THEORETICAL AND EFFECTIVE COMPLEXITY IN NATURAL LANGUAGE PROCESSING

Jean-Yves MORIN
AI Program
Department of Linguistics and Translation
University of Montreal
P.O.B. 6128, Station "Centre Ville"
Montreal (Quebec)
H3C 3J7
CANADA

FAX: (1-514) 343-2284
email: morinjy@iro.umontreal.ca

Abstract

In this paper, we first review some *theoretical complexity* results relevant to NLP and we show both their interest and inherent limitations. We then argue for a notion of *effective complexity* and, we try to identify effective *sources of complexity* and *sources of determinism* in natural language processing. Finally, we show how the former can be tamed by using the latter in order to guarantee effectiveness of language computations.

Theoretical complexity

In the last few years, there has been some interest in applying the techniques of algorithmics, and especially *complexity theory* (CT) in order to characterize the computational properties of modern grammatical formalisms: LFG (Berwick, 1982), GPSG (Barton, 1985), Barton et al., 1987), Ristad, 1986a, b, c, d, 1990b), 2-level morphology (Barton, 1986, Barton et al., 1987), prosodic morpho(phono)logy (Ristad, 1990a, 1994), etc.

Here is a summary of some the results of this "language complexity game," as Ristad, 1993) has called it.

(1) The UNIVERSAL RECOGNITION PROBLEM (URP)[1] for lexical functional grammars (Bresnan, 1983 ed.) is NP-hard (Berwick, 1982), Barton et al., 1987, ch. 4).

(2) URP for two-level morphology (Karttunen, 1983), Koskenniemi, 1983) is NP-complete (Barton, 1986), Barton et al., 1987, ch. 5).[2]

(3) URP for ID/LP grammars is NP-complete (Barton, 1985), Barton et al., 1987, ch. 7).

(4) URP for unordered CFGs is NP-complete (Barton et al., 1987, appendix A).

(5) URP for classical GPSG (Gazdar et al., 1985) is EXP-POLY-hard (Barton et al., 1987, ch. 8), Ristad, 1986a, b, c, d, 1990b).

(6) URP for R-GPSG is NP-complete (Barton et al., 1987, ch. 9), Ristad, 1990b).[3]

(7) The problem of *morpheme sequence generation* and *morpheme sequence recognition* for prosodic morphology ("prosodic composition"

---

[1] Barton et al. (1987) contrast the FIXED RECOGNITION PROBLEM (FRP) and the UNIVERSAL RECOGNITION PROBLEM (URP). They consider URP to be more representative. In the case of FRP, the language is fixed and the grammar does not constitute a parameter of the problem. In the case of URP, the grammar is a parameter of the problem. They argue that grammars should constitute a parameter, because of their importance.

[2] Koskenniemi & Church (1988)

[3] R-GPSG (for Revised GPSG) is a restricted version of GPSG characterized by limitations on almost all components: limitations on the depth of syntactic categories (unit feature closure: category-valued feature can take only atom-valued features as value), limitations on the length of ID-rules, limitations on the interaction of metarules (unit closure), simple defaults replacing both (FCRs and FSDs) and limitations on UIPs (especially the Head Feature Convention).

and "prosodic recognition" , in Ristad's terminology) are NP-complete (Ristad, 1990a, 1994).[4]

But, the applicability of these results is fairly limited for many reasons.

(A) Coarseness of CT. Complexity theory gives us only a very coarse-grained classification of the complexity (or cost) of computational problems in terms of bounds (*orders of complexity*) in the *worst-case* : O(higher bound), Ω (lower bound) and Θ (both higher and lower bound). It would be more useful to have a fine-grained characterization of problems for example, in terms of *average* (or *most frequent*) case(s), but such a characterization is not (yet) available.[5]

(B) Descriptive complexity. In the case of natural language computations, *descriptive complexity* (the complexity of grammars) seems much more important than *algorithmic complexity* (the complexity of the algorithms using them).[6] There

---

[4] These problems are defined in the following way by Ristad (1994):

"The Morpheme Composition Problem for prosodic morphology ("Prosodic Composition") is to decide whether the phonological correlates of a given set of morphemes can be composed into an executable phonological structure, according to a given morphological dictionary". (Ristad (1994: 193)

"Therefore, the Possible Word Problem for prosodic morphology ("Prosodic Recognition") is to decide whether a given sequence of phonemes is subsumed by the phonetic correlate of some combination of the morphemes listed in the morphological dictionary of a particular human language". (Ristad, 1994: 196)

On prosodic morphology, see, for instance, McCarthy (1981) or McCarthy & Prince (1990).

