UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

12.7 Analysis of mixed links depending on the syntactic result of the rules

12.7.1 Objectivization

12.7.1.1 Rules that cause ghosts

Rule 1

$$x = sv\underline{\wedge}o, \qquad y = s'v'o', \quad \Diamond(x,y) = sv(s'v'o')_o$$

Since the group $(s'v'o')$ is taking the focal place of o in the variable O, then $l(s'v'o') = l(o)$, upon which:

$$x = sv\underline{\wedge}o, \qquad y = s'v'o', \quad \Diamond(x,y) = sv(s'v'o')_{lo}$$

In order to join both structures, every time that $s' \in Z$, it is necessary a level assigner ghost. Then the rule is formulated again in the following way:

$$x = sv\underline{\wedge}o, \qquad y = s'v'o', \quad \Diamond(x,y) = sv\wr(s'v'o')_{lo}$$

Let's see what happens when a ULPS is inserted into the truncated pole $\begin{pmatrix} \\ S \end{pmatrix}$ of another one when $l(v) = q$. The suitable rule for this situation is:

$$x = sv\underline{\wedge}_q, \qquad y = s'v'o', \quad \Diamond(x,y) = sv\wr(s'v'o')_q$$

The result is the one verified in the example 100:

Example 100

x = *El gat vol # llet* $\qquad$ y = *Joan juga tennis*

x = `The cat wants # some milk` $\qquad$ y = `John plays tennis`

$\Diamond(x,y)$ = *El gat vol que Joan jugui a tennis*

$\Diamond(x,y)$ = `The cat wants that John plays tennis`

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

But if $l(v) \neq q$, then thanks to the action of the ghost, the link is also possible. For instance, for $l(v) = r$:

$$x = sv\underline{\wr}_{o_r}, \quad y = s'v'o', \quad \Diamond(x,y) = sv \wr (s'v'o')_r$$

Example 101

x = *El gat viu # a Barcelona* y = *En Joan juga a tennis*

x = `The cat lives # in Barcelona` y = `John plays tennis`

$\Diamond(x,y)$ = *El gat viu on Joan juga a tennis*

$\Diamond(x,y)$ = `The cat lives where John plays tennis`

The change of ghost adjusts the ULPS level to the v level in a way so that the operation is correct.

Rule 2

$$x = sv\underline{\wr}_o, \quad y = s'v'\#o', \quad \Diamond(x,y) = sv(s'v')_o$$

Since the group $(s'v')$ is taking the focal place of o in the variable O, then $l(s'v') = l(o)$:

$$sv\underline{\wr}_o, \quad y = s'v'\#o', \quad \Diamond(x,y) = sv(s'v')_{lo}$$

In order to join both structures, every time that $s' \in Z$, it is necessary a level assigner ghost:

$$x = sv\underline{\wr}_o, \quad y = s'v'\#o', \quad \Diamond(x,y) = sv \wr (s'v')_{lo}$$

Example 102 *For* $l(v) = q$

x = *El gat vol # llet* y = *Joan juga # tennis*

x = `The cat wants # some milk` y = `John plays # tennis`

$\Diamond(x,y)$ = *El gat vol que Joan jugui*

$\Diamond(x,y)$ = `The cat wants that John plays`

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

Example 103 *For* $l(v) = r$

x = *El gat viu # a Barcelona* y = *En Joan juga #a tennis*

x = The cat lives # in Barcelona y = John plays # tennis

$\Diamond(x, y)$ = *El gat viu on Joan juga*

$\Diamond(x, y)$ = The cat lives where John plays

However, if $\odot \notin \ell(v')$, then, as every time that a $v\#o$ cut exists, the resultant structure is not correct as we see in 104:

Example 104 *Per a* $\odot \notin^{-} \ell(v')$

x = *El gat vol # llet* y = *En Joan té # un gos*

x = The cat wants # some milk y = John has # a dog

$\Diamond(x, y)$ = **El gat vol que en Joan tingui*

$\Diamond(x, y)$ = *The cat wants that John has

12.7.1.2 Rules that cause infinitive structures

Rule 3

$$x = sv\llcorner o, \quad y = s'\#v'o', \quad \Diamond(x, y) = sv(v'o')_{l_O}$$

According to what we have said in 12.5.4, the level of a whole or fragmented ULPS is q. If a ghost does not arise, it is not possible to change the level assignment of the ULPS. Therefore, and since there cannot exist ghosts because $s' \notin Z_1$, then it is only possible:

$$x = sv\llcorner o_q, \quad y = s'\#v'o', \quad \Diamond(x, y) = sv(v'o')_q$$

Example 105

x = *El gat vol # llet* y = *Joan# juga tennis*

x = The cat wants # some milk y = John # plays tennis

$\Diamond(x, y)$ = *El gat vol jugar a tennis*

$\Diamond(x, y)$ = The cat wants to play tennis

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

However, if $q \notin \ell(v)$, then a disarrangement of syntactic level is produced among the elements of each one of the stage structures in the final result:

$$x = sv\underline{\alpha}_r, \qquad y = s'\#v'o', \quad \Diamond(x,y) = {}^*sv(v'o')_r$$

Example 106

x = *El gat viu # a Orta* $\qquad$ y = *Joan# juga tennis*

x = `The cat lives # in Orta` $\quad$ y = `John # plays tennis`

$\Diamond(x,y)$ = **El gat viu jugar a tennis*

$\Diamond(x,y)$ = `The cat lives (to play tennis)`$_r$

The outcome of this example is not correct in English because "`to play tennis`" cannot play the role r in a link of level.

Rule 4

$$x = sv\underline{\alpha}, \qquad y = s'\#v'\#o', \quad \Diamond(x,y) = sv(v')_o$$

In the same way as in 3, it must be rewritten in only one way:

$$x = sv\underline{\alpha}_q, \qquad y = s'\#v'\#o', \quad \Diamond(x,y) = sv(v')_q$$

Example 107

x = *El gat vol # llet* $\qquad$ y = *En Joan# juga # tennis*

x = `The cat wants # some milk` $\quad$ y = `John # plays # tennis`

$\Diamond(x,y)$ = *El gat vol jugar*

$\Diamond(x,y)$ = `The cat wants to play`

Considering that the link is impossible if $q \notin^{-} \ell(v)$:

Example 108

x = *El gat viu # a Orta* $\qquad$ y = *En Joan# juga # tennis*

x = `The cat lives # in Orta` $\quad$ y = `John # plays # tennis`

$\Diamond(x,y)$ = *El gat viu jugar*

$\Diamond(x,y)$ = `The cat lives to play`

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

12.7.2 Subjectivization

12.7.2.1 Rules that cause the appearance of ghosts

Rule 5

$$x = \underline{s}vo, \qquad y = s'v'o', \quad \Diamond(x,y) = (s'v'o')_s vo$$

The insertion of a whole ULPS in the place of focus s causes a readjustment in $PN(v)$ according to what we have explained in 12.5.4. Consequently, the rule is reformulated:

$$x = \underline{s}vo, \qquad y = s'v'o', \quad \Diamond(x,y) = (s'v'o')_s v_{3s} o$$

And because of s' is at Z_1, then the ghost $f\wr$ always arises in front of the inserted structure:

$$x = \underline{s}vo, \qquad y = s'v'o', \quad \Diamond(x,y) = \wr(s'v'o')_s v_{3s} o$$

Example 109

x = *L'arbre # és bonic* y = *Joan juga tennis*

x = `The tree # is nice` y = `John plays tennis`

$\Diamond(x,y)$ = *Que Joan jugui a tennis és bonic*

$\Diamond(x,y)$ = `That John plays tennis is nice`

Rule 6

$$x = \underline{s}vo, \qquad y = s'v'\#o', \quad \Diamond(x,y) = (s'v')_s vo$$

According to $PN(ULPS)$, the rule is reformulated:

$$x = \underline{s}vo, \qquad y = s'v'\#o', \quad \Diamond(x,y) = (s'v')_s v_{3s} o$$

With the appearance of $f\wr$:

$$x = \underline{s}vo, \qquad y = s'v'\#o', \quad \Diamond(x,y) = \wr(s'v')_s v_{3s} o$$

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

Example 110

x = *L'arbre # és bonic* y = *Joan juga # tennis*

x = `The tree # is nice` y = `John plays # tennis`

$\Diamond(x, y)$ = *Que Joan jugui és bonic*

$\Diamond(x, y)$ = `That John plays is nice`

This kind of focal insertion is a universal process. It always works from a syntactic point of view. However, it has some semantic problems that we have not considered to solve.

12.7.2.2 Rules that cause infinitive structures

Rule 7

$$x = \_s vo, \quad y = s'\#v'o', \quad \Diamond(x, y) = (v'o')_s vo$$

According to $PN(v) = 3s$, then

$$x = \_s vo, \quad y = s'\#v'o', \quad \Diamond(x, y) = (v'o')_s v_{3s} o$$

Without any ghost because s' is not found in the resultant string.

