Incremental discovery of sequential patterns for grammatical inference

Ramiro Aguilar RAMIRO@TEJO.USAL.ES
Instituto de Investigaciones en Informática, Universidad Mayor de San Andrés
Av. Villazón 1995, Monoblock Central. La Paz, Bolivia

Luis Alonso, Vivian López, María N. Moreno {LALONSO, VIVIAN, MMG}@USAL.ES
Departmento de Informática y Automática, Universidad de Salamanca,
Plaza de la Merced S/N, 37008 Salamanca, Spain

Abstract

In this work a methodology is described to generate a grammar from textual data. A technique of incremental discovery of sequential patterns is presented to obtain production rules simplified production rules, and compacted with bioinformatics criteria that make up a grammar that recognizes not only the initial data set but also extended data.

1. Introduction

The growing quantity of documentary information makes its analysis complex and tedious, so that automatic and intelligent methods for their processing are needed. To understand, "what say the data" is necessary to know the structure of the data language. In order to this, in (López & Aguilar, 2002) a general plan is defined that proposes a data mining method on text, to discover the syntactic-semantic knowledge of it. As part of all the proposed process a method is presented to obtain the grammar from the sequential patterns obtained in the text. This is the grammatical inference (GI) of the language of the text.

In this work a novel data mining process is described that combines hybrid techniques of association analysis and classical sequentiation algorithms of genomics to generate grammatical structures from a specific language. Subsequently, these structures are converted to *context-free grammars*. Initially the method applies to context-free languages with the possibility of being applied to other languages: structured programming, the language of the book of life expressed in the genome and proteome and even the natural languages.

1.1. Problem of grammatical inference

Grammatical inference (GI) is transversal to a number of fields including machine learning, formal languages theory, syntactic and structured pattern recognition, computational biology, speech recognition, etc. (de la Higuera, 2004).

Problem of GI is the learning of a language description from language data. The problem of context-free languages inference involves practical and theoretical questions. Practical aspects includes pattern and speech recognition; an approach of pattern recognition is the context-free grammatical (CFG) inference that built a set of patterns (Fu, 1974); another approach search the ability to infer CFGs from natural that would enable a speech recognizer to modify its internal grammar on the fly, thus allowing it to adjust to individual speakers (Horning, 1969). Theoretical aspects have importance as for the serious limitations of context-free languages (Lee, 1996) and, actually, to construct feasible algorithms of learning that imitate the model of the human language (this last point, can be problematic, but was one of the principal motivations for the early work in grammar inference (Horning, 1969)).

1.2. Language learning

The learning of a language also has to do with its identification. In the literature of the grammar inference, the attention focuses on the *identification in the limit*, this way, in each time t the machine that learns receives a information unit i_t on a language and output a hypothesis $H(i_1, ..., i_t)$; the learning algorithm is successful if after a finite amount of time, all its guesses are the same and are all a correct description in the language of the question. Another learning criteria

is the *exact identification using queries in polynomial time*, in this framework, the learning machine have access to the oracles that can answer questions, and must halt in polynomial time with a correct description of the language (Lee, 1996).

In the last 20 years, the inherent complexity present in the problem of grammatical inference, made unsuccesfull all the approaches (Miclet, 1986). The paper detailed in (de la Higuera, 2002) states that actually the algorithms with mathematical properties obtain better results than the algorithms with heuristic properties, but is when finite automata are used or, on the other hand, when algorithms of GIC learning are constructed, also it emphasizes that for the heuristic approach common "benchmark" does not exist and is then more difficult to compare and to evaluate the effectiveness of these methods. With the mentioned thing previously, our proposal has a data set available with which it is validated and we are open to other comparisons to improve or to ratify our work.

2. Techniques for the association analysis

Association analysis involves techniques that are different in its operations but all of them search relations among the attributes of a data set. Some techniques are:

- Association rules
- Discovery of sequential patterns, and
- Discovery of associations

2.1. Association rules

The association rules (AR) describe the relations of certain attributes with regard to others attributes in a database (DB). These rules identify cause-effect implications between the different attributes of the DB. For example, in the registers of products purchases, what article of purchase is identified as related to another; for instance: "the 80% of the people that buys diapers for baby, also buys talcum".

A rule have the form "if X then Y" or $X \Rightarrow Y$. X is called *antecedent* of the rule (in the example, "buys diapers"); Y is called *consequent* of the rule (in the example, "buys talcum").

