Check Digit System Based on Quasigroup String Transformation

Zahraddeen Abubakar Pindar¹, Sapiee Jamel², Abdulkadir Disina³, AbdulRahman Ghali⁴, Ku Nur Afiqah binti Ku Mahazer⁵, Mustafa Mat Deris⁶

Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia, Parit Raja, Batu Pahat 86400, Johor, Malaysia

Corresponding author: ¹deenpindar@gmail.com, ²sapiee@uthm.edu.my, ³academicdisina@gmail.com, ⁴aminuabdulrahman81@yahoo.com, ⁵ku_fiqa@yahoo.com, ⁶mmustafa@uthm.edu.my

Abstract
The advent of check digit methods has aided the detection of errors which are caused by human operations when information is typed wrongly. For instance, errors do often occur when numbers or characters are typed wrongly into a database and this may lead to many unwanted outcomes. Classical check digit system are usually based on basic arithmetic operations such as addition and modulo operation, and some of the classical check digit systems have been shown to be unable to detect certain types of errors. This paper proposes a check digit system based on the concept quasigroup string transformation (QTS). The error detection rate of the proposed check digit system was compared with other systems. The result of 95.56%, 57.78%, 79.67% 96% and 66.8% was obtained for ISBN, Luhn 10, UPC, Modified-UPC and Proposed Method respectively.

1. Introduction
Check digit has been in use extensively around the world for a long time. Errors which occur in machine communication are caused by human operations are detected by check digit systems [1]. For instance, in the process of keying-in codes into a database manually to store or retrieve information, while doing so errors do often occur and this may lead to many unwanted outcomes. A practical example is when a wrong UPC (Universal Product Codes) code is entered; this may result in purchase of an undesired product. Hence, it is important that these errors are detected and users are notified. This significant reason that led to the design of error detection methods; check digit methods. These check digit methods are designed to detect errors which occur while data is being keyed into an information system. Check digits are used in International Bank Account Number (IBAN) in bank account number used in many countries, International Standard Book Number (ISBN) numbers of books and Universal Product Codes (UPC) and Modified-UPC [2], [3] found in groceries [4].
Traditionally, string transformations are used as primitives for constructing encryption algorithms [5] and cryptographic hash functions [6]. In our recent work [7], we proposed a different kind of transformation and showed that the transformation can be used for constructing a compression for MAC algorithm [8]. The idea of this paper is an attempt to show that a check digit system can be constructed using quasigroup string transformation.

This paper is divided into three sections. The first section provides the introduction of the subject area. The second section discusses some prominent check digit methods. The third section discusses the proposed check digit method. And lastly the forth section provides the concluding remarks.

2. Related Work

2.1 Preliminaries

2.1.1 Permutation

Definition 2.1 [4][9]. A Permutation is a bijection \( \pi \) of a finite group \((Q,+)) onto itself \( \pi:Q\rightarrow Q \). The permutation a group \((Q,+)) is said to be non-commutative iff for all \( x,y \) in \( Q \) such that:

\[
x + \pi(y) \neq y + \pi(x) \rightarrow x\neq \pi(x)
\]

(1)

Proposition 2.1 : \( \pi \) is a non-commutative permutation on a finite group \( Q \) and satisfy for all \( u,v \in Q \) such that:

\[
\pi(u)+v=\pi(v)+u \rightarrow u=v
\]

(2)

Proof: The proposition is converse of definition. Thus, they are equivalent. Non-commutative permutations are highly efficient permutations.

2.1.2 Quasigroups

Definition 2.2 [10]. A quasigroup \((Q,\ast)) is an algebraic structure containing a set of elements \( Q \) together with a binary operation \( \ast \). For all \( a,b \) in \( Q \), there exist unique solution \( x,y \in Q \), such that:

\[
x \ast a = b \nonumber\]
\[
a \ast y = b
\]

(3)

In adopting the use of quasigroups, we consider quasigroups with special property, non-associativity. The significance of this category of quasigroups has been discussed extensively in [11]. Elaborate discussions on methods of generating quasigroups can be found in the works of Meyer [11] and Scielny [10].