Example 111

x = *L'arbre # és bonic* y = *En Joan # juga tennis*

x = `The tree # is nice` y = `John # plays tennis`

$\Diamond(x, y)$ = *Jugar a tennis és bonic*

$\Diamond(x, y)$ = `Playing tennis is nice`

In rules 7 and 8, Catalan and English follow different ways for the neutralization of $PN(v')$. English finishes $PN(v') = \sharp$ by means of the gerund, which is one of the forms in $\mathcal{V}$ that correspond to this condition, even though an infinitive result would also be accepted. However, Catalan always uses infinitive.

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

Rule 8

$$x = \_\underline{s}vo, \qquad y = s'\#v'\#o', \quad \Diamond(x,y) = (v')_s vo$$

With the following results after adjusting $PN(v) = PN(ULPS)$

$$x = \_\underline{s}vo, \qquad y = s'\#v'\#o', \quad \Diamond(x,y) = (v')_s v_{3s} o$$

Example 112

x = *L'arbre # és bonic* y = *En Joan # juga # tennis*

x = `The tree # is nice` y = `John # plays # tennis`

$\Diamond(x,y)$ = *Jugar és bonic*

$\Diamond(x,y)$ = `Playing is nice`

12.8 Generative power of mixed systems

12.8.1 Recursive rules

12.8.1.1 Rules recursive on the right: multiple object insertion

Rules 1 and 3, which insert ULPS with the group vo as an object of X, have the same generative result. If we take a terminal string obtained by means of rule 1 $-sv\wr(s'v'o')_{l_O}-$ as an X structure and we apply the rule again, the result is:

$$x = sv\wr(s'v'\_\underline{o'}), \qquad y = s''v''o'', \quad \Diamond(x,y) = sv\wr(s'v'\wr(s''v''o'')_{l_O})_{l_O}$$

Example 113

x = *El gat viu on en Joan juga # a tennis* y = *El conductor té tres flors*

x = `The cat lives where John plays # tennis` y = `The driver has three flowers`

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

$\Diamond(x, y)$ = *El gat viu on Joan juga a que el conductor té tres flors*
$\Diamond(x, y)$ = `The cat lives where John plays the driver has three flowers`

Rules 1 and 3 have the same results if in 3 all the ULPS accomplish that $q = l(v)$. Otherwise, the rule 3 becomes blockaded.

Technically, recursivity is infinite, whatever the level is to which the inserted ULPS has to be adapted, in a way that a structure is generated like the one in figure 12.2 for $n = 5$.

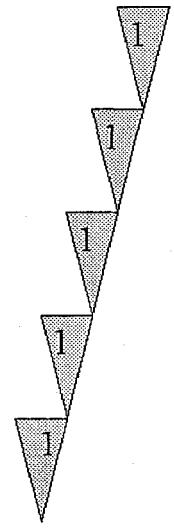


Figure 12.2: Greatest expansion of objectivizations with o at Z_1

The system increases in a stratum every time that a rule is applied, and therefore, a ULPS is added to the original one. This is to say, being n the number of strata, the number of sentences is equal to n.

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

12.8.1.2 Recursive rules on the left: multiple subject insertion

Rules 5 and 6 insert ULPS with the group *sv* in the position *s*, without the possibility of syntactic blockade.

Therefore, the generative power of these rules is the one we see in figure 12.3 for $n = 5$ where, being n the number of strata, the number of sentences is equal to n.

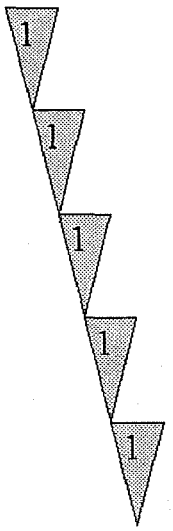


Figure 12.3: Greatest expansion of subjectivization with *s* at Z_1

12.8.2 Rules with blockade

12.8.2.1 Blockade on the right: blockade of object

Rules 2 and 4 are not recursive because they insert ULPS without *o* as an object of X. This absence of object in the resultant string prevents them from being applied again. These are rules that erase $\begin{pmatrix} o \\ O \end{pmatrix}$.

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

Therefore, their greatest generative power is minimal.

Figure 12.4: Greatest expansion of objectivizations without o at Z_1

Where the number of reaching levels is 2 and the number of forming sentences is also 2.

12.8.2.2 Blockade on the left: blockade of subject

Rules 7 and 8 delete $\begin{pmatrix} o \\ O \end{pmatrix}$. Therefore, now it cannot be applied any other cut of s nor the following insertion in its focal place.

As it happened with the erasers of o, the generative power of these rules is almost null and their application possibilities are reduced to an only one rule.

Figure 12.5: Greatest expansion of objectivizations without s at Z_1

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

12.9 Conclusions

12.9.1 Linguistic referent and generative power

Mixed systems create sentences that we have called *completive*. We refer to all those sentences that are inserted one into the other as an object or subject. However, substantivated relative structures are not included because they deserve a separate chapter within mixed systems.

By means of this molecular syntactic system, we can generate all the completive sentences that can be created in a language, which are divided into two groups:

- *Objectivated*: Υ is inserted in the focal place of o, having to assume its level.
- *Subjectivated*: Υ is inserted in the focal place of s.

These are the only systems of molecular syntax growing in two dimensions. The greatest expansion of these systems for a stratum n is equal to $2^{(n-1)}$, as we see in 12.6 for a level 8.

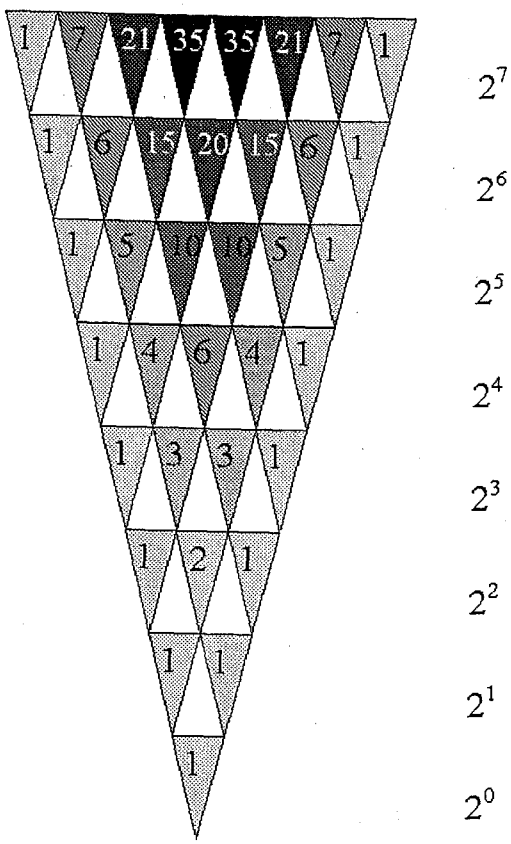


Figure 12.6: Greatest expansion of mixed systems.

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

12.9.2 Final reformulation of the rules

We use the terminology in strings, more synthetic:

Rule	x	y	$\Diamond(x,y)$	z	
1	$sv\llcorner o$	$s'v'o'$	$\Diamond(x,y)$	$sv\wr(s'v'o')_{lo}$	
2	$sv\llcorner o$	$s'v'\#o'$	$\Diamond(x,y$	$sv\wr(s'v')_{lo}$	
3	$sv\llcorner o$	$s'\#v'o'$	$\Diamond(x,y)$	$sv(v'_\sharp o')_q$	if $q \in \ell(v)$
				$^*sv(v'_\sharp o')$	if $q \notin \ell(v)$
4	$sv\llcorner o$	$s'\#v'\#o'$	$\Diamond(x,y)$	$sv(v'_\sharp)_q$	if $q \in \ell(v)$
				$^*sv(v'_\sharp o')$	if $q \notin \ell(v)$
5	$\lrcorner svo$	$s'v'o'$	$\Diamond(x,y)$	$\wr(s'v'o')_s v_3 so$	
6	$\lrcorner svo$	$s'v'\#$	$\Diamond(x,y)$	$\wr(s'v')_s v_{3s} o$	
7	$\lrcorner svo$	$s'\#v'o'$	$\Diamond(x,y)$	$(v'_\sharp o')_s v_{3s} o$	
8	$\lrcorner svo$	$s'\#v'\#o'$	$\Diamond(x,y)$	$(v'_\sharp)_s v_{3s} o$	

12.9.3 New structures introduced in this chapter

The main structure created in mixed links is the pole of three strata:

$$\begin{pmatrix} [s]v[o] \\ [S]\,V\,[O] \\ S \cup O \end{pmatrix}$$

For the first time, it has been necessary to make a reference to values PN and $^-\ell$ in a ULPS or a fragment of ULPS without any truncated pole where $\begin{pmatrix} v \\ V \end{pmatrix}$ exists.