The generation of the rule is supported by statistical and probabilistic aspects such as the support factor (f_s), confidence factor (f_c) and the expected confidence factor (f_e) defined as: $f_s = \frac{nr\_times\_rule}{nr\_total\_registers}$, $f_c = \frac{nr\_times\_rule}{nr\_times\_X}$ and $f_e = \frac{nr\_times\_Y}{nro\_total\_registers}$.

Table 1. Data set for association rules.

A	B	C	D	E	F
2	2	6	0	1	0.2
2	2	5	0	1	0.2
2	2	6	1	1	0.2
3	2	7	1	0	0.8
2	3	8	1	0	0.8
3	3	8	1	0	0.8
3	3	7	1	0	0.8

The minimum value of the support factor for the rules should be greater than a given threshold. If the confidence factor is greater than 0.5, then the rule appears, at least, in half the number of instances that means that the rule has certain sense. The difference between the support factor and the expected confidence factor should be minimum to assure the effectiveness of the rule.

For example, we consider the data of the table 1, a rule obtained is: $A = 2 \Rightarrow B = 2$ with $f_s = 0.43$, $f_c = 0.75$ and $f_e = 0.57$, means that the 75% of items whit $A = 2$ imply $B = 2$, besides in the 43% of all items complies that rule and $B = 2$ complies in the 57% of all items.

2.2. Discovery of associations

Similarly to the AR, the discovery of associations (DA) tries to find implications between different couples attribute-value so that the appearance of these determine a present association in a good quantity of the registers of the DB. To discover associations the following steps are carried out:

1. Associate an identifier to each transaction
2. Order sequentially the transactions according to its identifier
3. Count the occurrences of the articles creating a vector where each article is counted. The elements where the account is below of a "threshold", are eliminated
4. Combine in a matrix the transactions attribute-value and carry out the count of occurrences eliminating those elements that do not surpass the threshold
5. Repeat successively the steps 3 and 4 until no more transaction combinations are possible

Whit the data of the table 1, whit $threshold = 2$, the technique is applied as is observed in the figure 1 and the following associations are generated:

1. $A2 \Rightarrow B2 \Rightarrow E1 \Rightarrow F0.2$
2. $A3 \Rightarrow D1 \Rightarrow E0 \Rightarrow F0.8$
3. $B3 \Rightarrow D1 \Rightarrow E0 \Rightarrow F0.8$

The association 1 means: if the value of A and B is 2 and the value of E is 1 and the value of F is 0.2, then the registers with those characteristics can belong to a class. The other associations show the possible characteristics of the registers to belong to another class or behavior.

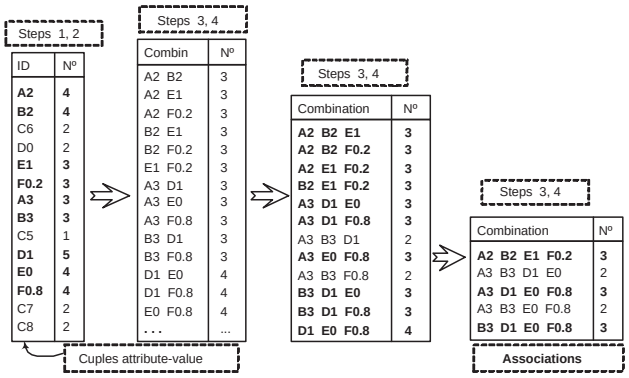

Figure 1. Discovery of associations in data of the table 1.

2.3. Discovery of sequential patterns

Discovery of sequential patterns (DSP) is very similar to the AR but search for patterns between transactions so that the presence of a set of items precede another set of items in a DB during a period of time. For example, if the data correspond to registers of articles purchased by clients, a description of what articles buys frequently a client can be obtained, and above all, which is the sequence of its purchase. Thus, the next time, the profile of the client would be known, and it will be able to predict the sequence of its purchase. This criteria can apply to another data control, for example, in the Bioinformatics context, when the data to treat correspond to the chain of nucleotides of the genome and sequences are discovered as the patterns that codify genes conform some protein (Fayyad et al., 1996) (Aguilar, 2003).

DSP have the following operation:

1. Identify the time related attribute
2. Considering the period of time when the sequential patterns are to be discovered, create an array ordered by the identifier of the transtaction
3. Create another array linking the articles of purchase of each client
4. According to the "support percent", infer the sequential patterns

The discovered patterns show instances of articles that appear in consecutive form in the data as is appreciated in the example of the figure 2.

