UNIVERSITAT ROVIRA I VIRGILI
GRAMMAR SYSTEMS: A FORMAL-LANGUAGE-THEORETIC FRAMEWORK FOR LINGUISTICS AND CULTURAL EVOLUTION
Maria Dolores Jiménez López
ISBN:978-84-691-1893-1/DL: T-351-2008

UNIVERSITAT ROVIRA Í VIRGILI

GRAMMAR SYSTEMS: A FORMAL~LANGUAGE~THEORETIC FRAMEWORK FOR LINGUISTICS AND CULTURAL EVOLUTION

DOCTORAL DISSERTATION

M.DOLORES JIMÉNEZ LÓPEZ

TARRAGONA~2000

It is natural to wonder which portion of the real world can be imitated in its behaviour by a set of meaningless symbols governed by formal rules. Would be possible to transform the whole real world in formal system? It seems that, in a very broad sense, we can answer in the affirmative: it is possible to suggest that the real world is nothing, in itself, but an extremely complicated formal system.

Douglas R. HOFSTADTER

UNIVERSITAT ROVIRA I VIRGILI
FACULTAT DE LLETRES
DEPARTAMENT DE FILOLOGIES ROMÀNIQUES

Grammar Systems: A Formal-Language-Theoretic Framework for Linguistics and Cultural Evolution

DOCTORAL DISSERTATION

Presented by

M. DOLORES JIMÉNEZ LÓPEZ

Supervised by

Dr. CARLOS MARTÍN VIDE

Tarragona, 2000

Acknowledgments

This thesis contains one name in its cover, but it does not belong at all to a single person. Without the help, support and encouragements given by many people I have crossed on my way during the four years of elaboration of this thesis, the results I present here would be inexistent. Thanks to all of them.

However, I would not be fair if I included everybody in the same 'level.' There are people that have given invaluable help for the fulfilment of this thesis and they deserve a special mention here.

First of all, I would like to express my deepest gratitute to my supervisor, Dr. Carlos Martín-Vide. Without his teaching, his help, his advices and his encouragement, this thesis would not have been written. Thank you very much for always having an answer for my questions.

I am deeply indebted to Dr. Erzsébet Csuhaj-Varjú who has been able to positively motivate me, who has made me to believe in what I was doing and who, with her comments and direct supervision, has made possible this dissertation. I will be always grateful to you.

I do not need to move country to continue giving thanks. If Dr. Csuhaj-Varjú support has been incalculable, her group -specially György Vaszil and Judit Csima- has helped also very much. Without having them everything would have been much more difficult. Köszönöm szépen for those days in Budapest.

Of course, I would like to thank everybody in SZTAKI (Computer and Automation Institute, Hungarian Academy of Sciences) for allowing me to work there, for make me feel part of that Institute and for making possible the direct collaboration with Dr. Csuhaj-Varjú and her group.

If my 'Hungarian people' have been so important, what to say about people from my research group? They have given me all the support I have needed, they have been always there when I have asked for their help (and even when I have not asked for it). Muchas gracias to members of GRLMC, and very specially to Rosa M. Hidalgo.

Other proper names have to be mentioned here, because they have contributed somehow to make this thesis possible with their teaching along several workshops during these four years, with their comments and with their wide knowledge about the topic: Dr. Gheorghe Păun, Dr. Alexandru Mateescu and Dr. Victor Mitrana.

Besides scientific support, the writing of this thesis would have been impossible without the unconditional support of my family that have stood me throughout this four years and have been patient enough to forgive my moods.

And last but not least, I have to acknowledge the financial support given by Direcció General de Recerca, Generalitat de Catalunya, through a FI Fellowship. And, of course, my gratefulness to Departament de Filologies Romàniques from Universitat Rovira i Virgili for allowing me to spend a long period of time in Hungary. Without that stay in Budapest, this thesis would be very different.

Tarragona, December 1999.

M. Dolores Jiménez López.

Contents

Objectives, Motivation, Organization

What this thesis is

> *'A lot of variants of Grammar Systems can be imagined, either suggested by 'practical' facts, or being natural from the mathematical point of view.'* [Csuhaj-Varjú *et al.*, 1994a, p. 137].

The above quotation could be considered the 'foundational stone' of this dissertation, since what we do in this thesis is, precisely, to define three new variants of Grammar Systems Theory suggested by 'practical' facts. In this case, we would like to show the possible *applicability* of that new theory to *linguistic* and *cultural* matters by proposing three new types of Grammar Systems.

Formal Language Theory was born in the middle of our century as a tool for modelling and investigating syntax of natural languages. After 1964, Formal Language Theory developed as a separate branch with specific problems, techniques and results and with an internal self-motivated life. So, Formal Languages, which started being a tool to be applied to natural languages, became rapidly a theory that studies formal systems independently of possible applications. In this thesis, we would like to perform just the op-

posite way: this is, we start with a new branch of Formal Language Theory -namely, Grammar Systems- that was born as a purely theoretical model, and we intend to show its possible applicability. Summing up, Formal Language Theory went from *application* to pure *theoretic* study, and we want to go from *theoretical model* to *application.*

Grammar Systems Theory has been widely investigated in the last decade and nowdays constitutes a well-developed formal theory that presents several advantages with respect to classical models. However, being a branch of Formal Languages, researchers in the field of Grammar Systems have concentrated mainly on theoretical aspects of the model. Although some attempts of application have been carried out, we could say that research on Grammar Systems Theory still being mainly theoretical. Taking into account results obtained from the formal study of the theory, its features, and its advantages with respect to other models, we consider that Grammar Systems Theory deserves to be studied from the point of view of its possible applications. This thesis aims to be an attempt of showing the potential applicability of Grammar Systems.

Theoretical studies in Grammar Systems have proved interesting results regarding its generative capacity, descriptional complexity, decidability properties, etc. Our goal in this thesis is to show that Grammar Systems Theory is not only a good theory from the formal point of view, but that it could be as well -due to its features- an useful tool for being applied to several matters.

In order to show the potential suitability of Grammar Systems Theory in different issues, we present three feasible examples. We apply this theory to the *architecture of natural language grammar,* to *conversation,* and to *cultural evolution.* We define three new variants of Grammar Systems that could account for the above three matters. With these three new types of Grammar Systems, we aim to show that it could be possible and useful to describe the above three issues in terms of Grammar Systems Theory and

that, therefore, results obtained from the *formal* study of Grammar Systems could cast light in several *practical* matters.

Summing up, to the question *'What is this thesis?'*, we can answer that it is an attempt to show the applicability of a full-developed theoretical model, namely Grammar Systems Theory.

The questions we answer in this thesis are basically:

- Can Grammar Systems Theory be applied to specific matters or it is only a good framework from the formal point of view?
- Can Grammar Systems Theory model real phenomena?

To find the answer to these questions:

- we apply Grammar Systems Theory to the *architecture of natural language grammar,* in order to show the possible suitability of this theory in accounting for *modularity, parallelism,* and *interaction* necessary in grammar.
- we apply Grammar Systems Theory to *conversational acts,* in order to see whether this theory could account for the ideas of *coordination, cooperation, interaction, emergence* and *dynamism* characteristic of conversation.
- we apply Grammar Systems Theory to *cultural evolution,* in order to check whether this theory is able to provide a formal language model that accounts for the basic mechanisms of cultural change.

The contributions of this thesis can be summarized as:

- Definition of a new type of Grammar Systems, called **Linguistic Grammar Systems,** that could account for the highly modular, parallel and interactive architecture of natural language grammar.

- Definition of a new type of Grammar Systems, called **Conversational Grammar Systems,** that could reveal itself as a suitable tool for providing a formal language description of conversation.
- Definition of a new type of Grammar Systems, called **Cultural Grammar Systems,** that could capture, in formal language terms, many features that researchers dealing with cultural change have pointed out about the dynamics of cultural evolution.
- Detailed description of the analogies that can be established between each of the formal frameworks we propose and the real facts they try to model.
- Evidence that Grammar Systems Theory is not only a good formal theory, but that it could be also an interesting theory from the point of view of its applications.
- Evidence that Grammar Systems Theory could reveal itself as a very useful tool in accounting for a wide range of linguistic matters -from syntax to pragmatics, including language change- and that it could describe, in a formal-language-theoretical fashion, the basic dynamics of cultural change.

What this thesis is not

> *'In order to exist, theory has to abstract away from practical details. Without abstraction and formalization no deep scientific results can be obtained.'* [Nijholt, 1988, p. 249].

It is very important, of course, to leave clear what this thesis is, but as important as to know what *it is* is to know what it *is not.*

As the previous quotation states, in order to have a theory it is necessary to abstract away from practical details. Without intending to build a theory, but with the simple aim of providing three variants of Grammar Systems that could show the possible applicability of this theory, we are constrained by the same requirements. We have to simplify matters we want to model in order to make them susceptible of being formalized. So, the reader should not expect to find, in this dissertation, specific and small details of any of the issues we deal with. Every formal framework we provide abstracts and simplifies very much the real fact it tries to capture. Each new variant of Grammar Systems we present here tries to fit with the most *basic* and *general* features of the matter it treates, and does not intend to account for small and concrete traits. It is worth to emphasize, therefore, that in this thesis we try to *formalize*, and, by definition, formalization means *simplification, abstraction* and *generalization* and not detail and particularity.

It should remain clear, also, from the very beginning that this thesis does not intend to explain concrete instances of any of the matters it deals with. We will not face the problem of how to generate a specific sentence or how to solve a syntactic problem by using Linguistic Grammar Systems; we will not take a real conversation in order to explain it by using Conversational Grammar Systems; and we will not try to explain an specific cultural change that has already taken place, by means of Cultural Grammar Systems. Since our principal and unique goal is to show the applicability of Grammar Systems Theory, we will limit ourselves to define the formal frameworks and to establish an analogy between them and the fact they could be applied to. Of course, we will try to make this analogy as detailed as possible in order to really show how every proposed variant of Grammar Systems could account for the specific matter it has built for, but we will not apply directly our frameworks to real instances of conversation, cultural change or sentence generation.

We should keep in mind that our objective here is to check whether Grammar Systems Theory can be applied or whether it is just a theoretical tool. In order to carry out such task we define three new frameworks and we explain how they could account for three different matters. But we do not go further, we do not carry out the direct application of such frameworks.

It is also very important to stress that this thesis is not just a mathematical *exercise*. This is, when we propose here three new variants of Grammar Systems and we state that they could account for several matters, we are not only trying to put in symbols what could be perfectly said in words. It is not just to formalize because of the pleasure of formalizing. What we try to do, by defining three new types of Grammar Systems, is to establish an analogy between the formal theory and the concrete fact we try to model, in such a way that, if the analogy has been well established, results formally obtained from the analysis of the formal model could be partially or totally translated to the real fact we are dealing with. This is, formalization should be viewed as the prior step for the obtaining of results that would be, otherwise, difficult to be reached.

Formalization has a long tradition in science, besides traditional fields as physics or chemistry, other scientific areas as medicine, cognitive and social sciences and linguistics have shown a tendency towards mathematization over the last decades. The use of mathematics -this is, formalization- in those fields has led to numerous results that would have been difficult to be obtained without such formalization. In this thesis, we want to fulfill just the first stage in such difficult task of obtaining results from a formal theory. We *formalize* in terms of Grammar Systems three different matters and we establish the respective analogies between the formal framework and the fact it tries to model. Next task (not carried out in this thesis) will be to study those formalizations in order to get results that could cast light in our understanding of the different matters faced here.

We have emphasized in the above section that, in contrast with what has been done, up to now, in the field of Grammar Systems, we do not intend to make a formal study of the theory, but that our goal here is to look for possible applications of it. It is true that in order to show such applicability we will formally define three new variants of Grammar Systems, but according to our initial goal, we will not carry out the formal study of any of those three new types of Grammar Systems. So, no theorem can be found in this thesis. No result about generative capacity, descriptional complexity, decidability properties etc. is included here. We would like to think that in this way we are following Gheorghe Păun assertion:

> *'Maybe not the results, but the eleborated conceptual framework, is the most important contribution of scientific activities, in certain moments of the development of disciplines. Grammar Systems are now in such a developmental stage that definitions rather than theorems are their major contribution to the whole theoretical computer science.'* Gheorghe Păun (cited by [Kelemen, 1999, p. 271]).

Why this thesis

Classical Theory of Formal Languages has been used to study several matters, from works in Linguistics to Economic Modelling, Developmental Biology, Criptography, Sociology, etc. So, the question is: if classical Formal Language Theory has been applied to such a wide range of fields, why not to look for applications of Grammar Systems Theory?

Grammar Systems, being a branch of Formal Language Theory, present the abstractness that has facilitated the application of Formal Languages to

many fields. However, this new theory provides in addition several advantageous features that lacks its 'older sister.' Notions as *modularity, cooperation, distribution, parallelism, interaction, emergent behavior,* etc. are in the base of Grammar Systems and constitute a real advantage whenever we try to apply this theory, since those same concepts are very vividly in several fields nowadays as, for instance, Computer Science, Artificial Intelligence or Cognitive Science.

Taking into account that the above-mentioned features of Grammar Systems Theory have revealed as very important in the study of natural languages, we have considered worth to check whether this theory can account for linguistic matters.

Natural languages could be imagined as a number of modules that interact in a nonsimple way, where the whole is more than the sum of the parts. It seems evident that understanding and generating sentences requires at least cooperative phonological, morphological, lexical, syntactic, referential, pragmatic, semantic... modules. If we speak about *interaction, cooperation* and *modularity* in natural languages, it seems natural to think that it could be possible to describe the architecture of natural language grammar in terms of Grammar Systems Theory.

Now if, with the same idea of Grammar Systems in mind, we face *conversational acts,* we realize that *coordination, cooperation, emergent behaviour,* and *dynamism* are again essential concepts in both areas. So, once more it seems natural to think that Grammar Systems Theory could offer suitable tools for providing a formal language account of conversation.

Architecture of grammar and conversation are, therefore, two possible areas where Grammar Systems theory could be applied. However, is Grammar Systems Theory only suitable in the field of Linguistics? In order to show that such theory could reveal as useful in a wide range of fields, and that, of course, it is not constrained to natural language study, we present our third

model: Cultural Grammar Systems. Moreover, taking into account that language is part of culture, and that models applied to the latter could reveal as useful in the former, we could say that if Cultural Grammar System reveals as a feasible framework to describe cultural evolution, it could be suitable as well in language change. In this way, we will complete our first aim of showing that Grammar Systems Theory could be a good model to be applied to a wide range of linguistic matters, from syntax to pragmatics including language evolution.

So, to the question *'Why this thesis?'*, we could answer: because we would like to face the challenge of showing that Grammar Systems Theory could be a very useful tool in several fields. Why do we think that such applicability is possible? Features as *modularity, cooperation, parallelism, interaction,* etc., as well as advantages with respect to classical models lead us to think that Grammar Systems Theory is not only an attractive theory from the formal point of view, but that it could be interesting as well with respect to its applications. And why linguistic and cultural matters? Because features important in the architecture of grammar, in conversational acts or in the dynamics of cultural evolution could be easily captured with Grammar System tools.

How this thesis is organized

This thesis is organized into five parts and one Appendix. Part I is necessary in order to understand the other four parts. Parts II, III and IV can be studied rather independently. Part V presents conclusions that follows from precendent parts. And Appendix offers examples of the different types of Grammar Systems presented in part I. Dependencies among different parts are showed in figure 0.1.

