space which is in contrast with this study that manipulates the text representation of the Quranic text or content.

Alginahi et al. (2013c) , Kamsin et al. (2014), Hakak et al. (2017), Alsmadi and Zarzour (2017) and Mazlan et al. (2018) used Unicode centric string matching approach or string matching approach to match or compare each string or letter from the word. Saada and Zhang, (2015) use a similar approach with the hexadecimal for compression but only representing one letter. Almazrooie et al. (2020) used one character two bytes to be coded and short ayat for testing. This study, however, extends the work to words, verses and chapters of the Quran using al-Fatihah and al-Baqarah, the longest surah as the

test case. Refer to Table 5.13 which summarizes the similarities and the differences for current application for digital Quran

Table 5.13: Similarities and Differences in the Current Application for Digital Quran.

Authors	Similarities	Differences
Alshareef & Saddik 2012	Authentication approach.	Searching algorithms were performed using 8 Diacritical Quranic phrases only the comparison was performed using 7 random Arabic words.
Sabbah & Selamat, 2013	The framework to detect and authenticate Quranic verses.	This work aims to propose a framework for accurate detection of Quranic verses from text whether it is none or fully diacritical using string matching techniques.
Alginahi et al., 2013c	Authentication approach.	Used Unicode centric string matching approach or string matching approach.
Kamsin et al., 2014	Authentication checking approach for Quran electronic version.	Used Unicode centric string matching approach or string matching approach.
Saada and Zhang, 2015	Hexadecimal representation for letters and compression.	Compression and Hexadecimal to represent only one letter.
Alsmadi & Zarzour (2017)	Authentication of Quranic verses Complete verse can be checked at a time.	Different verses were tested using different hashing approaches. Used Unicode centric string approach.
Hakak et al., 2017	Quran applications Using Unicode matching approach.	Used Unicode centric string matching approach.
Mazlan et al., 2018	Hexadecimal Conversion Algorithm by using the encoding approach.	Used Unicode centric string matching approach or string matching approach.
Almazrooie et al., 2020	Hexadecimal representation and coding method using UTF-8 to encode the text.	Used Unicode centric string matching approach or string matching and small ayah used.

The hexadecimal code Quran representation aims for a system that constructs the content structure if the digital Quran with this code. The translation has been done on the character level, word level, and verse level. Character level will be translated by extracting all the Arabic characters in the Quran regardless of the linguistic combinations. Word level will be translated by summation, combination and extracting all the duplicated words, generate a special ID for these words and verses use compressed sparse matrix technique for the content structure.

5.7.2 Content Integrity and Authentication

Testing on DQM and evaluation so far is by applying correct Quranic verses from Surah Al-Fatihah and Surah Al-Baqarah and the results showed promising space optimization and performed well. The next evaluation will be to test;

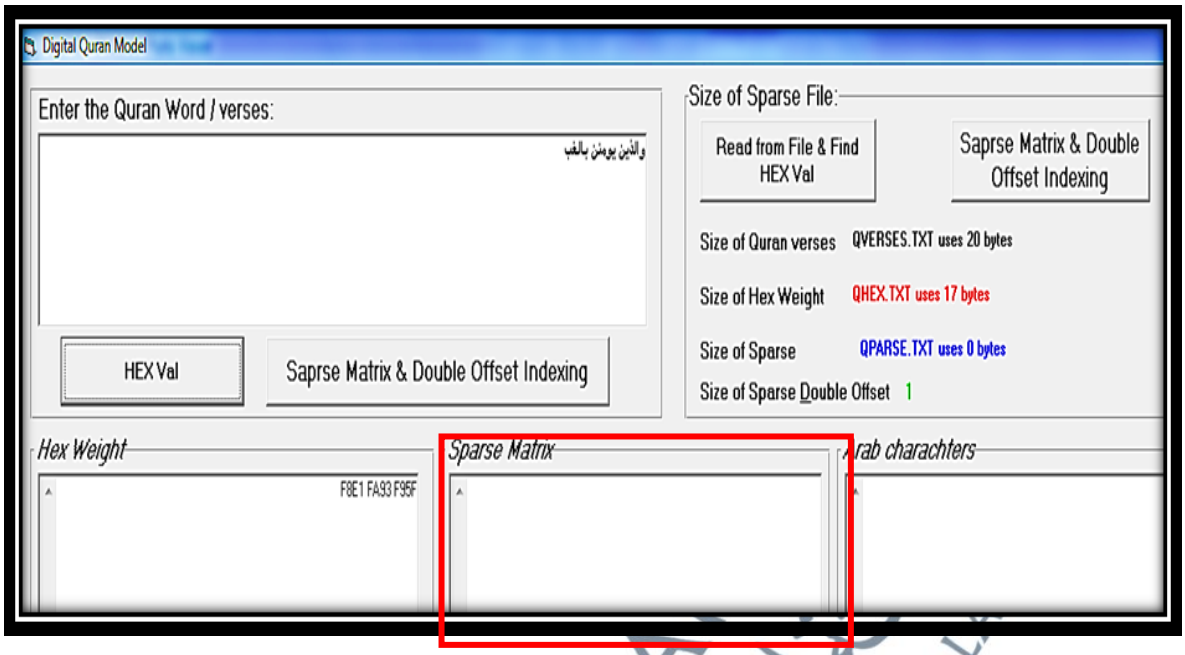
a. Missed letter text *(والذين يؤمنن بالغب)*

