Efficient Lossless Data Compression using Advanced Search Operators and Genetic Algorithms

Angel F. Kuri[1], Oscar Herrera[2]

[1] Instituto Autónomo de México,
Río Hondo No. 1, México D.F.
akuri@itam.mx
Centro de Investigación en Computación,
Av. Juan de Dios Batiz s/n, México D. F.
heoscar@yahoo.com

Abstract. Over the last decades several techniques for lossless data compression have been proposed (such as RLE[1], Huffman Coding[2], Arithmetic Coding[3], LZ[4], BWT[5], and PPM[6]). They all assume or construct a model of the source which generates the message and the compression ratio depends strongly on how accurately the source adheres to the premises. In this paper we propose a new method for efficient lossless data compression in which the original message generated by a non-ergodic source is transformed into another in which its behaves as if it were generated by an ergodic one. This is achieved by identifying a set of groups which display ergodicity when looked upon as independent symbols. The identification of such groups represents an NP Problem[7][8][9] and we, therefore, define a set of Advanced Search Operators which, when combined with a Non Traditional Genetic Algorithm, allow us to find a set of non correlated groups of symbols to achieve the desired transformation and achieve maximum compression.

Keywords. Data Compression, RLE, Huffman, LZ, BWT, PPM, ergodic, NP Problem, Search, Genetic Algorithm.

1 Introduction

RLE is a simple data compression technique restricted to messages with strings of repeated symbols called "runs". When there is no "runs" RLE easily give us negative compression. More sophisticated techniques such as Huffman Coding and its parent the Arithmetic Coding are based on the Shannon Information Theory [10] that presents a study for ergodic sources where entropy represents the theoretical limit for data compression. The best coding of Huffman technique is achieved when the average length of codes reaches the entropy assuming that the receptor knows the dictionary. However, when we consider the structure of the dictionary as an additional header the "best coding" may give us a negative compression. Besides, Huffman Codes does not give us an acceptable compression in case of equiprobable symbols and is unable to compress bits streams.

Adaptive Dictionary techniques such as LZ build a dictionary with groups of adjacents symbols, in consequence, they assume that source is ergodic and the maximum compression ratio is restricted to messages with these characteristics.

PPM techniques calculate the conditional probability of next symbol based on k previous symbols. That implies to correlate the current symbol with the k symbols which precede it.

In our model we transform a message from a non-ergodic source to other which can be view as it were generated by an ergodic source consequently can be optimally coded with some technique that assumes a ergodic source and so we achieve maximum compression.

We intend to correlate as is possible the symbols in a group at the same time we intend to uncorrelated the groups one each other. That groups of symbols have been called metasymbols and they are discussed in section 2.

Since we do not define a priori the number of groups, neither the number of symbols that compose each one and symbols are not necessarily adjacent we have a NP problem[11]. To find metasymbols we have proposed to use:

1. A Non Traditional Genetic Algorithm. The fitness measure not only represents a theoretical limit of compression (as entropy does) but also it considers in an intrinsic way the size of the dictionary (contrary to the entropy). Fitness measure represents the number of bits used to fully encode a message (header plus encoded message), in consequence when the fitness is less than the original size of the message we have a positive compression.
2. A set of Advanced Operators for Metasymbol Searching (AOMS).

Both of them are discussed in sections 3 and 4.

2 Metasymbols

To facilitate the description of metasymbol we should conceive a message as an array of symbols indexed from 0 to L-1 where L is the length or number of symbols that constitute the message. Before describing a metasymbol is necessary to explain two kind of groups which appear when we find groups of symbols.

2.1 Master Groups and Slave Groups

For simplicity we can think that a metasymbol is a pattern in the message, a group is an instance of that pattern, a master group is the first instance we found in the message (from left to rigth) and a slave group is a repeated group that previously has appeared as master.

Groups are written as a set of symbols s_z^{ij} where subindex z means the position in the original message, superindex i is a number of group and superindex j indicates a relative index into the group.