3. Grammars, languages and bioinformatics

3.1. Context-free grammar

A grammar $\mathcal{G}$ is defined like $\mathcal{G} = (\mathcal{N}, \mathcal{T}, \mathcal{P}, \mathcal{S})$, where $\mathcal{N}$ is the set of non terminals symbols, $\mathcal{T}$ is the set of terminals symbols or syntactic categories, $\mathcal{P}$ is the set of production rules and $\mathcal{S}$ is the initial symbol. The language of a grammar $\mathcal{L}(\mathcal{G})$ is the set of all terminal strings w that have derivations from the initial symbol. This is: $\mathcal{L}(\mathcal{G}) = \{w \; is \; in \; \mathcal{T}^* \mid \mathcal{S} \Rightarrow^* w\}$

A Context-Free Grammar (CFG) has production rules like $A \rightarrow \alpha$ where $A \in \mathcal{N}$ and $\alpha \in (\mathcal{N} \cup \mathcal{T})^*$. The substitution of A by α is carried out independently of the place in which appear A (Louden, 1997). The majority of the programming languages are generated by grammars of this type (enlarged with some contextual elements necessary for the language semantics)

3.2. Grammars and bioinformatics

Bioinformatics employs computational and data processing technologies to develop methods, strategies and programs that permit to handle, order and study the immense quantity of biological data that have been generated and are currently generated. For example, for the human genome (HG), the bioinformatics seeks to find meaning to the language of the more than 37.000 million peers A, C, T and G that have been compiled and stored in the "book of life".

They offer us the opportunity to understand the gigantic DB that contain the details of the circumstances of time and place in which the genes are activated, the conformation of the proteins that specify, the form in which they influence some proteins on others and the role that such influences can play in the diseases. Besides, what are the relations of the HG with the genomes of the model organisms, like the fly of the fruit, the mice and the bacteria? Will it be able to discover sequential patterns that show how are related

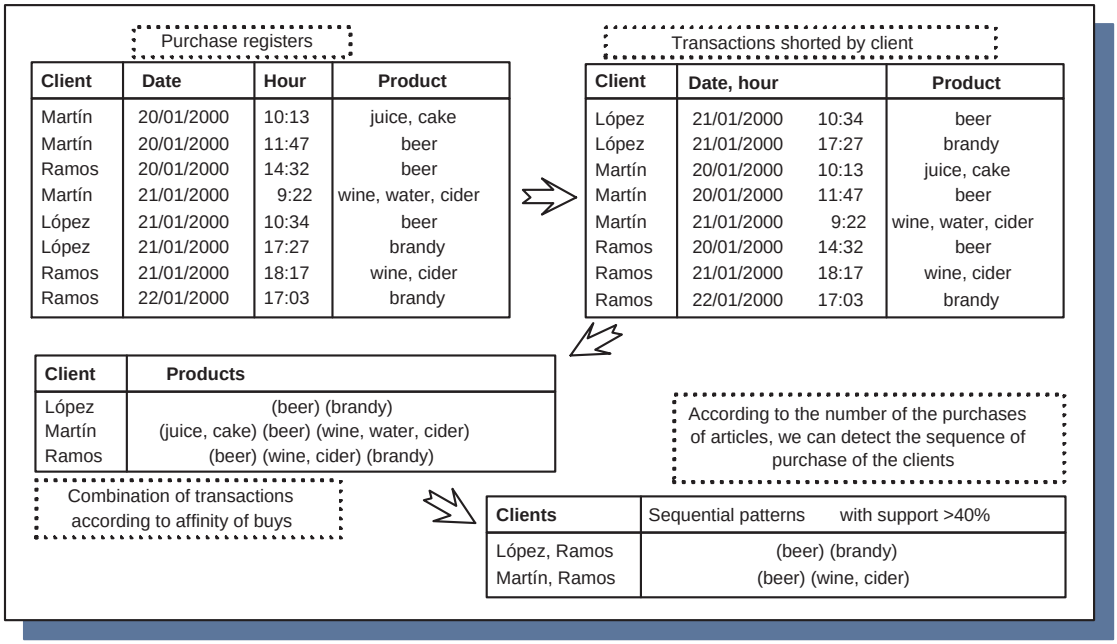

Figure 2. Discovery of sequential patterns in registers of products purchases (elaborated according to (Cabena et al., 1998)).

between itself the fragments of information? and will it be able to conform a grammatical structure that show the interpretation of the resultant set? If we are able to infer that structure for this type of language we will contribute to understand the real function of the structure of the DNA and we will understand slightly more than the questions presented.