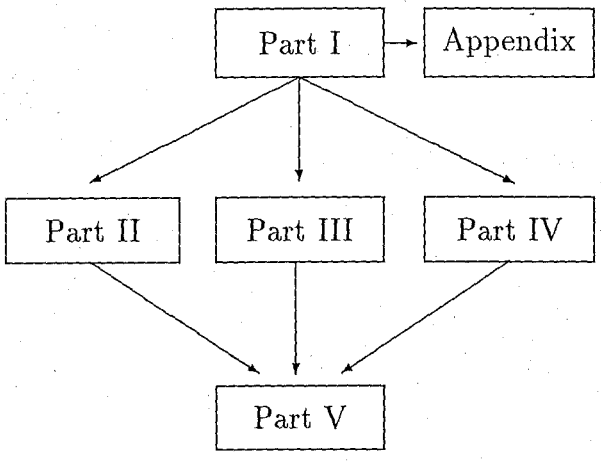

Figure 0.1: Overview of Parts

Parts II, III and IV have a similar structure. Each of them ends with a chapter in which we present partial *conclusions* and *future work,* and starts with and introductory chapter in which we answer three basic questions: *what?, why?* and *how?* With these questions we aim to make clear *what* is going to be presented in that part; which is the motivation for introducing the model that is presented (*why*); and *how* the part is organized into chapters.

Part I: Introduction. This part has three basic objectives: (1) to provide formal language prerequisites necessaries to understand formalization presented throughout the thesis; (2) to present Grammar Systems Theory, giving formal definitions of its basic types and of variants that will be referred to in the body of this dissertation; (3) and to review some of the advantageous features that make of Grammar Systems Theory a potential good candidate to be applied to several matters.

Part II: Linguistic Grammar Systems. This part presents Linguistic Grammar Systems. We will start by answering our three basic ques-

tions: *what?, why?, how?* After reviewing some questionable assumptions about grammar, we sketch two theories of parallel grammatical representations -namely, Sadock's Autolexical Syntax and Jackendoff's Architecture of Language Faculty- as a previous step of the introduction of our formal framework. Before presenting the formal definition of Linguistic Grammar System, we afford it informally, in order to make clear why we choose some specific Grammar Systems tools and not others. Once we have provided formal definitions and a formal example of Linguistic Grammar Systems, we present an informal example in order to sketch how our framework could account for generation of natural language structures. We close this part by offering some partial conclusions and future work.

Part III: Conversational Grammar Systems. This part is dedicated to the presentation of Conversational Grammar Systems. As always, we will start with a chapter in which we respond to our three questions. After having tried to get an idea of conversation -by looking for a definition, individuating its minimal units and summarizing some of its basic features- we will present some approaches used to model conversation in different research fields. Once we have got an approximate idea of what conversation is and how it has been approached from different perspectives, we will carry out the presentation of our formal framework. Such presentation will be done step by step, in order to cast light on every formal definition we make and with the objective of making clear the relationship between our formalization and the correspondent element or fact in conversation. Before offering conclusions and future work, we will put Conversational Grammar Systems versus conversation, trying to check how our formal framework could fit with some features that have been proposed as characteristic of face-to-face interactions.

Part IV: Cultural Grammar Systems. In this part we introduce the third variant of Grammar Systems Theory we propose in order to show its applicability: Cultural Grammar Systems. After the introductory chapter

in which we answer the usual questions (*what?, why? and how?*), we briefly review Cultural Algorithms as a theory that has greatly motivated our framework. Before facing the presentation of our formal model, we try to get and idea of *culture* by giving several definitions, identifying its units, individuating some of its properties and reviewing different models of gene-culture interaction. Presentation of Cultural Grammar Systems is divided into two steps: firstly, we build the model step by step, in an informal way, and trying to justify why we include every element and relationship; secondly, we provide the formal definition of our model, again justifying why we have opted for each formalization. After providing formal definitions of our framework, we present a sort of example by applying it to description of language evolution. We end this part, as always, with conclusions and future work.

Part V: Concluding Remarks. This part closes our dissertation by offering a brief summary of its content, presenting conclusions and giving some directions for future research.

Part I

Introduction

Chapter 1

Formal Language Prerequisites

Throughout this thesis we use the notation of Formal Language Theory. In this chapter we present some of the formal prerequisites necessaries for understanding the formalization presented in this work. For further information in the Theory of Formal Languages we refer to [Salomaa, 1973, Rozenberg & Salomaa, 1997, Révész, 1983, Wood, 1987].

A *set* is a collection of elements taken from some prespecified universe. A set is *finite* if it contains a finite number of elements and is *infinite* otherwise. A *sequence* of elements, from some universe, is a list of elements possibly, but not necessarily, with repetitions. Because it is ordered by position we can speak of the 1st, 2nd and i-th element.

A finite, nonempty set V of symbols or letters is called an *alphabet*. A *word* or a string over an alphabet V is a finite sequence of symbols from V. The *empty word* is denoted by λ and is the empty sequence of symbols. For an alphabet V, we denote by V^* the free monoid generated by V under the operation of concatenation, this is, the set of all words over V. V^+ denotes the set of all nonempty strings over V. The *length* of a string $x \in V^*$ (the number of symbol occurrences in x) is denoted by $|x|$. The number of occurrences in $X \in V^*$ of symbols $U \subseteq V$ is denoted by $|x|_U$.

Given two words x and y over V, their *catenation* is xy. $\lambda x = x\lambda = x$, for all x in V^*.

For two words x and y over V, x is a *prefix* of y if $y = xz$, for some z in V^*. $x = y$ if x is a prefix of y and $|x| = |y|$. x is a *proper prefix* of y if it is a prefix, $x \neq y$, and $y \neq \lambda$. *Suffix* and *proper suffix* are defined similarly. We say x is a *subword* of y if $y = wxz$, for some w and z in V^*. x is a *proper subword* if $x \neq y$ and $x \neq \lambda$.

Let Σ and Δ be alphabets. Then a mapping $h : \Sigma^* \to \Delta^*$ is said to be a *morphism* if $h(\lambda) = \lambda$ and $h(a_1...a_n) = h(a_1)h(a_2)...h(a_n)$, for all $a_1...a_n$ in Σ. A morphism is also known as a *homomorphism.*

A natural generalization of the replacement of symbols by words as found in morphisms is the replacement of symbols by sets of words, that is, *substitution.* Let $\mathcal{L}$ be a family of languages and Σ and Δ be two alphabets. Then a mapping $\sigma : \Sigma^* \to 2^{\Delta^*}$ is said to be a *substitution* if $\sigma(\lambda) = \{\lambda\}$ and $\sigma(a_1...a_n) = \sigma(a_1)...\sigma(a_n)$, for all $a_1...a_n$ in Σ. If additionally, for all a in Σ, $\sigma(a)$ is in $\mathcal{L}$, then we say that σ is an $\mathcal{L}$ *substitution.*

Given an alphabet V, a *language* L over V is a subset of V^*. Since languages are sets, the Boolean operations of *union, intersection* and *complement* are applicable and defined as usual. Given two languages L_1 and L_2, possibly over different alphabets, the *catenation* of L_1 and L_2 is denoted L_1L_2 and equals the set $\{x_1x_2 \mid x_1 \text{ in } L_1 \text{ and } x_2 \text{ in } L_2\}$.

A *Chomsky grammar* is a quadruple $G = (N, T, P, S)$ where N is the nonterminal alphabet, T is the terminal alphabet, $S \in N$ is the axiom (start symbol) and P is the set of rewriting rules (or productions), written as $x \to y$. A grammar is *context-sensitive* when its rules are of the form $x_1Ax_2 \to x_1wx_2$, x_1, x_2, w being strings over V_G, $A \in N$, $w \neq \lambda$. A grammar is *context-free* when all its rules are of form $A \to x$, where $A \in N$, $x \in V_G^*$. A grammar is called *regular* when it contains only rules of the form $A \to a$, $A \to aB$, $a \in T$, $A, B \in N$.

A direct *derivation* in G is denoted by $\Longrightarrow$ (by $\Longrightarrow_G$ when we need this information). The transitive (reflexive) closure of the relation $\Longrightarrow$ is denoted by $\Longrightarrow^+$ ($\Longrightarrow^*$).

The language generated by G, denoted by $L(G)$, is

$$L(G) = \{w \mid S \Longrightarrow_G^* w, w \in T^*\}$$

We denote by RE, CS, CF, LIN, REG the classes of unrestricted, context-sensitive, context-free, linear and regular grammars, respectively.

A rewriting system $G = (V, w, P)$ where V is an alphabet, $w \in P^+$ is an axiom and P is a finite set of rewriting rules $A \to v$, with $A \in V$, $v \in V^*$ is called a *pure context-free grammar*. Rules of G are applied in derivations as follows: for two strings $x, y \in V^*$ we say that x directly derives y in G, written as $x \Longrightarrow y$ iff $x = x_1 a x_2$, $y = x_1 v x_2$, $x_1, x_2 \in V^*$, and $a \to v \in P$. The generated language is defined by $L(G) = \{x \in V^* \mid w \Longrightarrow^* x\}$.

An 0L-system (Lindenmayer system) is a triple $G = (V, P, w)$ where V is an alphabet, $w \in V^+$ is the axiom and P is a finite set of productions of the form $a \to \alpha$, where $a \in V$, $\alpha \in V^*$, and P contains for each letter $a \in V$ at least one production of the form $a \to \alpha$.

For an 0L-system $G = (V, P, w)$ and for two strings $x, y \in V^*$ we say that x directly derives y in G, denoted by $x \Longrightarrow_G y$, if $x = a_1 a_2 \ldots a_n$ and $y = \alpha_1 \alpha_2 \ldots \alpha_n$, where $a_i \to \alpha_i$ is a production in P, $1 \leq i \leq n$.

The language generated by 0L-system G is $L(G) = \{x \mid w \Longrightarrow_G^* x, x \in V^*\}$.

A 4-tuple $G = (N, T, P, S)$ is said to be a *generalized semi-conditional grammar* if N, T, S are the set of nonterminals, the set of terminals, and the start symbol, respectively, and P is a finite set of productions of the form $p = (E, F) : A \to w$, where $E, F \subseteq (N \cup T)^+$ with E, F being finite sets

and $A \to w$ is context-free production; p can be applied to a setential form $x \in (N \cup T)^*$ if each element of E and no element of F is a subword of x and $A \to w$ can be executed. If E or F is the empty set, then no check of E or F, respectively, is made. E is said to be the set of permitting context conditions and F the set of forbidding context conditions of p.

Chapter 2

Grammar Systems Theory

2.1 Grammar Systems: A Formal Language Model for Multi-Agent Systems

Grammar System Theory is a recent and active branch in the field of Formal Languages that provides syntactic models for describing multi-agent systems at the symbolic level using tools from formal grammars and languages. The attempt of the 'parents' of the theory was, as themselves state in [Csuhaj-Varjú *et al.*, 1994a, p. 5], *'to demonstrate a particular possibility of studying complex systems*[1] *in a purely syntactic level'* or, what is the same, to propose a *grammatical* framework for multi-agent systems.

What to do if we face a situation in which we have to solve a problem or to perform a specific task having at hand only agents with limited capabilities that are not able to carry out the duty alone? Intuitively, it seems that a

[1] *Complex systems* are defined by Simon as those 'made up of a large number of parts that interact in a non-simple way. In such systems, the whole is more than the sum of the parts, not in an ultimate, metaphysical sense, but in the important pragmatic sense that, given the properties of parts and the laws of their interaction, it is not a trivial matter to infer the properties of the whole.' [Simon, 1981].

good idea is to tackle the problem using a group of agents, instead of just one. We can try in this way to achieve goals that are beyond the capability of single agents but that can be achivable by a group of them, to achieve the so-called *social goals.* Artificial Intelligence faces often situations of this kind and it has offered two different approaches to solve the question:

1. Distributing the information to cooperating parts. This approach is the one taken by the so-called **Distributed Artificial Intelligence** which is concerned with the collaborative solution of global problems by a distributed group of entities. Here several agents interact and cooperate in a common environment in order to solve a problem. Mutual sharing of information is necessary to reach a successful solution of the problem.

2. Creating a multi-agent system of *autonomous* agents. This approach gives rise to the so-called **Decentralized Artificial Intelligence** (cfr. [Demazeau & Muller, 1990]) which is concerned with the activity of *autonomous* agents in a multi-agent world. Here the functionality of an agent is viewed as an emergent property of its intensive interaction with its dynamic environment. Each autonomous agent may accomplish its own task, or cooperate with other agents, to perform its own individual task or a global social one. Each agent has its own individual existence which is not justified by the existence of other agents[2].

In these two approaches, notions such as *distribution, cooperation, synchronization, parallelism, communication,* etc. reveal themselves as very important. But these concepts are very common nowadays not only in Artificial

[2]Distributed and Decentralized AI have a common interest: the behaviour of distributed entities. However, while in the former a global task is initially defined, being the problem to design the distributed agent that will enable the performance of the global task; in the latter, decentralized autonomous entities are initially defined and our task is to study how these entities can achieve the solution of the problems.

Intelligence, but also in other areas such as Cognitive Science, Psychology, Computer Science and other related fields. Materializations of these ideas are parallel computers, distributed data bases, computer nets, etc. Similarly, psychologists speak about the modularity of mind and about the possibility that the mind functions in a distributed way. Taking into account that Formal Language Theory can be, in some sense, involved in those fields, an important challenge for it is to define a formal model for capturing these concepts. And here is where Grammar Systems appear, showing themselves as a way of modelling in terms of formal grammars and languages all the above notions.

Grammar Systems Theory could be thought of, mainly, as a formal theoretic model of Distributed Artificial Intelligence. However, the idea of Decentralized Artificial Intelligence could be found as a motivation of some special variants of Grammar Systems, such as *Colonies*, or in some subfields of the theory such as *Eco-Grammar Systems*.

But, what is a Grammar System? Roughly speaking, a Grammar System is a *set* of grammars working together, according to a specified protocol, to generate a language. Notice that while in classical Formal Language Theory *one* grammar (or automata) works individually to generate (or recognize) *one* language; here, instead, we have *several* grammars working together, in order to produce *one* language.

Defined in this way, it is easy to observe the relationship or analogy between this grammatical model and the so-called *Multi-Agent Systems* -defined as a *set* of computational agents that perform local problem solving and cooperatively interact to solve a *single* problem. If theory of Multi-Agent Systems is said to deal with *behavior*, *cooperation*, and *coordination* of intelligent agents that are solving a problem together, theory of Grammar Systems has been defined in [Dassow, Păun & Rozenberg, 1997] as the theory of *cooperation protocols*, taken into account the crucial importance the protocol of

cooperation has in this formal model.

Two basic classes of Grammar Systems have been distinguished:

1. **Cooperating Distributed Grammar Systems** (CD grammar systems, for short) which work in a sequential way, and

2. **Parallel Communicating Grammar Systems** (shortly, PC grammar systems) that function in parallel.

To these two basic types of Grammar Systems, several variants have been added in the relative short history of the theory. In this chapter, we will present formal definitions of the usual CD and PC Grammar Systems, and we will introduce some of the variants that will be of utility in the body of the thesis.

Up to now, we have reviewed some of the motivations that stand behind Grammar System Theory. Let us turn now to some historical notes. The theory was launched in 1988, when Cooperating Distributed Grammar Systems have been proposed as a syntactic model of the blackboard architecture of problem solving. One year later, Parallel Communicating Grammar Systems -highly motivated by the 'classroom model' of problem solving- are introduced as a grammatical model of parallelism. Since 1988 the theory has developed into several directions, motivated by several scientific areas. Other than Distributed and Decentralized Artificial Intelligence, Artificial Life, Molecular Computing, Robotics, Natural Language Processing, Ecology, Sociology, etc. have suggested some modifications of the basic model, and have given rise to the appearance of different variants and subfields of the theory. The important advantages that Grammar Systems present with respect to classical models encourage the application of this theory in several fields. The present dissertation tries to be an example of this applicability in the areas of natural languages and cultural evolution.

Let us examine now in a more detailed way the main types of Grammar Systems and some variants of them.