Here are some examples:

Master group:

$$\{ x_0^{0,0},\ y_1^{0,1},\ z_3^{0,3} \}$$

Slave groups:

$$\{ x_4^{1.0},\ y_5^{1.1},\ z_7^{1.3} \}$$
$$\{ x_8^{2.0},\ y_9^{2.1},\ z_{11}^{2.3} \}$$
$$\{ x_{12}^{3.0},\ y_{13}^{3.1},\ z_{15}^{3.3} \}$$

Metasymbol:

$$\{ x_0,\ y_1,\ z_3 \}$$

The difference between a master group and a metasymbol is that a metasymbol is a pattern without absolute indices referenced to the message. In a master group the subindices reference the position of the symbol in the original message, whereas, the subindices in a metasymbol represents the offset from the first symbol to any other into the metasymbol, that is why the first subindex always is 0.

Superindices j allows to compare groups to know whether a groups is slave of a master, i.e., all the superindices j of the two groups must match one to one.

The necessary number of metasymbols to codify a message is denoted by |M|, where $M = \{\alpha, \beta, \chi, \delta, \varepsilon, ...\}$. In what follows we use Greek letters to represent metasymbols. By definition, a metasymbol is considered different from another one if:

- The constituent symbols differ from those of any other metasymbol.

  Example: $\alpha = a_0 b_1 c_2$

  $\beta = a_0 e_1 f_2$

- They differ in the subindices of the symbols that compose them.
  Example:

  $\alpha = a_0 a_1 a_2$

  $\alpha = a_0 a_1 a_1$

- They differ in the number of symbols that constitute the metasymbol, called in what follows, length of the metasymbol.
  Example:

  $\alpha = a_0 b_1 c_2$

  $\beta = a_0 b_2 c_1 d_3$

  $\text{length}(\alpha) = 3$

  $\text{length}(\beta) = 4$

3. Non Traditional Genetich Algorithm: Vasconcelos Genetic Algorithm

The Vasconcelos Genetic Algorithm [12] use anular crossover. A variant of Vasconcelos has been proposed using PMX [13] proposed by Goldberg. Vasconcelos Genetic Algorithm is identified essentially because use a crossover between the best and the worst individuals in a population.

3.1 Coding of the genome

The genome is built as a permutation of the indices of the message using weight binary coding. For a message with length L we need $\log_2(L)$ bits for each index and $L^*\log_2(L)$ bits for the entire genome.

The groups in the genome are self-delimited because the indices of symbols which belongs to the same group appear ordered of ascending way (from left to right) and if appear an index less than its predecessor means a change of group.

We show an example for the message $x_0y_1A_2z_3x_4y_5B_6z_7x_8y_9C_{10}z_{11}$ where we have identified four metasymbols summarized in table 2 including the master and slaves groups associated to each metasymbol. The corresponding genome is:

1010 1000 1001 1011 0110 0100 0101 0111 0010 0000 0001 0011

Where is possible to identify the groups:

G1: 10
G2: 8,9,11
G3: 6
G4: 4,5,7
G5: 2
G6: 0,1,3

Comparing G2, G4 and G6 we know that G6 is a master group and its slaves are groups G2 and G4.

Groups G1, G3 and G5 are master groups because there is no repetition of them.

Table 1. Metasymbols, Master Groups and Slaves Groups for the message $x_0y_1A_2z_3x_4y_5B_6z_7x_8y_9C_{10}z_{11}$

Metasymbol	Master Groups	Slave Groups
$\alpha=x_0, y_1, z_3$	$x_0^{0.0}$, $y_1^{0.1}$, $z_3^{0.3}$	$x_4^{1.0}$, $y_5^{1.1}$, $z_7^{1.3}$ $x_8^{2.0}$, $y_9^{2.1}$, $z_{11}^{2.3}$
$\beta=A_0$	$A_2^{0.0}$	
$\chi=B_0$	$B_6^{0.0}$	
$\delta=C_0$	$C_{10}^{0.0}$	

3.2 Mutation

Note that change a bit from 0 to 1 or from 1 to 0 in the genome can produce an index out of range or duplicated, this is the reason to use other kind of mutation. Alternatively, we propose to permute two indices, so, we will never have indices out of range or duplicated and is no necessary to repair. Mutation is made with probability *Pmut*.