[5] Perrault (1984) had already made this point. An analysis of overall cost (or even redeemable cost), taking into account possible optimizations of complex but frequent cases (precompilation, memoization) might also be interesting.

[6] For example, given a $O(|G|^2 * n^3)$ bound (which is the actual bound for Earley's algorithm) in any parser with substantial coverage, the size of the grammar |G| can easily be larger than 10^6 symbols (recall that in CFG-based algorithms, the lexicon must be entirely spelled out, with all inflectional and

have been many interesting developments in the field of descriptive (or Kolmogorov) complexity recently.[7] But, to our knowledge, application of Kolmogorov complexity or related approaches to natural language computations (as opposed to formal languages)[8] has been fairly limited.[9] The reasons for this should be obvious. Descriptive (Kolmogorov) complexity deals with the shortest possible descriptions of objects.[10] In the case of natural language, it is very hard, if not impossible, to prove that something is the shortest possible description, even for a single phenomenon.[11]

(C) Most of the results concern grammatical formalisms.

Partiality of grammatical formalisms. It should be obvious that any grammatical formalism is *partial*, in the sense that it will always be possible to write grammars in a formalism which are not possible grammars of human languages. Trivially,[12] this

---

derivational morphology expanded) will almost always dominate the n^3 factor, except for large values of n ($n \geq 100$) seldom if ever encountered in practice.

[7] Cf. Kolmogorov (1965), Solomonoff (1964), Chaitin (1987) for classical works and Li & Vitányi (1993) or Watanabe (1992 ed.) for a rich sample of recent developments.

[8] Cf. Li & Vitányi (1993) and Boekee et al. (1982).

[9] Cf. Rissanen & Ristad (1994) for an application of the MDL (minimum description length) principle to the acquisition of metrical phonology.

[10] Furthermore, the theory of descriptive complexity makes use of sophisticated mathematical tools with which most linguists (including the present author) are not thoroughly familiar.

[11] If one were to adopt a principle-and-parameters approach, then it might be possible to define the shortest description of a given vector of binary parameters. The problem with this kind of approach is that a lot is hidden in the *interpretation* of the parameters and this would have to be spelled out in order for Kolmogorov complexity to be applicable.

can be done for any grammar G by reducing its lexical component to only one lexical form *f*, and then associating with this form all the lexical types $\{t_1, t_2, ..., t_m\}$ defined by the original grammar: $f \in \{t_1, t_2, ..., t_m\}$ thus creating a one-word, perfectly ambiguous grammar, where all sentences are strings of *f*s.

(D) The results concerning grammatical formalisms are essentially *negative*.

It has been shown, for example, that the URP (universal recognition problem) is NP-hard for classical LFG (lexical functional grammar) and EXP-POLY-hard for classical GPSG (Barton et al., 1987), thus showing that these two formalisms are potentially intractable (i.e., not inherently efficient, qua formalisms).

(E) The reductions used in these demonstrations are not perfectly *faithful*.

(i) They are not always spelled out rigorously in full detail.[13]

(ii) They are based on artificially constructed data, which could never appear in actual natural languages or descriptions thereof.[14]

(iii) They make crucial use of *empty categories*.[15]

(iv) They deal only with abstract *grammar formalisms* and do not take into account *substantive constraints*, which effective grammars also respect.

---

[12] Assuming, uncontroversially, that any grammatical formalism will allow lexical information to be represented in some way, relating some representation of form (phonological, graphemic, etc.) with grammatical information.

[13] Manaster-Ramer (1994) makes much the same point.

[14] It could even be argued that the fact that such data present difficulties for a given formalism is a quality, not a defect.

[15] Which, it should be noted, are not an essential component of LFG, GPSG or HPSG, as opposed to GB.