$PN(ULPS) = 3s$:

$$\left[\begin{pmatrix} s & v & o \\ S & V & O \end{pmatrix}\right]_{3s}$$

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

$$\left[\left(\begin{array}{c} v \\ V \end{array}\right)_{\sharp}\right]_{3s}$$

$^{-}\ell(ULPS) = (q)$:

$$\left[\left(\begin{array}{ccc} s & v & o \\ S & V & O \end{array}\right)\right]_{q}$$

$$\left[\left(\begin{array}{c} v \\ V \end{array}\right)_{r}\right]_{q}$$

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

Chapter 13

Recombination systems

Analyzing the way recombination systems work with basic stage strings has allowed us to study the behaviour of foci when facing new situations, to anticipate the arising of ghosts and to consolidate some rules for the readjustment of pieces within the new resultant string.

However, the operations applied from basic strings do not correspond to the mental production processes of linguistic expressions. If syntax works by cuting and joining, it does not do it just once but repeatedly, one time and another until a speech is built that can be monologic or dialogic, it is to say, with one or diverse participants.

In order to achieve this iteration, it is necessary to construct systems. We will do it starting from the following premises:

- The goal of a linguistic system is not to generate a lot, but to produce all those possible structures in a language.
- We start from axioms made up from only one pattern which we have called basic, assuming that complex linguistic structures are obtained by means of the recombination of other simple ones.
- Systems work with strings due to notational simplicity. In every opera-

tion where whole ULPS must be taken into account, rules have already been adapted.

13.1 Formulations in computer sciences

In order to construct a system, we will base on *H systems* –or splicing systems– according to the description given by (Păun, Rozenberg & Salomaa, 1998, pp. 148 ss). These authors define an *H system* starting from the pair (σ, L), where $\sigma = (V, R)$ is a H scheme and L is a language.

Extended H systems are a quadruple

$$\gamma = (V, T, A, R),$$

where V is an alphabet, $T \subseteq V$, $A \subseteq V^*$, and $R \subseteq V^*\#V^*\$V^*\#V^*$, where $\#, \$$ are special symbols not in V.

The difference between terminal and non-terminal vocabulary is not considered in linguistic systems because we will opt for constructing them with foci (which we consider terminal symbols) or with linguistic strings; it is to say $T = V$. Systems with such feature are called *non-extended H systems* in formal languages and they must be defined as a triplet:

$$\gamma = (V, A, R)$$

where V, A, R have the same meaning as in the previous expression.

Formalization of replication systems carried out by (Păun, Rozenberg & Salomaa, 1998, p. 261) is very similar to splicing formalization. This is the useful one as a theoretical foundation for recombination systems, whichever operations are included in them.

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

13.2 Specific features of recombination systems for natural languages

We have just affirmed that the definition of splicing systems in computer science is the base on which the construction of recombination systems is maintained. Nevertheless, we have also seen that the differences between molecular operations applied to formal languages and natural languages are great, not only due to the kind of stage strings used, but it is also needed to carry out a great number of adjustments to the resultant ones as we have been testing throughout the thesis. Therefore, it is advisable to formulate formal systems taking into account the following aspects:

- If it is necessary, according to the features of each kind of operation, some specific rules must be built - so as to be useful not only to basic strings –as we have seen up to now– but to any context.
- In the resultant string of almost every operation, ghosts arise whether of one kind or another.
- We need to foresee the strings readjustment when combining foci of two different stages.

13.2.1 Rules for stage strings x not necessarily basic

There can be rules of all the operations introduced up to now in the recombination systems: splicing, replication, controlled splicing, sticker links and mixed operations. We name the set of all the rules of these systems *recombination rules* (R_r), whereas the specific of each one of the methods we have mentioned previously are denoted in the following way: $R_{\vdash}$, $R_{\triangleright}$, $R_{\bowtie}$, R_{μ}, $R_{\diamond}$.

It is advisable to carry out a very important assessment related to the behaviour of the recombination within the systems: being u a focus or group

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

of foci, u is equivalent to u^n. We have already seen that in σ, ω and ξ, structures, but it is extensive to any gathering with $f \smile$. It is to say, for the purpose of the application of $u = u^n$ rules when $u^n = u^n \cup f \smile^{(n-1)}$.

Considering their specific features, $R_{\bowtie}$, R_{μ}, $R_{\Diamond}$ do not need adjustments to act in systems where x is not probably basic.

However $R_{\vdash}$ as well as $R_{\rhd}$, universal and arbitrary, need a reformulation so as to be able to operate in any context.

Let's establish, then, the new rules under the following conditions:

- String x as a string z of the previous step.
- String y always basic.

13.2.1.1 Splicing rules

The basic rules are not enough for a system, because although the string y will always be composed according to the pattern SVO, the string x will be, in every step, the product z of the previous operation. Therefore, whereas all the cuts that can be carried out in y are limited to $\varnothing\#s$, $s\#v$, $v\#o$, $o\#\varnothing$, in x we have to take into account all the possible combinations of four elements taken two by two. In y we consider the possibility of cut $\varnothing\#s$ and $o\#\varnothing$ in an end, it is to say, to eliminate or to include the whole string. In x, because of foci combinations cannot be foreseen, so as to be able to delete or splice integrally the structure we have introduced the cuts $\varepsilon\#\varnothing$ and $\varnothing\#\varepsilon$. Apart from this context, the focus of cut ε has been avoided in order to prevent the discontrolling of the system.

Rules, arising from these criteria, called $R_{\vdash}$, are:

$$1\ \frac{z_1 s \mid v z_2}{z_1' \varnothing \mid s' z_2'} \qquad 2\ \frac{z_1 s \mid v z_2}{z_1' s' \mid v' z_2'} \qquad 3\ \frac{z_1 s \mid v z_2}{z_1' v' \mid o' z_2'} \qquad 4\ \frac{z_1 s \mid v z_2}{z_1' o' \mid \varnothing' z_2'}$$

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

5	z_1v	oz_2	6	z_1v	oz_2	7	z_1v	oz_2	8	z_1v	oz_2
	$z_1'\varnothing$	$s'z_2'$		$z_1's'$	$v'z_2'$		$z_1'v'$	$o'z_2'$		$z_1'o'$	$\varnothing'z_2'$
9	z_1s	sz_2	10	z_1s	sz_2	11	z_1s	sz_2	12	z_1s	sz_2
	$z_1'\varnothing$	$s'z_2'$		$z_1's'$	$v'z_2'$		$z_1'v'$	$o'z_2'$		$z_1'o'$	$\varnothing'z_2'$
13	z_1v	vz_2	14	z_1v	vz_2	15	z_1v	vz_2	16	z_1v	vz_2
	$z_1'\varnothing$	$s'z_2'$		$z_1's'$	$v'z_2'$		$z_1'v'$	$o'z_2'$		$z_1'o'$	$\varnothing'z_2'$
17	z_1o	oz_2	18	z_1o	oz_2	19	z_1o	oz_2	20	z_1o	oz_2
	$z_1'\varnothing$	$s'z_2'$		$z_1's'$	$v'z_2'$		$z_1'v'$	$o'z_2'$		$z_1'o'$	$\varnothing'z_2'$
21	z_1s	oz_2	22	z_1s	oz_2	23	z_1s	oz_2	24	z_1s	oz_2
	$z_1'\varnothing$	$s'z_2'$		$z_1's'$	$v'z_2'$		$z_1'v'$	$o'z_2'$		$z_1'o'$	$\varnothing'z_2'$
25	z_1v	sz_2	26	z_1v	sz_2	27	z_1v	sz_2	28	z_1v	sz_2
	$z_1'\varnothing$	$s'z_2'$		$z_1's'$	$v'z_2'$		$z_1'v'$	$o'z_2'$		$z_1'o'$	$\varnothing'z_2'$
29	z_1o	vz_2	30	z_1o	vz_2	31	z_1o	vz_2	32	z_1o	vz_2
	$z_1'\varnothing$	$s'z_2'$		$z_1's'$	$v'z_2'$		$z_1'v'$	$o'z_2'$		$z_1'o'$	$\varnothing'z_2'$
33	z_1o	sz_2	34	z_1o	sz_2	35	z_1o	sz_2	36	z_1o	sz_2
	$z_1'\varnothing$	$s'z_2'$		$z_1's'$	$v'z_2'$		$z_1'v'$	$o'z_2'$		$z_1'o'$	$\varnothing'z_2'$
37	$z_1\varnothing$	εz_2	38	$z_1\varnothing$	εz_2	39	$z_1\varnothing$	εz_2	40	$z_1\varnothing$	εz_2
	$z_1'\varnothing$	$s'z_2'$		$z_1's'$	$v'z_2'$		$z_1'v'$	$o'z_2'$		$z_1'o'$	$\varnothing'z_2'$
41	$z_1\varepsilon$	$\varnothing z_2$	42	$z_1\varepsilon$	$\varnothing z_2$	43	$z_1\varepsilon$	$\varnothing z_2$	44	$z_1\varepsilon$	$\varnothing z_2$
	$z_1'\varnothing$	$s'z_2'$		$z_1's'$	$v'z_2'$		$z_1'v'$	$o'z_2'$		$z_1'o'$	$\varnothing'z_2'$