3.3 Crossover

Similarly to the mutation, annular crossover can produce indices out of range or duplicated, then crossover is implemented using PMX. Crossover is made with probability *Pcross*.

4 Advanced Operators for Metasymbol Searching

We define a set of operators which allows to manipulate a message in order to find metasymbols.

Definition. ε is a empty symbols used to indicate absence of symbol.

Arr(msg). Take a message and associate consecutive indices starting with 0 y ending with L-1.

|Arr(msg)|. The number of symbols different of ε contained in Arr(msg) .

$f_{Arr}(s_i)$. Calculate the frequency of symbol s_i in the array Arr.

$F_{(Obj)}( )$. Calculate the distribution of frequencies of symbols contained in an object such as a matrix, an array, acolumn of a matrix, etc., and is represented by a set of pair $(s_i, f(s_i))$.

Roulette$(F_{(Obj)}( ))$. Given a distribution of frequencies of symbols s_i select any s_i using proportional selection.

TH( ). Return a random number between 1 and L inclusive which will act as a threshold for the minimum number of repetitions of a group.

Pos(s_i). Is an array of indices where symbol s_i appears in the array Arr.

Pos$(s_i)_k$. The k-th element of *Pos*(s_i).

Right(Arr, n_k). An array of size L which first n_k symbols are the last n_k symbols of Arr and the rest is completed with empty symbols ε. The subindices starts with 0 and finish with L −1. Each symbol has associated a pair of superindices (n_k, i) with i=0,1,2,..., L-1.

$Vsi_{pos(si)k}$ = Right(Arr,n_k) where n_k = L-pos$(s_i)_k$

SMsi={$Vsi_{pos(si)k}$, k=0,1,2,...,f(si)-1}

Sel$(Cj^*, s_i^*)_{SMsi}^{Pcol}$. Because SMs_i can be view as a matrix it means to get from SMs_i the column Cj that contains the symbol s_i with the highest frequency, minimum j and that has not been selected previously. s_i^* means all the instances of symbol s_i contained in column Cj. *Pcol* is the probability of selecting any other column and symbol which do not comply with the restrictions previously described.

$|s_i^*|$ The number of instances of symbol s_i contained in column Cj

Elim$(SMsi\ s_{e1,e2,...,en})$= SMsi replacing s_{e1}^{ij} with a empty symbol ε_{e1}^{ij} and whenever s_{e1}^{ij} dont be the j-th element of Vk in SM_A

Elim$(Arr\ s_{e1,e2,...,en})$= Arr replacing s_{e1}^{ij} with ε_{e1}^{ij}

Elim$(Arr_{Pos(si)})$= SM_A replacing $s_{Pos(si)k}^{ij}$ with ε_{e1}^{ij} where k=0,1,2,...f(s_i)-1

Restrict$(SM_A\ s_{e1,e2,...,en})$ ={V_k of SM_A such that e_i belongs to V_k}

AddColumn(Cj)*. Each time we apply *Sel*$(Cj^*, s_i^*)_{SMsi}$ we get a column and its number of column j which are stored in an array of columns.

EmptyArr( ). Reset the array of columns Cj*.

MatrixofMasterAndSlaves( ). After apply several times *Sel*$(Cj^*, s_i^*)_{SMsi}$ we get an array of columns that is possible to order under j in ascendant order and join them to construct a matrix. That matrix contains master and slaves groups as rows.

GetMasterAndSlaves(MatrixofMasterAndSlaves). Given a matrix with groups is possible to identify the master group comparing the first subindex of all of them. The group which have the minimum subindex is the master group.

GetMetasymbol( ). Given a master group is possible to get a metasymbol in the next way:

Master group:

$\{ s_{z0}^{i,j,0}, \; s_{z1}^{i,j,1}, \; s_{z2}^{i,j,2}, \ldots, s_{zn}^{i,j,n} \}$

Metasymbol:

$\{ s_{j0}, \; s_{j1}, \; s_{j2}, \ldots, s_{jn} \}$

Once defined the operators we can describe the Algorithm for searching metasymbols in a compact way.

