FINITE INDUCTIVE SEQUENCE: FACTORING TECHNIQUE

A THESIS
SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF MASTER OF SCIENCE
IN THE GRADUATE SCHOOL OF THE
TEXAS WOMAN'S UNIVERSITY

COLLEGE OF ARTS AND SCIENCES

BY
QUN LU, MASTER OF SCIENCE

DENTON, TEXAS
DECEMBER 1997

TEXAS WOMAN'S UNIVERSITY
DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCE
DENTON, TEXAS

December 1997

To the Dean for Graduate Studies and Research:

I am submitting herewith a thesis written by Qun Lu entitled "Finite Inductive Sequence: Factoring Technique." I have examined this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Mathematics/Computer Science.

Mona Cherri

Mona Cherri, Major Professor

We have read this thesis and recommend its acceptance:

W.D. Zimmermann

Huguette Tran

Turner Hogan

Department Chair

Accepted

Leslie M Thompson

Dean for Graduate Studies and Research

Abstract

"Finite Inductive Sequence: Factoring Technique"

This thesis considers finite sequences of symbols extracted from an alphabet set. In this case, the alphabet set is the positive integers. The underlying hypotheses is that such finite sequence is finitely inductive(FI). Finitely inductive implies that a symbol at any position can be determined by the symbols preceding it. The technique used is called FI - Factoring. FI is primarily used to learn about the presence of relationships between symbols of arbitrary sequences. Note that indicating the presence of a relationship does not necessarily provide information about the nature of that relationship. To gain that knowledge further analysis is required.

Consequently, the pattern recognition technique involves factoring a sequence of data into a series of small sequences called implicants. The collection of implicants is then form a ruling. This ruling is used to match other sequences, and sequences of residual. FI factoring technique focuses on direct analysis of the structure of individual sequences.

Beginning with a given finite sequence of symbols, the factoring algorithm will describe the underlying structure of each sequence. Each input sequence which in this case is finite, is characterized by function tables or ruling describing the structure of each sequence.

By Qun Lu

December, 1997

Texas Woman's University

Denton, Texas

TABLE OF CONTENTS

CHAPTER I

INTRODUCTION

1.1 Problem Statement

This thesis considers finite sequences of symbols extracted from a given alphabet set, and attempts to describe their structure. In general, finite sequences are finitely inductive. Finite induction implies that every symbol at any position in the sequence can be predicted by the preceding p-symbols for some positive integer p. The largest value of such p is called the inductive base.

For this purpose and considering only positive integers for symbols, the factoring method is implemented.

1.2 Thesis Organization

This thesis is organized as follows: Chapter I gives an introduction of the problem; Chapter II provides the conceptual background of finite induction and explains each of these terms: FI, inductive base, function table, factoring, ruling, implicant, and type I storage system. Chapter II concludes by giving the reader a basic knowledge about FI and the techniques used in solving the problem; Chapter III explains the factoring technique which is illustrated by examples; Chapter IV describes the C++ program, lists running results, and includes a brief explanation of outcomes; and Chapter V shows applications of the factoring technique.

CHAPTER II

FINITE INDUCTIVE SEQUENCES

2.1 Introduction

This chapter explains the basic concepts of finite induction including an introduction to finite inductive sequences (FI), inductive base, function table, factoring, ruling, implicant and type I storage system.

2.2 Definitions

Definition 2.2.1:

A finite inductive sequence (FI) over a given alphabet is a sequence of symbols extracted from this set satisfying the following condition: the choice of a letter at any one point is uniquely determined by the choice of the preceding n letters, for some fixed positive number $n \geq 1$ (Cherri, 1996). The least such an integer n is called the inductive base. For example, all finite sequences are finitely inductive. Each FI sequence is represented by a finite set of subsequences with length equal or smaller than an inductive base. The collection is called a function table (Case and Fisher, 1984). In general, an FI sequence can be represented by more than one function table. A possible explanation would be that partitioning a sequence into subsequences of a given length or smaller, depends on the nature of the sequence itself. Note that some sequences with a given value of inductive base cannot be factored.

For example, if the alphabet set is {0, 1} and the sequence is

0010001000011001000100001 1...

It becomes clear that this sequence cannot be describe in one function table, with inductive base 2. Some sequences may contain subsequences of length greater than the inductive base. Therefore, selecting a good value of an inductive base is important (see Chapter 3). The following is an example of a finitely inductive sequence with inductive base 2 over the alphabet {1, 2, 3, 4, 5} and its function table which gives deterministic inductive inference.

<u>Example 2.2.1:</u> The following sequence:

1 4 4 5 3 1 2 1 2 1 2 1 2 1 2 1 2 ...

is described by the function table:

Table A
1 4 → 4
4 4 → 5
4 5 → 3
5 3 → 1
3 1 → 2
1 2 → 1
2 1 → 2

Clearly, after the starting segment 1 4 is given the rest is uniquely determined by this function table.

Hence, from the above discussion a finite inductive sequence is determined by its starting segment and its function table. A given row in the function table is called an <u>implicant</u> (Cherri, 1996). The left side is called the antecedent and the single symbol on the right side the consequent. Referring to example 2.2.1, the antecedents are words of length two; the conclusion is that the sequence has an inductive base less than or equal to two. For the sequence in example 2.2.1, it is not possible to make such a table in which the antecedents are shorter. Therefore, the inductive base is two. If the segment 1 4 4 5 3 is deleted then the rest sequence is periodic.