b. Non-Quran text *(الدراسة القديمة)*

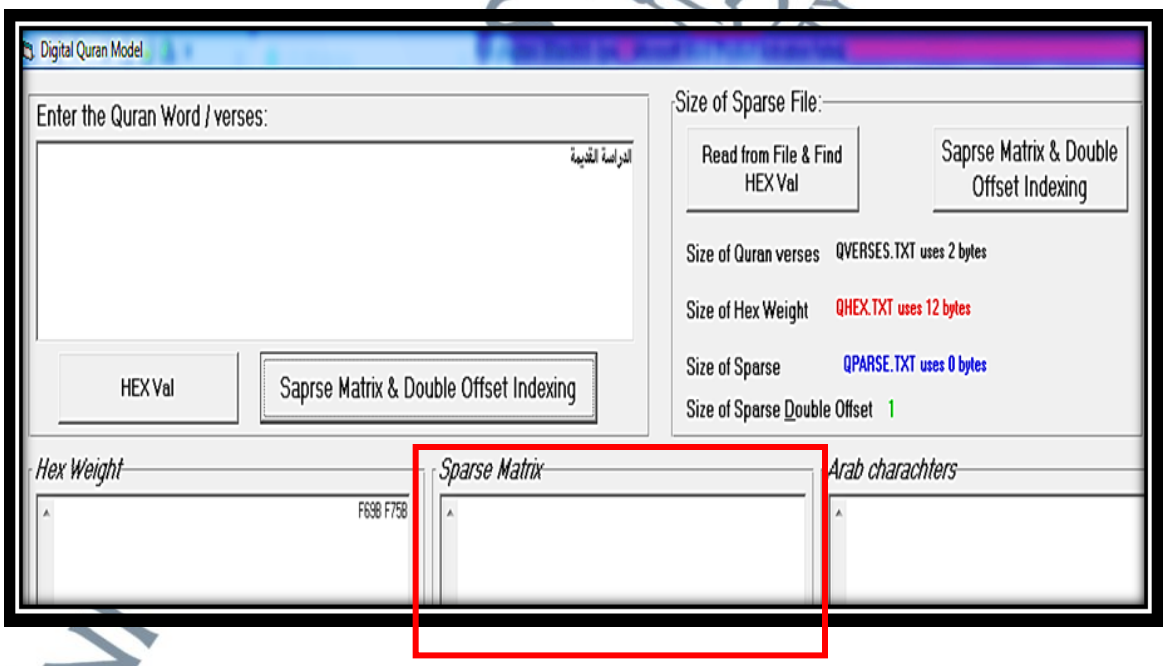
c. Non-Quran text with words exists in the Quran. *(الله اعلم كثيرا وقليلا)*

The DQM did not accept such input of fake and incorrect verses and words it gives incomplete results and will not issues the sparse matrix. These results demonstrate the authenticity and content integrity of DQM as presented in Figures 5.16, 5.17 and 5.18.