Nonetheless, the mere fact that such reductions are possible within a grammatical formalism is indicative. It allows us to discover that some constraints (substantive or formal) are implicitly respected by effective descriptions but are not explicitly stated either as part of the formalism itself or as substantive constraints attached to it. For instance, for GPSG (Gazdar et al., 1985) and HPSG (Pollard & Sag, 1994), one can mention:

(a) the implicit limitation on the *length* of (the right-hand side of) ID-rules (GPSG) or ID-schema (HPSG);

(b) *functional* constraints on the contents of ID-rules or ID-schema[16]

(c) *endocentricity* of ID-rules or ID-schema[17]

(d) dispensability of *empty categories*[18]

Empty categories are not an essential component of information-based grammatical theories (as opposed to configurational theories, like GB). In information-based grammatical theories, global dependencies are linked to lexical expectations, which do not have to be computed but just searched. A trace empty node, which can be hypothesized just about anywhere in GB, corresponds to an element of a SUBCAT list, which is part of stored lexical entries and reduced only by actually occuring elements (complements, or fillers).

(e) *universal projection of lexical information*

The *universal projection of lexical information* states that all lexical categories are projected, not only major categories (but minor categories do not have *autonomous*

---

[16] Daughter nodes in ID-constraints (rules or schema) are typed (e.g. lexical, head, complement, adjunct, filler, etc.)

[17] Although this is not much of a constraint by itself, as Kornai & Pullum (1990) have demonstrated for all versions of X-bar theory, it is sufficient to exclude some perverse use of ID-rules.

[18] Cf. Pollard & Sag (1994, ch. 9).

projections).[19] This simply captures the intuition that all lexical items in a string carry grammatical information, i.e., there are no *useless* words. It also avoids *projection paradoxes* (i.e. is a noun phrase an NP or a DP?) and the proliferation of *functional projections*, characteristic of GB approaches (where distinct information has to correspond to distinct nodes), with all the empty nodes they presuppose.

---

[19] That is, the projection of a minor category must be unified with that of a major category. For instance, in a sequence DET(erminer) QUANT(ifier) CLASS(ifier) N, all four categories have a projection DETP, QUANTP, CLASSP and NP, but only NP is autonomous. Therefore, there is only one NP node holding the grammatical information of all four projections :

DETP = QUANTP = CLASSP = NP.

Cf. Morin (1989).

(f) *off-line parsability* (or *bounded projection.*)

Off-line parsability (Kaplan & Bresnan, 1983) excludes non branching cyclic derivations, which could lead to undecidability. A grammar is off-line parsable if it does not allow derivations of the form: $A \Rightarrow^* A \Rightarrow^* \alpha$ (or, in terms of parse trees, cyclic trees of the form: $[_A \ldots [_A \alpha] \ldots]$, where ... contains only further brackets.) If such derivations (or parse trees) were allowed, the same string α could be assigned an infinite number of structures and parse trees could be infinitely deep for a given string.[20]

A grammar obeys bounded projection if and only if :

(i) any local tree admitted by the grammar is either a projection (i.e., $[_{X^i} \ldots X^{i-1} \ldots]$), an adjunction (i.e., $[_{\alpha'} \alpha \beta]$) or a coordination (i.e., $[_{\alpha'} \alpha_1 \alpha_2 \ldots \alpha_n]$) tree,

(ii) projections are bounded: there is a maximal value (2 in our model) of max for any projection X^{max}) and

(iii) it does not allow empty categories.

This constraint is much stronger than OLP, it thus also guarantees decidability.

Also, on the positive side, CT results help us identify some aspects of grammatical formalisms (e.g., empty categories, empty derivations) as potentially problematic. Empty categories allow a complex hypothesis space to grow indefinitely, independently of the length of the input. Empty derivations allow derivation trees to

---

[20] Shieber gives a formal definition of OLP, which is more general than the traditional LFG (Kaplan & Bresnan, 1983: 266) one, while being applicable to abstract constraint-based grammars (where, informally, tree-nodes are labeled by trees). τ / <0> is, informally, the label of the root of tree τ and ρ is a monotonic weakening function (like subsomption).

"Definition 57 A grammar *G* is *off-line parsable* if and only if there exists a finite-ranged function ρ on models such that $\rho(M) \leq M$ for all M and there are no parse trees τ admitted by *G* such that $\rho(\tau$ / <0>) = $\rho(\tau'$ / <0>), for some τ' a sub-parse tree of τ with identical yield". (Shieber, 1992: 81)

Haas (1989) defines a constraint of depth-boundedness, which is stronger than OLP, but weaker than bounded projection.

grow indefinitely, independently of the length of the input. Moreover, it encourages us to consider natural language computations at a more abstract level than the usual *algorithmic level*, in Marr's (1980) terminology, namely the *computational level*, where problems are defined purely in terms of input and output, independently of the specific algorithms and data structures used.