1 2 1 2 1 2 1 2 1 2 1 2 ...

1 2 is called the period and the initial sequence is called eventually periodic.

Definition 2.2.2:

A sequence $X_1X_2 \ldots X_n \ldots$ is eventually periodic if and only if there exist positive integers N and p such that $X_n = X_{n+p}$ for all n greater than N. The least such positive p is the period.

Every FI sequence is eventually periodic (Gaines 1976, Andrea 1973). The non-periodic part added, is called the transient. If the sequence is infinite, the transient is the shortest starting segment that if deleted will leave a periodic sequence; otherwise, for the finite case the transient is the whole sequence.

Definition 2.2.3:

An FI pair is the pair consisting of the starting segment s and the function table F (i.e. (s, F)). A sequence is finitely inductive, which is equivalent to the existence of an FI pair (Cherri, 1996).

This representation of a pair is not unique. We select the one in which the lengths of the antecedents is least. The least such length is the inductive base b, and the representation is minimal. To reconstruct the original sequence, a final segment or subsequence is required. Consequently, the sequence reconstructed by the FI pair cannot contain a segment of length b which is not an antecedent except as a terminal segment. If the sequence is infinite, every segment of length b is necessarily an antecedent of some implicant in the function table. In both cases antecedents have to be distinct.

Definition 2.2.4:

The extended period or "Eperiod" is the sum of the length of the transient and the period length of the periodic part.

In the case of no transient the Eperiod is equal to the period, and in the case of finite sequences there is no periodic part, the entire sequence is called the transient, and the Eperiod is equal to the length of the sequence (Cherri 1996, Case and Fisher 1984).

Definition 2.2.5:

A function table is said to be in reduced form if each implicant is in reduced form. An implicant is in reduced form whenever it is irredundant as to length. Therefore, it cannot be shortened without changing globally the table. To set ideas, consider the following example.

Example 2.2.2: The FI sequence:

1121:0120122:0120122:0120122:...

The colons are not part of the alphabet, but for periodicity and to help read clearly the sequence, and the following function tables:

Table A	Table B	Table C
(1) 112 → 1 (2) 120 → 1 (3) 210 → 1 (4) 101 → 2 (5) 012 → 0 (6) 120 → 1 (7) 201 → 2 (8) 012 → 2 (9) 122 → 0 (10) 220 → 1	(1) 11210 → 1 (2) 12101 → 2 (3) 21012 → 0 (4) 10120 → 1 (5) 01201 → 2 (6) 12012 → 2 (7) 20122 → 0 (8) 01220 → 1 (9) 12201 → 2 (10) 22012 → 0 (11) 20120 → 1	(1) 11 → 2 (2) 112 → 1 (3) 21 → 0 (4) 0 → 1 (5) 01 → 2 (6) 1012 → 0 (7) 12012 → 2 (8) 22 → 0 (9) 22012 → 0

Three different function tables representing a single finite sequence with inductive base 3, 5, and 5. Table C is the reduced function table. At this point antecedents do not have to have the same length. In this table each antecedent occurs only once, and each implicant is reduced. The inductive base can be calculated from the reduced form table by finding the length of the longest antecedent. To avoid anomalies any FI sequence is characterized by the finitely inductive pair (F, s) where F is the reduced function table and s is the starting segment.

For the sequence the starting segment is 11210. It is clear that the reduced form of a function table can be constructed efficiently. Also, if the reduced form function table is given with the starting segment, the original sequence can be reconstructed. Note that the starting segment for the above sequence can be reduced to 11. The following are the steps that we need to take to recover the entire sequence:

1.	112	(By applying Rule 1 in Table C)
2.	1121	(By applying Rule 2 in Table C)
3.	11210	(By applying Rule 3 in Table C)
4.	112101	(By applying Rule 4 in Table C)
5.	1121012	(By applying Rule 5 in Table C)
6	11210120	(By applying Rule 6 in Table C)
7.	112101201	(By applying Rule 4 in Table C)
8.	1121012012	(By applying Rule 5 in Table C)
9.	11210120122	(By applying Rule 7 in Table C)
10.	112101201220	(By applying Rule 8 in Table C)
11.	1121012012201	(By applying Rule 4 in Table C)
12.	11210120122012	(By applying Rule 5 in Table C)
13.	112101201220120	(By applying Rule 9 in Table C)

Then repeat indefinitely steps 7 through 13. Hence, the entire sequence is recovered by successively generating larger and larger segments of the FI sequence by applying one implicant each time. It is clear that the ruling is selected from the function table, and at that step it is the only ruling that you can apply. The problem of selecting which ruling to apply at what time is nonexistent.

Definition 2.2.6:

Type I storage system consists of a number of "levels." Each level has storage for function table (Case and Fisher, 1984). The method of decomposing any sequence is called factoring. The information stored in a Type I storage system is called a ruling (Cherri, 1996). For this thesis, only finite function tables are considered.

This chapter introduced the basic knowledge underlying FI, function tables, and Type I storage system. The next chapter we will describe the technique used to generate the reduced function table.

CHAPTER III

FACTORING TECHNIQUE

3.1 Introduction

In this chapter, we explain the factoring technique through examples, with emphasis on varying the inductive base and number of levels to generate one-level and multi-level ruling. In addition, shorter and longer inductive bases are used to resolve a specific problem. Several theoretical results such as theorems are listed and their proof is referenced.