One of the applications of the bioinformatics is the farmacology, offering reviving solutions to the old model for the creation of new medicines. It is worth to note that, one of the more elementary bioinformatics operations consists of the search of resemblances between a fragment of DNA recently arranged and the already available segments of diverse organisms (remember and associate this with the DSP). The finding of approximate alignments permits to predict the type of protein that will specify such sequence. This not only provides trails on pharmacological designs in the initial phases of the development of medicines, but suppresses some that will constitute un resolving "puzzles". A popular series of programs to compare sequences of DNA is BLAST (Basic Local Alignment Search Tool) (Altschul et al., 1990) (Altschul et al., 1997) whose mechanism of comparison applied in the development of the new medicine is shown in the plan of the figure 3.

4. Data mining procedure for the grammatical inference

The idea considers the experiences acquired (Aguilar, 2003), the literature and the existing theories (Mitra & Acharya, 2003) (Louden, 1997) (Moreno, 1998), carrying out the prosecution on data that are not structured in relations or tables with differentiated attributes but those are codified as a finite succession of sentences.

The data mining procedure has the following phases:

- Language generation by means of an context-free grammar. This language will be the source of data
- Codification of the strings of the language regarding its syntactic categories
- Dispensing with the initial grammar, discovery of sequential patterns on the codified language. This discovery, called "incremental", is a combination of the operation of the DSP and of the operation of the search of identical sequences. With this, patterns of sequences will be found that then will be replaced by an identifier symbol
- Replace the discovered sequences by their identifiers. With the previous thing the identifier is stored and the sequence as a production rule

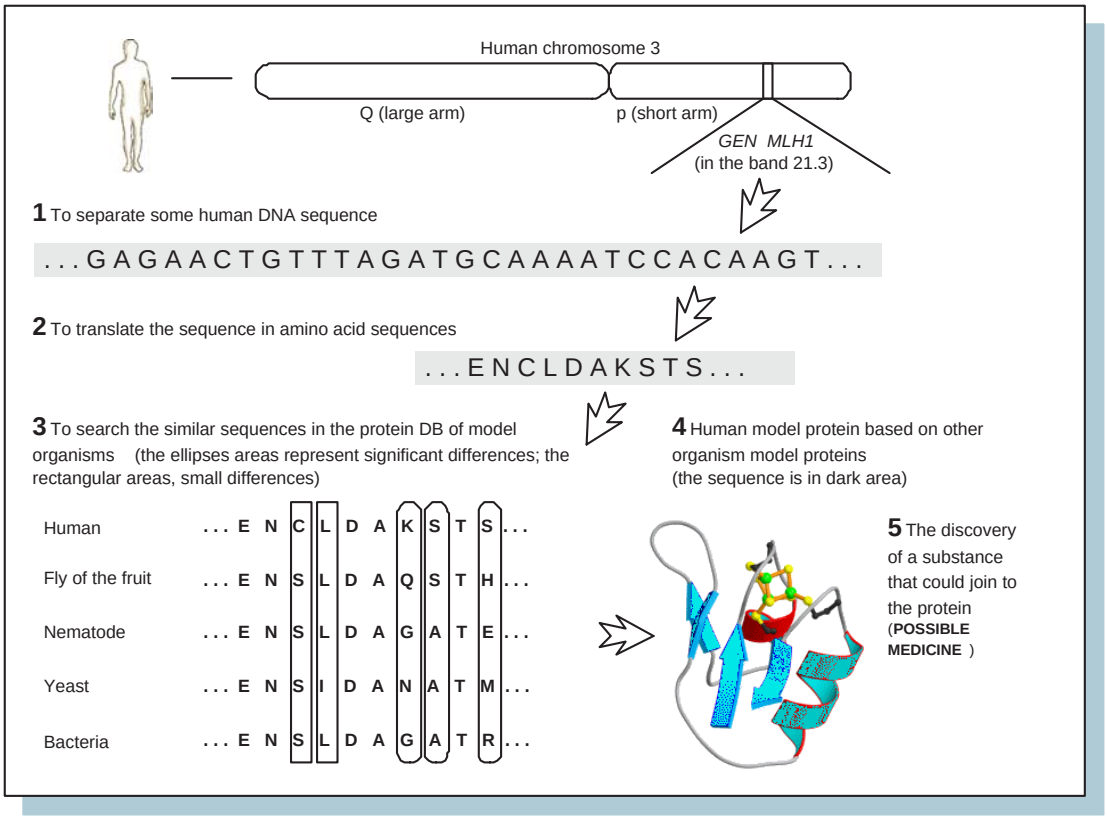

Figure 3. Utilization of the bioinformatics in the farmacology (elaborated according to (Howard, 2004)).