1. *Arr(msg).*
2. EmptyArr( ).
3. $F_{Arr}( )$
4. *DELIM= Roulette*($F_{Arr}( )$).
5. *Pos(DELIM).*
6. SMsi
7. th=*TH( )*
8. *AddColumn(0)*
9. *for i=0 to L-1 do* F_{Cj}
10. $\mathrm{Sel}(Cj^*, s_i^*)_{Smsi}^{Pcat}$
11. if $|s_i^*| <$ Th go step 17
12. *AddColumn(Cj*).*
13. *Elim(SMsi* s_i^**)*
14. *Elim(Arr* s_i^**)*
15. SM_{ij}=*Restrict*(SM_{ij} $s_{e1,e2,\ldots,en}$) ={V_k of SM_A such that e_i belongs to V_k}
16. repeat step 9 to 14
17. *MatrixofMasterAndSlaves( ).*
18. *GetMasterAndSlaves(MatrixofMasterAndSlaves).*
19. *GetMetasymbol( ).*
20. Repeat step 2 to step 19 while |*Arr(msg)*| >*0*

This Algorithm has been summarized in a Operator Called *CatastroficMutation( )* used in the Genetic Algorithm. CatastroficMutation is made with probability *Pcat.*

In the next example we illustrate how is possible to find metasymbols from the message xyAzxyBzxyCz.

Given the message msg ="xyAzxyBzxyCz"

L=12

Arr=$x_0y_1A_2z_3x_4y_5B_6z_7x_8y_9C_{10}z_{11}$

F ={(x,3),(y,3),(A,1),(z,3),(B,1),(C,1)}

Pos(x) ={0,4,8}

Vx_0= Right(Arr,12)

Vx_4= Right(Arr,8)

Vx_8= Right(Arr,4_k)

SMsi={ $\{x_0^{0.0}\; y_1^{0.1}\; A_2^{0.2}\; z_3^{0.3}\; x_4^{0.4}\; y_5^{0.5}\; B_6^{0.6}\; z_7^{0.7}\; x_8^{0.8}\; y_9^{0.9}\; C_{10}^{0.10}\; z_{11}^{0.11}\}$,
$\{x_4^{1.0}\; y_5^{1.1}\; B_6^{1.2}\; z_7^{1.3}\; x_8^{1.4}\; y_9^{1.5}\; C_{10}^{1.6}\; z_{11}^{1.7}\; \varepsilon^{1.8}\; \varepsilon^{1.9}\; \varepsilon^{1.10}\; \varepsilon^{1.11}\}$,

$\{x_8^{2,0}\ y_9^{2,1}\ C_{10}^{2,2}\ z_{11}^{2,3}\ \varepsilon^{1,4}\ \varepsilon^{1,5}\ \varepsilon^{1,6}\ \varepsilon^{1,7}\ \varepsilon^{1,8}\ \varepsilon^{1,9}\ \varepsilon^{1,10}\ \varepsilon^{1,11}\}$

}

Let Th=3

AddColumn(0)

F_{C0}={(x,3)}, F_{C1}={(y,3)}, F_{C2}={(A,1),(B,1), (C,1)}, F_{C3} ={(z,3)}, F_{C4}={(x,2)}, F_{C5}={(y,2)}, F_{C6}={(B,1),(C,1)}, F_{C7}={(z,2)}, F_{C8}={(x,1)}, F_{C9}={(y,1)}, F_{C10}={(C,1)}, F_{C11}={(z,1)}

Sel(Cj, s_i*)*$_{Smsi}^{Fcol=0}$ returns Cj*=1, s_i*=$y_{1,5,9}$

AddColumn(1).