3.2 Factoring Technique

Beginning with a given pattern or a finite sequence of symbols, this procedure will describe the underlying structure of the sequence. The solution used is by induction over the length of the antecedent. First, we select antecedents of length one, or all possible subsequences of length two (i.e., only the subsequences within the given sequence). From left to right apply these implicants, and keep the ones that will not produce a contradiction. Repeat the process until the length of the antecedent reaches the inductive base value k or until the sequence is exhausted (whichever comes first) .

Consider the following example, using an inductive base of 3:

Sequence: 41212312341212312341212312341212312341...

Subsequences of length two:

12, 21, 23, 31, 34, 41.

Since 21 and 23 exist concurrently we have a contradiction. The same thing applies for 31 and 34. Therefore, retain only 12, and 41.

Subsequences of length three:

121, 212, 123, 231, 312, 234, 341, 412.

Register 212, 312, 341, and 412.

Subsequences of length four:

1212, 2123, 1231, 2312, 3123, 1234, 2341, 3412, 4121.

Hold only 1212, 2123, 2312, 3123, 2341, 3412, 4121.

The process terminates here, and the first level function table is:

Level 1 Function Table:

Level 1	
1 → 2 4 → 1 21 → 2 31 → 2 34 → 1 41 → 2 121 → 2	212 → 3 231 → 2 312 → 3 234 → 1 341 → 2 412 → 1

Applying each implicant to the original sequence by removing the consequent, the remaining sequence, which we will call first residual is:

414141414…

Repeat the same process for the first residual and call the function table at this point the second level function table.

Level 2 Function Table:

Level 2
4 → 1 1 → 4

and the second residual is nil, and the process stops.

Hence, the factorization of the above sequence is given by the collection of function tables at all levels. Therefore, the ruling is:

Level 1		Level 2
1 → 2	121 → 2	4 → 1
4 → 1	212 → 3	1 → 4
21 → 2	231 → 2	
31 → 2	312 → 3	
34 → 1	234 → 1	
41 → 2	341 → 2	
	412 → 1	

Note that, it is possible to factor concurrently a finite set of FI sequences.

Now, we explain the representation of a finite sequence into a finitely inductive pair (F, s). In general, one such pair may not be sufficient to represent the finite sequence. Depending on the structure of the sequence itself and some restrictions such as the length of the inductive base, representing a given finite sequence may require an ordered set of finitely inductive pairs $(F_1, s_1), (F_2, s_2), \ldots, (F_n, s_n)$. The finite ordered set S $= \{(F_i,s_i) | \ i = 1 \ldots n\}$ is called a ruling (Cherri, 1996). A level i is an ordered pair (F_i, s_i).

At each level a partial sequence is generated. These sequences are interrelated by deletion of certain occurences of symbols. This system of sequences generated by the ruling is developed by an appeal principle: if the function table in one level cannot give the prediction of next symbol, then an appeal to the next higher level is mode. The main advantage of this type of representation is that with one level the inductive base may have to be a lot longer than that with multiple levels. A ruling having a certain inductive base b in each level may represent a sequence having a much longer inductive base.

To illustrate this, we consider the following example:

Example 3.2.1: A ruling

Let $S = \{(F_1,s_1), (F_2,s_2), (F_3,s_3), (F_4,s_4)\}$, and

The Function Tables

F_1	F_2	F_3	F_4
$0 \rightarrow 1$	$0 \rightarrow 2$	$12 \rightarrow 1$ $10 \rightarrow 0$ $20 \rightarrow 0$	$11 \rightarrow 2$ $2 \rightarrow 0$ $0 \rightarrow 2$

and

$s_1 = 11,$ $s_2 = 11,$ $s_3 = 11,$ $s_4 = 11.$

The partial sequences generated by the ruling on each level are as follows:

(spaces and colons are used to aid in reading only)

L4 112: 0 2:0 2:0 2:0 ...

L3 1121:0 0 2:0 0 2:0 0 2: ...

L2 1121:0 20 22:0 20 22:0 20 22: ...

L1 1121:0120122:0120122:0120122: ...

These sequences are constructed concurrently by the appeal principle. Except for the first two columns, all columns are constructed sequentially from left to right. This process is called continuation. Initially, the partial sequences in all levels are equal to the initial segments given s_i. Level 1 will contain the original complete sequence.

At this point, we observe each of the following:

1. The ruling of above example is called a type 1-1-2-2 ruling, (i.e., the inductive base in level 1 is 1, in level 2 is 1, in level 3 is 2, and in level 4 is 2).
2. The ruling represents the sequence in level 1.
3. The respresentation of an FI sequence by a set of simpler FI pairs is called a <u>Factorization</u>.

3.3 Factorization

A ruling is called a "factorization" when it stands in this relation to some sequence. The factorization of a sequence is <u>proper</u> if and only if the inductive base in each level is less than the inductive base of the original sequence. A sequence is said to be <u>factorable</u> if and only if it has a proper factorization; otherwise it is said to be non-factorable. Each storage system should be able to store the function tables and starting chain for each of the levels. In addition, it should have a register for each level which can store any word of length equal to the inductive base in that level. Finally, the system should be able to perform the <u>continuation</u> (Cherri, 1996). One step in continuation is determined as follows: let i be the least integer such that the word in the register in level i is an antecedent w of some implicant $w \rightarrow q$ in the function table for level i. The symbol q is "pushed into the register in this level from the right" for level i and all levels below (i.e., the contents of these registers are shifted once to the left and the new symbol on the right is enforced to be q).