13.2.1.2 Replication rules

Rules $R_{\triangleright}$ for systems are the following:

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

1) $x_1s \mid vx_2$ / $\#s'\#$

2) $x_1s \mid vx_2$ / $\#s'v'\#$

3) $x_1s \mid vx_2$ / $\#s'v'o'\#$

4) $x_1s \mid vx_2$ / $\#v'o'\#$

5) $x_1s \mid vx_2$ / $\#o'\#$

6) $x_1s \mid vx_2$ / $\#v'\#$

7) $x_1v \mid ox_2$ / $\#s'\#$

8) $x_1v \mid ox_2$ / $\#s'v'\#$

9) $x_1v \mid ox_2$ / $\#s'v'o'\#$

10) $x_1v \mid ox_2$ / $\#v'o'\#$

11) $x_1v \mid ox_2$ / $\#o'\#$

12) $x_1v \mid ox_2$ / $\#v'\#$

13) $x_1s \mid sx_2$ / $\#v'\#$

14) $x_1s \mid sx_2$ / $\#s'v'\#$

15) $x_1s \mid sx_2$ / $\#s'v'o'\#$

16) $x_1s \mid sx_2$ / $\#v'o'\#$

17) $x_1s \mid sx_2$ / $\#o'\#$

18) $x_1s \mid sx_2$ / $\#v'\#$

19) $x_1v \mid vx_2$ / $\#s'\#$

20) $x_1v \mid vx_2$ / $\#s'v'\#$

21) $x_1v \mid vx_2$ / $\#s'v'o'\#$

22) $x_1v \mid vx_2$ / $\#v'o'\#$

23) $x_1v \mid vx_2$ / $\#o'\#$

24) $x_1v \mid vx_2$ / $\#v'\#$

25) $x_1o \mid ox_2$ / $\#s'\#$

26) $x_1o \mid ox_2$ / $\#s'v'\#$

27) $x_1o \mid ox_2$ / $\#s'v'o'\#$

28) $x_1o \mid ox_2$ / $\#v'o'\#$

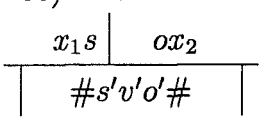
29) $x_1o \mid ox_2$ / $\#o'\#$

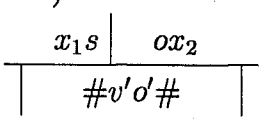
30) $x_1o \mid ox_2$ / $\#v'\#$

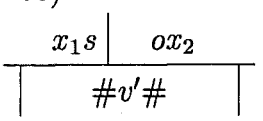
31) $x_1s \mid ox_2$ / $\#s'\#$

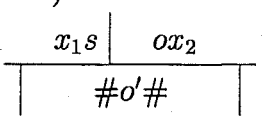
32) $x_1s \mid ox_2$ / $\#v'\#$

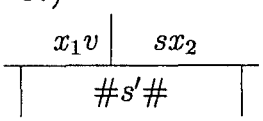
UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

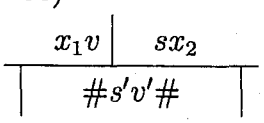
33) 

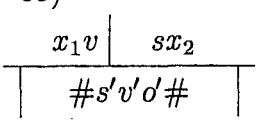
34) 

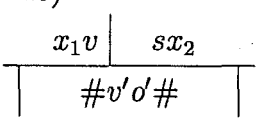
35) 

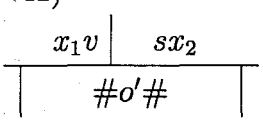
36) 

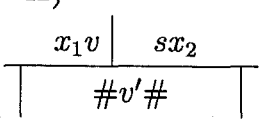
37) 

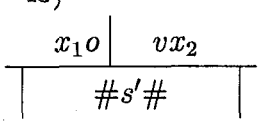
38) 

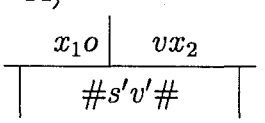
39) 

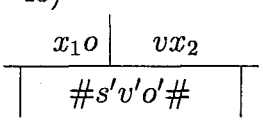
40) 

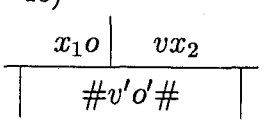
41) 

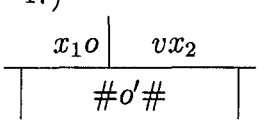
42) 

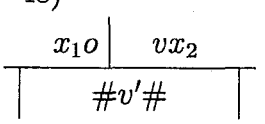
43) 

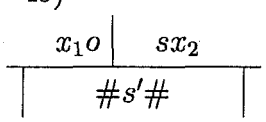
44) 

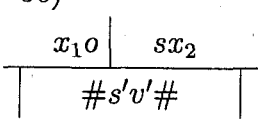
45) 

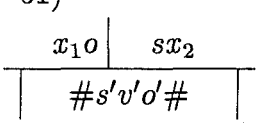
46) 

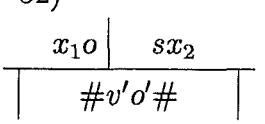
47) 

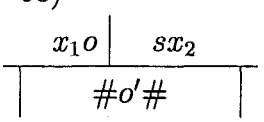
48) 

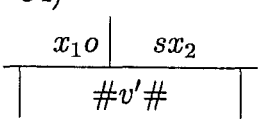
49) 

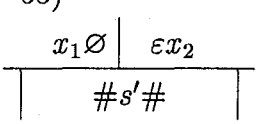
50) 

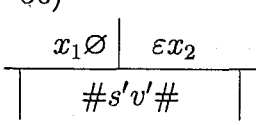
51) 

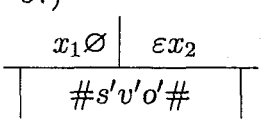
52) 

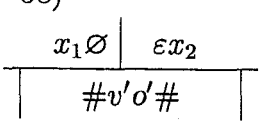
53) 

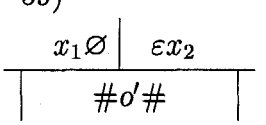
54) 

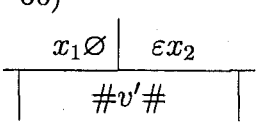
55) 

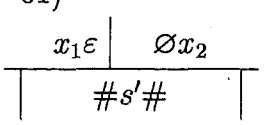
56) 

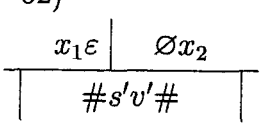
57) 

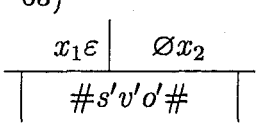
58) 

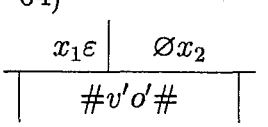
59) 

60) 

61) 

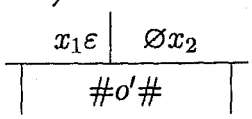
62) 

63) 

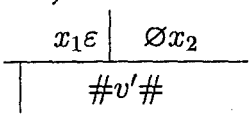
64) 

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

65)



66)



All these rules have been formulated for convenience when operating within the systems. Other rules that have not been foreseen in this graphic can be invented. The suitability of the creation of new rules will be manifested when a kind of systems where y is not necessarily basic is constructed.

13.2.2 Ghosts

We name F to the set of kinds of ghosts which are capable of emerging in a system. In recombination systems the maximum extension of $F = f \cdot \cup f\star \cup f \sim \cup f\wr$. Depending on the kind of rules included in a system, some specific kinds of ghosts will be required. Actually, there is enough with $F = \{f \cdot \cup f\star\}$, for splicing, replication and controlled splicing; whereas sticker systems need $F = f \sim$ and mixed systems need $F = f\wr$.

13.2.2.1 Appearance rules

The appearance rules of ghosts R_f, are already known and they are formulated, for instance: $(s)(s) \rightarrow s \cdot s$.

The embracing of a specific ghost in a system can be restricted by limiting the contexts where it can appear or, on the contrary, permitting the appearance wherever it is possible by non including simply any specific rule.

13.2.2.2 Rules of ghosts readjustment

We have already affirmed that groups of foci $(u_1...u_n)^m$ $n, m > 1$ behave just like these same groups for $m > 1$. It is to say, connections do not break the friction groups.