2.2 Cooperating Distributed Grammar Systems

The initial motivation for defining Cooperating Distributed Grammar Systems (CD Grammar Systems, for short) comes for the blackboard model of problem solving as it is shown in [Csuhaj-Varjú & Dassow, 1990], the paper where CD Grammar Systems were introduced. It should be noted, however, that the first appearance of the syntagm 'cooperating grammar system' was in [Meersman & Rozenberg, 1978] with motivations related to two-level grammars. A similar idea appears already in [Abraham, 1972] where compound and serial grammars are considered. Modular grammars, introduced in [Atanasiu & Mitrana, 1989], present also a similar motivation.

A CD Grammar System consists of a finite set of generative grammars that cooperate in the derivation of a common language. Component grammars generate the string in turns (thus, sequentially), under some cooperation protocol. Work of such a system can be described as follows: initially the axiom is the common sentential form. At each moment in time, one grammar (and just one) is active, this is, it rewrites the common string, while the rest of grammars that make up the system are inactive. Conditions under which a component can start/stop its activity on common sentential form are specified by a cooperation protocol. Terminal strings generated in this way form the language of the system.

As it has been already pointed out, in the architecture of a CD Grammar System one can easily recognize the structure of the blackboard model as it is used in problem solving. Blackboard model, defined as a highly structured

special case of opportunistic problem solving, is described in [Nii, 1989] as consisting of three major components:

1. *Knowledge sources.* Knowledge needed to solve the problem is partitioned into knowledge sources, which are kept separate and independent.

2. *Blackboard data structure.* Problem solving state data are kept in a global database, the *blackboard.* Knowledge sources produce changes in the blackboard that lead incrementally to a solution to the problem. Communication and interaction among knowledge sources take place solely through the blackboard.

3. *Control.* Knowledge sources respond opportunistically to changes in the blackboard. There is a set of control modules that monitor changes in the blackboard and decide what actions to take next. Criteria are provided to determine when to terminate the process. Usually one of the knowledge sources indicates when the problem solving process is terminated, either because an acceptable solution has been found or because the system cannot continue further for lack of knowledge or data.

An analogy can be drawn between CD Grammar Systems and the above model of problem solving: the common sentential form is the blackboard; component grammars correspond to knowledge sources; protocol of cooperation encodes control on the work of knowledge sources; rewriting of a non-terminal symbol can be interpreted as a developmental step on the information contained in the current state of the blackboard; and, finally, a solution to the problem corresponds to a terminal word.

Tha basic model of CD Grammar Systems presents sequentiallity in its work and homogeneity in the cooperation protocol. However, variants have

been introduced that present some parallelism in their functioning ('teams') or that change the initial homogeneity in heterogeneity of the modes of co-operation ('hybrid systems'). Basic model has been extended, also, by the addition of 'extra' control mechanisms.

Let us introduce now the formal definitions of CD Grammar Systems.

Definition 2.2.1 *A Cooperating Distributed Grammar System of degree n, $n \geq 1$, is a construct*[3]

$$\Gamma = (N, T, S, G_1, \ldots, G_n),$$

where,

- *N, T are disjoint alphabets,*
- *$S \in N$ is the axiom,*
- *$G_i = (N, T, P_i)$, $1 \leq i \leq n$, the so-called components of the system Γ, are context-free grammars without axiom where,*
  - *N is the non-terminal alphabet,*
  - *T is the terminal alphabet,*
  - *P_i is a finite set of context-free rules over $N \cup T$.*

Definition 2.2.2 *Let Γ be a CD Grammar System as defined above. Let $x, y \in (N \cup T)^*$. Then we write $x \Longrightarrow_{G_i}^{k} y$, for $1 \leq i \leq n$, iff there are words $x_1, x_2, \ldots, x_{k+1}$ such that*

[3]In [Csuhaj-Varjú *et al.*, 1994a] the basic type of CD Grammar System is defined as $\Gamma = (T, S, G_1, \ldots, G_n)$, and each G_i as (N_i, T_i, P_i). However, the main part of papers in the field as well as [Dassow, Păun & Rozenberg, 1997], present the definition we give here that appears in [Csuhaj-Varjú *et al.*, 1994a, p. 137] as an 'other class of CD Grammar System.'

(i) $x = x_1$, $y = x_{k+1}$,

(ii) $x_j \Longrightarrow_{G_i} x_{j+1}$, *i.e.* $x_j x'_j A_j x''_j$, $x_{j+1} = x'_j w_j x''_j$, $A_j \to w_j \in P_i$, $1 \leq j \leq k$.

Moreover, we write

$x \Longrightarrow^{\leq k}_{G_i} y$ *iff* $x \Longrightarrow^{k'}_{G_i} y$ *for some* $k' \leq k$,

$x \Longrightarrow^{\geq k}_{G_i} y$ *iff* $x \Longrightarrow^{k'}_{G_i} y$ *for some* $k' \geq k$,

$x \Longrightarrow^{*}_{G_i} y$ *iff* $x \Longrightarrow^{k}_{G_i} y$ *for some* k,

$x \Longrightarrow^{t}_{G_i} y$ *iff* $x \Longrightarrow^{*}_{G_i} y$ *and there is no* $z \neq y$ *with* $y \Longrightarrow^{*}_{G_i} z$.

Any derivation $x \Longrightarrow^{k}_{G_i} y$ corresponds to k direct derivation steps in succession by grammar G_i. $\leq k$-derivation mode corresponds to a time limitation, since agent can perform *at most* k derivation steps. $\geq k$-derivation mode represents competence, since it requires agent to perform *at least* k derivation steps. $*$-mode denotes and arbitrary derivation: agent *can* work on the sentential string as long as it wants to do it. And finally, t- mode stands for a terminal derivation, where agent *must* perform derivation steps as long as it can.

Definition 2.2.3 *Let* Γ *be a CD Grammar System, and denote* $D = \{*, t\} \cup \{= k, \leq k, \geq k \mid k \geq 1\}$. *The language generated by the system* Γ *in the derivation mode* $f \in D$ *is*

$$L_f(\Gamma) = \{w \in T^* \mid S \Longrightarrow^{f}_{P_{i_1}} w_1 \Longrightarrow^{f}_{P_{i_2}} \ldots \Longrightarrow^{f}_{P_{i_m}} w_m = w, m \geq 1, 1 \leq i_j \leq n, 1 \leq j \leq m\}$$

For an example of CD Grammar Systems, see Appendix, p. 495. Information about the generative capacity of Cooperating Distributed Grammar Systems can be found in [Csuhaj-Varjú *et al.*, 1994a, pp. 36-72], in

[Dassow, Păun & Rozenberg, 1997, p. 162] and in [Mihalache, 1998a, p. 16]. Other than [Csuhaj-Varjú *et al.*, 1994a, Dassow, Păun & Rozenberg, 1997], for more information about CDGS, the reader can see papers referred in the bibliography as [Csuhaj-Varjú, 1991a, Csuhaj-Varjú, 1991b, Dassow, 1995b, Csuhaj-Varjú, Dassow & Kelemen, 1991, Csuhaj-Varjú & Kelemen, 1989] [Dassow & Kelemen, 1991, Bordihn & Csuhaj-Varjú, 1996] [Mihalache & Mitrana, 1999].

2.2.1 Controlled Cooperating Distributed Grammar Systems

As we have seen in the above section, only very weak stop conditions are present in usual CD Grammar Systems. This makes very simple cooperation among agents, since there is no mechanism (other than stop conditions) that controls the sequence of work of those components. To cover this lack, the basic model has been enlarged by adding *control mechanisms*. In [Csuhaj-Varjú *et al.*, 1994a], three different types of such mechanisms have been introduced:

1. **external control**, when, for example, some sequence of components is fixed in advance;

2. **internal control**, which establishes some specific conditions on the sentential form that determine start/stop conditions;

3. **memories**, where agents compute some additional values (not depending on current sentential form) that determine whether or not they can start/stop their actions.

CD Grammar Systems with External Control

Adding an *external control* is one of the ways that has been proposed to generalize CD Grammar Systems with control mechanisms. This type of control fixes in advance the order in which grammars that build up the system have to participate in the rewriting of sentential form.

Definition 2.2.4 *Let Γ be a CD Grammar System as defined in Definition 2.2.1 and let $f \in \{*, t, 1, 2, \ldots, \leq 1, \leq 2, \ldots, \geq 1, \geq 2, \ldots\}$. Further let $U = (W, E)$ be a directed graph the nodes of which are labelled by components. Then the language $L_f^U(\Gamma)$ generated by Γ controlled by U in the f-mode is defined as the set of all words $z \in T^*$ for which there is a derivation*

$$S = w_0 \Longrightarrow_{G_{i_1}}^{f} w_1 \Longrightarrow_{G_{i_2}}^{f} w_2 \Longrightarrow_{G_{i_3}}^{f} \ldots \Longrightarrow_{G_{i_m}}^{f} w_m = z,$$

such that, for $1 \leq j \leq m-1$,

$$(G_{i_j}, G_{i_{j+1}}) \in E$$

The reader can find an example of CDGS with External Control in Appendix, p. 497. For information about the generative capacity of CDGS with External Control we refer to [Csuhaj-Varjú *et al.*, 1994a, p. 75], where it is offered a theorem that characterizes the generative power of CDGS with control by graphs.

CD Grammar Systems with Internal Control

The second type of control mechanism added to CD Grammar Systems is *internal control.* This is done by some specific conditions on the sentential form that determine start/stop conditions for participation of components in rewriting process. We can consider this type of control as a *dynamic* one -

since it takes into consideration current state of sentential form- in contrast with the *external control* that can be considered *static* -since state of the sentential form does not influence the sequence of components that has been established in advance. Let us see, now, formal definition of a CD Grammar System with internal control.

Definition 2.2.5 *A dynamically controlled CD Grammar System* Γ *is an* $(n+2)$*-tuple*

$$\Gamma = (T, G_1, G_2, \ldots, G_n, S)$$

where

- $T \subseteq \bigcup_{i=1}^{n} T_i$ *is the terminal alphabet*
- *for* $1 \leq i \leq n$, $G_i = (N_i, T_i, P_i, \pi_i, \rho_i)$ *is a grammar or component with start condition* π_i *and stop condition* ρ_i, *i.e., for* $1 \leq i \leq n$,
  - N_i *and* T_i *are disjoint alphabets,*
  - $V_i = N_i \cup T_i$,
  - P_i *is a finite set of rewriting rules,*
  - π_i ρ_i *are predicates on* V^*.
- $S \in N$ *is the axiom.*

Definition 2.2.6 *Let* Γ *be a dynamically controlled CD Grammar System as in Definition 2.2.5. Then we define the language* $L(\Gamma)$ *generated by* Γ *as the set of all words* $Z \in T^*$ *for which there is a derivation*

$$S = w_0 \Longrightarrow^*_{G_{i_1}} w_1 \Longrightarrow^*_{G_{i_2}} w_2 \Longrightarrow^*_{G_{i_3}} \ldots \Longrightarrow^*_{G_{i_r}} w_r = z$$

such that, for $1 \leq j \leq r$,

$$\pi_{i_j}(w_{j-1}) = \textit{ true and } \rho_{i_j}(w_j) = \textit{ true},$$

and for $f \in \{t, 1, 2, \ldots, \leq 1, \leq 2, \ldots, \geq 1, \geq 2, \ldots\}$ *we define the language* $L_f(\Gamma)$ *generated by* Γ *in the* f*-mode as the set of all words* $z \in T^*$ *such that there is a derivation*

$$S = w_0 \Longrightarrow^f_{P_{i_1}} w_1 \Longrightarrow^f_{P_{i_2}} w_2 \Longrightarrow^f_{P_{i_3}} \ldots \Longrightarrow^f_{P_{i_r}} w_r = z$$

such that, for $1 \leq j \leq r$, $\pi_{i_j}(w_{j-1}) = true$.

In the definition of $L_f(\Gamma)$ the stop condition $\rho_i(w_j) =$ true is replaced by the stop condition which is naturally given by f-mode.

Definition 2.2.7 *Let* Γ *be a dynamically controlled CD Grammar System as in Definition 2.2.5. We define some special types of predicates. We say that the predicate* σ *on* V^* *is of*

- *type* (a) *iff* $\sigma(w) = true$ *for all* $w \in V^*$,
- *type* (rc) *iff there are two subsets* R *and* Q *of* V *and* $\sigma(w) = true$ *iff* w *contains all letters of* R *and* w *contains no letter of* Q,
- *type* (K) *iff there are two words* x *and* x' *over* V *and* $\sigma(w) = true$ *iff* x *is a subword of* w *and* x' *is not a subword of* w,
- *type* (K') *iff there are two finite subsets* R *and* Q *of* V^* *and* $\sigma(w) = true$ *iff all words of* R *are subwords of* w *and no word of* Q *is a subword of* w,
- *type* (C) *iff there is a regular set* R *over* V *and* $\sigma(w) = true$ *iff* $w \in R$.

Intuitively, (a) means that the grammar can start/stop at any moment, (rc) means that it can start/stop only if some letters are present/absent in the current sentential form, and (K), (K') and (C) denote such cases where some global context conditions have to be satisfied by the current sentential form.

For an example of CDGS with Internal Control, the reader is referred to Appendix, p. 499. Information about the generative capacity of CDGS with Internal Control can be found in [Csuhaj-Varjú *et al.*, 1994a, pp. 88-100].

CD Grammar Systems with Memories and Registers

Till now we have seen CD Grammar Systems that are able to communicate only through the blackboard. That is, they do not have additional devices to exchange information. CD Grammar Systems enlarged with tools for communication have been considered. CD Grammar Systems with *memories* are an example.

A CD Grammar System with memories is a finite set of semi-conditional grammars, each of them associated with a stack called *memory*. Throughout derivation process, component grammars send messages to each other and use their memories for storing/erasing messages they receive. In this variant of CD Grammar System, changes in current sentential form are controlled both by contents of memories and context conditions associated with productions. Thus, a direct derivation step consists of testing some parts of the memory, checking sentential form, executing a rule, updating current contents of the memory and sending messages to other cooperating grammars. Formally,

Definition 2.2.8 *A CD Grammar System with memories (mCD Grammar System, for short) is a tuple*

$$\Gamma = (T, G_1, ..., G_n, S),$$

where

- $T \subseteq \bigcup_{i=1}^{n} T_i$ *is the terminal alphabet,*
- $S \in N_i$ *for some* i, $1 \leq i \leq n$ *is the axiom,*
- $G_i = (N_i, T_i, P_i)$, *for* $1 \leq i \leq n$, *are semi-conditional grammars where,*
  - N_i *and* T_i *are finite nonempty disjoint sets,*
  - P_i *is a finite set of productions of the form*

$$p = (c, (r, q) : A \rightarrow w, d, m_1, \ldots, m_n),$$

where $d, m_1, \ldots, m_n$ *are words over* $N_i \cup T_i$, c *is either the empty string or it is a nonempty word over* $N_i \cup T_i$ *and* $(r, q) : A \rightarrow w$ *is a semi-conditional production.*

We say that c *is the memory contents checkword,* r *and* q *are the permitting and the forbidding context conditions of* p; r *and* q *are strings in* $(N_i \cup T_i)^+$ *or they are equal to the empty set (when* $r = \emptyset$ *or* $p = \emptyset$, *then the corresponding condition is ignored, it is assumed to hold for all the strings in* $(N_i \cup T_i)^*$*);* $A \rightarrow w$ *is a context-free production, the core of* p, d *is the string deleted from the memory of* G_i *and* $m_1, \ldots, m_n$ *are the messages to be sent to the memory of* $G_1, \ldots, G_n$, *respectively.*

Definition 2.2.9 *Let* $\Gamma = (T, G_1, ..., G_n, S)$ *be an mCD Grammar System. A tuple* $\alpha = (w, v_1, \ldots, v_n)$, *where* $w_i, v_i \in V^*$, $1 \leq i \leq n$ *is said to be a configuration of* Γ; w *is said to be a current sentential form and* v_i, $1 \leq i \leq n$, *is said to be the current contents of the* i*-th memory (the memory of the* i*-th component).*

A configuration α *is said to be an initial configuration if* $\alpha = (S, \epsilon, \ldots, \epsilon)$, *and it is said to be a final configuration if* $\alpha = (w, \epsilon, \ldots, \epsilon)$, *where* $w \in T^*$

Direct derivation steps are controlled both by context conditions and by content of memories.