The following is example of how to perform the continuation.

Example 3.3.1: Continuation

Function Table:

F_1	F_2	F_3	F_4
0 → 1 10 → 1 20 → 1	0 → 2	12 → 1 10 → 0 21 → 0	11 → 2 2 → 0 0 → 2

$s_1 = 11$, $s_2 = 11$, $s_3 = 11$, and $s_4 = 11$.

Steps in Continuation

	Step0	Step1	Step2	Step3	Step4	Step5	Step6	Step7	Step8	Step9
L4	11	12	12	20	20	20	20	20	20	02
L3	11	12	21	10	10	10	00	00	00	02
L2	11	12	21	10	10	02	20	20	20	22
L1	11	12	21	10	01	12	20	01	12	22

and if this is repeated indefinitely, then the right most symbol steps through the sequence corresponding to that level. The remaining symbols from the starting segment are placed at the beginning. Note that the original sequence corresponds to level 1

1 1 2 1: 0 1 2 0 1 2 2: 0 1 2 0 1 2 2 ...

It is important to note the following factorization theorem.

3.4 Factoring Theorems

Factoring Theorem 1:

The total number of implicants in a factorization of a given FI sequence is approximately equal to number of implicants in the given FI sequence, provided that all function tables are in reduced form(case and Fisher 1984).

Although this seems to be a negative result for factoring, the lengths of the antecedents may be shorter by orders of magnitude than those in the original sequence. To relate each level function table to the function table of the original sequence is difficult.

In a factorization of a given FI sequence, the function table for the lowest level is a subset of the function table of the given sequence; provided all function tables are in reduced form.

To explain this, it suffices to remember that the sequence generated by continuation in level 1 is the same as the original sequence and since the function table in the lowest level is reduced form, it consists of reduce form implicants of the original sequence. Therefore, the set inclusion is established.

Factoring Theorem 2:

Let S be an FI sequence, and I a set of implicants of S. The "residual sequence" R obtained from S by applying (or pushing down) the set of implicants I is defined to be that sequence obtained from S by deleting all occurrences of consequents of elements of I.

If the function table for an FI sequence is in reduced form, the residual sequence obtained by applying a set of implicants is the same as the sequence obtained by deleting the occurrence of the consequences of these implicants. In addition, the set of implicants can be reconstructed from the set of occurences of these deleted symbols (Cherri, 1996).

Up to this point, the method explained is an attempt to find a factorization of a given FI sequence so that the inductive base in each level is less than or equal to a given positive integer k. Our goal is to find an efficient optimal way of describing the structure if such sequences, and use these structures to find the best match. The existence and unqueness of such factorization is of major interest. Next, we will state one of the main results of factorization for special binary sequences.

Factoring Theorem 3:

Any binary periodic FI sequence having period less than 2^k, has a factorization in which the inductive base in each level is less than k.

The proof is by induction on the period length and is found in the appendix of Case and Fisher (1984).

3.5 Factoring Examples

In this section we will list few examples of factoring specific sequences. We will start with a periodic sequence.

Example 3.5.1: Factoring a periodic FI sequence

Sequence, 22: 0101120122: 0101120122: 0101120122: …

The reduced function table of the given sequence:

F_1	
22	$\rightarrow$ 0
0	$\rightarrow$ 1
2201	$\rightarrow$ 0
101	$\rightarrow$ 2
1012	$\rightarrow$ 0
1201	$\rightarrow$ 2
2012	$\rightarrow$ 2

and assuming that the inductive base is two. Therefore, the set of implicants that we can apply is {22 $\rightarrow$ 0 and 0 $\rightarrow$ 1} and, the first residual is:

22: 02022: 02022: 02022: …

the function table for the first residual:

F_2	
22	$\rightarrow$ 0
0	$\rightarrow$ 2

the second residual:

02: 02: 02: 02: 02: …

the function table for the second residual:

F_3	
0	$\rightarrow$ 2
2	$\rightarrow$ 0

The resulting factorization is given by the three function tables as follows:

Function tables:

F1	F2	F3
22 → 0 0 → 1	22 → 0 0 → 2	0 → 2 2 → 0

and, the starting segments

$s_1 = 22$ $s_2 = 22$ and $s_3 = 22$

Therefore, the resulting system of sequences at different levels are:

L3 02: 0 2: 0 2: 0 2: …

L2 22: 0 20 22: 0 20 22: 0 20 22: …

L1 22:0101120122: 0101120122: 0101120122: …

Here the factorization is constructed with inductive base in each level less than or equal to two. The following example is a sequence which has a long string of occurrence of the same symbols or pattern as sequentially factorable.

Example 3.5.2: A sequentially factorable sequence.

The sequence is:

1 222 222 222: 1 222 222 222: 1 222 222 222: …

and, the sequence factorization is given by the following table.

Function table 3.5.3:

F1	F2	F3	F4	F5
$1 \rightarrow 2$ $12 \rightarrow 2$	$1 \rightarrow 2$ $12 \rightarrow 2$	$1 \rightarrow 2$ $12 \rightarrow 2$	$1 \rightarrow 2$ $12 \rightarrow 2$	$1 \rightarrow 2$ $2 \rightarrow 1$

and, the starting segments are:

$s_1 = 1$, $s_2 = 1$, $s_3 = 1$, $s_4 = 1$, and $s_5 = 1$.