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

On the other hand, rules of ghosts readjustment have already been formulated in a systematic way in 8.6.5.2. For connectors:
being (uu) any given group of friction, and g any ghost $\neq$ ",":

$$\begin{aligned} \forall \quad & (uu)^n = ((uu)^n, f.^{n-1}), \\ & \sigma = (s^n, f.^{n-1}), \\ & \omega = (o^n, f.^{n-1}), \\ & f.^1 \ldots f.^{n-2} = ","; \ f.^{n-1} = g \end{aligned}$$

And for bounder ghosts:

Being $u = sv[o]$, g any bounder ghost $\neq$ ",":

$$\begin{aligned} \forall \quad & (u)^n = ((u)^n, f\star^{n-1}), \\ & f\star^1 = g; \ f\star^2 \ldots f\star^{n-2} = ","; \ f \smile^{n-1} = \text{"and"} \end{aligned}$$

13.2.3 Rules of string readjustment

These rules do not appear explicitly in linguistic systems, because they are automatically activated when foci are in contact.

13.3 Types of recombination systems

Considering the great differences that exist between an *H System* and a recombination system, we will call them in a different way in order to avoid comparisons. We name recombination systems Θ *Systems.*

It is needed to distinguish different kinds of Θ *Systems* according to the rules used and according to the generative power:

1. According to the rules:

   (a) restricted systems,

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

(b) extended systems.

2. According to the generative power:

   (a) simple systems,

   (b) complex systems,

   (c) parallel systems:

      i. dialogic systems,

      ii. monologic systems.

13.3.1 Depending on the rules: restricted systems / extended systems

Systems with rules, so that:

- $\forall r \in (R_r \subset \Re)$, $r \in R_{\vdash}$, are called restricted systems in $\vdash$,
- $\forall r \in (R_r \subset \Re)$, $r \in R_{\rhd}$, are called restricted systems in $\rhd$,
- $\forall r \in (R_r \subset \Re)$, $r \in R_{\bowtie}$, are called restricted systems in $\bowtie$,
- $\forall r \in (R_r \subset \Re)$, $r \in R_{\mu}$, are called restricted systems in μ,
- $\forall r \in (R_r \subset \Re)$, $r \in R_{\Diamond}$, are called restricted systems in $\Diamond$.

Systems in which non of those conditions are accomplished are extended.

In other words, *restricted* Θ *systems* are those in which there is only one kind of recombination rules; *extended* Θ *systems* are those in which there are more than one kind of recombination rules.

It is very difficult to formulate a restricted system in $\bowtie$ and in μ, which would be very similar to the multioperations that we have already described in chapters 10 and 11.

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

However, it is possible to construct easily restricted systems in $\vdash$, in $\rhd$, and even in $\Diamond$. The first two produce exactly the same, and therefore, in this sense, it is equivalent a restricted system in $\vdash$, in $\rhd$ or one where there are rules of both operations. A restricted system in $\Diamond$ has some specific generative features that have already been described in chapter 12.

13.3.2 According to generative power

We have affirmed that there are three kinds of systems depending on their generative power: simple, complex and parallel systems.

We will classify those three groups according to two criteria:

- Depending on the number of resultant strings at the end of the derivation.
  - Systems that produce an only terminal string:
    - simple,
    - complex.
  - Systems that produce multiple terminal strings:
    - parallel.
- Depending on the possible gathering of results.
  - Non-pushdown systems:
    - simple.
  - Pushdown systems:
    - complex,
    - parallell.

Now, we will study each one of these kinds of systems taking into account their distinctive features.

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

13.4 Simple systems

Those are directly related to the *non-extended H systems*, although they differ in all the components that have already been mentioned: specific rules, ghosts and readjustment of strings.

A *Simple* Θ *system* is defined as 5-uple:

$$\theta = (V, A, S, \Re, F)$$

where $V = \left(\{s_1...s_n\} \in \mathcal{S}, \{v_1...v_n\} \in \mathcal{V}, \{o_1...o_n\} \in \mathcal{O}\right)$ is an alphabet, $A \subseteq V^*$ is a set of axioms constructed as an ordered triplet $(u \in \mathcal{S}, u \in \mathcal{V}, u \in \mathcal{O})$, $S \in A$ is the initial axiom, $\Re \subseteq R_r \cup R_f$ is a set of rules, F is a set of ghosts.

Let's see now some examples:

Example 114 *We want to generate* $z = svo(\cdot svo)^n$, $n \geq 2$

In order to obtain it, we will construct a *simple* Θ *system* restricted in $\vdash$.

$$\theta = (V, A, \Re, S, F = f\cdot)$$

where

$A =$ $(a_1...a_n \| a_n = s_n v_n o_n, \quad n > 1)$

$S =$ a_1

$R_{\vdash} =$ 41

$R_f =$ $(svo)(svo) \rightarrow svo \cdot svo$

From where it is obtained:

1. $s_1v_1o_1\$s_2v_2o_2 \vdash_{41} s_1v_1o_1 \cdot s_2v_2o_2$

2. $s_1v_1o_1s_2v_2o_2\$s_3v_3o_3 \vdash_{41} s_1v_1o_1 \cdot s_2v_2o_2 \cdot s_3v_3o_3$

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

It is to say $a_1 \vdash_{41^n} svo(\cdot svo)^n$

And so on up to infinite, so that this system can bring about the connection of all the simple sentences *svo* of a language.

Example 115 *We want to obtain* $sv(\cdot\ sv)^n o(\cdot o)^m; n, m \geq 1$.

In order to do it, a *simple* Θ *system* restricted in $\triangleright$ can be constructed in a way that:

$$\gamma = (V, A, \Re, S, F = f\cdot)$$

where

$A =$ $(a_1 ... a_n || a_n = s_n v_n o_n, \quad n > 1)$

$S =$ a_1

$R_{\triangleright} =$ $11, 8$

$R_f =$ $(sv)(sv) \rightarrow sv \cdot sv,$

$(o)(o) \rightarrow o \cdot o$

From where we obtain, among others, the following derivation:

1. $s_1 v_1 o_1 \$ s_2 v_2 o_2 \triangleright_8 s_1 v_1 \cdot s_2 v_2 o_1$

2. $s_1 v_1 \cdot s_2 v_2 o_1 \$ s_3 v_3 o_3 \triangleright_8 s_1 v_1 \cdot s_2 v_2 \cdot s_3 v_3 o_1$

3. $s_1 v_1 \cdot s_3 v_3 \cdot s_2 v_2 o_1 \$ s_4 v_4 o_4 \triangleright_8 s_1 v_1 \cdot s_2 v_2 \cdot s_3 v_3 \cdot s_4 v_4 o_1$

4. $s_1 v_1 \cdot s_4 v_4 \cdot s_3 v_3 \cdot s_2 v_2 o_1 \$ s_5 v_5 o_5 \triangleright_{11} s_1 v_1 \cdot s_2 v_2 \cdot s_3 v_3 \cdot s_4 v_4 o_5 \cdot o_1$

It is to say $a_1 \triangleright_{(8^n, 11^m)} sv(\cdot sv)^n o(\cdot o)^m$, for $n, m \geq 1$.

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

13.5 Complex systems

Simple systems are very difficult to control, and their generative power usually exceeds what is desirable. For instance, we want to reach $svo(\cdot vo)^n \star s(\cdot s)^m vo$ for $n, m \geq 1$. With a simple system, it can be obtained such as $s(\cdot s)^n vo(\cdot vo)^m \star s(\cdot s)^i vo(\cdot vo)^j \star \ldots$ for $n, m, i, j \geq 1$ with the same rules used to reach that string.

If the goal is to obtain a string where the same rule can be applied in two or more contexts or is recursive on the right, it is advisable to limit its extension by planning systems that produce parts so as to recombine them later. This is what complex systems try to do.

A *complex* Θ *system* is a 7-uple

$$\theta = (V, A, S, \Re, G, F, P)$$

where $V = \left(\{s_1 \ldots s_n\} \in \mathcal{S}, \{v_1 \ldots v_n\} \in \mathcal{V}, \{o_1 \ldots o_n\} \in \mathcal{O}\right)$ is an alphabet, $A \subseteq V^*$ is a set of axioms constructed as an ordered triplet $(u \in \mathcal{S}, u \in \mathcal{V}, u \in \mathcal{O})$, $S = (Sa, Sb \in A)$ is the set of initial axioms, $\Re \subseteq R_r \cup R_f$ is a set of rules, F is the set of ghosts, $G = G_a \$ G_b \subseteq \{V \cup F\}^*$ is an objective string, P is a pile, symbol \$ simply means to apply a recombination operation without referring explicitely to any of them.