Definition 2.2.10 *Let $\Gamma = (T, G_1, ..., G_n, S)$, $n \geq 1$, be an mCD Grammar System and let $\alpha = (w, v_1, \ldots, v_n)$ and $\alpha' = (w', v_1', \ldots, v_n')$ be two configurations of Γ. We say that α derives α' directly, denoted by*

$$\alpha \Longrightarrow_\Gamma \alpha',$$

if there is a component G_i of Γ, for some i, $1 \leq i \leq n$, with a production

$$p = (c, (r, q) : A \to v, d, m_1, \ldots, m_n),$$

such that the following conditions hold:

(i) $w = xAy$ *and* $w' = xvy$ *for some* $x, y \in (N_i \cup T_i)^*$;

(ii) if $r \neq \emptyset$, *then* $w = u_1 r u_2$ *for some* $u_1, u_2 \in (N_i \cup T_i)^*$;

(iii) if $q \neq \emptyset$, *then* $w \neq u_1' q u_2'$ *for any* $u_1', u_2' \in (N_i \cup T_i)^*$;

(iv) $v_i = c\bar{v}_i$ *for some* $\bar{v}_i \in (N_i \cup T_i)^*$;

(v) $v_k' = m_k v_k$ $(v_k' = v_k m_k)$ *for all* k, $1 \leq i, k \leq n$ *with* $i \neq k$, *if the k-th memory is organized as a stack;*

(vi) $v_i' = m_i v_i''$ *and* $d v_i'' = v_i$ *or* $(v_i' = v_i'' m_i)$ *if the i-th memory is organized as a stack.*

These conditions can be explained as follows: conditions (i), (ii) and (iii) mean that a production of a semi-conditional grammar will be executed. Condition (iv) describes how current content of the memory is tested. Conditions (v) and (vi) describe how messages are sent and how the local memory of the component is updated by erasing some unnecessary information.

Definition 2.2.11 *Let $\Gamma = (T, G_1, ..., G_n, S)$, $n \geq 1$, be an mCD Grammar System as the defined above, then the language generated by Γ is*

$$L(\Gamma) = \{w \mid (S, \epsilon, \ldots, \epsilon) \Longrightarrow_{\Gamma}^{*} (w, \epsilon, \ldots, \epsilon), w \in T^*\}$$

An example of CDGS with memories is offered in Appendix, p. 501. For results on the generative power of CDGS with memories, the reader can see [Csuhaj-Varjú *et al.*, 1994a, pp. 109-115].

We have considered CD Grammar Systems with memories where each component of the system has associated with it a memory which changes when derivation steps are performed and whose content constrains the start/stop of an agent. Now, it is also useful to consider a *global memory*-a register- shared by all the agents during the derivation of a CD Grammar System. In this case, the content of the register is a certain number. The application of an agent's rule results in a change of the content of the register. The switching from one agent to another is only possible if the register is empty. Formally,

Definition 2.2.12 *A CD Grammar System with and additive register is a construct*

$$\Gamma = (T, G_1, G_2, \ldots, G_r, v_1, v_2, \ldots, v_r, S),$$

where T, $G_1, G_2, \ldots, G_r$ and S are defined as an usual CD Grammar System and, for $1 \leq i \leq r$, v_i is a mapping from P_i into the set $\mathbf{Z}$ of integers.

Definition 2.2.13 *Let $\Gamma = (T, G_1, G_2, \ldots, G_r, v_1, v_2, \ldots, v_r, S)$ be a CD Grammar System with an additive register as defined in Definition 2.2.12. Let $(x, \alpha), (y, \beta) \in V_G^* \times \mathbf{Z}$, $1 \leq i \leq r$, and $p \in P_i$. Then we say*

$$(x, \alpha) \Longrightarrow_p (y, \beta)$$

iff

$$x = xAx, y = xwx, p = A \to w, \text{ and } \beta = \alpha + v_i(p).$$

Definition 2.2.14 *Let* $\Gamma = (T, G_1, G_2, \ldots, G_r, v_1, v_2, \ldots, v_r, S)$ *be a CD Grammar System with an additive register as in Definition 2.2.12. The language generated by* Γ *is defined as*

$L(\Gamma) = \{z \mid z \in T^*$ *and there are words* $x_1, x_2, \ldots, x_{n-1}$ *and components* $H_1, H_2, \ldots, H_n$ *such that* $(S,0) \Longrightarrow_{H_1}^* (x_1,0) \Longrightarrow_{H_2}^* (x_2,0) \Longrightarrow_{H_3}^* \ldots \Longrightarrow_{H_{n-1}}^* (x_{n-1},0) \Longrightarrow_{H_n}^* (z,0)\}$.

In Appendix, p. 504, can be found an example of CDGS with registers. Results on generative capacity of CDGS with register can be found in [Csuhaj-Varjú *et al.*, 1994a, p. 121].

2.2.2 Hybrid CD Grammar Systems

Every variant of CD Grammar Systems that has been presented up to now can be called *homogenous*, in the sense that all of them consist of a set of grammars of the same type working in the same way. Maybe, it could be more realistic to consider *heterogeneous* systems where agents were different, with different capabilities and where they were allowed to work in different ways. The answer to this new idea is the introduction of *hybrid CD Grammar Systems* where components work using different derivation modes.

Definition 2.2.15 *A hybrid CD Grammar System is a construct*

$$\Gamma = (N, T, S, (G_1, f_1), \ldots, (G_n, f_n)), n \geq 1,$$

where

- $(N, T, S, G_1, \ldots, G_n)$ *is a usual CD Grammar System,*
- $f_i \in D$, $1 \leq i \leq n$.

Definition 2.2.16 *Let* $\Gamma = (N, T, S, (G_1, f_1), \ldots, (G_n, f_n)), n \geq 1$, *be a hybrid CD Grammar System as above, then the language generated by* Γ *is*

$$L(\Gamma) = \{w \in T^* \mid S \Longrightarrow_{P_{i_1}}^{f_{i_1}} w_1 \Longrightarrow_{P_{i_2}}^{f_{i_2}} \ldots \Longrightarrow_{P_{i_m}}^{f_{i_m}} w_m = w, m \geq 1,$$
$1 \leq i_j \leq n, 1 \leq j \leq m\}$.

For information about the generative capacity of Hybrid Systems, the reader is referred to [Dassow, Păun & Rozenberg, 1997, p. 165] and [Csuhaj-Varjú *et al.*, 1994a, p. 147]. An example of Hybrid CDGS can be found in Appendix, p. 505.

2.2.3 Teams

Teams in grammars systems introduce *parallelism* in the model. They can be placed in the middle between CD Grammar Systems and PC Grammar Systems. At one extreme in the continuum of this theory we have CD Grammar Systems where components work sequentially, this is, at each unit in time only one agent is active, while the others are waiting; at the other extreme, we find PC Grammar System in which all the components are simultaneously active; and in the middle of this continuum, we find *teams* of CD Grammar Systems where *several* components are active simultaneously while some others are waiting for their turn.

Definition 2.2.17 *A prescribed team CD Grammar System (of variable size) is a construct*

$$\Gamma = (N, T, S, P_1, P_2, \ldots, P_n, Q_1, Q_2, \ldots, Q_m),$$

where

- $(N, T, S, P_1, P_2, \ldots, P_n)$ *is a CD Grammar System*
- $Q_i \subseteq \{P_1, P_2, \ldots, P_n\}$, $1 \leq i \leq m$, *is called a prescribed team.*

Definition 2.2.18 *Given a prescribed team CD Grammar System as above, a team* $Q_i = \{P_{i_1}, P_{i_2}, \ldots, P_{i_{s_i}}\}$, $1 \leq s_i \leq n$ *and* $1 \leq i \leq m$, *is used in a one-step derivation as follows. It is said that* x *derives* y *in one step, for* $x, y \in (N \in T)^*$, *by using team* Q_i, *written as* $x \Longrightarrow_{Q_i} y$ *iff*

(i) $x = x_1 A_1 x_2 A_2 \ldots x_{s_i} A_{s_i} x_{s_i+1}$,

(ii) $y = x_1 y_1 x_2 y_2 \ldots x_{s_i} y_{s_i} y_{s_i+1}$,

(iii) $x_l \in (N \in T)^*, 1 \leq l \leq s_i + 1$,

(iv) $A_k \rightarrow y_k \in P_{i_k}, 1 \leq k \leq s_i$

At any derivation step, from each component of the team being used one of its productions is used in parallel with all the other members of the team. As a team is a set, no order of the components is assumed and hence the rewritten symbols A_1 to A_{s_i} can appear in x in any order.

Definition 2.2.19 *Let* $\Gamma = (N, T, S, P_1, P_2, \ldots, P_n, Q_1, Q_2, \ldots, Q_m)$, *be a prescribed team CD Grammar System. Then the following modes of derivation can be defined for* Γ. *For* $x, y, z \in (N \in T)^*$, $k \geq 1$ *and* $1 \leq i \leq m$,

$x \Longrightarrow_{Q_i}^{\leq k} y$ *iff* $x \Longrightarrow_{Q_i}^{0} y$ *or* $x \Longrightarrow_{Q_i}^{k'} y$ *for some* $k' \leq k$,

$x \Longrightarrow_{Q_i}^{=k} y$ *iff* $x \Longrightarrow_{Q_i}^{k} y$,

$x \Longrightarrow_{Q_i}^{\geq k} y$ *iff* $x \Longrightarrow_{Q_i}^{k'} y$ *for some* $k' \geq k$,

$x \Longrightarrow_{Q_i}^{*} y$ *iff* $x \Longrightarrow_{Q_i}^{0} y$ *or* $x \Longrightarrow_{Q_i}^{k} y$ *for some* k,

$x \Longrightarrow_{Q_i}^{t_0} y$ *iff* $x \Longrightarrow_{Q_i}^{*} y$ *and there is no* z *such that* $y \Longrightarrow_{Q_i} z$,

$x \Longrightarrow_{Q_i}^{t_1} y$ *iff* $x \Longrightarrow_{Q_i}^{*} y$ *and for no component* $P_{i_j} \in Q_i$ *and no* z *there is a derivation* $y \Longrightarrow_{P_{i_j}} z$,

$x \Longrightarrow_{Q_i}^{t_2} y$ *iff* $x \Longrightarrow_{Q_i}^{*} y$ *and there is a component* $P_{i_j} \in Q_i$ *for which there is no derivation* $y \Longrightarrow_{P_{i_j}} z$.

The first four modes of derivation are the natural extension of modes in CD Grammar System to teams of grammars. The last three are variants of t-mode. In the case of mode t_0, the work of a team ends successfully when no further derivation step can be done as a team. In the case of mode t_1, the work ends when no component of the team can apply one of its productions any longer. And finally, in the mode t_2, the work of a team ends when there is at least one component that cannot longer apply one of its productions.

Definition 2.2.20 *Let* $\Gamma = (N, T, S, P_1, P_2, \ldots, P_n, Q_1, Q_2, \ldots, Q_m)$, *be a prescribed team CD Grammar System. The language generated by* Γ, *operating in mode* $f \in \{*, t_0, t_1, t_2\} \in \{\leq k, = k, \geq k \mid k \geq 1\}$, *is*

$L_f(\Gamma) = \{z \in T^* \mid S \Longrightarrow_{Q_{i_1}}^{f} w_{i_1} \Longrightarrow_{Q_{i_2}}^{f} \ldots \Longrightarrow_{Q_{i_p}}^{f} w_{i_p} = z, 1 \leq i_j \leq m, 1 \leq j \leq p\}$

In [Dassow, Păun & Rozenberg, 1997, p. 168], it is presented a theorem that summarized results about the generative capacity of Teams. The reader is referred to [ter Beek, 1996a, ter Beek, 1996b, Mateescu, 1995] for more information about the topic. For an example, see Appendix, p. 506.

2.3 Parallel Communicating Grammar Systems

Parallel Communicating Grammar Systems (PC Grammar Systems, for short) were introduced in [Păun & Sântean, 1989] as a grammatical model of *parallelism* in the broad sense.

Informally, a PC Grammar System consists of several usual grammars, each of them having its own sentential form. In each time unit (a common clock divides the time in units in a uniform way for all the components) each component uses a rule, rewriting the associated sentential form. Cooperation among agents takes place thanks to the so-called *query symbols* that permit communication among components. When a component introduces a query symbol for another component, the latter one sends its current sentential form to the former which replaces the query symbol with the string received. One component is distinguished as the *master,* and the language generated by it (with or without communication) is the language of the system

If CD Grammar System were considered a grammatical model of the blackboard system in problem solving, PC Grammar System can be thought of as a formal representation of the *classroom model.* Let us take the blackboard model and let us make in it the following modifications:

- allow each knowledge source to have its own 'notebook' containing the description of a particular subproblem of a given problem;
- allow each knowledge source to operate only on its own 'notebook' and let there exist one distinguished agent which operates on the 'blackboard' and has the description of the problem;
- and finally, allow agents to communicate by request the content of their own 'notebook'.

These modifications on the blackboard model lead us to 'classroom model' of problem solving where the classroom leader (the master) works on the blackboard, pupils have particular problems to solve using their notebooks, they can communicate in each step of problem solving, and global problem is solved through such cooperation on the blackboard.

As it happens with CD Grammar Systems, an easy analogy can be established between PC Grammar Systems and the classroom model: pupils correspond to grammars which make up the system, and their notebooks to the sentential forms. Production sets of grammars encode knowledge of pupils. Distinguished agent corresponds to the 'master'. Rewriting a nonterminal symbol can be interpreted as a developmental step of the information contained in the notebooks. A partial solution, obtained by a pupil corresponds to a terminal word generated in one grammar, while solution of the problem is associated to a word in the language generated by the 'master.'

We have seen in the case of CD Grammar System that modifications to the basic model have been introduced given rise to different new variants. In PC Grammar Systems we find variants as well. Modifications in the type of components or in the way of communicating strings produce several new types of Parallel Communicating Grammar Systems.

Let see now formal definition of a PC Grammar System.

Definition 2.3.1 *A Parallel Communicating Grammar System of degree n, $n \geq 1$, is a construct,*

$$\Gamma = (N, K, T, G_1, \ldots, G_n)$$

where,

- *N, T, K are mutually disjoint alphabets,*

- $K = \{Q_1, Q_2, \ldots, Q_n\}$ *are called query symbols and they are associated in a one-to-one manner to components* $G_1, ..., G_n$,
- $V_\Gamma = N \cup K \cup T$,
- $G_i = (N \cup K, T, P_i, S_i)$, $1 \leq i \leq n$, *the so-called components of the system, are usually Chomsky grammars where,*
  - N *is the non-terminal alphabet,*
  - $K = \{Q_1, Q_2, \ldots, Q_n\}$ *is the set of query symbols,*
  - T *is the terminal alphabet,*
  - P_i *is a finite set of rewriting rules over* $N \cup K \cup T$, *for* $1 \leq i \leq n$,
  - $S_i \in N$ *is the axiom, for* $1 \leq i \leq n$.