The logical step that follows factorization, is matching. The matching procedure must answer each of the following questions: first, how does this representation help us distinguish one sequence from another, and how efficient is this representation in finding the closest match? Several criterias can be used to evaluate closeness, and to find a best match.

Clearly, from earlier discussion we can conclude that not all finitely inductive sequences are factorable. But, it was shown (case and Fisher 1984) that the factorable sequences form a large proportion of the totality of finitely inductive sequences. Case and Fisher (1984) classified the non-factorable sequences in two classes that they named pseudo-random and raw counts. To learn more about these sequences refer to Case and Fisher (1984). If the proportion of non-factorable sequences to all FI sequences (having a certain inductive base) was too large then this study is mathematically inadequate. But, Case and Fisher (1984) showed that this proportion is very small.

CHAPTER IV

PROGRAM AND RESULT

4.1 Introduction

In this chapter, we implement the C++ program, present running results, and discuss outcomes.

4.2 Problem Solving Using C++ Program

The purpose of this program is to factor sequences of positive integers. This program is written in C++.

Input:

A sequence of positive integers.

The maximum length for each level (Inductive Base).

The number of levels.

Output:

The ruling of levels.

Residual for each levels.

Data Structures:

A one-dimensional array of strings representing sequence numbers (sequence).

A one-dimensional array of residuals (residual).

Main Level 0

```
Read data
Input Inductive Base and Number of Levels
Factor
Print ruling and residuals
```

To open the input file, we will make our program more flexible by prompting the user for a file name and reading it in as a string.

Open For Input (Inout: someFile) Level 1

```
Prompt user for name of file
Read fileName
Associate fileName with stream someFile,
   and try to open it
IF file could not be opened
   Print error message
```

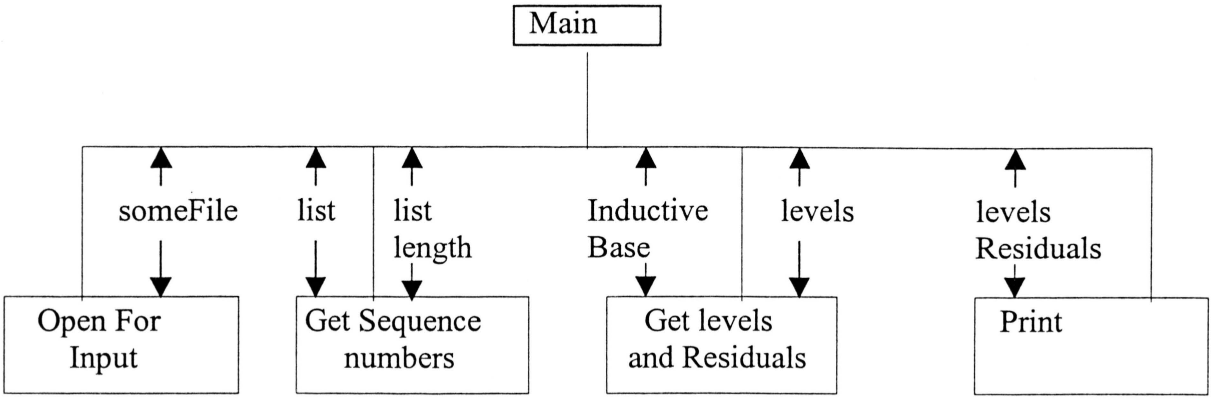
Get sequence numbers

```
Set length = 0
Read numbers from Input File
WHILE NOT EOF on Input File
     Get number into sequence list,
          incrementing length
      Let residual equals sequence list
      Read number from Input File
```

Get Levels and Residuals

```
If Inductive Base < Length Of Sequence
   Then Do
           Set Level = 1
           Set implicant length to 2
             Generate all implicants with length 2
             Increment implicant length
             Loop until length of implicant = the inductive base
             Generate Residual
Print level ruling and Residual
  Loop until number of levels = Input of Number of Levels
```

Module Structure Chart:



4.3 Running Results

Sequence 1: 1 1 2 0 2 1 1 3

Number of Levels = 3, and **Inductive Base** = 2

Ruling:

Level 1	Level 2	Level 3
0 → 2 12 → 0 20 → 2 02 → 1 21 → 1	2 → 3 11 → 2 12 → 3	1 → 1
Residual = 1123	Residual = 11	Residual = 1

Sequence 2 : 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0

Number of Levels = 3, and **Inductive Base** = 1

Level 1 - residual = 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0

Level 2 - residual = 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0

Level 3 - residual = 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0

No implicant were generated for this inductive base, so we have to change the inductive base as following.