Elim(SMsi s_i)$_1$

SMsi={ $\{x_0^{0,0}\ y_1^{0,1}\ A_2^{0,2}\ z_3^{0,3}\ x_4^{0,4}\ \varepsilon_5^{0,5}\ B_6^{0,6}\ z_7^{0,7}\ x_8^{0,8}\ \varepsilon_9^{0,9}\ C_{10}^{0,10}\ z_{11}^{0,11}\}$,
$\{x_4^{1,0}\ y_5^{1,1}\ B_6^{1,2}\ z_7^{1,3}\ x_8^{1,4}\ \varepsilon_9^{1,5}\ C_{10}^{1,6}\ z_{11}^{1,7}\ \varepsilon^{1,8}\ \varepsilon^{1,9}\ \varepsilon^{1,10}\ \varepsilon^{1,11}\}$,
$\{x_8^{2,0}\ y_9^{2,1}\ C_{10}^{2,2}\ z_{11}^{2,3}\ \varepsilon^{1,4}\ \varepsilon^{1,5}\ \varepsilon^{1,6}\ \varepsilon^{1,7}\ \varepsilon^{1,8}\ \varepsilon^{1,9}\ \varepsilon^{1,10}\ \varepsilon^{1,11}\}$ }

Elim(Arr s_i).* Arr= $x_0\varepsilon_1A_2z_3x_4\varepsilon_5B_6z_7x_8\varepsilon_9C_{10}z_{11}$

SM_{si}=Restrict(SM_{si} $s_{c1,c2,\ldots,cn}$)

SMsi={ $\{x_0^{0,0}\ y_1^{0,1}\ A_2^{0,2}\ z_3^{0,3}\ x_4^{0,4}\ \varepsilon_5^{0,5}\ B_6^{0,6}\ z_7^{0,7}\ x_8^{0,8}\ \varepsilon_9^{0,9}\ C_{10}^{0,10}\ z_{11}^{0,11}\}$,
$\{x_4^{1,0}\ y_5^{1,1}\ B_6^{1,2}\ z_7^{1,3}\ x_8^{1,4}\ \varepsilon_9^{1,5}\ C_{10}^{1,6}\ z_{11}^{1,7}\ \varepsilon^{1,8}\ \varepsilon^{1,9}\ \varepsilon^{1,10}\ \varepsilon^{1,11}\}$,
$\{x_8^{2,0}\ y_9^{2,1}\ C_{10}^{2,2}\ z_{11}^{2,3}\ \varepsilon^{1,4}\ \varepsilon^{1,5}\ \varepsilon^{1,6}\ \varepsilon^{1,7}\ \varepsilon^{1,8}\ \varepsilon^{1,9}\ \varepsilon^{1,10}\ \varepsilon^{1,11}\}$ }

F_{C0}={(x,3)}, F_{C1}={(y,3)}, F_{C2}={(A,1),(B,1), (C,1)}, F_{C3} ={(z,3)}, F_{C4}={(x,2)}, F_{C5}={}, F_{C6}={(B,1),(C,1)}, F_{C7}={(z,2)}, F_{C8}={(x,1)}, F_{C9}={}, F_{C10}={(C,1)}, F_{C11}={(z,1)}

Sel(Cj*, s_i*)$_{Smsi}^{Fcol=0}$, returns Cj*=3, s_i*=$y_{3,7,11}$

AddColumn(3).

Elim(SMsi s_i)$_3$

SMsi={ $\{x_0^{0,0}\ y_1^{0,1}\ A_2^{0,2}\ z_3^{0,3}\ x_4^{0,4}\ \varepsilon_5^{0,5}\ B_6^{0,6}\ \varepsilon_7^{0,7}\ x_8^{0,8}\ \varepsilon_9^{0,9}\ C_{10}^{0,10}\ \varepsilon_{11}^{0,11}\}$,
$\{x_4^{1,0}\ y_5^{1,1}\ B_6^{1,2}\ z_7^{1,3}\ x_8^{1,4}\ \varepsilon_9^{1,5}\ C_{10}^{1,6}\ \varepsilon_{11}^{1,7}\ \varepsilon^{1,8}\ \varepsilon^{1,9}\ \varepsilon^{1,10}\ \varepsilon^{1,11}\}$,
$\{x_8^{2,0}\ y_9^{2,1}\ C_{10}^{2,2}\ z_{11}^{2,3}\ \varepsilon^{1,4}\ \varepsilon^{1,5}\ \varepsilon^{1,6}\ \varepsilon^{1,7}\ \varepsilon^{1,8}\ \varepsilon^{1,9}\ \varepsilon^{1,10}\ \varepsilon^{1,11}\}$ }