2.2 Common Errors in Check Digit

There are five types of error that have been analyze in this thesis which are single digit error, single transposition digit error, jump transposition digit error, twin digit error and jump twin digit error [2], [3]. Table 1 gives the relative frequency of error occurrences.

2.2.1 Single Digit Error

This error occurs when only a single digit in the whole string of numbers or alphabets is typed wrongly. Example is when 123 is typed as 124. Here the digit 3 is typed as 4.

2.2.2 Single Transposition Error

This type of error occurs when the \( n \)th and the \((n+1)\)th digits are replaced by each other respectively. Example is when 123 is typed as 132. Here the digit 2 is transposed with 3.
2.2.3 Twin Error
This type of error occurs when two identical digits are incorrectly typed as two other identical digits. Example is when 11 is typed as 22.

| Error type         | Form      | Relative frequency |
|--------------------|-----------|--------------------|
| Single error       | \(a \rightarrow b\) | 79.1%               |
| Single transposition error | \(ab \rightarrow ba\) | 10.2%               |
| Jump transposition | \(abc \rightarrow cba\) | 0.8%               |
| Twin error         | \(aa \rightarrow bb\) | 0.5%               |
| Jump twin error    | \(aca \rightarrow bcb\) | 0.3%               |

2.2.4 Jump Transposition Error
This type of error occurs when the \(n\)th and \((n+2)\)th digits are replaced by each other respectively. Example is when 132 is typed as 231.

2.2.5 Jump Twin Error
This type of error occurs when the \(n\)th and \((n+2)\)th digits are identical and are mistyped as two other identical digits such as when 131 is typed as 232.

2.3 Check Digit Methods
This section discusses some prominent check digit methods.

2.3.1 Parity Check Method
This method was developed by Peter Luhn [1]. It had no weight. The code word is summed up and modulo with 10. The parity check is computed using \((a_1+a_2+a_3+a_4+a_5+a_6+a_7) \equiv 0 \pmod{10}\). The check digit equation is defined in Equation 4.

\[
\sum_{k=1}^{n} a_{n-k} \equiv 0 \pmod{10} \tag{4}
\]

This check digit method has certain limitations. It can be observed that transposition errors in the form of Equation \(\alpha\) and \(\beta\) will not be detected by this check digit method due to commutative law for addition.

\[
(a_1+a_2) \mod 10 = (a_2+a_1) \mod 10 \tag{5}
\]

\[
(a_1+a_2+a_3) \mod 10 = (a_3+a_2+a_1) \mod 10 \tag{6}
\]

This means that when digits are transposed from \(a_1+a_2\) to \(a_2+a_1\), the error will not be detected. Hence, to solve this concern a weighted parity is required in the check digit system in order to detect the adjacent errors and jump transposition errors.

2.3.2 Weighted Parity Method
Conventional check digit methods such as Universal Product Code (UPC) and International Standard Book Number (ISBN) impose weights on every digit before adding. The weighted parity method is adopted as a solution the problem parity check digit method; that is solving the problem of adjacent and
jump transposition error. The final result after summation is modulo by 10. Let the sequence of weight be $(w_{n-1}, w_{n-2}, \ldots, w_2, w_1)$ and the number to which the check digit will be assigned to be $a_{n-1}, a_{n-2}, \ldots, a_2, a_1$, such that the check digit $a_n$ is computed using Equation 7.

$$\sum_{k=1}^{n} a_{n-k} \cdot w_{n-k} \equiv 0 \pmod{10}$$

(7)

where $a_{n-k}$ is the information digit and $w_{n-k}$ is the weight.