Number of Levels = 4, and **Inductive Base** = 2

Ruling:

Level 1	Level 2	Level 3	Level 4
10 → 0 01 → 1 10 → 0	10 → 0 01 → 1 10 → 0	10 → 0 01 → 1 10 → 0	10 → 0 01 → 1 10 → 0

Level 1 - residual = 11111111000000011111110000000

Level 2 - residual = 1111111100000011111000000

Level 3 - residual = 111111110000011111100000

Level 4 - residual = 11111111000011110000

Number of Levels = 4, and **Inductive Base** = 4

Ruling:

Level 1	Level 2	Level 3	Level 4
10 → 0 01 → 1 110 → 0 100 → 0 011 → 1 1110 → 0 1100 → 0 1000 → 0 0001 → 1 0011 → 1 0111 → 1 1110 → 0 1100 → 0 1000 → 0	10 → 0 01 → 1 110 → 0 100 → 0 001 → 1 011 → 1 1110 → 0 1100 → 0 1000 → 0 0001 → 1 0011 → 1 0111 → 1	10 → 0 00 → 1 01 → 1 110 → 0 100 → 1 001 → 1 011 → 0 1110 → 0 1100 → 1 1001 → 1 0011 → 0 0110 → 0	

Level 1 - residual = 1111111100000111100000

Level 2 - residual = 11111111001100

Level 3 - residual = 111111110

Level 4 - residual = 111111110

Number of Levels = 3, and **Inductive Base** = 6

Ruling:

Level 1			Level 2			Level 3
10 → 0	11110 → 0	110000 → 0	10 → 0	1110 → 0	001110 → 0	
01 → 1	11100 → 0	100000 → 0	01 → 1	11110 → 0	011100 → 0	
110 → 0	11000 → 0	000001 → 1	110 → 0	11000 → 1		
100 → 0	10000 → 0	000011 → 1	100 → 0	10001 → 1		
001 → 1	00001 → 1	000111 → 1	000 → 1	00011 → 1		
011 → 1	00011 → 1	001111 → 1	001 → 1	00111 → 0		
1110 → 0	00111 → 1	011111 → 1	011 → 1	01110 → 0		
1100 → 0	01111 → 1	111110 → 0	1110 → 0	111110 → 0		
1000 → 0	11110 → 0	111100 → 0	1100 → 0	111100 → 0		
0001 → 1	11100 → 0	111000 → 0	1000 → 1	111000 → 1		
0011 → 1	111110 → 0	110000 → 0	0001 → 1	110001 → 1		
0111 → 1	111100 → 0		0011 → 1	100011 → 1		
1110 → 0	111000 → 0		0111 → 0	000111 → 0		

Level 1 - residual = 11111111000111000

Level 2 - residual = 111111110

Level 3 - residual = 111111110

Number of Levels = 3, and **Inductive Base** = 7

Ruling:

Level 1			Level 2		Level 3
10 → 0	00001 → 1	1110000 → 0	10 → 0	111110 → 0	
01 → 1	00011 → 1	1100000 → 0	00 → 1	111100 → 1	
110 → 0	00111 → 1	1000000 → 0	01 → 1	111001 → 1	
100 → 0	01111 → 1	0000001 → 1	110 → 0	110011 → 0	
001 → 1	111110 → 0	0000011 → 1	100 → 1	100110 → 0	
011 → 1	111100 → 0	0000111 → 1	001 → 1	1111110 → 0	
1110 → 0	111000 → 0	0001111 → 1	011 → 0	1111100 → 1	
1100 → 0	110000 → 0	0011111 → 1	1110 → 0	1111001 → 1	
1000 → 0	100000 → 0	0111111 → 1	1100 → 1	1110011 → 0	
0001 → 1	000001 → 1	1111110 → 0	1001 → 1	1100110 → 0	
0011 → 1	000011 → 1	1111100 → 0	0011 → 0		
0111 → 1	000111 → 1	1111000 → 0	0110 → 0		
1100 → 0	001111 → 1	1110000 → 0	11110 → 0		
11110 → 0	011111 → 1	1100000 → 0	11100 → 1		
11100 → 0	1111110 → 0	1000000 → 0	11001 → 1		
11000 → 0	1111100 → 0		10011 → 0		
10000 → 0	1111000 → 0		00110 → 0		
Residual = 11111111001100			Residual = 111111110		Residual = 111111110

Number of Levels = 3, and **Inductive Base** = 8

Ruling:

Level 1				Level 2	Level 3
10 → 0	10000 → 0	1111110 → 0	11111000 → 0	1 → 1	
01 → 1	00001 → 1	1111100 → 0	11110000 → 0	11 → 1	
110 → 0	00011 → 1	1111000 → 0	11100000 → 0	111 → 1	
100 → 0	00111 → 1	1110000 → 0	11000000 → 0	1111 → 1	
001 → 1	01111 → 1	1100000 → 0	10000000 → 0	11111 → 1	
011 → 1	111110 → 0	1000000 → 0	00000000 → 1	111111 → 1	
1110 → 0	111100 → 0	0000001 → 1	00000001 → 1	1111111 → 1	
1100 → 0	111000 → 0	0000011 → 1	00000011 → 1		
1000 → 0	110000 → 0	0000111 → 1	00000111 → 1		
0001 → 1	100000 → 0	0001111 → 1	00001111 → 1		
0011 → 1	000001 → 1	0011111 → 1	00011111 → 1		
0111 → 1	000011 → 1	0111111 → 1	00111111 → 1		
11110 → 0	000111 → 1	11111111 → 0	01111111 → 1		
11100 → 0	001111 → 1	11111110 → 0			
11000 → 0	011111 → 1	11111100 → 0			
Residual = 11111111				Residual = 1	Residual = 1

<u>**Sequence 3 :**</u> 1 2 3 4 5 6 7 8 9 0

Number of Levels = 2, and **Inductive Base** = 1

Ruling:

Level 1	Level 2
1 → 2 2 → 3 3 → 4 4 → 5 5 → 6 6 → 7 7 → 8 8 → 9 9 → 0	
Residual = 1	Residual = 1