Those systems generate, first G_a, later G_b and finally we apply to them the rule of recombination that the system establishes. It is to say, the system works as many times as fragments have been made in the objective string. Each whole derivation -with its own rules or with the same ones- is called a, b, c... Its result is kept in a pile. When all the semi-objectives have been obtained, they are recombined.

Example 116 *Generation of the string:* $svo(\cdot vo)^n \star s(\cdot s)^m vo$ *for* $n, m \geq 1$

To do that, we construct a *complex* Θ *system* restricted in $\vdash$:

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

$$\theta = (V, A, \Re, G = G_a\$G_b, F = f \cdot \cup f\star)$$

We are dealing with a restricted Θ system in $\vdash$ where:

$A =$ $(a_1...a_n || a_n = s_n v_n o_n,\ n > 1)$
$Sa =$ a_1
$Sb =$ a_1
$Ga =$ $svo(\cdot vo)^n, n \geq 1$
$Gb =$ $s(\cdot s)^m vo, m \geq 1$
$aR_{\vdash} =$ 42
$bR_{\vdash} =$ 1
$abR_{\vdash} =$ 41
$R_f =$ $(svo)(svo) \rightarrow svo \star svo$
$(vo)(vo) \rightarrow vo \cdot vo$
$(s)(s) \rightarrow s \cdot s$

θa:

1. $s_1v_1o_1\$s_2v_2o_2 \vdash_{42} s_1v_1o_1v_2o_2$
2. $s_1v_1o_1v_2o_2\$s_3v_3o_3 \vdash_{42} s_1v_1o_1v_2o_2v_3o_3$

θb:

1. $s_1v_1o_1\$s_2v_2o_2 \vdash_1 s_1s_2v_2o_2$
2. $s_2s_2v_1o_1\$s_3v_3o_3 \vdash_1 s_1s_2s_3v_3o_3$
3. $s_1s_2s_3\$s_4v_4o_4 \vdash_1 s_1s_2s_3s_4v_4o_4$

Due to the fact that the pile has stored both results, we can obtain $\theta a\$b$:

$s_1v_1o_1v_2o_2v_3o_3\$s_1s_2s_3s_4v_1o_1 \vdash_{41} s_1v_1o_1v_2o_2v_3o_3 \star s_1s_2s_3s_4v_4o_4$

we have achieved the string $s(vo)^3 \star s^4vo$, which belongs to the objective.

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

13.6 Parallel systems

13.6.1 Features

The defined features of parallel systems are the following:

- Two systems, simple or complex, collaborate to form joined structures.
- They have piles capables of storing multiple results.
- They link the strings of the piles, but they do not cause a linguistic recombination.

A *Parallel* Θ *system* is defined as a 4-uple

$$\Theta = (\theta_1, \theta_2, P, E)$$

where θ_1 i θ_2 are two systems, P is a pile, E is a set of stickers.

We may imagine parallel Θ systems where collaborators were not two but n systems θ. Thus, a parallel Θ system would be defined as an $n+2$-uple:

$$\Theta = (\theta_1, \theta_2 \ldots \theta_n, P, E)$$

13.6.2 Kinds of parallel systems

There are two kinds of parallel systems:

- *Dialogic* $\rightarrow$ development on two sides:
  - each system carries out alternately one operation whose result is stored and, furthermore, it acts as a stage string in so that the other system carries out another one,

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

- the results of θ_1 and θ_2 are stored in the pile in the order in which they have been obtained,
- the axioms gathered in the pile are joined by means of the symbol $\natural$ that, in fact, isolates every resultant string,
- the first axiom, that initiaties the whole derivation, is also stored. We call it *activator axiom.*

- *Monologic* $\rightarrow$ development on one side
  - the results of θ_2 act as a string y –which, obviously, is not basic by definition– for the system θ_1,
  - only the results of θ_1 are stored in the pile,
  - the axioms gathered in the pile are linked, something that is denoted by means of $\frown$,
  - the first and unique axiom of θ_1 is stored as if it were a resultant string. We call it *activator axiom.*

13.6.3 Dialogic parallel systems

A *Dialogic parallel* Θ *system* is defined as:

$$\Theta = (\theta_1, \theta_2, P, E\natural)$$

where θ_1, θ_2 are simple systems because they must only carry out one operation each time at their turn, but they can be restricted or irrestricted. P is the pile, whose position is taken by the axiom P_1 of θ_1, which acts as a base of the pile. $E\natural$ is the sticker symbol which is useful to relate strings z.

The functioning of *Dialogic parallel* Θ *systems* is shown in figure 13.1.

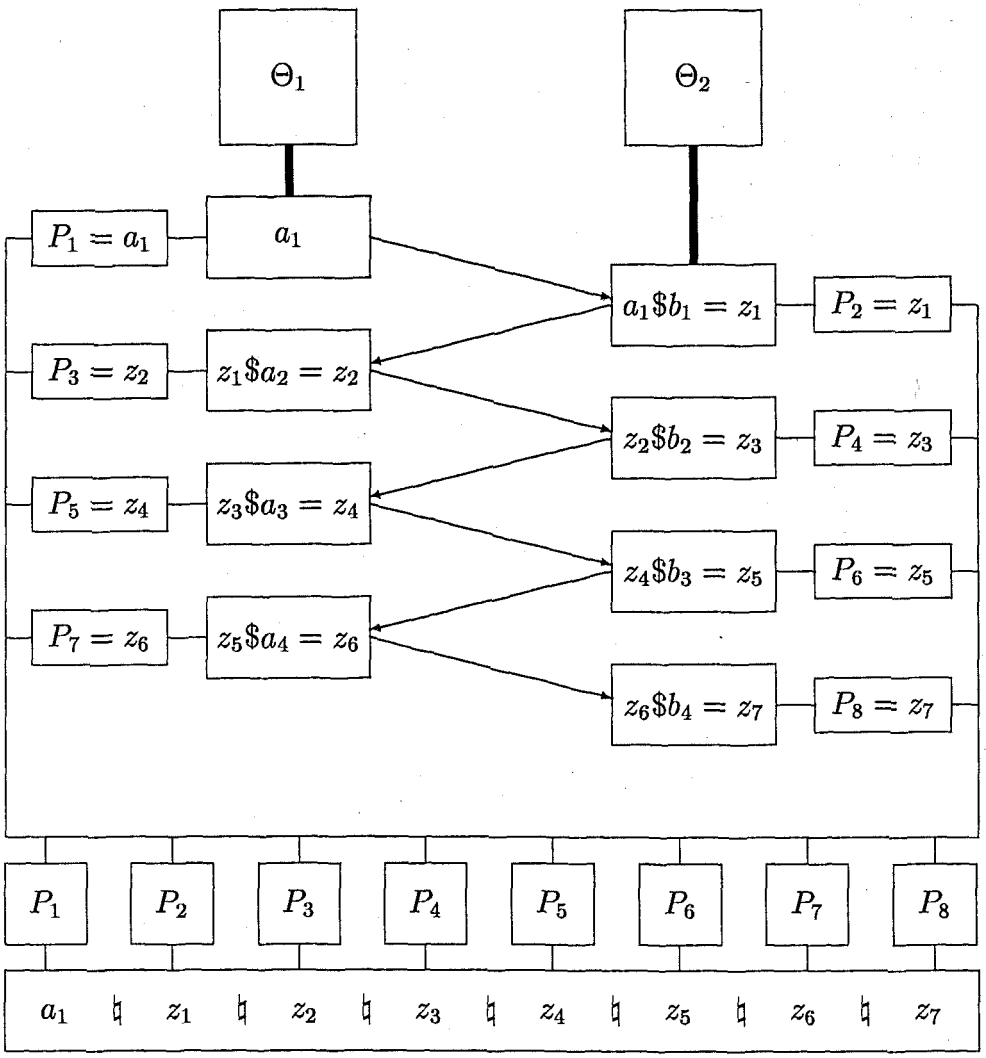


Figure 13.1: Dialogic parallel systems Θ

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

Example 117 *We see the mechanism of a dialogic parallel system:*

$$\Theta = (\theta_1, \theta_2, P, E\natural)$$

We define θ_1 as a restricted system in $\Diamond$:

$$\theta_1 = (V, A, S, \Re, F)$$

where $V = \big(\{s_1...s_n\} \in \mathcal{S}, \{v_1...v_n\} \in \mathcal{V}, \{o_1...o_n\} \in \mathcal{O}\big)$ is an alphabet, $A \subseteq V^*$ is a set of axioms constructed as an ordered triplet $(u \in \mathcal{S}, u \in \mathcal{V}, u \in \mathcal{O})$, $S \in A$ is the initial axiom, $\Re \subseteq R_\Diamond \cup R_f$, $F = f\wr$.

$$\begin{aligned} A = &\quad (a_1 \dots a_n = s_n v_n o_n,\ n \geq 1) \\ S = &\quad a_1 \\ R_\Diamond = &\quad 5, \end{aligned}$$