- Repeat the two previous steps until all the sentences of the language are replaced by identifiers

4.1. Language generation

We consider the CFG $\mathcal{G}_{æ}$ proposed in (Louden, 1997) about the generation of arithmetic expressions $\mathcal{G}_{æ} = (\mathcal{N}, \mathcal{T}, \mathcal{P}, \mathcal{S})$ where $\mathcal{N} = \{\text{Exp, Num, Dig, Op}\}$, $\mathcal{T} = \{0, 1, +, *\}$,

$\mathcal{P}$: Exp $\rightarrow$ Exp Op Exp | (Exp) | Num

Num $\rightarrow$ Dig$^+$

Dig $\rightarrow$ 0 | 1

Op $\rightarrow$ + | *

and $\mathcal{S}$ =Exp.

We can modify the formalism of this CFG of the following form:

$\mathcal{G}_{æ} = (\mathcal{N}, \mathcal{T}, \mathcal{P}, \mathcal{S})$ where $\mathcal{N} = \{\text{E, d, b, o, a, c}\}$, $\mathcal{T} = \{0, 1, +, *, (,)\}$,

$\mathcal{P}$: E $\rightarrow$ E o E | a E c | n

d $\rightarrow$ b$^+$

b $\rightarrow$ 0 | 1

o $\rightarrow$ + | *

a $\rightarrow$ (

c $\rightarrow$)

and $\mathcal{S}$ =E, what does not change in essence the character of the original grammar.

With the previous criteria, a sample of the language generated by $\mathcal{G}_{æ}$ can be seen in the figure 4, point (i). It is noted that each line corresponds to a sentence accepted by the grammar.

4.2. Language codification

Considering the language that is generated with $\mathcal{G}_{æ}$, all the symbols of $\mathcal{T}$ can be codified with the symbols of $\mathcal{N}$, only for this particular case the symbols to be used are {b, o , a, c} as syntactic categories. See the figure 4, point (ii).

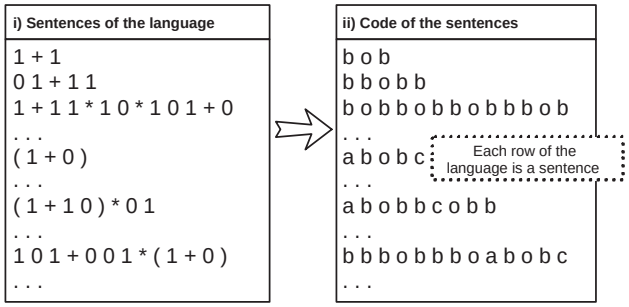

Figure 4. Language of arithmetic expressions on which its grammar is inferred.

4.3. Incremental discovery of sequential patterns and associations

The hybrid discovery of sequential patterns applied to codified languages seeks key subsequences in the sentences of the language. Each subsequence $\mathbf{q}$ has a length $\mathbf{w_q}$ that indicates the number of symbols that possesses. In this particular case $1 \leq \mathbf{w_q} \leq 5$ and $\mathbf{Q}$ is defined as a string of length $\mathbf{w_Q}$. By convention, in the codified language many sentences exist that conform the population of the language. The idea consists of finding subsequences, to identify them with a symbol and to replace with that symbol the appearances of the subsequences in the sentences of the population, all the previous procedure of repetitive form until each sentence is identify by a single symbol.

The detailed steps are:

1. For all the sentences, While $\mathbf{w_Q} > 1$ do:

- **1.1.** For $\mathbf{w_q} = 1..5$ do:
  - **1.1.1.** For all the sentences:
    - **(a)** Make $\mathbf{q}$ from then $\mathbf{w_q}$ first symbols of $\mathbf{Q}$
    - **(b)** Compute the global scoring $\mathbf{g_q}$ of $\mathbf{q}$ defined as $\mathbf{g_q} = \sum_{i=1}^{cant} \mathbf{p_q^i}$, where $\mathbf{p_q} = \frac{\mathbf{w_q} * nr\_apparitions\_of\_q\_in\_Q}{\mathbf{w_Q}}$ is the scoring of $\mathbf{q}$ in $\mathbf{Q}$
  - **1.1.2.** End For
- **1.2.** End For
- **1.3.** Selecting the subsequence $\mathbf{q}_*$ of greater global scoring
- **1.4.** If $\mathbf{q}_*$ has more than one symbol, then replacing all the consecutive appearances of that symbol by itself. Thus the production rule is created $\alpha \rightarrow \alpha^+$ (in this particular case, this are replaced all the bb by d, to see figure 5)
- **1.5.** If $\mathbf{q}_*$ has more than one symbol, then replace all the appearances of $\mathbf{q}_*$ in the sentences $\mathbf{Q}$ creating the production rule $A \rightarrow contained\_of(\mathbf{q}_*)$. The symbol A is generated consecutively so that the following time that another rule production is created, is utilized $B, C, ...$ and so on (to see figure 5)

