I. INTRODUCTION

In a civilization, human labor of one type is exchanged for human labor of another type. For example, when one eats at a restaurant, one exchanges money earned from one’s own labor for the labor of people who cook, raise animals and vegetables, deliver goods, etc. This exchange is made possible by a salary, which establishes a value for labor of different types. There are many ways to represent a value; however, in virtually all number systems, symbols from some finite set are combined to represent potentially large values. The decimal number system, in which there are ten symbols, has found wide use; many people believe this is due to the ten fingers of the human species. A similar situation exists in computers. Since conventional logic elements easily realize two values, computers use binary number systems. The prospect of multiple-valued logic suggests the need to understand number systems with more than two digits.

In spite of the recent use of binary systems, there is evidence that suggests the Chinese used the binary number system 5000 years ago, Newman [15, p. 2422].

An interesting discovery was made at the site of Mohenjo-Daro, an ancient city near the Indus River (Morrison and Morrison [14]). Irregularly shaped stones were found, each twice the weight of another. Archeologists believe that these were used by merchants in the Harappan culture for an equal-arm balance 4000 years ago. For some object on one side of a balance in equilibrium, the stones on the other side represent the binary value of the object’s weight. Fig. 1 below shows an example of how such a system might work.
Redundant Multiple-Valued Number Systems
The formation of this system is simple. First, two small stones are found, each of identical weight; a balance scale is used to determine that their weights are identical. Interestingly, the two equal-weight stones found near the Indus River, each weighed slightly less than an ounce. Then, using a balance scale again, a third stone is found equal to the weight of the two equal-weight stones. These three stones are then used to find a fourth, which has weight identical to their combined weight, etc.. Thus, the stones’ weight occurs in the proportion - 1, 1, 2, 4, 8, 16,... .

III. COMPARISON OF NUMBER SYSTEMS

A. STANDARD BINARY NUMBER SYSTEM

In the standard binary number system, a number \( N \) has a value given as

\[
N = a_n2^{n-1} + \ldots + a_22^2 + a_12^1 + a_02^0,
\]

where \( a_i \in \{0,1\} \). Fig. 2 below shows an example of the 4-bit standard binary number system. The table shows all binary 4-tuples and their corresponding value. The histogram plots the number of representations for each numerical value. From this, it can be seen that there is exactly one 4-bit representative for each of the non-zero integers, 0, 1, 2, ..., 15.

\[
N = a_32^3 + a_22^2 + a_12^1 + a_02^0, \text{ where } a_i \in \{0,1\}.
\]

Fig. 3 shows the 4-bit nega-binary number system. As with the standard binary number system, every number has a unique representative 4-tuple. It is interesting that there are twice as many negative integers as there are positive integers. This statement is true of general nega-binary number systems, when the number of bits is even. If the number of bits is odd, the number of positive integers is twice the number of negative integers plus 1. The disparity between the number of negative numbers and positive numbers is a significant disadvantage.

\[
N = a_3(-2)^3 + a_2(-2)^2 + a_1(-2)^1 + a_0(-2)^0, \text{ where } a_i \in \{0,1\}.
\]

B. NEGA-BINARY NUMBER SYSTEM

In the nega-binary number system, both positive and negative integers are represented. In this system, the number representation is the same as the binary number system with the base 2 replaced by -2. That is,

\[
N = a_n(-2)^{n-1} + \ldots + a_2(-2)^2 + a_1(-2)^1 + a_0(-2)^0.
\]

Fig. 4 below shows the 4-bit two’s complement number system.

\[
N = a_3(-2)^3 + a_2(-2)^2 + a_1(-2)^1 + a_0(-2)^0, \text{ where } a_i \in \{0,1\}.
\]
number system. As with all number systems shown so far, all representations are unique. In this system, there is one more negative number than positive number.