Thus, the weighted check digit method is an improvement over the parity check method. However, the UPC check digit method does not detect all jump transposition errors. For single errors to be detected, choice of $w_i$ must be relatively prime to 10 as in Equation 8.

$$\text{GCD}(w_i, 10) = 1$$

(8)

However $(w_i \cdot w_{i+1})$ is even, thus $(w_i \cdot w_{i+1})$ is a zero divisor in $\mathbb{Z}_{10}$. Hence, this implies that transpositions such as $\ldots x_i x_{i+1} \ldots \rightarrow \ldots x_i x_{i+1} \ldots \rightarrow \ldots x_{i+1} x_i \ldots$, where $x_i = a_i, w_i$ and $x_{i+1} = a_{i+1} w_{i+1}$, will not be detected when $|x_i - x_{i+1}| = 5$. The ISBN code takes non-numerical digits into account to represents the 10th digit in base 11 and represent the use the character “X” if the check digit is “10”. Example the book Impressionism and Post-Impressionism: The heritage, Leningrad, The Pushkin Museum of Fine Arts, Moscow, and The National Gallery of Art, Washington has ISBN 0-517-66562-X, where X is the check digit.

3. Proposed Check Digit Method

It can be clearly observed that the check digit systems discussed in the related work section are based on simple arithmetic operations. This paper proposes a new check digit system based on quasigroup string transformation ($T_f$). The motivation for developing a transformation based check digit systems stems from our earlier work [7], where the transformation function ($T_f$) served as a primitive for a keyed-hash function [8]. The transformation function acts is a function is sensitive to changes. This property is best explained in Example 1. The transformation function ($T_f$) is composed of two sub-function; forward ($T_A$) and reverse ($T_R$) transformation. The forward transformation takes in input string and processes it in a forward direction. The reverse transformation on the other hand takes in the output of the forward transformation and processes it in a reverse. The output results of the forward and reverse transformations are XORed to generate the result for final transformation ($T_f$). We give a formal definition of forward and reverse transformation in Definitions 2.3 and 2.4.

**Definition 2.3** Given that $(Q, *)$ is a non-associative quasigroup and elements $a_1, a_2, \ldots a_n$ be input of transformation $T_A$ and a leader $l_i$, where $a_1, a_2, \ldots a_n, l_i \in Q$. A forward transformation $T_A$ is defined as:

$$T_A(a_1 \ldots a_n) = \begin{cases} b_1 = l_1 \cdot \pi(a_1) , \\ b_i = b_{i-1} \cdot \pi(a_i), \quad i = 2 \ldots n \end{cases}$$

(9)

**Definition 2.4** Let output of forward transformation $T_A$ be input of reverse transformation $T_R$ and elements $c_1, c_2, \ldots c_n$, where $c_1, c_2, \ldots c_n \in Q$. A reserve transformation $T_A$ is defined as:

$$T_R(b_1 \ldots b_n) = \begin{cases} c_n = \pi(b_n) \cdot l_2 , \\ c_{n-i} = \pi(c_{n-i}) \cdot b_{n-(i-1)}, \quad i = 1 \ldots n \end{cases}$$

(10)

The leaders’ $l_1$ and $l_2$ for both forward and reverse transformations respectively are computed as follows:

$$l_1 = (((((c \cdot a_1) \cdot a_2) \cdot \ldots \cdot a_{n-1}) \cdot a_n)$$

$$l_2 = (((c \cdot b_1) \cdot b_2) \cdot \ldots \cdot b_{n-1}) \cdot b_n$$
where \( c = 0 \)

The final transformation output is computed by XORing the output of \( T_A \) and \( T_B \), as given in Equation 11.

\[
T_F = d_1, d_2, \ldots, d_n = T_A(a_1, a_2, \ldots, a_n) \oplus T_B(a_1, a_2, \ldots, a_n) \tag{11}
\]