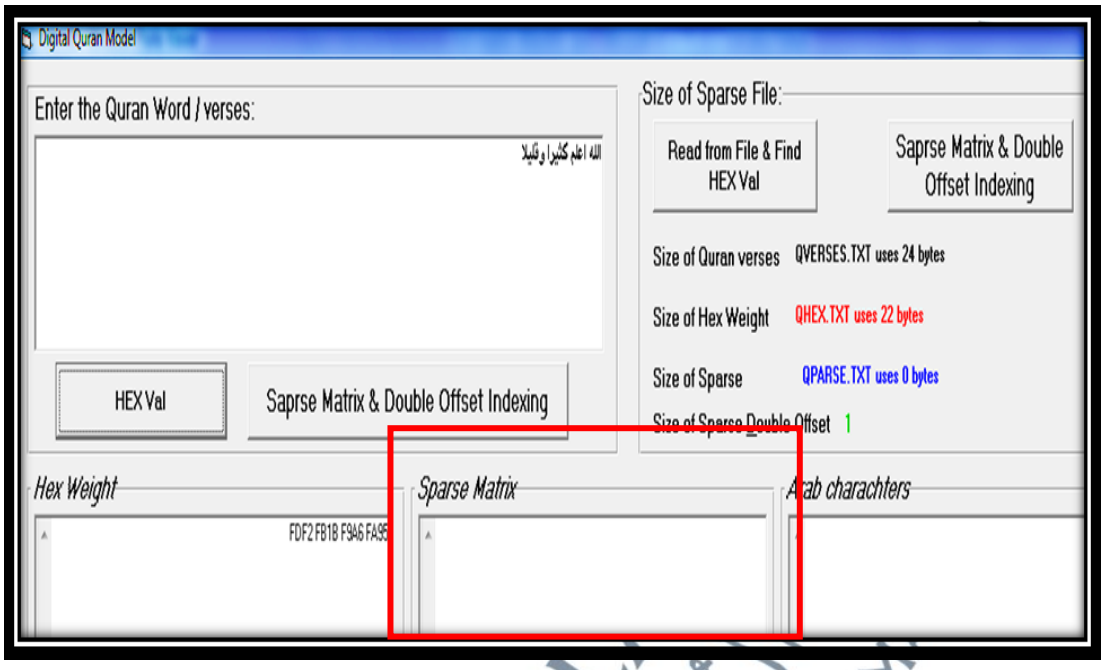
UNIVERSITI SAINS ISLAM MALAYSIA
ISLAMIC SCIENCE UNIVERSITY OF MALAYSIA



Figures 5.16: Testing with Missed Letter Text shows no sparse matrix was constructed Unmatched input text with the Look Up Table



Figures 5.17: Testing with Non-Quran Text shows no sparse matrix was constructed since it does not match the Quran words in the Look Up Table



Figures 5.18: Testing with Non-Quran Text that include Words that Exist in Quran shows no sparse matrix was constructed since it does not match any Quran verses cross checked with the Look-Up Table

This is the primary procedure in which the input is the text extracted from the source, and the output is the portions of the text that have been identified as Quranic quotations using a mixture of four detection algorithms. The next section has a detailed description of this procedure.

To improve detection accuracy in these instances, additional strategies were used, as stated in the following paragraphs:

The first additional strategy is to seek for typical Arabic article opening or ending words that signal that the following section of text is a quote or a verse. Additionally, distinct verb tenses (Past/Present), and the poem may conclude with any ending phrase or may not end at all; in other words, there are no restrictions governing the beginning and

ending phrases. These scenarios create an enormous number of possible possibilities and combinations. Especially the most common situation, in which the author of the article cites the verse(s) using the chapter (Surah) name or a number and the verse number enclosed in brackets separated by a comma.

Additionally, a position index of filtered Quranic words; this inverted index comprises a list of all spots in the Holy Quran where each filtered Quranic word is referenced. By removing non-Quranic terms found by the DR technique, this technique improves not only the identification of verses and quotations in a non-diacritical text but also the accuracy of the Diacritical Ratio technique in a completely decorated text (Alshareef & El Saddik, 2012). This approach necessitates the calculation of the weights of letters, diacritics, and filtered Quranic words. However, it may be used in conjunction with standard string matching algorithms. Additionally, this methodology may be viewed as a straightforward authentication method that can be used with non-diacritic text to verify that the Quranic words in a quote are in the right sequence as they appear in the holy Quran.

The figures 5.16, 5.17 and 5.18 show that authenticity and content integrity of DQM could successfully detect missed letter text ***(والذين يؤمنن بالغب)***, Non-Quranic text ***(الدراسة القديمة)***, and Non-Quranic text with words exist in the Quran ***(الله اعلم كثيرا وقليلا)***.

5.7.3 Experts Evaluation

The role of the expert is to ensure that the method and the proposed DQM were relevant in this thesis and reviewing the prospective methodological development. The experts in Islamic studies specifically Quranic studies to validate and confirm that the output of the Quranic text from the Hexadecimal conversion form is without error or zero error as required by Quran publication,

The Quran experts were also involved in database collection, experimentation, and evaluation of the proposed DQM. Interviews were conducted to gauge their perception on the implementation of the DQM. They gave their views of the whole problem. Besides that, they also approve the digital Quran content protection as the research scope, where DQM implementation was able to detect non-Quranic text and do not produce the Hexadecimal conversion. The two key tasks were (i) to confirmed that there were zero error in the Quran display using DQM with the storing and displaying mechanism and (ii) to confirmed that the DQM implementation were able to detect non Quranic text as shown in Figure 5.16 ,5.17 and 5.18. The domain experts involved were

a. Dr. Bashar Igried Deb Alkhawaldeh, Department Chair Faculty of Prince Al-Hussein Bin Abdallah II for Information Technology, The Hashemite University, Zarqa, Jordan.
b. Dr. Mohammad Al Khaldy, assistant professor at Khwarizmi University Technical College (KUTC) in Jordan.

5.8 Conclusion

The DQM model was implemented using Java programming language and Visual Basic for the proof of concept of with all the for key algorithms involved. Assessment on the approach was conducted to evaluate the storage optimization achievement in terms of words and verses conversion, sparse matrix structure with unique ID Look up table for unique Quran words and compressed sparse matrix content structure using double offset indexing. The content integrity was also evaluated to ensure the authenticity of the digital Quran using this new approach of space management efficiency. Experts were consulted throughout the study to ensure the process are legitimate and the outcome is agreeable with the policies of digital Quran publishing.

UNIVERSITI SAINS ISLAM MALAYSIA
جامعة العلوم الإسلامية الماليزية
ISLAMIC SCIENCE UNIVERSITY OF MALAYSIA