Elim(Arr s_i).* Arr= $x_0\varepsilon_1A_2\varepsilon_3x_4\varepsilon_5B_6\varepsilon_7x_8\varepsilon_9C_{10}\varepsilon_{11}$

SM_{si}=Restrict(SM_{si} $s_{c1,c2,\ldots,cn}$)

SMsi={ $\{x_0^{0,0}\ y_1^{0,1}\ A_2^{0,2}\ z_3^{0,3}\ x_4^{0,4}\ \varepsilon_5^{0,5}\ B_6^{0,6}\ \varepsilon_7^{0,7}\ x_8^{0,8}\ \varepsilon_9^{0,9}\ C_{10}^{0,10}\ \varepsilon_{11}^{0,11}\}$,
$\{x_4^{1,0}\ y_5^{1,1}\ B_6^{1,2}\ z_7^{1,3}\ x_8^{1,4}\ \varepsilon_9^{1,5}\ C_{10}^{1,6}\ \varepsilon_{11}^{1,7}\ \varepsilon^{1,8}\ \varepsilon^{1,9}\ \varepsilon^{1,10}\ \varepsilon^{1,11}\}$,
$\{x_8^{2,0}\ y_9^{2,1}\ C_{10}^{2,2}\ z_{11}^{2,3}\ \varepsilon^{1,4}\ \varepsilon^{1,5}\ \varepsilon^{1,6}\ \varepsilon^{1,7}\ \varepsilon^{1,8}\ \varepsilon^{1,9}\ \varepsilon^{1,10}\ \varepsilon^{1,11}\}$ }

MatrixofMasterAndSlaves( ).

Cols= $\begin{bmatrix} x_0^{0,0} \\ x_4^{1,0} \\ x_8^{2,0} \end{bmatrix} \begin{bmatrix} y_1^{0,1} \\ y_5^{1,1} \\ y_9^{2,1} \end{bmatrix} \begin{bmatrix} z_3^{0,3} \\ z_7^{1,3} \\ z_{11}^{2,3} \end{bmatrix}$

GetMasterAndSlaves(MatrixofMasterAndSlaves).
Master group: $\{x_0^{0,0}\ y_1^{0,1}\ z_3^{0,3}\}$
Slave groups: $\{x_4^{1,0}\ y_5^{1,1}\ z_7^{1,3}\}$
$\{x_8^{2,0}\ y_9^{2,1}\ z_{11}^{2,3}\}$
GetMetasymbol( ).
$\{x_0\ y_1\ z_3\}$
In a similar way we can find the metasymbols $\{A_0\}$, $\{B_0\}$ and $\{C_0\}$.

5 Experiments

Several experiments have been tested to probe the efficiency of the algorithm, here we show the result of a experiment with the message

"AAAAAAAAAABBBBBBBBBBCCCCCCCCCCDDDDDDDDDDEEEEEEEEEE
FFFFFFFFFGGGGGGGGGGHHHHHHHHHHIIIIIIIIIIJJJJJJJJJJKKKKKKKKKK
LLLLLLLLLLMMMMMMMMMMNNNNNNNNNNOOOOOOOOOOPPPPPPPPPP
QQQQQQQQQQRRRRRRRRRRSSSSSSSSSSTTTTTTTTTTUUUUUUUUUUV
VVVVVVVVVWWWWWWWWWXXXXXXXXXXYYYYYYYYYYZZZZZZZ
*ZZZZ*aaaaaaaaaabbbbbbbbbbccccccccccddddddddddeeeeeeeeeeffffffffffgggggggggg
hhhhhhhhhhiiiiiiiiiijjjjjjjjjjkkkkkkkkkklllllllllmmmmmmmmmmnnnnnnnnnnoooooo
ooooppppppppppqqqqqqqqqqrrrrrrrrrrssssssssssttttttttttuuuuuuuuuuvvvvvvvvvvwww
wwwwwwwxxxxxxxxxxyyyyyyyyyy*zzzzzzzzzz*"

than can be compressed successfully with RLE whereas Huffman Technique provides negative compression (-1.44 %).