The pseudorandom property of transformation function \( T_F \) begins at the process of generating the leader \( l_1 \) and \( l_2 \). Intuitively, changing a value in a given input will result to a different leader. For instance, let the \( l \) be the leader generated using transformation \( T_A \) using the input \( a_1, a_2, \ldots, a_n \). A slight change in the input, say \( a_1, a_2, \ldots, a_i, \ldots, a_n \), will result to a different leader \( l' \) with high probability. This can be easily proven, given that all the elements are multiplied under a non-associative quasigroup operation “\( \oplus \)”, using Equations 9 and 10. The summation of transformation output \( T_F (d_1, d_2, \ldots, d_n) \) defined XOR operation “\( \oplus \)” defines a single check digit system.

\[
\mu = d_1 \oplus d_2 \oplus \ldots \oplus d_{n-1} \oplus d_n \tag{12}
\]

where \( \mu \) is the check digit and \( d_1, d_2, \ldots, d_n \) is output of transformation \( T_F \).

The method will work best for systems with Hexadecimal numbers due to the XOR operation used in Equation 11 and 12. Example of such systems are International Standard Audiovisual Number (ISAN) which enables the identification of any kind of audiovisual works and International Mobile Equipment Identifier (MEID) code which is unique for each mobile station.

**Example 2.1**

Let \((Q, \cdot)\) non-associative quasigroup and 4029FCDA and 4029FCD1 be two slightly different information digits where the difference is found the last character, and \( \pi(x) \) be a random non-associative permutation for the transformation \( T_F \).

\[
\pi(x) = \begin{pmatrix}
0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & A & B & C & D & E & F \\
C & 6 & 3 & E & 2 & D & 5 & 9 & 8 & B & F & 1 & 7 & 4 & A & 0
\end{pmatrix}
\]

The non-associative quasigroup is constructed using the function in Equation 13.

\[
x \cdot y = x \oplus g(y) \tag{13}
\]

Where the function \( g(x) \) is a non-associative permutation defined as:

\[
g(x) = \begin{pmatrix}
0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & A & B & C & D & E & F \\
4 & 6 & 2 & F & D & 7 & 8 & 5 & B & 3 & E & A & 1 & 0 & C
\end{pmatrix}
\]

We omit the structure of the quasigroup due to its large size and simply give the function to generate the quasigroup.

**Table 2:** Computing different inputs in Transformation Function \( T_F \)

| Transformation stages | Input 4 0 2 9 F C D A | Input 4 0 2 9 F C D 1 |
|-----------------------|-----------------------|-----------------------|
| Forward \( (T_F) \)   | \( T_A(4029FCDA) = 0A315DC8 \) | \( T_A(4029FCD1) = 71A8C43A \) |
| Reverse \( (T_R) \)   | \( T_R(0A315DC8) = CE419F92 \) | \( T_R(71A8C43A) = 9E7745AF \) |
| Final \( (T_F) \)     | \( 0A315DC8 \oplus CE419F92 = C470C25A \) | \( 71A8C43A \oplus 9E7745AF = EFDF8195 \) |
| Check Digit \( \mu \) | \( C \oplus 4 \oplus 7 \oplus 0 \oplus C \oplus 2 \oplus 5 \oplus A = 6 \) | \( E \oplus F \oplus D \oplus F \oplus 8 \oplus 1 \oplus 9 \oplus 5 = 9 \) |

Hence, it can see from Table 2 that changing a single character will significantly affect the outcome of the transformation, and by extension resulting to a new check digit, 4029FCDA-6 and 4029FCD1-9.
4. Experimentation
Error detection test was carried out to examine the error detection capability of the proposed method in its ability to in relation to other check digit methods. The results are summarized in Table 3. Four hundred and fifty (450) test-data were used in the comparative testing. The test-data were generated randomly, given that there are no standardized datasets to test the error detection capability of a check digit method. Each test-data contains twelve (12) digits. This is done in-order to meet the standardized UPC digit length. Fifty test-data was used to carry out each individual test (error).