Definition 2.3.2 *Given a PC Grammar System* $\Gamma = (N, K, T, G_1, \ldots, G_n)$ *as above, for two n-tuples* $(x_1, x_2, \ldots, x_n)$, $(y_1, y_2, \ldots, y_n)$, $x_i, y_i \in V_\Gamma^*$, $1 \leq i \leq n$, *we write* $(x_1, \ldots, x_n) \Longrightarrow (y_1, \ldots, y_n)$ *if one of the next two cases holds:*

(i) $|x_i|_K = 0$, $1 \leq i \leq n$, *and for each* i, $1 \leq i \leq n$, *we have* $x_i \Longrightarrow y_i$ *in grammar* G_i, *or* $x_i \in T^*$ *and* $x_i = y_i$;

(ii) there is i, $1 \leq i \leq n$, *such that* $|x_i|_K > 0$; *then, for each such* i, *we write* $x_i = z_1 Q_{i_1} z_2 Q_{i_2} \ldots z_t Q_{i_t} z_{t+1}$, $t \geq 1$, *for* $z_j \in V_\Gamma^*$, $|z_j|_K = 0$, $1 \leq j \leq t+1$; *if* $|x_{i_j}|_K = 0$, $1 \leq j \leq t$, *then* $y_i = z_1 x_{i_1} z_2 x_{i_2} \ldots z_t x_{i_t} z_{t+1}$ *[and* $y_{i_j} = S_{i_j}$, $1 \leq j \leq t$*]; when, for some* j, $1 \leq j \leq t$, $|x_{i_j}|_K \neq 0$, *then* $y_i = x_i$; *for all* i, $1 \leq i \leq n$, *for which* y_i *is not specified above, we have* $y_i = x_i$.

In words, an n-tuple $(x_1, \ldots, x_n)$ directly yields $(y_1, \ldots, y_n)$ if:

(i) either no query symbols apperars in $x_1, \ldots, x_n$, and then we have a componentwise derivation, $x_i \Longrightarrow_{G_i} y_i$, $1 \leq i \leq n$ (one rule is used in each component G_i), excepting the cases when x_i is terminal, $x_i \in T^*$, and then $y_i = x_i$, or

(ii) in the case of query symbols appearance, a *communication step* is performed as these symbols impose: each occurence of Q_j in x_i is replaced by x_j, providing x_j does not contain query symbols. More exactly, a component x_i (containing query symbols) is modified only when all its occurrences of query symbols refer to strings without query symbols occurrences. In a communication operation, communicated string x_j replaces query symbol Q_j (we say that Q_j is *satisfied* in this way); after that, grammar G_j resumes working from its axiom. Communication has priority over effective rewriting. If some query symbols are not satisfied at a given communciation step, they will be satisfied at next one (providing they ask for strings without query symbols at that moment) and so on. No rewriting is possible when at least a query symbol is present.

Note that rules $Q_I \rightarrow x$ are never used, hence we shall assume that such rules do not appear.

We have denoted in the same way, by $\Longrightarrow$, both the componentwise derivation steps and the communication steps. As usual, by $\Longrightarrow^*$ we shall denote the reflexive transitive closure of this relation, that is the sequences of intercalated derivations and communications steps.

The work of a PC Grammar System is blocked in two cases:

1. when a component x_i of the current n-tuple $(x_1, ..., x_n)$ is not terminal with respect to G_i, but no rule of G_i can be applied to X_i (this can happen both due to rules in P_i and to communication), and

2. when a *circular query* appears: if G_{i_1} introduces Q_{i_2}, G_{i_2} introduces Q_{i_3} and so on, until $G_{i_{k-1}}$, which introduces Q_{i_k}, and G_{i_k} which introduces Q_{i_1}, then no derivation is possible (communication has priority), but no communication (in this cycle) is possible (only strings without query symbols occurrences are communicated).

Definition 2.3.3 *The language generated by a PC Grammar System* Γ *as above is*

$$L(\Gamma) = \{x \in T^* \mid (S_1, S_2, \ldots, S_n) \Longrightarrow^* (x, \alpha_2, \ldots, \alpha_n), \alpha_i \in V_\Gamma^*, 2 \leq i \leq n\}$$

So, we start from the n-tuple of axioms $(S_1, \ldots, S_n)$ and proceed by repeated rewriting and communication steps, until component G_1 produces a terminal string. Notice that in $L(\Gamma)$ we retain the strings generated in this way on the first component, without care about strings generated by $G_2, \ldots, G_n$. Component G_1 is called the *master* of the system.

PC Grammar Systems we have presented in the above definitions are systems without any restriction imposed on the use of query symbols, that is, each component G_i is allowed to introduce a symbol Q_j (obviously, when G_i introduces the symbol Q_i, the derivation is blocked by query circularity).

Definition 2.3.4 *Let* $\Gamma = (N, K, T, G_1, \ldots, G_n)$ *be a PC Grammar System. If only* G_1 *is allowed to introduce query symbols (formally,* $P_i \subseteq (N \in T)^* \times (N \in T)^*$ *for* $2 \leq i \leq n$*), then we say that* Γ *is a* **centralized** *PC Grammar System; in contrast, the unrestricted case is called* **non-centralized**.

Definition 2.3.5 *A PC Grammar System* $\Gamma = (N, K, T, G_1, \ldots, G_n)$ *is said to be* **returning** *(to axiom) if, after communicating, each component whose string has been sent to another component returns to axiom. A PC Grammar System is* **non-returning** *if in point (ii) of Definition 3.3.2, the words written in brackets,*

$$[y_{i_j} = S_{i_j}, 1 \leq j \leq t]$$

are erased. (That is, after communicating, the grammar G_{i_j} *does not return to* S_{i_j}*, its axiom, but continues processing the current string.)*

In [Csuhaj-Varjú *et al.*, 1994a, pp. 158-176], [Mihalache, 1998a, pp. 16-28] and [Dassow, Păun & Rozenberg, 1997, pp. 180-186] results on PCGS generative capacity can be found. An example of Parallel Communicating Grammar Systems is offered in Appendix, p. 510.

2.3.1 PC Grammar Systems with "Renaming"

Parallel Communicating Grammar Systems with "Renaming" appears in 1996 with the aim of solving some problems in the generation of specific non-context-free structures present in natural languages [Păun, 1996b]. Consider the following three languages

$$L_1 = \{xx \mid x \in \{a, b\}^*\}$$

$$L_2 = \{a^n b^n c^n \mid n \geq 1\}$$

$$L_3 = \{a^n b^m c^n d^m \mid n, m \geq 1\}$$

All three can be generated using PC Grammar Systems with context-free components; L_1 and L_2 can be generated even with right-linear systems; and

a small number of components suffices for the generation of them (for formal details cfr. [Csuhaj-Varjú *et al.*, 1994a, Păun, 1995d, Chiţu, 1997]).

If we now compare

$$L_3 = \{a^n b^m c^n d^m \mid n, m \geq 1\}$$

$$L_3' = \{a^n b^m a^n b^m \mid n, m \geq 1\}$$

we will see that L_3 is difficult to be generated -PC Grammar Systems that generate it are complex and have an intricate work- whereas L_3' is easy to be generated (cfr. [Păun, 1995d]). Clearly, the difficulty of L_3 rests on the correlation of a^n with c^n and b^n with d^n where $a \neq c$ and $b \neq d$ while in L_3' case letters that are correlated (a with a, and b with b) are equal. This fact suggested the variant of PC Grammar System that we are introducing in this section and that seems useful from a 'natural language' point of view: PC Grammar Systems with "Renaming."

Definition 2.3.6 *A PC Grammar System with "Renaming" is a construct*

$$\Gamma = (N, K, T, G_1, \ldots, G_n, h_1, \ldots, h_m)$$

where,

- N, T, K *are mutually disjoint alphabets,*
- $K = \{Q_1, Q_2, \ldots, Q_n\}$ *are called query symbols and they are associated in a one-to-one manner to the components* $G_1, ..., G_n$,
- $V_\Gamma = N \cup K \cup T$,
- $G_i = (N \cup K, T, S_i, P_i)$, $1 \leq i \leq n$, *the so-called components of the system, are usually Chomsky grammars where,*
  - N *is the non-terminal alphabet,*

- $K = \{Q_1, Q_2, \ldots, Q_n\}$ *is the set of query symbols,*
- T *is the terminal alphabet,*
- $S_i \in N$ *is the axiom, for* $1 \leq i \leq n$.
- P_i *is a finite set of rewriting rules over* $N \cup T \cup K \cup K'$, *for* $1 \leq i \leq n$, *where* $K' = \{[h_j, Q_i] \mid 1 \leq i \leq n, 1 \leq j \leq m\}$ *and each* $[h_j, Q_i]$ *is a symbol.*

- $h_j : (N \cup T)^* \to (N \cup T)^*$, $1 \leq j \leq m$, *are weak codes such that:*

  (i) $h_j(A) = A$, *for* $A \in N$,

  (ii) $h_j(a) \in T \cup \{\lambda\}$, *for* $a \in T$.

A PC Grammar System with "renaming" works in the same way as an usual PC Grammar System (cfr. Definition 2.3.2) with only one difference: whenever a $[h_j, Q_i]$ symbol appears in the sentential form of any of the components of the system, it must be replaced not by w_i, but by $h_j(w_i)$.

The language generated by a PC Grammar System with "renaming" is, as in the case of usual PC Grammar Systems defined above, the language of the 'master'.

Examples of PCGS with Renaming can be found in Appendix, p. 511.

2.3.2 PC Grammar Systems with Communication by Command

Motivated by data flow of WAVE-like architectures of highly parallel processors, by Connection machines, by Boltzmann machines and by other parallel devices, PC Grammar Systems with communication by command (shortly, CCPC) were introduced in [Csuhaj-Varjú, Kelemen & Păun, 1996].

In contrast with PC Grammar Systems presented up to now, where communication is done *by request*, in this new variant of Parallel Communicating Grammar Systems, communication is done *by command.* This means that when a string derived in a component of the system corresponds to another component -this is, matches the pattern associated to another component- it is sent to this latter component. So, we do not need appearance of a query symbol in the sentential form of a component to get the string of another component. In this case every component sends its sentential form whenever it realizes that the string it has matches the language selector of another component of the system.

The working of a CCPC does not differ very much from the functioning of an usual PC Grammar System. We start from an initial configuration where each component has its own start symbol and proceed by a sequence of alternating rewriting and communicating steps. As we have already said, a sentential form can be communicated only if it matches the language selector associated with the addressee component. After communication, the component either can return to its axiom or it can continue with its current string. As in usual PC Grammar Systems, communication has priority over rewriting. And, as it can be expected, the language of the system is the language generated by the component designed as the 'master.'

Let us look now at the formal definition of these systems.

Definition 2.3.7 *A PC Grammar System with communication by command is a construct*

$$\Gamma = (N, T, (S_1, P_1, R_1), \ldots, (S_n, P_n, R_n)), n \geq 1,$$

where

- *N and T are disjoint sets,*

- *N is the non-terminal alphabet,*
- *T is the terminal alphabet,*
- *(S_i, P_i, R_i), $1 \leq i \leq n$, are the components of the system, where*
  - *$S_i \in N$ is the axiom,*
  - *P_i is the set of rewriting rules over $N \cup T$,*
  - *$R_i \subseteq (N \cup T)^*$ is the selector language of the component i (R_i will be either specified as a regular set or by a pattern[4]).*

Let us now look at the functioning of a CCPC Grammar System. We start from a initial configuration $(S_1, \ldots, S_n)$. At any step, the state of the process will be described by an n-tuple $(x_1, ..., x_n)$ of strings over $N \cup T$. The system will modify the current configuration either:

1. by **rewriting** steps. A rewriting step will be performed componentwise. We can consider two possibilities:

   (a) each component has to use a rule, rewriting its current sentential form, except those components whose string is terminal which are

[4]Having an alphabet A of *constants* and an alphabet V of *variables*, a **pattern** is a string over $A \cup V$.

For a pattern π over A and V, the *language* associated to π, denoted by $L_A(\pi)$, consists of all the strings obtained from π by consistently replacing the variable occurrences by non-empty strings over A ('consistently' means that all the occurrences of the same variable are replaced by the same string).

For example, if $A = \{a, b\}$ and $V = \{X_1, X_2\}$, then $\pi = aaX_1X_1bX_2bb$ is a pattern. It defines the set of all the strings over A starting with two occurences of a, continuing with a replication of any string over A, one occurrence of b, any string over A, and ending with a double occurrence of b. Therefore, $x = aa(abb)(abb)b(bab)bb$ has the form specified by this pattern, while $y = aaabbababbabbb$ has not. So, we have $L_A(\pi) = \{aawwbzbb \mid w, z \in \{a, b\}^+\}$.

allowed to do nothing. It can be considered a *minimal* derivation strategy: one rewrites exactly until a target language is matched, or

(b) each component has to perform a *maximal* derivation step, this is, the rewriting step is accomplished when no further rewriting is possible on the sentential form of that component.

In both cases, at the end of the step, one checks whether or not a communication step can be performed. That is, one checks if currently generated strings match selector languages R_i. In the second case, a communication step *must* be performed, otherwise the system cannot perform another derivation step. In the first case, if no communication is to be performed, another rewriting step can be done.

In between those two cases we may also consider different derivation modes as k, $\leq k$, $\geq k$ steps, like in CD Grammar Systems.

2. or by **communication** steps. In order to define a communication step we have to specify rules for:

   - Defining the string to be communicated. This problem has at least two solutions:

     (a) communication *without* splitting. In this case only complete current sentential forms are considered as messages. If $x_i \in R_j$, then x_i as a whole is transmitted to the component j;

     (b) communication *with* splitting. In this case, for every decomposition $x_i = x_{i,1}x_{i,2}\ldots x_{i,k}$, $k \geq 1$, such that $x_{i,j} \in R_{s_j}$, for some $1 \leq s_j \leq n$, $1 \leq j \leq k$, component i will produce the messages $x_{i,1}, x_{i,2}, \ldots, x_{i,k}$ which will be transmitted to the matching components $s_1, s_2, \ldots, s_k$.

- Solving the conflicts at target components. This problem can have several solutions. Basically, the received messages can be
  (a) *adjoined* to the current string of the addressee, or
  (b) they can *replace* the string of the addressee.

  In both cases we can consider that
  (a) *only one* of the received messages is taken into account. We face here again two possibilities:
    i. either we choose *nondeterministically* one message, or
    ii. we consider a *priority* relation, for instance, defined by the natural ordering of the component sending messages.
  (b) or *all* the received messages concatenated in a specified order.
- Defining the next string for components which have transmitted messages. As in the case of usual PC Grammar System, a component which has submitted its string to other component, without receiving new strings, can
  (a) either *return* to its axiom, or
  (b) *continue* to process its current string.