Our method find just one metasymbol α, where

$\alpha=\{A_{10}B_{10}C_{20}D_{10}E_{40}F_{50}G_{60}H_{70}I_{80}J_{90}K_{100}L_{110}M_{120}N_{130}O_{140}P_{150}Q_{160}R_{170}S_{180}T_{190}U_{200}V_{210}$
$W_{220}X_{230}Y_{240}Z_{250}a_{260}b_{270}c_{280}d_{290}e_{300}f_{310}g_{320}h_{330}i_{340}j_{350}k_{360}l_{370}m_{380}n_{390}o_{400}p_{410}q_{420}r_{430}s_{440}t_{450}u_{460}v_{470}w_{480}x_{490}y_{500}z_{510}\}$

that is repeated 10 times. Now, the message encoded with metasymbols looks like $\alpha\alpha\alpha\alpha\alpha\alpha\alpha\alpha\alpha\alpha$. and we can compress up to 84.28 % using Huffman Coding and including the dictionary. A full description about the implementation of this method can be found in [14].

6 Conclusions

The selecting of the basic symbols (called metasymbols) in a message is important because determines the maximum compression ratio we can achieve. We showed that is possible to find metasymbols and apply a entropy coder to the new message to get better compression ratios.

Advanced Search Operators combined with Vasconcelos Genetic Algorithm have been used successfully to find patterns in an approach the to associated NP problem. Because we don not define a priori the number of metasymbols, neither the number of symbols which constitute them, we think in a kind of unsupervised clustering which can be of interest in data mining.

7 References

[1] Nelson, M., Jean L., (1995). *The Data Compression Book*, Second Edition, M&T Books Redwood City, CA.

[2] Huffman, D. A. (1952). *A method for the construction of minimum-redundancy codes*. Proc. Inst. Radio Eng. 40 (9): 1098–1101.

[3] Witten, I. H., R. Neal,and J. G. Cleary, (1987). *Arithmetic coding for data compression*,. Communications of the ACM 30 (6): 520-540.

[4] Ziv, J., and Lempel, A., (1977). *A Universal Algorithm for Sequential Data Compression*. IEEE Trans. on Inf. Theory IT-23.3, 337-343.

[5] Burrows M., and Wheeler, D. J., (1994). *A block-sorting lossless data compression algorithm*, Digital Syst. Res. Ctr., Palo Alto, CA, Tech. Rep. SRC 124.

[6] Cleary, J. G. and Witten, I. H. (1984). *Data compression using adaptive coding and partial string matching*. IEEE Transactions on Communications, Vol. 32, No. 4, 396-402.

[7] Paturi, R., Rajasekaran, S., and Reif, J.H. (1989). *The Light Bulb Problem*. In Second Work shop on Computational Learning Theory.

[8] Steven, P. (1996). *A Hybrid Local Search Algorithm for Low Autocorrelation Binary Sequences*. Technical Report, Department of Computer Science, National University of Ireland at Cork.

[9] Auluck, F.C. and D.S. Kothari, (1946). *Statistical mechanics and the partitions of numbers*, Proceedings of the Cambridge Philosophical Society 42.

[10] Shannon, C.E., (1948). *A Mathematical Theory of Communication*, Bell Sys. Tech. *J. 27*, 379-423, 623-656.

[11] Kuri, A., Herrera O., (2003). Metrics for Symbol Clustering from a Pseudoergodic Information Source, IEEE Proceedings of ENC2003.

[12] Kuri, A., (1999). A Comprehensive Approach to Genetic Algorithms in Optimization and Learning, Editorial Politécnico.

[13] Goldberg, D., (1989). Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley Publishing, Company.

[14] Kuri, A., Herrera, O., (2003). *Lossless Scheme for Data Compression using Metasymbols*, Memorias del XII Congreso Internacional de Computación - CIC2003.