D. ONE’S COMPLEMENT NUMBER SYSTEM

The one’s complement number system is the first example of a redundant number system. Specifically, 0 has two representatives, usually called 0 and -0. With any number represented as

\[ N = a_n1(-2)^{n+1} + \ldots + a_2a_2 + a_1a_1 + a_0a_0 + a_{n-1} \]

0 is 00...0 and -0 is 11...1. Fig. 5 below shows the 4-bit one’s complement number system. The two representatives of 0 are shown as a line of length 2 at value 0.

\[ N = a_j(-2)^{j+1} + a_2a_2 + a_1a_1 + a_0a_0 + a_{n-1}, \text{ where } a_j \in \{0,1\}. \]

Figure 5. The 4-Bit One’s Complement Number System.

E. FIBONACCI NUMBER SYSTEM

The Fibonacci number system is the second example of a redundant number system. In such a number system, numbers have the value

\[ N = a_n1F_{n+1} + \ldots + a_2F_4 + a_1F_3 + a_0F_2, \]

where \( F_i = F_{i-1} + F_{i-2} \) and \( F_2 = F_1 = 1 \). That is, instead of powers of 2 or -2, Fibonacci numbers are used as the base. In such a number system, there are many redundant representatives. Fig. 6 shows that, for the 4-bit Fibonacci number system, there are four redundant representatives.

\[ N = a_jF_j + a_2F_4 + a_1F_3 + a_0F_2, \text{ where } a_j \in \{0,1\} \text{ and } F_i \text{ is the } i\text{th Fibonacci number}. \]

Figure 6. The 4-Bit Fibonacci Number System.

F. FIBONACCI NUMBER SYSTEM WITH NO ADJACENT 1’S

Zeckendorf [16] showed that if one retains only those binary patterns in the Fibonacci number system without pairs of 1’s, then, each integer is uniquely represented. Fig. 7 shows this. In the table of Fig. 7, 4-tuples with at least one pair of 1’s are overlayed with a line. Removing all 4-bit patterns with pairs of 1’s, yields a number system in which 0, 1, ..., and 7 are uniquely represented.

\[ N = a_jF_j + a_2F_4 + a_1F_3 + a_0F_2, \text{ where } a_j \in \{0,1\} \text{ and } F_i \text{ is the } i\text{th Fibonacci number}. \]

Figure 7. The 4-Bit Fibonacci Number System in Which No Pairs of Adjacent 1’s Exist.

The absence of pairs of adjacent 1’s makes this number system useful in systems that retrieve and store data serially. For example, Davies [6] describes a CD-ROM system in which codewords without pairs of 1’s are used. Kautz [12] proposed codes that could be transmitted without an accompanying clock, requiring there be enough 0 to 1 and 1 to 0 transitions to delineate where bits begin and end.
Brown [1] showed that if one retains only those binary patterns in the Fibonacci number system with the most 1’s, then, in the resulting system, each integer is uniquely represented. Stated differently, each representation with the most number of 1’s is unique. Fig. 8 shows this for 4-bit patterns. In this case, the numbers 0, 1, ..., and 11 are uniquely represented.

\[
N = a_3F_4 + a_2F_3 + a_1F_2 + a_0F_0, \quad \text{where } a_i \in \{0,1\} \text{ and } F_i \text{ is the } i\text{th Fibonacci number. } F_i = F_{i-1} + F_{i-2} \text{ and } F_1 = F_2 = 1.
\]

A natural extension of the Fibonacci number system represents negative integers as well as positive integers. Let \( F_i = (-1)^{n+1}F_i \), and let

\[
N = a_{n-1}F_{n-1} + \ldots + a_2F_4 + a_1F_3 + a_0F_2,
\]

where \( a_i \in \{0,1\} \) and \( F_i \) is a Fibonacci number. Negative values of \( N \) occur when the sum of negative Fibonacci numbers exceeds the sum of positive numbers in the representation. The case of \( n = 4 \) is shown in Fig. 9. Note that this number system has many redundant representatives.