Definition 2.3.8 *Let* $\Gamma = (N, T, (S_1, P_1, R_1), \ldots, (S_n, P_n, R_n)), n \geq 1$ *be a CCPC Grammar System working with maximal derivations as rewriting steps, communicating without splitting the strings, replacing the strings of the target component by a concatenation of the received messages, in the order of the system components, and returning to axioms after communication. For such a system we define a* **rewriting step** *by* $(x_1, \ldots, x_n) \Longrightarrow (y_1, \ldots, y_n)$ *iff*

$x_i \Longrightarrow^* y_i$ *in* P_i *and there is no* $z_i \in (N \cup T)^*$ *such that* $y_i \Longrightarrow z_i$ *in* P_i *(if* $x_i \in T^*$*, then* $y_i = x_i$*, otherwise* $x_i \Longrightarrow^+ y_i$*).*

A **communication step** *denoted by* $(x_1, \ldots, x_n) \vdash (y_1, \ldots, y_n)$ *is defined as follows. Let*

$$\delta_i(x_i, j) = \begin{cases} \lambda, & if \quad x_i \notin R_j \quad or \quad i = j, \\ x_i, & if \quad x_i \in R_j \quad and \quad i \neq j, \end{cases}$$

for $1 \leq i, j \leq n$,

$$\Delta(j) = \delta(x_1, j)\delta(x_2, j) \ldots \delta(x_n, j),$$

for $1 \leq j \leq n$ *(this is the "total message" to be received by the* j*-th component), and*

$$\delta(i) = \delta(x_i, 1)\delta(x_i, 2) \ldots \delta(x_i, n),$$

for $1 \leq i \leq n$ *(this is, the "total message" sent by the* i*-th component, a power of* x_i *indicating to how many targets the* i*-th components sends a message). Then for* $1 \leq i \leq n$, *we define*

$$y_i = \begin{cases} \Delta(i), & if \quad \Delta_i \neq \lambda, \\ x_i, & if \quad \Delta_i = \lambda \quad and \quad \delta(i) = \lambda \\ S_i & if \quad \Delta(i) = \lambda \quad and \quad \delta(i) \neq \lambda \end{cases}$$

In words, y_i is either the concatenation of the received messages, if any exists, or it is the previous string, when this component is not involved in communications, or it is equal to S_i, if this component sends messages but it does not receive messages. Observe that a component cannot send messages to itself.

The language generated by the CCPC Grammar System in this way can be defined in the two following modes:

1. as the set of all the terminal strings generated by a designed component of the system, its *master* (as in usual PC Grammar Systems where the first component is designed, by convention, as being the master),

2. or as the set of all the terminal strings generated by any component of the system.

In the formalization we will offer here, we consider the first case, this is, when the language of the system is the one generated by the first component (the *master*).

Definition 2.3.9 *Let* $\Gamma = (N, T, (S_1, P_1, R_1), \ldots, (S_n, P_n, R_n))$, $n \geq 1$ *be a CCPC Grammar System as above, then the language generated by* Γ *is defined as follows:*

$L(\Gamma) = \{w \in T^* \mid (S_1, \ldots, S_n) \Longrightarrow (x_1^{(1)}, \ldots, x_n^{(1)}) \vdash (y_1^{(1)}, \ldots, y_n^{(1)}) \Longrightarrow (x_1^{(2)}, \ldots, x_n^{(2)}) \vdash (y_1^{(2)}, \ldots, y_n^{(2)}) \Longrightarrow \ldots \Longrightarrow (x_1^{(s)}, \ldots, x_n^{(s)})$, *for some* $s \geq 1$ *such that* $w = x_1^{(s)}\}$

An example of PCGS with communication by command is offered in Appendix, p. 513. In [Dassow, Păun & Rozenberg, 1997, p. 192] can be found a theorem that summarizes results on generative capacity of PCGS with communication by command.

2.4 Eco-Grammar Systems

We have defined grammars systems as grammatical models of multi-agent architectures, stressing in this way the relationship between that theory and models coming from Artificial Intelligence. Now, if Grammar Systems are related to AI (as they effectively are), a subfield of this theory, the so-called Eco-Grammar Systems, is closely related to AL or, what is the same, Artificial

Life -a recent research area in computer science with the aim of creating and studying *'man-made system that exhibit behaviors characteristic of natural living system'* [Langton, 1989]. Therefore, Eco-Grammar System, introduced in [Csuhaj-Varjú *et al.*, 1994b], can be thought of as a formal language theoretical framework model of systems of Artificial Life. The model provides a syntactical framework for ecosystems, this is, for communities of evolving agents and their interrelated environment.

Briefly, an Eco-Grammar System can be defined as a multi-agent system where different components, apart from interacting among themselves, interact with a special component called 'environment.' So, within an Eco-Grammar System we can distinguish two types of components: *environment* and *agents*. Both are represented at any moment by a string that identifies current state of the component. These strings change according to sets of evolution rules (Lindenmayer systems). Interaction of agents and environment is carried out through agents actions performed on the environmental state by the application of some productions (rewriting rules) from the set of action rules of agents.

An Eco-Grammar System can be thought of as a generalization of both CD Grammar Systems and PC Grammar Systems. If we superpose a CD Grammar System and a PC Grammar System, we obtain a system consisting of grammars having both individual strings (like in PC Grammar Systems) and a common string (like in CD Grammar Systems). If we call this common string *environment* and we mix the functioning of CD and PC Grammar Systems, letting each component to work on its own string and also on environmental string, something similar to an ecosystem is obtained. If to all of this, we add one more grammar, expressing evolution rules of the environment, and we make evolution of agents depend on environmental state, the thing we obtain is nothing else that an **Eco-Grammar System.**

The concept of Eco-Grammar System is based on six postulates formu-

lated according to properties of Artificial Life (cfr. [Csuhaj-Varjú *et al.*, 1994b, Csuhaj-Varjú, 1995]):

1. An *ecosystem* consists of an *environment* and a set of *agents.* Both state of the environment and states of agents are described by strings of symbols of given alphabets.

2. In an ecosystem there is a *universal clock* which marks time units, the same for all the agents and for environment, according to which the evolution of agents and of the environment is considered.

3. Both environment and agents have characteristic *evolution rules*, which are in fact Lindenmayer systems, hence are applied in a parallel manner to all the symbols describing agents and environment; such a (rewriting) step is done in each time unit.

4. Evolution rules of environment are *independent* on agents and on the state of the environment itself. Evolution rules of agents *depend* on the state of the environment (at a given moment, a subset of applicable rules is chosen from a general set associated to each agent).

5. Agents act on the environment according to *action rules*, which are pure rewriting rules used sequentially. In each time unit, each agent uses one action rule which is chosen from a set depending on the current state of the agent.

6. *Action has priority over evolution* of the environment. In a given time unit exactly the symbols which are not affected by action (in the environment) are rewritten (in parallel manner) by evolution rules.

The main features of an Eco-Grammar System, pointed out above, are captured in figure 2.1.

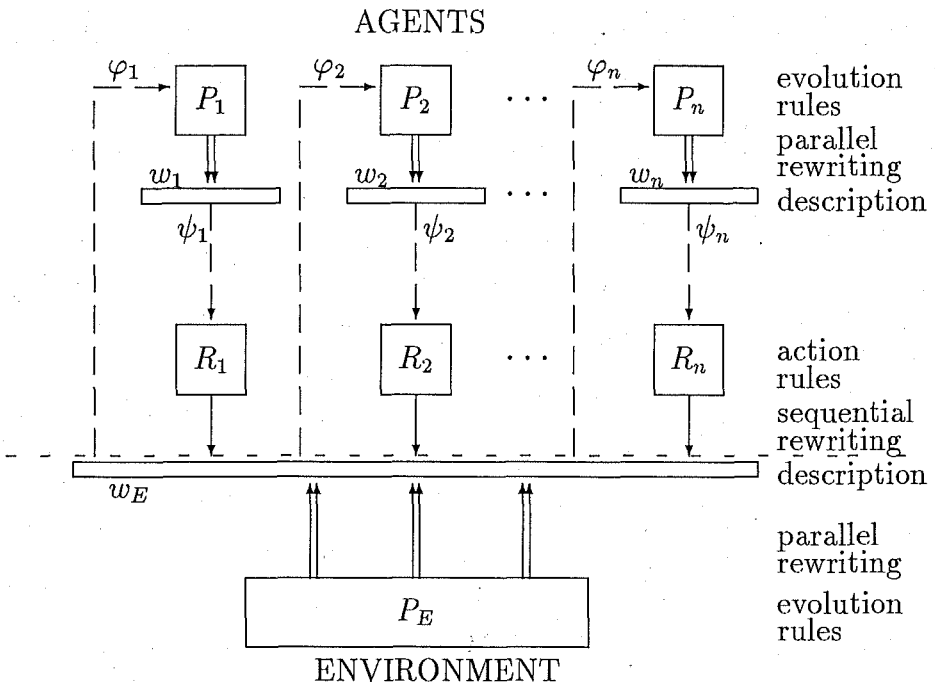

Figure 2.1: Eco-Grammar System

Let us see now what an Eco-Grammar System is and how it works in a formal way.

Definition 2.4.1 *An Eco-Grammar System (an EG-system, for short) is an* $n+1$*-tuple*

$$\Sigma = (E, A_1, \ldots, A_n),$$

where

- $E = (V_E, P_E)$,
  - V_E *is a finite alphabet,*

- *P_E is a finite set of 0L rewriting rules over V_E.*

- *$A_i = (V_i, P_i, R_i, \varphi_i, \psi_i)$ for i, $1 \leq i \leq n$, where*

  - *V_i is a finite alphabet,*
  - *P_i is a finite set of 0L rewriting rules over V_i,*
  - *R_i is a finite set of rewriting rules of the form $x \longrightarrow y$ with $x \in V_E^+$, $y \in V_E^*$,*
  - $\varphi_i : V_E^* \longrightarrow 2^{P_i}$,
  - $\psi_i : V_i^+ \longrightarrow 2^{R_i}$.

The above components are interpreted as follows:

- E corresponds to the environment with an alphabet V_E and a set of evolution rules P_E.

- $A_i = (V_i, P_i, R_i, \varphi_i, \psi_i)$ for i, $1 \leq i \leq n$, corresponds to an agent (the i-th agent), with an alphabet V_i, a set of evolution rules P_i, a set of action rules R_i. Mapping φ_i, depending on the state of the environment, selects current evolution rules for A_i, and mapping ψ_i, depending on the current state of A_i, selects rules for current action.

Definition 2.4.2 *A state of an Eco-Grammar System $\Sigma = (E, A_1, \ldots, A_n)$ is an $(n+1)$-tuple*

$$\sigma = (w_E, w_1, w_2, \ldots, w_n),$$

where $w_E \in V_E^*$ and $w_i \in V_i^*$, $1 \leq i \leq n$; w_E is the state of the environment, and w_i is the state of i-th agent, $1 \leq i \leq n$.

Functioning of Eco-Grammar System is done by changing its states, that is, by modifying those strings that identify agents and environment at that moment. Both actions and evolutions contribute to change current state of the system.

Definition 2.4.3 *Let $\sigma = (w_E, w_1, w_2, \ldots, w_n)$ be a state of EG-system $\Sigma = (E, A_1, \ldots, A_n)$. Agent A_i is said to be active in state σ if the set of its current action rules, this is $\psi_i(w_i)$, is nonempty.*

By an action of an active agent A_i in state σ we mean an application of an action rule r, $r \in \psi_i(w_i)$, to the environmental state w_E.

A simultaneous action of agents $A_{i_1}, \ldots, A_{i_r}$, $\{i_1, \ldots, i_r\} \in \{1, \ldots, n\}$ being active in state σ, onto the environment is a parallel derivation step

$$w_E \Longrightarrow w'_E$$

such that

$$w_E = x_1 u_1 x_2 u_2 \ldots u_r x_{r+1}$$

and

$$w'_E = x_1 v_1 x_2 v_2 \ldots v_r x_{r+1},$$

where $u_j \to v_j \in \psi_{i_j}(w_{i_j})$, $1 \leq j \leq r$, and $x_i \in V_E^*$, $1 \leq i \leq r+1$.

Definition 2.4.4 *Let $\sigma = (w_E, w_1, w_2, \ldots, w_n)$ be a state of EG-system $\Sigma = (E, A_1, \ldots, A_n)$. We say that w'_i is an actual evolution of agent A_i in state w_i, if w'_i can be derived from w_i by productions of $\varphi_i(w_E)$, in 0L-manner, $1 \leq i \leq n$.*

For two states w_E and w'_E of the environment we say that w'_E is an evolution of w_E if w'_E can be derived from w_E by productions of P_E in 0L-manner.

A change of a state of an Eco-Grammar System means an evolution of the state of every agent and an evolution of the environment at each place except some distinguished ones where currently active agents perform simultanously an action.

Definition 2.4.5 *Let $\sigma = (w_E, w_1, w_2, \ldots, w_n)$ and $\sigma' = (w'_E, w'_1, w'_2, \ldots, w'_n)$ be two states of EG-system $\Sigma = (E, A_1, \ldots, A_n)$. We say that σ changes into σ', written as*

$$\sigma \Longrightarrow_\Sigma \sigma',$$

iff the following conditions hold:

(i) w'_E arises from w_E by an evolution affected by all the active agents in state σ, that is

$w_E = z_1x_1z_2x_2\ldots z_mx_mz_{m+1}$ *and* $w'_E = z'_1y_1z'_2y_2\ldots z'_my_mz'_{m+1}$ *such that*

- $z_1x_1z_2x_2\ldots x_mz_{m+1} \Longrightarrow z_1y_1z_2y_2\ldots y_mz_{m+1}$ *is a simultaneous action of all the agents $A_{i_1}, \ldots, A_{i_m}$, $\{i_1, \ldots, i_m\} \subseteq \{1, \ldots, n\}$, that are active in state σ and*
- $z'_1z'_2\ldots z'_{m+1}$ *is an evolution of $z_1z_2\ldots z_{m+1}$.*

(ii) w'_i an is an evolution of A_i in state w_i, $1 \leq i \leq n$.

In words, the next state of the agent is determined only by its evolution rules, while the next state of the environment depends both on its own evolution rules and on interactions with currently active agents.

Definition 2.4.6 *For a given Eco-Grammar System Σ and an initial state σ_0 we define the set of state sequences of Σ by*

$$Seq(\Sigma, \sigma_0) = \{\{\sigma_i\}_{i=0}^{\infty} \mid \sigma_0 \Longrightarrow_\Sigma \sigma_1 \Longrightarrow_\Sigma \sigma_2 \Longrightarrow_\Sigma \ldots\}.$$

The set of environmental state sequences is

$$Seq_E(\Sigma, \sigma_0) = \{\{w_{Ei}\}_{i=0}^{\infty} \mid \{\sigma_j\}_{j=0}^{\infty} \in Seq(\Sigma, \sigma_0), \sigma_j = (w_{Ej}, w_{1j}, \ldots, w_{nj})\}.$$

The set of states of the j-th agent A_j is defined by

$$Seq_j(\Sigma, \sigma_0) = \{\{w_{jk}\}_{k=0}^{\infty} \mid \{\sigma_k\}_{k=0}^{\infty} \in Seq(\Sigma, \sigma_0), \sigma_k = (w_{Ek}, w_{1k}, \ldots, w_{jk}, \ldots, w_{nk})\}.$$

The language of the environment is

$$L_E(\Sigma, \sigma_0) = \{w_E \in V_E^* \mid \{\sigma_j\}_{j=0}^{\infty} \in Seq(\Sigma, \sigma_0), \sigma_j = (w_E, w_1, \ldots, w_n)\}.$$

and the language of the i-th agent is

$$L_i(\Sigma, \sigma_0) = \{w_i \in V_i^* \mid \{\sigma_j\}_{j=0}^{\infty} \in Seq(\Sigma, \sigma_0), \sigma_j = (w_E, w_1, \ldots, w_i, \ldots, w_n)\}.$$

for $i = 1, 2, \ldots, n$.

$Seq(\Sigma, \sigma_0)$ corresponds to the behavior of the system, that is to the set of possible sequences of states, following directly each other, starting from the initial state. $Seq_E(\Sigma, \sigma_0)$ and $Seq_i(\Sigma, \sigma_0)$ are the corresponding sets of sequences of the states of the environment and that states of the i-th agent. $L_E(\Sigma, \sigma_0)$ and $L_i(\Sigma, \sigma_0)$ correspond to those states of the environment and those states of i-th agent that are reachable from the initial state of the system.

An example of Eco-Grammar System can be found in Appendix, p. 515. For results about the generative capacity of Eco-Grammar Systems, we refer to [Csuhaj-Varjú *et al.*, 1996, Dassow & Mihalache, 1995], and [Dassow, Păun & Rozenberg, 1997, p. 199]. More information about Eco-Grammar Systems can be found in [Păun, 1995a, Csuhaj-Varjú, 1995] [Csuhaj-Varjú, 1996a, Dassow, 1995a, Kelemenová, 1999, Mihalache, 1995] [Csima, 1998, Csima, 1999].