2. Returning to step 1 noting that with 1.4 and 1.5 changes the size of the sentences of the population of the language

With the previous procedure production rules are generated that recognize the sentences of the language. The production rules number can be considerable so that we apply a particular method of simplification of grammar.

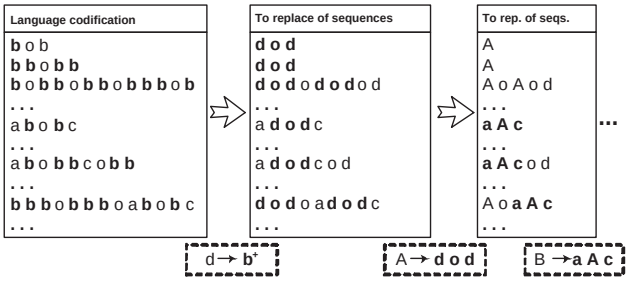

Figure 5. Hybrid discovery of sequential patterns for the context-free languages.

5. Experiments

5.1. Rules similarity

Considering the language $\mathcal{L}_{æ}$ of arithmetic expressions[1], we apply the hybrid algorithm of DSP and the production rules of the figure 6 were obtained. With the right hand of rules, that conform the sequential patterns of the language, a *substitution matrix* is computed that it is observed in figure 7, this matrix shows the similarity values between terminal symbols. Similarity among a pair of consecutive symbols is related with the apparition frequency of the symbols in the language (is a matrix like BLOSUM matrix (Henikoff & Henikoff, 1992)). Subsequently, is possible to make alignments among those sequences by compact them.

In the substitution matrix $m(i, j)$ each row i and each column j correspond with a non terminal symbol of the production rules generated. The symbols are put

[1]The corpus can be observed in `http://www.geocities.com/ramirohp/corpusae.html`

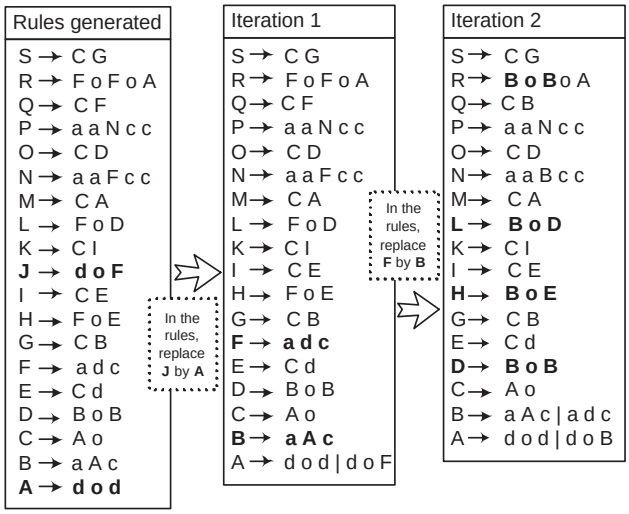

Figure 6. Production rules generated and some iterations in its simplification.

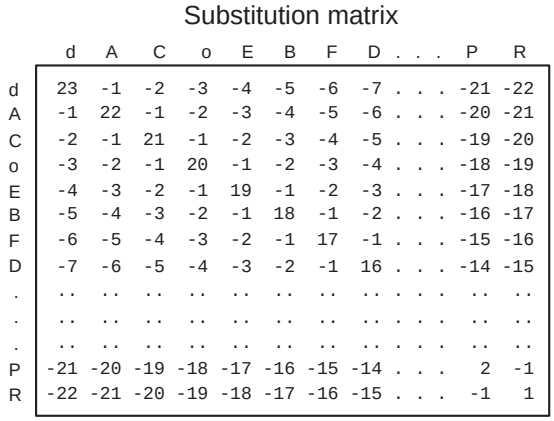

Figure 7. Substitution matrix for the rules generated.

according its apparition frequency, this is, first d, after A, C, o and so on. For $\mathcal{L}_{æ}$ themselves it generated 19 symbols *A, B,..., S* that join with the symbols of the codification *d, o, c* and *a* they conform 23 non terminal symbols (in the bioinformatics context, the symbols would correspond to the amino acids). The values of the matrix denote the importance of the alignment among the not terminal symbols; for example, $m(d, d) = 23$ denotes a degree of high similarity between both symbols; $m(d, A)$ denotes a degree of similarity of -1.