Figure 8. The 4-Bit Fibonacci Number System in Which the Most 1’s Exist.

H. NEGA-FIBONACCI NUMBER SYSTEM

Another extension of the Fibonacci number system represents negative integers as well as positive integers. Let \( F_i = (-1)^{n+1}F_i \), and let

\[
N = a_{n-1}F_{n-1} + \ldots + a_2F_4 + a_1F_3 + a_0F_2,
\]

where \( a_i \in \{0,1\} \) and \( F_i \) is a Fibonacci number. Negative values of \( N \) occur when the sum of negative Fibonacci numbers exceeds the sum of positive numbers in the representation. The case of \( n = 4 \) is shown in Fig. 9. Note that this number system has many redundant representatives.

Figure 9. The 4-Bit Nega-Fibonacci Number System.

I. NEGA-FIBONACCI NUMBER SYSTEM WITHOUT ADJACENT 1’S

It has been shown Bunder [3] that, in the nega-Fibonacci number system, if the tuples with adjacent 1’s are removed, then the remaining tuples represent distinct values. Fig. 10 below shows that, for the 4-bit system, there are 8 remaining tuples, representing the values -4 through 3.

\[
N = a_3F_4 + a_2F_3 + a_1F_2 + a_0F_0, \quad \text{where } a_i \in \{0,1\} \text{ and } F_i = (-1)^{n+1}F_i \text{ for } F_i \text{ is the } i\text{th Fibonacci number. } F_1 = F_{i+1} + F_{i+2} \& F_i = F_2 = 1. \quad N = a_{-3}F_{-3} + a_{-2}F_{-2} + a_{-1}F_{-1} + a_0.
\]

Figure 10. The 4-Bit Nega-Fibonacci Number System.

J. TRIBONACCI NUMBER SYSTEM

A natural extension of the Fibonacci number system is the tribonacci number system, Capocelli et al [5] and Fraenkel [8]. In this system, each number has the form

\[
N = a_{n-1}F_{n-1} + \ldots + a_2F_4 + a_1F_3 + a_0F_2,
\]

where \( F_i = F_{i+1} + F_{i+2} + F_{i+3} \text{ and } F_1 = 2^1, F_2 = 2^1, \text{ and } F_3 = 3^0. \) Fig. 11 shows that, for the 4-bit tribonacci number system, there is one redundant representative.

Figure 11. The 4-Bit Tribonacci Number System.
K. RESTRICTED TRIBONACCI NUMBER SYSTEM

Fraenkel [8] has shown that if there are no groups of three or more consecutive 1’s, then the representations are unique. Fig. 12 shows that, for the 4-bit tribonacci number system, eliminating 4-tuples with three or more consecutive 1’s yields a number system in which 0, 1, ..., 12 have unique representations.

\[
N = a_1F_2 + a_2F_4 + a_3F_3 + a_0F_2, \text{ where } a_i \in \{0, 1\} \text{ and } F_i \text{ is the } i\text{th Tribonacci number.}
\]

\[
F_i = F_{i-1} + F_{i-2} + F_{i-3}, \text{ and } F_0 = 0, F_1 = 1, F_2 = 1.
\]

This result is correct. Indeed, Fraenkel [8] has shown that in number systems where \( F_i \) is given as

\[
F_i = F_{i-1} + F_{i-2} + \ldots + F_{i-m},
\]

for \( i > m+1 \) and \( F_i = 2^i \), for \( 1 < i \leq m+1 \), then if there are no more than \( m-1 \) consecutive 1’s, the representations are unique.