And θ_2 as an extended system:

$$\theta_2 = (V, B, S, \Re, F)$$

where $V = \big(\{s_1...s_n\} \in \mathcal{S}, \{v_1...v_n\} \in \mathcal{V}, \{o_1...o_n\} \in \mathcal{O}\big)$ is an alphabet, $B \subseteq V^*$ is a set of axioms constructed as an ordered triplet $(u' \in \mathcal{S}, u' \in \mathcal{V}, u' \in \mathcal{O})$, $S \in B$ is the initial axiom, $\Re = R_\vdash \cup R_\triangleright \cup R_f$,$F = f \cdot \cup f\star$.

$$\begin{aligned} B = &\quad (b_1 \dots b_n \| b_n = s'_n v'_n o'_n,\ n \geq 1) \\ S = &\quad b_1 \\ R_\vdash = &\quad 1 \\ R_\triangleright = &\quad 10 \end{aligned}$$

The system always starts by θ_1

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

ACT	θ_1	$s_1v_1o_1$
STEP 1	θ_2	$s_1v_1o_1 \vdash_1 s_1 \cdot s'_1v'_1o'_1$
STEP 2	θ_1	$s_1s'_1v'_1o'_1 \Diamond_5 \wr (s_2v_2o_2)s'_1v'_1o'_1$
STEP 3	θ_2	$\wr(s_2v_2o_2)s'_1v'_1o'_1 \rhd_{10} \wr(s_2v_2o_2)s'_1v'_1v'_2o'_2o'_1$
STEP 4	θ_1	$\wr(s_2v_2o_2)s'_1v'_1v'_2o'_2o'_1 \Diamond_5 \wr \big(\wr (s_3v_3o_3)v_2o_2\big)s'_1v'_1v'_2o'_2o'_1$

Meanwhile, the pile is filled with string z produced by each one of the operations.

P	Loading
P_1	$s_1v_1o_1$
P_2	$s_1v_1o_1 \natural s_1 \cdot s'_1v'_1o'_1$
P_3	$s_1v_1o_1 \natural s_1 \cdot s'_1v'_1o'_1 \natural \wr (s_2v_2o_2)s'_1v'_1o'_1$
P_4	$s_1v_1o_1 \natural s_1 \cdot s'_1v'_1o'_1 \natural \wr (s_2v_2o_2)s'_1v'_1o'_1 \natural \wr (s_2v_2o_2)s'_1v'_1v'_2o'_2o'_1$
P_5	$s_1v_1o_1 \natural s_1 \cdot s'_1v'_1o'_1 \natural \wr (s_2v_2o_2)s'_1v'_1o'_1 \natural \wr (s_2v_2o_2)s'_1v'_1v'_2o'_2o'_1 \natural$ $\wr\big(\wr (s_3v_3o_3)v_2o_2\big)s'_1v'_1v'_2o'_2o'_1$

The final result is the last one that stores the pile at the moment in which the system stops:

$s_1v_1o_1 \natural s_1 \cdot s'_1v'_1o'_1 \natural \wr(s_2v_2o_2)s'_1v'_1o'_1 \natural \wr(s_2v_2o_2)s'_1v'_1v'_2o'_2o'_1 \natural \wr\big(\wr(s_3v_3o_3)v_2o_2\big)s'_1v'_1v'_2o'_2o'_1$

13.6.4 Monologic parallel systems

A *Monologic parallel system* Θ is a 5-uple:

$$\Theta = (\theta_1, \theta_2, P, E \frown, TH)$$

where θ_1, θ_2 is defined as simple systems, restricted or irrestricted. P is the pile and $E \frown$ is the sticker symbol which is useful to link strings z of θ_1. TH is the relater between θ_2 and θ_1. In figure 13.2, for instance, $TH : w^{2n+1} \in$

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

$\theta_2 \rightarrow y_{(n+1)} \in \theta_1$, for $n \geq 0$, , taking into account that in monologic parallel systems the resultant strings of θ_2 are denoted w to avoid confusions.

Example 118 *We see a monologic parallel system* Θ *that can be adapted to figure 13.2:*

$$\Theta = (\theta_1, \theta_2, P, E \frown, TH)$$

where $TH = w^{2n+1} \in \theta_2 \rightarrow y_{(n+1)} \in \theta_1$,

We define θ_1 as:

$$\theta_1 = (V, A, S, \Re, F)$$

where $V = \left(\{s_1...s_n\} \in \mathcal{S}, \{v_1...v_n\} \in \mathcal{V}, \{o_1...o_n\} \in \mathcal{O}\right)$ is an alphabet, $A \subseteq V^*$ is a set of one axiom constructed as an ordered triplet $(u \in \mathcal{S}, u \in \mathcal{V}, u \in \mathcal{O})$, $S \in A$ is the initial axiom, $\Re \subseteq R_{\triangleright} \cup R_{\vdash} \cup R_f$, $F = f\cdot$, where

$$\begin{aligned} A = &\quad (a_{10} \,\|a_{10} = s_{10}v_{10}o_{10}) \\ S = &\quad a_{10} \\ R_{\vdash} = &\quad 7 \\ R_{\triangleright} = &\quad 1 \end{aligned}$$

And θ_2 as:

$$\theta_2 = (V, A, S, \Re, F)$$

where $V = \left(\{s_1...s_n\} \in \mathcal{S}, \{v_1...v_n\} \in \mathcal{V}, \{o_1...o_n\} \in \mathcal{O}\right)$ is an alphabet, $A \subseteq V^*$ is a set of axioms constructed as an ordered triplet $(u \in \mathcal{S}, u \in \mathcal{V}, u \in \mathcal{O})$, $S \in A$ is the initial axiom, $\Re = R_{\vdash} \cup R_{\diamond} \cup R_{\triangleright} \cup R_f$, $F = f\cdot \cup f\wr$.

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

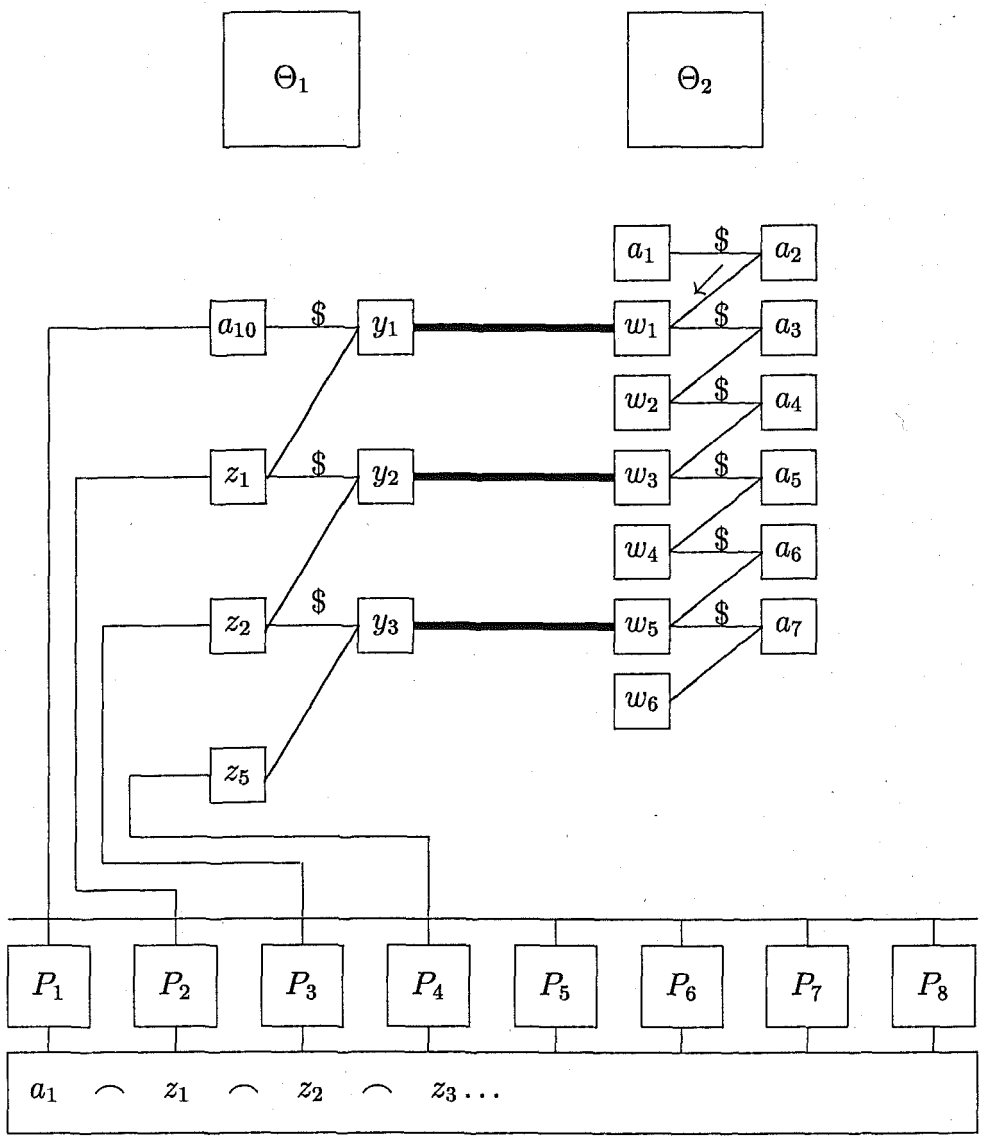


Figure 13.2: Monologic parallel systems Θ

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

$A = \quad (a_1...a_n \,\|a_n = s_n v_n o_n,\ per\ n \geq 1)$
$S = \quad a_1$
$R_{\vdash} = \quad 3, 27, 42$
$R_{\Diamond} = \quad 6$
$R_{\rhd} = \quad 4, 56$