5.2. Rules simplification and compaction

With the right parts of the productions rules (where the first rules generated have greater importance) we search *similar sequences* to compact them.

The steps are:

- The sequence β that can be compacted with the sequence α is activated with the similarity function f; $f(\alpha, \beta) = \begin{cases} 1 & si \ \frac{\sum_{i=1}^{n} m(\alpha_i, \beta_i)}{\sum_{i=1}^{n} m(\alpha_i, \alpha_i)} > \theta; \\ 0 & si \ e.o.c. \end{cases}$
  where n is the minimal length between the sequences α and β, θ is a threshold or *similarity factor* with value 0.4 in this particular case
- The similar sequences are compacted and will be derived by a single non terminal symbol. The remaining non terminal symbol should be replaced for the previous one in all the right parts of the rules
- Repeat the previous steps until there are no similar sequences

For example, for the language $\mathcal{L}_{æ}$ the rules **dod** and **aAc** are not similar since $f(\mathbf{dod}, \mathbf{aAc}) = 0$ since $\frac{\sum m(\mathbf{dod},\mathbf{aAc})}{\sum m(\mathbf{dod},\mathbf{dod})} = \frac{-10-2-11}{23+20+23} = \frac{-23}{66} = -0.35$ is not greater than 0.40. Nevertheless, the rules **dod** and **doF** are similar since, $\frac{\sum m(\mathbf{dod},\mathbf{doF})}{\sum m(\mathbf{dod},\mathbf{dod})} = \frac{23+20-6}{23+20+23} = \frac{37}{66} = 0.56$. This way, the generated rules are simplifying and compacting iteratively (figures 6 and 8) until a grammar is built $\mathcal{G}'_{æ} = (\mathcal{N}', \mathcal{T}', \mathcal{P}', \mathcal{S}')$ where $\mathcal{N}' = \{$S, R, E, D, B, A, d, b, o, a, c$\}$, $\mathcal{T}' = \{0, 1, +, *, (,)\}$,

$\mathcal{P}'$: S → R | E | D | B | A | d

R → DoA

E → Cd | CB | CE | CA | CD

D → BoB | BoE | BoD

C → Ao

B → aAc | adc

A → dod | doB

d → b^+

b → 0 | 1

o → + | *

a → (

c →)

and $\mathcal{S}'$ =S.