L. FIBONACCI-LIKE NUMBER SYSTEM

Another natural extension of the Fibonacci number system is a Fibonacci-like number system Klein [13], where each number has the form

\[
N = a_{n-1}F_{n-1} + \ldots + a_2F_4 + a_1F_3 + a_0F_2,
\]

where \( F_i = F_{i-1} + F_{i-m} \) for \( i > m+1 \), and \( F_i = i-1 \), for \( 1 < i \leq m+1 \), where \( m \geq 2 \). Fig. 13 shows a 4-bit Fibonacci-like number system. This is redundant; five duplicate representations exist.

\[
N = a_0F_5 + a_1F_4 + a_2F_3 + a_3F_2, \text{ where } a_i \in \{0, 1\} \text{ and } F_i \text{ is the } i\text{th Fibonacci-like number.}
\]

\[
F_i = F_{i-1} + F_{i-2} + \ldots + F_{i-m}, \text{ where the digits, } F_i, \text{ are } 0, 1, \ldots, m-1, \text{ and the basis is given as}
\]
Fig. 15 shows a 3-digit system, where there are two redundant representatives. Here, there are two redundant representatives. For example, we expect that the restriction - no adjacent 1’s - reduces the number of 1’s in comparison to the number of 0’s. We are interested in the extent to which this happens.

Our first result shows the distribution of non-redundant representations in the Fibonacci, tribonacci, quadranacci, etc. number systems. Specifically, Fig. 17 shows the number of representatives verses the number of 1’s in 16-bit representations. That is, we consider 16-bit tuples who value is given by

\[ N = a_1 F_4 + a_2 F_3 + a_3 F_2, \text{ where } a_i \in \{0,1,2\} \]

which represents the Fibonacci, tribonacci, and quadranacci number systems, respectively. It is interesting that the Fibonacci number system has very few representatives (in which pairs of 1’s are not allowed) compared to the tribonacci number system (in which triples of 1’s are not allowed). By way of comparison, the distribution of 1’s in the standard binary number system (which corresponds to \(m = \infty\)) is shown. Table 1 below...

### IV. MEASURING REDUNDANCY

The question is important when we restrict the tuples allowed in the system. For example, we are interested in the extent to which the restriction - no adjacent 1’s - reduces the tuples available to represent numbers in the Fibonacci number system. We measure redundancy in two ways: 1) the number of tuples available to represent values and 2) the percentage of various digits over all representations.
shows the proportion of bits that are 1 in the allowed representatives when the number of bits is large.

<table>
<thead>
<tr>
<th>( m )</th>
<th>Proportion of 1’s</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.2764</td>
</tr>
<tr>
<td>3</td>
<td>0.3816</td>
</tr>
<tr>
<td>4</td>
<td>0.4337</td>
</tr>
<tr>
<td>5</td>
<td>0.4621</td>
</tr>
<tr>
<td>6</td>
<td>0.4782</td>
</tr>
<tr>
<td>7</td>
<td>0.4875</td>
</tr>
<tr>
<td>8</td>
<td>0.4929</td>
</tr>
<tr>
<td>( \infty )</td>
<td>0.5000</td>
</tr>
</tbody>
</table>

Table 1.[4] The proportion of bits that are 1 in various number systems.

Fig. 18 shows the distribution of 1’s for the generalized Fibonacci number systems. In these systems, a representative represents a number as in (1), where \( F_i = F_{i-1} + F_{i-m} \) for \( i > m+1 \), and \( F_i = i-1 \), for \( 1 \leq i \leq m+1 \), where \( m \geq 2 \). As in the previous example, \( m = 2 \) corresponds to the Fibonacci number system. It is interesting, that, in this case, that there are significantly more representatives when \( m = 2 \) than when \( m = 3 \).

![Diagram](image)

Figure 18. [4] The distribution of non-redundant representatives in 16-bit number systems.

Table 2 shows the proportion of digits that are 0, 1, \( ... \), \( m-1 \) in \( m \)-nacci number systems.