<u>**Sequence 4 :**</u> 1 2 3 4 5 6 1 2 3 4 5 6 7 1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8 9 1 2 3 4 5 6 7 8 9 0

Number of Levels = 2, and **Inductive Base** = 1

Ruling:

Level 1	Level 2
1 → 2 2 → 3 3 → 4 4 → 5	
Residual = 117178178917890	Residual = 117178178917890

Number of Levels = 4, and **Inductive Base** = 3

Ruling:

Level 1		Level 2	Level 3	Level 4
1 → 2	91 → 2	11 → 7	91 → 9	1 → 1
2 → 3	123 → 4	71 → 7	111 → 9	11 → 1
3 → 4	234 → 5	81 → 7	119 → 1	
4 → 5	345 → 6	91 → 7	191 → 9	
5 → 6	561 → 2	117 → 1	919 → 0	
12 → 3	612 → 3	171 → 7		
23 → 4	671 → 2	717 → 8		
34 → 5	712 → 3	781 → 7		
45 → 6	781 → 2	817 → 8		
61 → 2	812 → 3	891 → 7		
71 → 2	891 → 2	917 → 8		
81 → 2	912 → 3			
Residual = 117178178917890		Residual = 1119190	Residual = 111	Residual = 1

<u>**Sequence 5 :**</u> 0 0 0 0 1 0 0 0 1 1 0 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 1 1 0 1

Number of Levels = 2, and **Inductive Base** = 2

Level 1 - residual = 0 0 0 0 1 0 0 0 1 1 0 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 1 1 0 1

Level 2 - residual = 0 0 0 0 1 0 0 0 1 1 0 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 1 1 0 1

No implicant were generated for this inductive base, so we have to change the inductive base as following.

Number of Levels = 5, and **Inductive Base** = 3

Ruling:

Level 1	Level 2	Level 3	Level 4	Level 5
010 → 0	010 → 1	010 → 0	010 → 0	010 → 0
100 → 0	101 → 1	011 → 1	011 → 1	100 → 1
	100 → 0			

Level 1 - residual = 0000101100000111100011111101

Level 2 - residual = 0000100001110011111101

Level 3 - residual = 000010001110011111101

Level 4 - residual = 00001001100111101

Level 5 - residual = 00001010011101

Number of Levels = 8, and **Inductive Base** = 3

Ruling:

Level 1	Level 2	Level 3	Level 4
010 → 0 100 → 0	010 → 1 101 → 1 100 → 0	010 → 0 011 → 1	010 → 0 011 → 1

Level 5	Level 6	Level 7	Level 8
010 → 0 100 → 1	101 → 0 100 → 1 011 → 1 111 → 0 110 → 1	1 → 0 01 → 0 10 → 1 001 → 0 010 → 1 101 → 0	

Level 1 - residual = 00001011000011110001111101

Level 2 - residual = 0000100001110011111101

Level 3 - residual = 000010001110011111101

Level 4 - residual = 00001001100111101

Level 5 - residual = 00001010011101

Level 6 - residual = 000010101

Level 7 - residual = 00001

Level 8 - residual = 00001

Number of Levels = 4, and **Inductive Base** = 4

Ruling:

Level 1	Level 2	Level 3	Level 4
010 → 0	010 → 1	010 → 0	
100 → 0	101 → 1	100 → 1	
0010 → 0	100 → 0	011 → 1	
0100 → 0	0000 → 1	111 → 0	
0110 → 0	0010 → 1	110 → 1	
1100 → 0	0101 → 1	0001 → 0	
0111 → 1	1011 → 0	0010 → 0	
	0110 → 0	0100 → 1	
	1100 → 0	1001 → 1	
	1000 → 1	0011 → 1	
	0011 → 1	0111 → 0	
		1110 → 1	

Level 1 - residual = 00001011000111000111111101

Level 2 - residual = 0000010011101

Level 3 - residual = 000001

Level 4 - residual = 000001

Number of Levels = 4, and **Inductive Base** = 5

Ruling:

Level 1		Level 2		Level 3	Level 4
010 → 0	01100 → 0	010 → 0	00100 → 0	0 → 0	
100 → 0	11000 → 0	0000 → 1	01000 → 1	00 → 0	
0010 → 0	00000 → 1	0010 → 0	10001 → 1	000 → 0	
0100 → 0	00111 → 1	0100 → 0	00011 → 0		
0110 → 0	11100 → 0	1000 → 1	00110 → 0		
1100 → 0	00111 → 1	0110 → 0	01100 → 1		
0111 → 1		1100 → 1	11001 → 1		
00010 → 0		1001 → 1	10011 → 1		
00100 → 0		0111 → 1	00111 → 1		
01000 → 1		1110 → 1	01111 → 1		
10001 → 1		00001 → 0	11111 → 0		
00110 → 0		00010 → 0	11110 → 1		

Level 1 - residual = 00001000110011111101

Level 2 - residual = 0000

Level 3 - residual = 0

Level 4 - residual = 0

Sequence 6 : 1 1 2 1 0 1 2 0 1 2 2 0 1 2 0 1 2 2 0 1 2 0 1 2 2

Number of Levels = 3, and **Inductive Base** = 4

Ruling:

Level 1		Level 2	Level 3
0 → 1 11 → 2 21 → 0 10 → 1 01 → 2 20 → 1 22 → 0 112 → 1 121 → 0 210 → 1 101 → 2 120 → 1 201 → 2	122 → 0 220 → 1 201 → 2 1121 → 0 1210 → 1 2101 → 2 1012 → 0 0120 → 1 1201 → 2 0122 → 0 1220 → 1 2201 → 2	2 → 0 0 → 2 11 → 2 12 → 0 20 → 2 02 → 0 112 → 0 120 → 2 202 → 0 020 → 2 1120 → 2 1202 → 0 2020 → 2	1 → 1
Residual = 1120202		Residual = 1	Residual = 1

Sequence 7 : 1 2 1 2 3 1 2 3 4 1 2 1 1 3 1 2 3 4 1 2 1 2 3 1 2 3 4 1 2 1 2 3 1 2 3 4 1

Number of Levels = 5, and **Inductive Base** = 2

Ruling:

Level 1	Level 2	Level 3	Level 4	Level 5
4 → 1 31 → 2 34 → 1 41 → 2 11 → 3 13 → 1	4 → 1 21 → 2 23 → 1 31 → 3 13 → 4 34 → 1 11 → 3	21 → 3 13 → 1 31 → 2 23 → 2 32 → 3	2 → 1 12 → 1 21 → 3	1 → 2

Level 1 - residual = 121231341134123134123134

Level 2 - residual = 121312323

Level 3 - residual = 1213

Level 4 - residual = 12

Level 5 - residual = 1

<u>**Sequence 8 :**</u> 2 2 0 1 0 1 1 2 0 1 2 2 0 1 0 1 1 2 0 1 2 2 0 1 0 1 1 2 0 1 2 2

Number of Levels = 3, and **Inductive Base** = 3

Ruling:

Level 1	Level 2	Level 3
0 → 1 22 → 0 20 → 1 10 → 1 11 → 2 220 → 1 010 → 1 101 → 1 011 → 2 112 → 0 120 → 1 012 → 2 122 → 0	0 → 2 22 → 0 20 → 2 02 → 0 220 → 2 202 → 0 020 → 2	2 → 2
Residual = 22020202	Residual = 22	Residual = 2

4.4 Conclusion

In conclusion, it is clear that varying the inductive base as well as the number of levels is a necessary step towards accomplishing good output. At time and for a specific small number of sequences, this factoring process fails. For example, the binary sequence (Sequence 5) could not be factored for a short inductive base and a small number of levels. The residual is the original sequence. Hence, modification of factoring algorithm is necessary to generate a ruling.

CHAPTER V

USES OF THE FACTORING TECHNIQUE AND CONCLUSION

5.1 Uses Of The Factoring Technique

Primarily, FI is used to learn about the presence of relationships between symbols of arbitrary sequences. Note that indicating the presence of a relationship does not necessarily provide information about the nature of that relationship. An immediate consequence of identifying relationships would be to find if the same relationships exist in other sequences. Thus, a matching and pattern identification algorithms will follow.

Secondly, the factorization process can be extended to a set of sequences. Simultaneous factoring of a finite set of sequences requires a set of starting chains rather than just one. This method of storing a set of closely related sequences is plausible and powerful, and it can be used to identify and discriminate between sequential patterns. Here, implicants are generated for all sequences in the set, that have minimal length antecedent. The factorization process is identical to the one used for one sequence, but instead of generating one residual, a sequence of residuals is generated at each level of factorization. The implicants generated at each level is done by adjoining implicants generated at that level for each individual sequence. Therefore, an implicant may have a shorter antecedent with respect to another, and a set of sequences may have some implicants in common.

Ultimatelly, parallel factoring of a set of sequences gives rise to a sort of cluster analysis. This method can be used to differentiate between two or more sequences and to find common subpatterns. It also gives an efficient, non-redundant coding for the set of sequences. It seems like this process would factor out the common part of the set of sequences leaving exactly enough information to discriminate between them. Therefore, a class of sequences can be generated and for a new given sequence we can determine what class it belongs to. If the newly presented sequence contradicts the function tables of the original set, then we will classify it as unknown to the system (or foreign). Hence, at hand a better method to recognize patterns.

Finally, the factorization process relies upon details of deterministic inductive inferences, so it only applies to data having some amount of causality. Also, it can be used to extract causal information from mixed data. This process presents an alternative method to build structures, and gives a new view of the raw data.

5.2 Conclusion

It seems that this model applied to one or any sequence of observations is of interest. By using this technique, we are hoping to efficiently classify patterns into different classes according to their main structure. In the factoring technique, the main pattern is distinguished from the subpatterns. A small local change will not influence the global framework.

REFERENCES

Andrea, J. H. 1973. Work on a "river" program: PUSS, Man-Mach. Studies Report VC-DSE/3, pp. 6-43.

Case, J., and P. Fisher. 1984. Learnable Sequences / A Prealgebra

Case, J., and P. Fisher. 1984. Long term memory module. Bulletin of Mathematical Biology 46 (2).

Cherri, Mona Y. 1996. Automatic Speech Recognition Using Finite Inductive Sequences.

Dale,N., Weems C., and Headington M. 1997. Jones and Bartlett publishers. Programming and Problem Solving with C++.

Gaines, J., L. Lamel, W. Fisher, J.Fisus, D. Pallet, and N. Dahlgren. 1990. Acoustic-phonetic Continuous Speech Corpus: DARPA, the TIMIT database. NISTIR 4930.