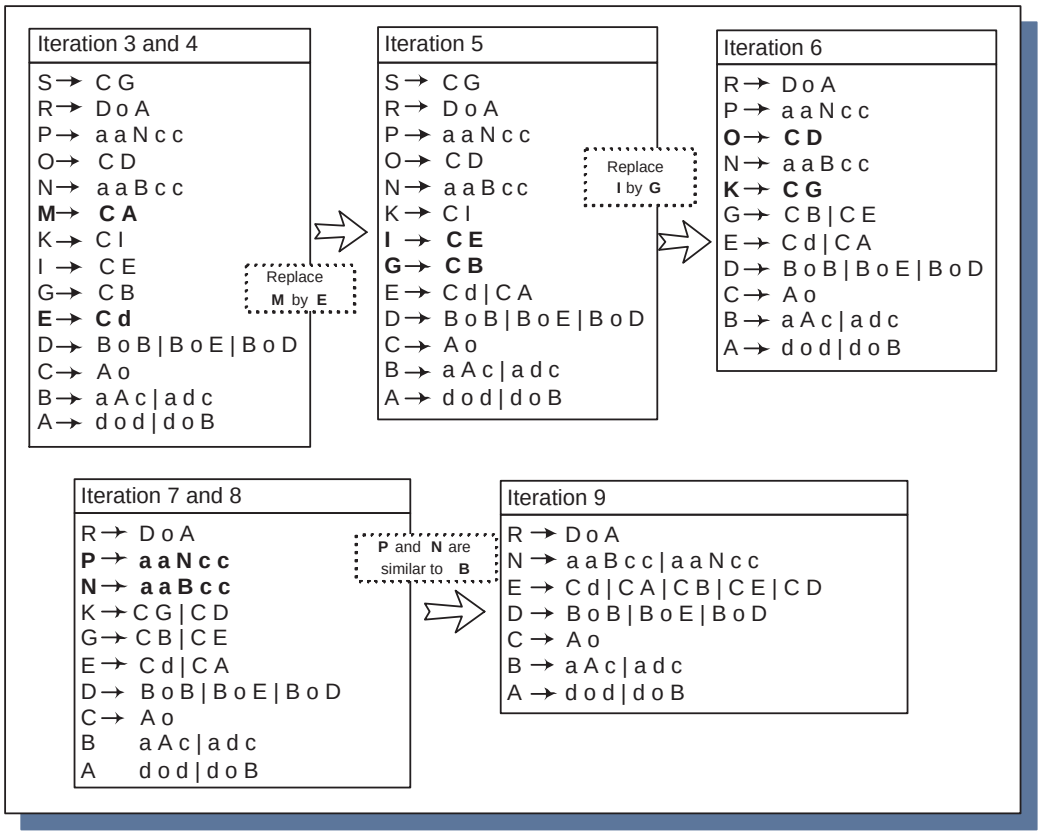

Figure 8. Simplification and compaction of the production rules generated.

6. Conclusions

In the experiments, a language $\mathcal{L}_{æ}$ has been considered generated by predetermined context-free grammar $\mathcal{G}_{æ}$ and the syntactic categories **b**, **o**, **a** and **c** were known beforehand; but later none of the properties of that grammar were utilized to generate the set of production rules that then conformed the grammar $\mathcal{G}'_{æ}$. The approach extends to processing of data that are believed to have a grammatical structure that could be generated automatically. We could imagine to find somewhat similar for the genome, for the proteome or for the natural languages, the doubt is served.

References

Aguilar, R. (2003). *Minería de datos. Fundamentos, técnicas y aplicaciones*. Salamanca: University of Salamanca.

Altschul, S. F., Gish, W., Miller, W., Meyers, E. W., & Lipman, D. J. (1990). Basic local alignment search tool. *Molecular Biology*, *215*, 403–410.

Altschul, S. F., Madden, T. L., Schaffer, A. A., Zhang, J., Zhang, Z., Miller, W., & Lipman, D. J. (1997). Gapped blast and psi-blast: a new generation of protein database search programs. *Nucleic Acids Research*, *25*, 3389–3402.

Cabena, P., Hadjinian, P., Stadler, R., Verhees, J., & Zanasi, A. (1998). *Discovering data mining. From concept to implementation*. Prentice Hall.

Corlett, L. (2003). *Web content mining: a survey* (Technical Report). Department of Computer Science, California State University.

de la Higuera, C. (2002). Current trends in grammatical inference. *Proceedings of Joint Iapr International Workshops Sspr 2000 and Spr 2000* (pp. 130–135). Lecture Notes in Artificial Intelligence, Springer-Verlag.

de la Higuera, C. (2004). A bibliographical study of grammatical inference. *Pattern Recognition*.

Fayyad, U., Piatetsky-Shapiro, G., Smyth, P., & Uthurusamy, R. (1996). *Advances in knowledge discovery and data mining*. Salamanca: MIT Press.

Fu, K.-S. (1974). *Syntactic methods in pattern recognition.* Academic Press.

Henikoff, S., & Henikoff, J. G. (1992). Amino acid substitution matrices from protein blocks. *Proc. National Academic Science*, *89*, 10915–10919.

Horning, J. J. (1969). *A study of grammatical inference* (Technical Report 139). Computer Science Department, Stanford University.

Howard, K. (2004). La fiebre de la bioinformática. *Investigación y ciencia: nueva genética*, *38*, 79–82.

Lee, L. (1996). *Learning of context-free languages: a survey of the literature* (Technical Report). Computer Science Department, Stanford University.

López, V., & Aguilar, R. (2002). Minería de datos y aprendizaje automático en el procesamiento del lenguaje natural. *Workshop de minería de datos y aprendizaje, IBERAMIA 2002* (pp. 209–216). Sevilla: University of Sevilla.

Louden, K. C. (1997). *Compiler construction. Principles and practice.* International Thomsom Publishing Inc.

Lucas, S. (1994). Structuring chromosomes for context free grammar evolution. *Proceedings of first IEEE International Conference on Evolutionary Computation* (pp. 130–135).

Miclet, L. (1986). *Structural methods in pattern recognition.* Chapman and Hall.

Mitra, S., & Acharya, T. (2003). *Data mining. Multimedia, soft computing and bioinformatics.* John Wiley and sons.

Moreno, A. (1998). *Linguística computacional.* Madrid: Editorial Síntesis.