2.5 Final Remarks

The presentation of Grammar System Theory we have made in this chapter does not intend to be an exhaustive one. The great development the theory has undergone in its decade or so of existence prevented us from completeness. It is not possible to collect in an introduction, without being excessive, all the variants of Grammar Systems that have arised since that 1988 when the theory was first introduced. We refer to [Csuhaj-Varjú *et al.*, 1994a] for the state of the theory till the middle of 1992. For newer results and a extensive bibliography on the field [Dassow, Păun & Rozenberg, 1997] can be seen.

Just to conclude, we can name some of the variants that have not been referred to in the body of the chapter and that are vividly in the ambit of Grammar Systems Theory. In the area of CD Grammar System we can see that the model has been extended to computational devices different from the generative mechanisms that were presented in the original definition: accepting grammars ([Fernau & Holzer, 1996]) or automata ([Dassow & Mitrana, 1999]) are examples of this extension. Another variant of Grammar Systems, intermediate between CD and PC, are *Stratified Grammar Systems* introduced in [Csuhaj-Varjú *et al.*, 1993].

Colonies are another special variant of Cooperating Distributed Grammar Systems. Colonies -proposed in [Kelemen & Kelemenová, 1992]- can

be thought of as grammatical models of multi-agent systems motivated by Brooks' subsumption architectures[5] [Brooks, 1991]. They describe language classes in terms of behavior of collections of very simple, purely reactive, situated agents with emergent behavior. Formally, a colony is set of regular grammars which generate finite languages and operate -without any explicitly predefined strategy of cooperation- on a common sentential form: the symbolic environment. The environment is quite passive, its state changes only as result of acts agents perform on it. Because of the lack of any predefined strategy of cooperation, each component participates in the rewriting of current strings whenever it can participate in it. The behavior of a colony -this is, the language- is defined as the set of all the strings which can be generated by the colony from a given starting string. For formal definitions, results and new variants on this area we refer to [Kelemen, 1991, Kelemen, 1992a, Kelemen, 1993a, Kelemen, 1998, Kelemenová & Csuhaj-Varjú, 1993] [Kelemenová & Csuhaj-Varjú, 1994, Dassow, Kelemen & Păun, 1992] [Martín-Vide & Păun, 1998b, Martín-Vide & Păun, 1999, Baník, 1996] [Martín-Vide & Păun, 1998a, Kelemenová & Csuhaj-Varjú, 1992].

Another highly frequent notion in the area of Grammar Systems Theory in the last few years is **Networks of Language Processors** (NLP, for short). This notion has been introduced in [Csuhaj-Varjú & Salomaa, 1997, Csuhaj-Varjú, 1996c] with the aim of presenting a general paradigm for capturing the common properties of distributed architectures. Thus, *network of language processors* is a collective, global term for describing several parallel, distributed, communicating architectures in Formal Language Theory. Models that fall under this notion are typically composed by a finite number of grammars that derive their own sentential form in parallel and whose work

[5]Subsumption architectures were suggested for computation whose purpose is to program intelligent, situated, embodied agents as collections of competitive behaviors without any central locus of control of the behavior or central representation of the agents' inner or outer environment.

is organized in a communicating system in order to generate a language. As it can be seen such architectures are just the ones studied in the area of Grammar Systems, specifically in the subfield of PC Grammar Systems.

Roughly speaking, an NLP is a set of language processors[6] located in the nodes of a network (a virtual graph). Each node rewrites its own sentential form and informs the other of its activity via communication of strings. Rewriting and communication take place alternatively. The functioning of the network consists, therefore, in the change of its configuration or its states by means of rewriting and communication steps. Differences in the way of communication and in the language associated with the network give rise to a wide variety of NLP. For further information on Networks of Language Processors see [Csuhaj-Varjú, 1996c, Csuhaj-Varjú, 1997, Csuhaj-Varjú, 1998], [Csuhaj-Varjú & Salomaa, 1997].

If we turn now to the area of PC Grammar Systems we can see also the appearance of a great number of variants. Different types of communication protocols: what, how, and when to communicate; different type components of the system (context-sensitive grammars, L systems...); and other modifications in the basic model we have presented above give rise to a wide variety of PC Grammar Systems. It is beyond our scope to present all these new variants in this thesis. We will limit ourselves to introduce, whenever it will be necessary, the variants that are used in the present work. To know about the new types of PC Grammar Systems, the reader can consult the references given in [Dassow, Păun & Rozenberg, 1997] as well as articles included in the bibliography at the end of this dissertation as [Mihalache, 1998c, Păun, 1996a,

[6] A language processor is a symbol processing device, it can be considered as a grammar in a generalized sense. It consists of an alphabet of symbols, a finite language that codes operations on strings built up from symbols of the alphabet (the set of rules) and a relation that describes how to produce, by applying elements from the rule set, new strings from some old one. Generative grammars, contextual grammars, splicing schemes can be examples of language processors

Freund *et al.*, 1995, Mihalache, 1994b, Mihalache, 1996],
[Vaszil, 1998, Vaszil, 1999, Mitrana, Păun & Rozenberg, 1994].

Chapter 3

Advantages of Using Grammar Systems

We have already stated our intention of applying Grammar Systems Theory to the description of some linguistic and cultural matters. Since the very beginning, we have stressed the idea that Grammar Systems Theory presents many advantages over classical formal models. And these advantages are, precisely, the reason that lead us to suggest the possible usefulness of this new grammatical tool in accounting for several issues in natural languages and culture.

Even though in this dissertation we will concentrate only in linguistic and cultural matters, this does not mean that Grammar Systems Theory could be useful solely in those fields. The many advantages this theory presents make of it a potential good tool to be applied to several other fields.

In this chapter we intend to review some of the features that characterize and define Grammar Systems and that, at the same time, have revealed themselves as crucial notions in several fields.

Notions as *modularity*, *distribution*, *cooperation*, *parallelism*, *interaction* or *emergent behavior* are very present in fields as Computer Science, Artificial

Intelligence or Cognitive Science. Those same concepts seems to have great importance when we try to describe the functioning of natural languages. All these notions are in the very base of Grammar Systems Theory and constitute a real advantage when we try to apply the theory to different matters.

Together with the above features, another good advantage Grammar Systems offer when applied to natural languages is the possibility of obtaining non-context-free structures while using context-free grammars. This advantage is evident if we take into account all the problems that the generation of non-context-free constructions has caused in Linguistics.

In what follows we will analyze one by one all the above-mentioned notions. In every section we will stress the importance that each of those concepts has in different fields, in order to show how vivid they are and to enhance, thereby, the advantage that the presence of those features in the theory of Grammar Systems entails.

3.1 Modularity

One of the main advantages -if not the most important one- of Grammar Systems Theory is its idea of **modularity**. *Modularity* has shown to be a very important idea in a large range of fields. Cognitive Science, Natural Language Processing, Computer Science and, of course, Linguistics are, among others, examples of fields where modular models have been proposed. To get an approximate idea of the current significance of modularity we will briefly review some of those fields that advocate for modular approaches.

3.1.1 Modularity in Cognitive Science

According to [Stillings, 1987], it is a commonplace belief in Cognitive Science that complex computational systems are at least weakly decomposable into

components. In general, modular theories in Cognitive Science propose a number of independent but interacting cognitive 'modules' that are responsible for each cognitive domain. Specific arrangement of those modules usually varies in each theory, but generally each mental module encapsulates a definable higher mental function. There may be, for example, separate structures for spatial reasoning, mathematical ability, musical talent, phonological skill, oral lexicon, written lexicon, nonverbal thought, and verbal thought to name a few.

Even though the idea of modularity has been implicit in Cognitive Science for a long time, it is with the publication of *The Modularity of Mind* [Fodor, 1983] when those implicit ideas that had been current over the previous two decades crystallized into a single recognizable hypothesis: the mind is not a seamless, unitary whole whose functions merge continously into one another; rather, it comprises a number of distinct, specialized, structurally idiosyncratic modules that communicate with other cognitive structures in only very limited ways.

According to [Fodor, 1983], we can establish a functional taxonomy of the psychological processes which recognizes three different types of systems:

1. *Sensory transducers* that convert physic energy that incide over sensorial receptors in mental representations;

2. *Input systems* or specific domain processing systems, whose work is to obtain a representation from the stimuli and information suministred by the transducers;

3. and *Central systems* or processes of general domain whose work consist of unifying information from different modules and to perform intelligent tasks as belief formation, inferences realization, decision making, problem solving, reasoning, etc.

From the previous taxonomy follows that information from the external environment passes first through a system of sensory transducers, which transform the data into formats that each special-purpose input system can process. Each input system, in turn, outputs data in a common format suitable for central, domain-general processing.

Crucial in the above taxonomy is that *input systems* are characterized as *modular*. The modularity of input systems is defined by several properties that are present in those systems and that differentiate them from the central systems considered as non-modular. Properties to which Fodor refers and that characterize modular systems are the following:

- *Domain specificity;*
- *Mandatoriness;*
- *Lack of access to intermediate representations by other systems;*
- *Autonomy;*
- *Innately specification;*
- *Shallow output;*
- *Information encapsulation;*
- *Fixed neural architecture;*
- *Speed;*
- *Susceptibility to characteristic breakdowns.*

It is worth to stress that it is the *co-occurrence* of all the properties pointed out above what defines a module or an input system. Alone, particular properties do not necessarily entail modularity.

Central systems, in contrast are:

- *No domain specific;*
- *More subject to voluntary control;*
- *Unencapsulated;*
- *Typically neurally scattered;*
- *Operate with a semantically much richer set of representations;*
- *Slow.*

The above ideas briefly summarize the thesis of modularity defended by Fodor. But Fodor's theory is not by far the only one about modularity of mind. In fact, in the 1980s there starts a new trend represented by authors as Chomsky, Marr, Garfield, Jackendoff and, of course, Fodor himself, who defend the non-homogeneous character of mind.

Chomsky, for example, is very clear in recognizing the modularity of mind when he states that:

> *'Every complex biological system we know is highly modular in its internal structure. It should not be a terrible surprise to discover that the human mind is just like other complex biological systems: that it is composed of interacting subsystems with their specific properties and character and with specific modes of interaction among the various parts.'* [Chomsky, 1984, p. 16].

The same clear idea is reflected in his question

> *'Is the mind organized into distinct cognitive faculties with their specific structures and principles, or are there uniform principles of learning, accomodation, assimilation, abstraction, induction, strategy, or whatever that simply apply to different stimulus*

> *materials to provide our knowledge of the behavior of objects in physical space, our knowledge that certain strings of words do or do not have certain meanings, and so on?'* [Chomsky, 1984, p. 47].

and the adamant answer to it:

> *'The available evidence seems to me to favor a modular approach.'* [Chomsky, 1984, p. 47].

Marr is another representant of what [Garfield, 1987] has called the *scientific paradigm* of modularity in contemporary Cognitive Science. Marr defends that idea in his development of a theoretical framework for exploring the nature of modularity in the visual system (cfr. [Marr, 1982, Arbib, 1987]). Marr's theory of vision contains two distinct claims about modularity:

1. the first claim is that low-level or early visual processes constitute a module in Fodor's strong sense.

2. the second claim is that there are *submodules* within the low-level vision system. This second claim about modularity is that low-level vision is a two-stage process and that each of the stages can be seen as a module that is further broken down into submodules.

A somehow different approach to modularity is *Representational Modularity* proposed by Jackendoff (cfr. [Jackendoff, 1997]). The overall idea of *representational modularity* is that mind/brain encodes information in some finite number of distinct representational formats or 'languages of the mind.' Each of these 'languages' is a formal system with its own set of primitives and principles of combination, so that it defines an infinite set of expressions. For each of these formats, there is a module of mind/brain responsible for

it. For example, phonological structure and syntactic structure are distinct representational formats, with distinct primitives and principles of combination. *Representational modularity* therefore establishes that architecture of mind/brain needs separate modules for those two encodings. Each of these modules is domain specific and informationally encapsulated in Fodor's sense.

Representational modules differ from Fodorian ones in that they are individuated by the representations they process rather than by their function as faculties for input or output. This is, they are individual levels of representation, rather than being an entire faculty.

In addition to *representation modules*, Jackendoff considers a system of *interface modules*. An interface module communicates between two levels of encoding, say L_1 and L_2, by carrying out a partial translation of information in L_1 into information in L_2 form, or, what is the same, by imposing a partial homomorphism between L_1 and L_2 information. An interface module is domain specific[1] and informationally encapsulated[2].

The main part of researchers we have referred to make reference, as a way of showing the modularity of mind, to language faculty. This is the case of Fodor, Chomsky or Jackendoff, for example. These authors speak of language faculty as one of the modules of mind, so they face the modularity of language faculty from one of two possible points of view: the *external modularity*. They refer to grammar as an independent cognitive module, as a system that operate only on a specific domain of information and that has its own primitives and principles. But it is also possible to tackle modularity

[1]The phonology-to-syntax interface module, for example, knows only about phonology and syntax, not about visual perception or general purpose audition.

[2]The phonology-to-syntax module takes whatever phonological inputs are available in the phonology representation module, maps the appropriate parts of them into syntactic structures, and delivers them to the syntax representation module, with no help or interface from, say, beliefs about the social context.

of grammar from another perspective: *internal modularity*[3]. If we consider that language system is internally modular we are defending that grammar itself is made up by independent and interacting components. This last view is the one adopted in Linguistics and Natural Language Processing.

3.1.2 Modularity in Linguistics

> *'What we loosely call 'knowledge of language' involves in the first place knowledge of grammar and beyond that other cognitive systems that interact with grammar: conceptual systems with their specific properties and organizing principles may be quite different in character from the 'computational' language faculty; pragmatic competence might be a cognitive system distinct and differently structured from grammatical compentence; these systems may furthermore be composed of distinct though interacting components.'* [Chomsky, 1980, p. 90].

It seems to be a general agreement that linguistic system has a modular nature. The complexity of linguistic behavior seems to be the consequence of the interaction of a number of independent systems (e.g. phonology, morphology, syntax, semantics, pragmatics...) each of which is characterized by its own primitives and operations.

Modular approach to grammar has been shown to have important consequences for the study of language. This has led many grammatical theories to use modular models. The idea of having a system made up of several independent components (syntax, semantics, phonology, morphology, etc.) -each

[3]It is worth to remind that the opposite of modularity is *homogeneity*, and that a system can be either externally or internally homogeneous. Notice that these are all independent notions. A system could be internally homogeneous, but externally modular, or internally modular but externally homogeneous.

one governed by its own rules and having its own primitives, and interacting among them- seems to be a good choice to account for linguistic matters. We could cite several modular approaches in Linguistics, from Chomsky's Generative Grammar to Autolexical Syntax [Sadock, 1985] or Jackendoff's view of the Architecture of Language Faculty [Jackendoff, 1997], just to name a few.

One of the main advantages of modular grammar -as [Frazier, 1988] points out- is that they can reduce delays imposed by monolithic or nonmodular grammars. This reduction of delays is due to the fact that, in modular grammars, subsystems of grammatical principles can be applied independently of each other and, sometimes, in a parallel fashion.

One of the most important characteristics of Generative Grammar has been to concive linguistic knowledge as a differentiated module of mind. This is, as an autonomous capacity independent of the rest of cognitive systems. But, Chomsky's modularity concept does not stop there, he applies the concept of modularity to grammar itself in order to internally distinguish different components of language system. We can say -following [Sciullo, 1997]- that in Chomsky modularity of grammar resides in the coexistence of autonomous components (phonological, syntactic, morphological, semantic) as well as in the interaction of given sets of principles (Theta theory, case theory, binding theory, etc.) at given levels of representations (D-structure, S-structure, Logical form, etc.). However, we do not find modularity only in Principles and Parameters framework [Chomsky, 1981]. In the Minimalist program [Chomsky, 1995], it is provided a general framework for the formalization of a fully modular theory of grammar, as well. In that framework, grammar includes a small set of interacting modules, components and interface levels.