| Errors          | ISBN | Luhn 10 | UPC | Modified-UPC | Proposed Method |
|-----------------|------|---------|-----|--------------|-----------------|
| Single digit    | 100% | 100%    | 100%| 100%         | 96%             |
| Single transposition | 88.90% | 0%     | 100%| 100%         | 97%             |
| Jump transposition | 100% | 0%     | 0% | 92%          | 95%             |
| Twin            | 88.90% | 88.9%   | 88.89% | 96%          | 24%             |
| Jump twin       | 100% | 100%    | 88.90% | 92%          | 22%             |
| Total accuracy  | 95.56% | 57.78%  | 79.67% | 96%          | 66.8%           |

Based on the results from Table 3, Modified-UPC method has an average total accuracy of 96%. UPC method, Luhn and ISBN have an average total accuracy of 79.67%, 57.78% and 95.5% respectively. Lastly, the proposed method has accuracy 66.8%. While the proposed method has the least error detection capability as shown in Table 3, the method has an error detection accuracy of 96% and 97% for single digit and single transposition errors, which have the highest occurrences as shown in Table 1.

5.0 Conclusion
Check digit systems are used to detect errors in while keying-in data into a system. Check digit systems are usually constructed using simple equations. A typical example is the Luhn, UPC and ISBN check digit systems that based on simple arithmetic operations. This paper attempts to show that a check digit system can be constructed based on quasigroup transformation function (QTS). While the results in Table 3 shows that the proposed check digit system has the least accuracy, however it does show that the total accuracy of proposed check is greater than 50%. The future work of this research is to enhance the proposed check digit system in order to increase the error detection rate. This can be achieved by using different quasigroups and permutations at different states of transformations (forward and reverse transformation). Similarly, other properties of quasigroups such as Isotopes could be utilized in order increase the error detection capability of the system.

Acknowledgment
This research was supported by the Office for Research, Innovation, Commercialization and Consultancy (ORICC), Universiti Tun Hussein Onn Malaysia (UTHM) under Project Vot No. U194.

References
[1] J. a. Gallian, “Error detection methods,” ACM Comput. Surv., vol. 28, no. 3, pp. 504–517, 1996.
[2] Z. A. Pindar, “UTHM certificate verification using microtext double check digit.” Diss. Universiti Tun Hussein Onn Malaysia, pp. 1–53, 2014.
[3] Z. A. Pindar, A. H. Disina, and S. Jamel, “An Enhanced UPC Mod 10 Check Digit Method,” Int. J.
Knowl. Based Comput. Syst., vol. 3, no. 2, pp. 1–6, 2015.

[4] A. Ecker and G. Poch, “Check character systems,” Comput. (Vienna/New York), vol. 37, no. 4, pp. 277–301, 1986.

[5] S. I. Marnas, L. Angelis, and G. L. Bleris, “All-Or-Nothing Transforms Using Quasigroups All-Or-Nothing Transforms,” in Ist Balkan Conference in Informatics, 2003, pp. 183–191.

[6] A. Mileva and S. Markovski, “Quasigroup String Transformations and Hash Function Design,” in ICT Innovations 2009, 2010, pp. 367–376.

[7] Z. Pindar, S. H. Jamel, A. Disina, and M. M. Deris, “Compression Function Based on Permutations Quasigroups,” ARPN J., vol. 11, no. 22, pp. 1–8, 2016.

[8] Z. A. Pindar, S. H. Jamel, M. Aamir, and M. M. Deris, “PinTar: A New Keyed Hash Function based on Pseudorandom 2 n -to- n bit Compression Function,” in 4th International Conference on Information and Network Security (ICINS 2016), 2016.

[9] R.-H. Schulz, “A note on check character systems using Latin squares,” Discrete Math., vol. 97, pp. 371–375, 1991.

[10] K. O. Scielny, “Generating quasigroups for cryptographic applications,” Int. J. Appl. Math. Comput. Sci., vol. 12, no. 4, pp. 559–570, 2002.

[11] K. A. Meyer, A new message authentication code based on the non-associativity of quasigroups. Iowa State University, 2006.