Even though it is guessed that both systems work in parallel, we propose a simplification by following this process:

1. to carry out the whole derivation θ_2,

2. to take w_{2n+1} for $n \geq 0$ as strings y of θ_1,

3. to carry out the derivation θ_1.

We start derivation θ_2:

	θ_2		
STEP	OPERATION	w	y in θ_1
STEP 1	$\vdash_3$	w_1: $s_1 o_2$	y_1
STEP 2	$\rhd_{56}$	w_2: $s_3 v_3 s_1 o_2$	
STEP 3	$\vdash_{27}$	w_3: $s_3 v_3 o_4$	y_2
STEP 4	$\Diamond_6$	w_4: $\wr(s_5 v_5)_s v_3 o_4$	
STEP 5	$\rhd_4$	w_5: $\wr(s_5 v_5)_s v_6 o_6 v_3 o_4$	y_3
STEP 6	$\vdash_{42}$	w_6: $\wr(s_5 v_5)_s v_6 o_6 v_3 o_4 v_7 o_7$	

Once the results that must act as strings y in θ_2 are obtained, the derivation is carried out:

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

θ_1

STEP	x	y	Operation	z
STEP 1	$s_{10}v_{10}o_{10}$	s_1o_2	$\rhd_1$	$s_{10}s_1v_{10}o_{10}$
STEP 2	$s_{10}s_1v_{10}o_{10}$	$s_3v_3o_4$	$\rhd_1$	$s_{10}s_1s_3v_{10}o_{10}$
STEP 3	$s_{10}s_1s_3v_{10}o_{10}$	$\wr(s_5v_5)_S v_6o_6v_3o_4$	$\vdash_7$	$s_{10}s_1s_3v_{10}o_6v_3o_4$

While the pile is being filled:

STEP	Pile
Base	$s_{10}v_{10}o_{10}$
P_2	$s_{10}v_{10}o_{10} \frown s_{10}s_1v_{10}o_{10}$
P_3	$s_{10}v_{10}o_{10} \frown s_{10}s_1v_{10}o_{10} \frown s_{10}s_1s_3v_{10}o_{10}$
p_4	$s_{10}v_{10}o_{10} \frown s_{10}s_1v_{10}o_{10} \frown s_{10}s_1s_3v_{10}o_{10} \frown s_{10}s_1s_3v_{10}o_6v_3o_4$

The final result is the one that stores in the last step of the derivation: $s_{10}v_{10}o_{10} \frown s_{10}s_1v_{10}o_{10} \frown s_{10}s_1s_3v_{10}o_{10} \frown s_{10}s_1s_3v_{10}o_6v_3o_4$

13.7 Conclusions

Recombination systems are rudimentary approaches to mental processes of language generation. In short, the following comparisons can be carried out:

- Simple systems → thought.
- Complex systems → thought.
- Monologic parallel systems → speech.
- Dialogic parallel systems → talk.

We may say that simple systems are a suitable analogy of thought because they work with no definite direction and without limitations in their rules, adding more and more simple structures to a derivative line that can be hardly

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

prevented. In fact, the strings that can be the goal of a simple system must be clearly simple because any recursive rule has to ve avoid in order to obtain a concret sequence, jast as happens in any evolution not guided by thought.

Complex systems have been defined as simple systems that work with little objectives so as to recombine them later. This is a good analogy of what an individual thought directed towards a specific objective means, or at least it can feign it quite well.

Parallel monologic systems have been built so as to be able to simulate monologues, it is to say, framents of controlled an reasoned talk, with specific objectives, in which not every element is internal, but that the external world or other thoughts already formed provide the material so as to end up formulating the talk.

Finally, parallel dialogic systems feign to simulate a talk in which, on both sides, elements are furnished that are usually interrelated or come from formulations done outside.

This nature of inner (talking) and exterior (context) interaction that exists in parallel systems is what justifies the inclusion of not basic stage strings that can be the result of other derivations, as we may perfectly notice in the mechanism of supplying of stage strings in the monologic systems (fig. 13.2).

In order to complete such analogies, among other things, there should be constucted what we name *sintactic molecular systems*, which are obtained after linking *recombination systems* + *evolutive rules.*

Except for the parallel systems, which work without any specific objective, the others try to produce specific strings. We name *minimal system* for a particular string, the minimal existing system that can generate that string. A classification of minimal systems can be made:

- Simple:
  - Restricted,

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

  - Extended.

- Complex:
  - Restricted,
  - Extended.

From where a linguistic classification of structures emerges, between:

- Compact: are those produced minimally by simple systems,
  - restricted compact: are those produced minimally by restricted simple systems,
  - extended compact: are those produced minimally by extended simple systems.
- Articulated: are those produced minimally by complex systems
  - restricted articulated: are those produced minimally by restricted complex systems,
  - extended articulated: are those produced minimally by extended complex systems.

This classification of linguistic structures, whose each domain is nowadays unknown, does not correspond to any of the existing ones up to now, and it may help to rearrange the whole recombination view.

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

Chapter 14

Conclusions

In chapter 4 we have pointed out the following **goals** in the development of molecular recombination methods for syntax:

1. To study the *linguistic complexity* with the tools offered by molecular model.

2. *To simulate the generating mechanisms* of complex syntactic structures.

3. To check whether the *domain of structures* that can be obtained by means of recombination methods is comparable to the domain of complex structures of the language.

In order to obtain these goals, one has relied on the following **theoretical stands**:

a) *Minimal axiomatic structures of the language.* They have been established in the second part of the thesis in order to provide the basic element with which to work throughout the whole work. The *basic ULPS* has been determined as a stage for molecular operations.

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

b) *Recombination methods,* which have been extracted from genetic phenomena and are based on the intuitive mechanisms of cutting and splicing. Throughout this part of the thesis we have used the following ones:

- splicing: link of two structures with blunt ends,
  - * strict splicing,
  - * replication,
  - * controlled splicing;
- sticker links: joint of two structures with cohesive ends;
- mixed links: joint of a structure with blunt end with anoter one with staggered end.

From all these methods, strict splicing as well as recombination and sticker links had already been experimented on computational sciences. However, controlled splicing and mixed links have been introduced due to their adjustment to the linguistic study without being previously formalized out of this field.

Due to the fact that genetic approach had not been tested by syntax up to now, has provoked some very novel **results**. Those can be included in these four points, following the guide of goals that we had indicated:

1. Redefinition of some syntactic concepts from the new molecular point of view.
2. Generation of recombination structures.
3. Reorganization of the domain of syntactic complexity.
4. Cooperation of systems.

Now we will examine and systematize the conclusions we have reached in each one of these four points:

UNIVERSITAT ROVIRA I VIRGILI
MOLECUAR COMPUTING METHODS FOR NATURAL LANGUAGE SYNTAX
Gemma Bel Enguix
ISBN:978-84-691-1896-2/D.L:T-352-2008

Redefinition of some syntactic concepts from the new molecular outlook

Answering to the first goal of studying the syntactic phenomena from a new outlook, genetic focusing leads to a revision of some traditional concepts. It offers a quite independent and renewing perspective with enough theoretical bases so as to suggest a reflection on the engineering of language which is materialized in the introduction of new syntactic concepts.

- Double stratification of syntactic structures with different components:
  - ULPS,
  - pattern,
  - pole,
  - focus.
- Linguistic strings: classification and definition.
- Ghosts: typology and appearance rules.
- Relations and focal assemblies with concepts such as:
  - the set ℓ for v and $^{-}\ell$ for o,
  - osmosis,
  - the groups σ, ω, ϖ, ξ.
- Strings and ghosts readjustment.

Generation of recombination structures

In goal 2, our purpose was to simulate those mechanisms that produce complex syntactic structures.