A clear modular conception of grammar is defended also by Jackendoff, who adopts the following assumption:

'I will assume an overall organization of the mental informa-

> *tion structure involved in a language as follows. This organization includes three autonomous levels of structure: phonological, syntactic and conceptual. Each of these has its own characteristic primitives and principles of combination and its own organization into subcomponents, such as segmental phonology, intonation contour, and metrical grid in phonology, and D-structure, S-structure, Phonetic Form and Logical Form (or counterparts in other theories) in syntax. Each of the levels is described by a set of formation rules that generates the well-formed structures of the level.'* [Jackendoff, 1990, p. 17].

The same modular idea is defended by the author some years later in his *The Architecture of Language Faculty* [Jackendoff, 1997].

In [Harnish & Farmer, 1984], authors propose also a modular theory in which language system is analyzed into distinct but interacting subsystems of principles which apply independently and under very general constraints. Harnish and Farmer defend the idea that linguistic system is modular both in the internal and external sense.

Also [Farmer, 1984] defines a theory based on the hypothesis that language system is modular in nature. Starting from this startpoint, a theory of language is built as a system of rules and representations factorable into independent but interacting subsystems.

And finally, we could not end this brief review of modular theories in Linguistics without making reference to Sadock's *Autolexical Grammar* [Sadock, 1991]. Autolexical Grammar is clearly a modular theory since it recognizes the existence of various components. But the modularity found in Sadock's theory is somehow different from what is found in Generative Linguistics, for example. Here, modules are individual grammars of various dimensions of representation (syntax, morphology, semantics). According to

Sadock, the existence of those independent levels -each of which characterized by an autonomous grammar- contrasts with the modularity of Government and Binding, for example, where modules are systems of principles, constraints or rules that cross-cut the several levels of representation. In other words, while Sadock's modules are informationally encapsulated in the terminology of Fodor, modules of GB grammar are not.

According to all the above-mentioned, it seems that the idea of modularity is highly present in Linguistics. But the defence of a modular nature of grammar goes beyond theoretical Linguistics, being also very important in the field of Natural Language Processing.

3.1.3 Modularity in Natural Language Processing

> *'There are different ways to organize a generation system to gain more flexibility. The ideal organization lets each module contribute the choices to do with its area concern, without having to make other decisions that would be better made by some other module.'* [Allen, 1987, p. 504].

The above quotation clearly defens as ideal organization in *Natural Language Generation* the modular one. But modularity is not only adequate in *Natural Language Generation*, but also in *Natural Language Understanding*. We can find modular theories of language comprehension which consist of several distinct processing subsystems, each with its own properties and its own characteristic information sources (cfr. [Frazier, 1988]). [Sabah, 1997a] sees modularity as a must in Natural Language Understanding systems. And [Allen, 1987] refers to the modularity of *language understanding* differetiating three different phases of processing:

1. *parsing phase* when a sentence is processed into a structural description using syntactic and morphological knowledge;

2. *semantic interpretation phase* when structural description is mapped, using semantic knowledge, into a logical form which represents literal meaning of the sentence;

3. and *contextual interpretation phase* when this representation is mapped into a final representation of the effects of understanding the sentence.

In [Clifton & Ferreira, 1987] it is defended the idea that there may be specialized subcomponents in language-understanding systems, identified as Fodorian modules. Even though each module should operate autonomously, some limited intercommunication among modules ought to exist in order to get a plausible system. Modularity and interaction is also defended in [Altman, 1987].

So, in general we can say that Natural Language Processing takes advantage from the idea of modularity. To consider a modular structure where several language modules have limited information and operate in an independent fashion will facilitate very much the task of processing natural language, as has been already stressed by several researchers working in the field. So, again, modularity reveals itself as a very important notion.

In this section we have stressed the important role of modularity in several different fields like Cognitive Science, Linguistics or Natural Language Processing. Also in Computer Science the idea of modular decomposition seems to be a pervasive one, according to [Stein, 1997]. But of course, there are many researchers in the above fields who reject modularity. In this way, for example, we can see how *connectionist* models of mind have been proposed as alternative of modular theories. *Postmodular* systems are offered for Cognitive Robotics in [Stein, 1997]. In the area of psycholinguistics, Marslen-Wilson & Komisarjevsky declare themselves *against modularity* arguing that

> *'The thesis is seductive, entertaining, perhaps even heuristically useful. But as a basis for the construction of explanatory the-*

> *ories of human psycholinguistic performance it is fundamentally misleading.'* [Marslen-Wilson & Komisarjevsky, 1987, p. 61].

Karmiloff-Smith goes *beyond modularity*, and in [Karmiloff-Smith, 1992] defends the thesis of *modularization* over the prespecified modules defended by Fodor.

However, in spite of opinions against, modularity is a crucial concept in many research areas, nowadays. But if there is an area where the concept has revealed itself as very important this is for sure the study of natural languages. As we have suggested above, we can think of grammar as made up of different components (modules) working independently (with its own rules and primitives) and cooperating among them to reach a common aim: the generation of language. That concept of modularity is very close to the one present in Grammar Systems Theory. We have already said, that a Grammar System is a set of grammars -this is, different modules- working together in order to generate a language (cfr. chapter 2). So, it is not difficult to get an idea of a Grammar System where each of its components is a module representing an autonomous system (syntax, semantics, phonology...) of our grammar. Those different modules would work independently, but interacting among them whenever they feel it is necessary.

Summing up, *modularity* has revealed itself as a very useful trait in Linguistics, Computer Science, Cognitive Science and other related fields. This feature has helped to make easier tasks as natural language processing, for example. In general, it seems that modular systems present clear advantages over non-modular ones. Grammar Systems Theory shares these advantages, since it can be presented as a *modular* theory where knowledge of the system is distributed among different *modules* (grammars) that cooperate among them to generate the language of the system. So, Grammar Systems present the advantages of *modular* models. And it is precisely the modular nature of this grammatical tool one of the reasons that suggest the possible application

of that new formal device to a large range of problems and, specifically, to the study of natural languages since, as we have seen, this field is one of the great beneficiaries of the modular approach.

3.2 Distribution & Cooperation

Distribution and **cooperation** are two important notions in Computer Science, Artificial Intelligence and Cognitive Psychology. In those fields it is usual to deal with complex tasks *distributed* among a set of 'processors' that *work together* in a well defined way. Computer nets, computers with parallel architecture, distributed data bases and knowledge systems can be seen as practical aspects of distribution and cooperation. Even, it is quite probable that human brain works in a distributed fashion if we take into account the large number of competences the brain uses simultaneously at every moment.

But problems of distribution and cooperation are not unique to computer systems, of course. They exist at multiple levels of activity in a wide range of situations. People pursue their own goals through communication and cooperation with other people. Animals interact, cooperate with each other, and form communities. Particles interact with each other and compose different types of material. Summing up, we can see a prevalence of cooperation in human societies in particular and in the world in general.

A good sign of the relevance of *distribution* and *cooperation* in Artificial Intelligence, Computer Science and other related fields is the rise of research lines as the so-called *Cooperative Distributed Problem-Solving* (cfr. [Durfee, Lesser & Corkill, 1989]). This distributed/cooperative type of problem-solving studies how a loosely linked network of problem solvers can work together to solve problems that are beyond their individual capabilities. Each problem-solver in the network is capable of sophisticated problem solving and

can work independently, but problems faced by each of them cannot be completed without cooperation. Cooperation is necessary because no problem-solver has sufficient expertise, resources and information to solve a problem. Among the reasons that can lead to *cooperation*, [Durfee, Lesser & Corkill, 1990] point out the following ones:

- to improve performance by working in parallel;
- to increase the variety of solutions by allowing agents to form local solutions without being overly influenced by other agents;
- to increase the confidence of a (sub)solution by having agents rederive (verify) each other's results, possibly using different problem-solving expertise and data;
- to increase the probability that a solution will be found, despite agents failures, by assigning important tasks to multiple agents;
- to reduce the amount of unnecessary duplication of effort by letting agents to recognize and avoid useless redundant activities;
- to improve the overall problem solving by permitting agents to exchange predictive information;
- to reduce communication by being more selective about what messages are exchanged;
- to improve the use of computing resources by allowing agent to exchange tasks to better balance the computational load;
- to improve the use of individual agent expertise by allowing agent to exchange tasks so that a task is performed by the most capable agents;
- to minimize the time agents must wait for results from each other by coordinating activity.

Similar reasons, and the suitability shown by cooperative distributed architectures, have led to apply distributed models to the field of Natural Language Processing. In text analysis, for example, we can find models as the one presented in [Stefanini & Warren, 1996]. *TALISMAN* is defined as a distributed architecture for text analysis in French, that includes linguistic agents that correspond either to classical levels in linguistics (morphology, syntax, semantics) or to complex language phenomena. The system is based on direct communication between agents and uses mailboxes for sending messages with an asynchronous mode of communication. Possible interactions between agents have to be regulated and this is done by means of interaction protocols. The main goal of [Stefanini & Warren, 1996] is to show that complex linguistic phenomena like coordination, ellipsis, or negation, can be defined and processed in a distributed architecture.

CARAMEL model proposed in [Grau, Sabah & Vilnat, 1994] is also an example of distributed architecture in Natural Language Understanding. The model proposed by those authors intends to be applied to several different topics such as text understanding, dialogue management, making abstracts or intelligent tutoring systems. Having this wide scope, a very modular and flexible architecture is necessary. And cooperative distributed systems offer such flexibility.

And finally, in the field of Cognitive Science, an example of distribution and cooperation is Minsky's society of mind. Minsky has developed a hypothesis based on viewing human mind as an organized society of intercommunicating agents, the so-called 'society of mind.' Mental activities appear from interactions between agents organized in some local hierarchies. He tries to explain how minds work. In Minsky words, the *society of mind* is

> *'That scheme in which each mind is made of many smaller processes. These we'll call agents. Each mental agent by itself can only do some simple thing that needs no mind or thought at*

> *all. Yet when we join these agents in societies -in certain very special ways- this leads to true intelligence.'* [Minsky, 1985, p. 17].

If cooperation and distribution have revealed as very important topics in all the above fields, they are not less significative in Grammar Systems Theory. As it could be not otherwise, *distribution* and *cooperation* are very present in Grammar Systems. The importance of cooperation in the theory can be summarized by simply defining Grammar Systems as *'the theory of cooperation protocols.'*

In chapter 2, we have defined a Grammar System as a *set* of grammars working together in order to generate a language. So if a Grammar System is a *set* of grammars is because the knowledge of the system is *distributed* in different sources (grammars, in this case). And that grammars work together to generate a language means nothing else than they *cooperate* among them to reach a common goal. Therefore, distribution and cooperation are in the very base of Grammar Systems and constitute definitely another good reason to consider the application of this new theory to several fields, specially to the study of natural language.

3.3 Parallelism & Interaction

If we want to realize about how important is **parallelism** in our world, we just need to look at the *'most successful computing device'* [Smith, 1991], namely the *human mind.* The parallel processing carried out by human mind is evident if we consider that -even though most of our conscious experience is serial- we can perform a set of operations consciously (generating sentences, for example) while we perform, in parallel, another set of tasks without conscious attention (driving a car or typing). But we do not need to look at two

operations performed at the same time -like speaking and driving- to realize about the parallelism of the human mind. Many cognitive processes exhibit degrees of parallelism. Our eyes, ears and other senses simultaneously process vast amounts of information in parallel, for example.

In *Computer Science* many procedures - in particular those that try to emulate human cognitive processes- call for parallel processing, either by requiring concurrently executable subtasks or relying on collective decision making. This imperative led to the appareance of *Distributed Parallel Computing.* Computer networks, distributed data bases, highly parallel computers, parallel logic programming languages [Nijholt, 1994], etc. present a new philosophy of computing, very different from the traditional Turing-von Neumann sequential one.

Parallel Distributed Processing systems, known as *PDP*, have been defined by Rumelhart and McClelland [Rumelhart & McClelland, 1986]. PDP are *parallel* in that many very simple processors operate collaboratively, and they are *distributed* in that information possessed and manipulated by the system is spread over a network of interconnections. Parallel computational models have been applied in several domains. Language (visual and auditory work recognition, production, lexical access, and parsing), vision, motor control and knowledge representation, are some examples. Much of the interest the paradigm has aroused is due to its apparent success in providing new computational solutions to hard problems, and to its apparently biological plausibility, since its processing units are supposed to work like neurons.

If we have a look, now, at *Natural Language Processing* we can see that *parallelism* and *interaction* are also present there. While initially human sentence processing was explained by means of *serial models*, now researchers have turned to highly *parallel* ones. Serial models used to adopt a 'syntax-first' approach, where syntactic processing of the sentence had to be over before semantic processing began, which in turn preceded pragmatic process-

ing. In such a models, information of lower-levels could not be used to correct decisions at higher levels with the consequent explosion of syntactic possibilities. That situation led to the preference of *parallel and interactive* models, where a system were capable of using any type of knowledge at any moment it needed it, without being constrained by a serial and hierarchical structure. Marslen and Wilson reported about some experiments that gave psychological evidence that processing at each level of natural language description could constrain and guide simultaneous processing at other levels, defending in this way the parallelism in language processing:

> *'... sentence perception is most plausibly modelled as a fully interactive parallel process: that each word, as it is heard in the context of normal discourse, is immediately entered into the processing system at all levels of description, and is simultaneously analysed at all these levels in the light of whatever information is available at each level at that point in the processing of the sentence.'* (cited by [Nijholt, 1993, p. 5]).

According to [Nijholt, 1993], we can distinguish different types of *interactive parallel sentence processing models*:

- **Heterarchical systems** where processing tasks are assigned to different processors or processes and in which every knowledge source interacts with every other.
- **Blackboard models** where multiple knowledge sources can process in parallel and co-operatively by means of a globally accessible 'blackboard' on which they can write and read intermediate results and through which they communicate and interact.
- Models which consider **words as active agents** that interact with each other and possibly with other knowledge sources in order to obtain a meaning representation of a sentence.

- **Connectionist** or **neural network** approach where, in a huge network of extremely simple processing elements, language processing is 'coded' into spreading of activation and converging of activation towards a pattern that somehow represents the meaning of a sentence.

So, from the above follows that we can understand interaction and parallelism at different fine-grained levels. But if there is something clear is that human sentence processing seems to be a parallel process. It seems to be hardly plausible to think that we process a sentence step by step in a serial fashion. Therefore any model that intends to be considered as a model of human sentence processing should possess two adjectives: *parallel* and *interactive*.

But parallelism is present not only in Computer Science and Natural Language Processing, it has revealed itself important in Linguistics, as well.

Traditionally, in *Linguistics* we have found a hierarchical view of grammar where the different organizational dimensions of language are seen as 'levels' obtainable from one another in a certain fixed order. This is to say, representations of a high-level component are passed to a lower-level component that modifies them and passes the resulting representation on to a lower component. Functioning of grammar is, therefore, hierarchical and serial and it presents lack of autonomy of different components or modules, since each component needs the string of a higher one in order to start its own work. The history of Generative Grammar, for instance, shows a large number of attempts proposing alternative hierarchizations of grammar. The result is a great diversity of opinions with regard which component must pass information to which one, which is generative and which interpretative, etc. Each of the models proposed has its merits and its drawbacks, of course. But it is precisely this situation the one that leads to think that maybe the point is not to look for a hierarchization or another, but to search for a system with parallel, autonomous components that work independently to account