<table>
<thead>
<tr>
<th>( m )</th>
<th>Proportion of 0’s</th>
<th>Proportion of 1’s, 2’s, ... ( m-1 )’s</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.7236</td>
<td>0.2764</td>
</tr>
<tr>
<td>3</td>
<td>0.8057</td>
<td>0.3816</td>
</tr>
<tr>
<td>4</td>
<td>0.8492</td>
<td>0.4337</td>
</tr>
<tr>
<td>5</td>
<td>0.8762</td>
<td>0.4621</td>
</tr>
<tr>
<td>6</td>
<td>0.8948</td>
<td>0.4782</td>
</tr>
<tr>
<td>7</td>
<td>0.9084</td>
<td>0.4875</td>
</tr>
<tr>
<td>8</td>
<td>0.9188</td>
<td>0.4929</td>
</tr>
<tr>
<td>( \infty )</td>
<td>1-1/( m )</td>
<td>1/( m )</td>
</tr>
</tbody>
</table>

Table 2.[4] The proportion of bits that are 0, 1, \( ... \), \( m-1 \) in various number systems

\[ N = a_m 2^{m-1} + ... + a_2 2^2 + a_1 2^1 + a_0 2^0, \]

where \( a_i \in \{0,1,2,3\} \). This number system is redundant; for example, 101 = 021 = 5. Addition, in this system, is done without carries across long strings of digits. Specifically, carries occur across only three digits. As a result, there is little delay associated with carry. To illustrate, consider the addition of 10231 + 11111. Both 10231 and 11111 are 31 and their sum is 62. The first step is to perform the addition digit by digit, where there is no carry in nor out of each digit. This is illustrated by the first three lines in Fig. 19 below. Notice that the sum, \( Z \), has a digit, 4, that is not allowed in this system. The next step is to convert all digits into their binary equivalent. The maximum digit is 6 (3+3), and so three bits are necessary to represent such a number. The next three lines in Fig. 17 show the binary equivalent of the sum digits. These bits are then added, forming a sum \( S \). In Fig. 19, this is the seventh line. Note that the largest digit in \( S \) is 3, since we add at most three 1 bits. The resulting sum is equivalent to \( Z \) and the digits are between 0 and 3. Thus, it is a number in the number system.

\[ A = \begin{array}{cccc} 1 & 0 & 2 & 3 \\ B = \begin{array}{cccc} 1 & 1 & 1 & 1 \end{array} \end{array} \]

\[ Z = A + B = \begin{array}{cccc} 2 & 1 & 3 & 4 & 2 \end{array} \]

\[ \begin{array}{c} C_0 = \begin{array}{cccc} 0 & 1 & 1 & 0 \end{array} \\ C_1 = \begin{array}{cccc} 1 & 0 & 1 & 0 \end{array} \\ C_2 = \begin{array}{cccc} 0 & 0 & 0 & 1 \end{array} \end{array} \]

\[ S = \begin{array}{cccc} 0 & 1 & 0 & 3 & 1 & 1 & 0 \end{array} \]

Figure 19 [11]. An example of addition in a redundant number system.
In performing the addition, it is important to note that carries occur across three digits only. Fig. 20 shows an adder which performs the addition shown in the example of Fig. 19. This shows clearly the extent to which the carries affect the sum. The longest path in this circuit occurs through a (wired) summation gate, an MVL-to-Binary-Converter, and another summation gate.

![Diagram of MVL to Binary Converter and summation gate](image)

**Figure 20 [11].** An example circuit for the addition of redundant numbers.

It is interesting to note that addition is done in constant time; that is, it is done in time independent of the number of digits.

**VI. CONCLUSIONS**

We have considered number systems spanning from binary to multiple-valued and from non-redundant to redundant. While non-redundant number systems represent more numbers than redundant number systems, the latter has significant advantages with respect to transmission or storage of data and with respect to high-speed arithmetic operations.

**REFERENCES**

ABSTRACT

We survey number systems in which the digits are multiple-valued and the representations are redundant. In a redundant number system, there is at least one value that can be written in at least two ways. As a basis of comparison, we consider also non-redundant binary number systems, including the standard binary number system. We compare systems on the basis of redundancy - that is, how many redundant numbers exist.