There are also some results, concerning specific language computation problems, which purport to be defined more or less independently of any given formalism.

(I) URP for agreement grammars (simple grammars embodying both agreement and lexical ambiguity) is NP-complete (Barton et al., 1987, ch. 3, Ristad & Berwick, 1989).

(II) The anaphora resolution problem is NP-complete (Ristad, 1993).

These reductions seem to suggest that natural language computations are inherently NP-complete. How can we explain then (unless $P = NP$) that actual language processing by humans is normally quite efficient? One would have to resort to mysterious performance factors which would not degrade performance (like the more usual performance limitations), but, on the contrary, improve it, acting as they would as *oracles* or *accelerators*.[21]

---

[21] As a matter of fact, Ristad's position on this problem is not very clear (as Manaster-Ramer (1994) also notes in his review of Ristad, 1993). On the one hand, he virulently attacks the traditional competence-performance distinction, while, on the other hand, he uses something quite similar to account for the fact that natural language computations are not intractable, after all. A much simpler way out would be to assume that humans do not use only *linguistic* knowledge in language computations, but other sources of information, which act as sources of determinism, counteracting the sources of complexity present in natural languages.

Effective complexity

A different (but complementary) approach to the study of complexity tries to identify inherent *sources of complexity* in natural languages, as opposed to sources of complexity which are simply artifacts of the particular formalisms used.

It also tries to identify means to effectively cope with complexity problems and to reduce the disastrous effects of such complex computations by insulating them in precisely defined locations to avoid complex interaction dependencies, or by identifying *sources of determinism* that effectively constrain natural language computations.

Sources of grammatical complexity[22]

We can identify many inherent sources of grammatical complexity in natural languages.

First, there is *lexical complexity*.

The number of lexical items in any wide-coverage model is quite large ($\geq 10^n$, where $4 \leq n \leq 6$, using conservative estimates). The information associated with each of these items is complex.[23] Furthermore, lexical items can be ambiguous, the same form being associated with many types.[24] The structure of the lexicon itself can also be quite complex (with defaults, simple or multiple inheritance, lexical types, lexical rules, etc.). But an interesting feature of lexical information is that most of it can be

---

[22] There is no room here to discuss semantic and pragmatic complexity.

[23] In an explicit (but purely linguistic) lexicon, like the DEC for French, (Mel'cuk et al., 1984-...), for example, each lexical entry corresponds to pages of (fairly succinctly coded) information.

[24] Here, we use the word 'type' in an informal sense, to refer to any particular combination of grammatical information.

precompiled and stored.[25] Thus, at runtime, lexical retrieval can produce, in linear time, all the types associated with a given form. If a form is ambiguous, a disjunctive type will be retrieved.

Then, there is syntactic complexity.

At the level of phrase structure, any natural language exhibits a large number of grammatical constructions (local trees, in an information-based framework), with many possible values for each of the constituents (nodes in the local tree) of any construction. There is also phrase structure ambiguity: active ambiguity (many possible constituents for a given type of object) and passive ambiguity (many possible types of which a given type can be a constituent).

At the relational level, there are grammatical, thematic and rhematic relations, as well as binding relations (global dependencies, control, rection, anaphora, etc.).

Sources of determinism: natural partitions

If all these sources of complexity could interact freely, and thus combine multiplicatively, language computations would obviously be intractable. But, we would like to suggest that there are also inherent sources of determinism in natural languages, which make it possible to partition the space of objects and constraints in

---

[25] Most of it, but not all of it. For mildly inflected languages, like the Romance languages, storing precompiled inflected forms seems to be feasible and could result in an order of magnitude increase of the size of the lexicon. (We can mention that, at the end of the nineteenth century, Bescherelle hand-compiled all the inflected forms of some 8000 French verbs, resulting in a two volume dictionary of verbal forms.) Recall that most of the space in the lexicon is used for *grammatical information*, not for forms, and inflected forms share most of this information. For highly inflected languages, (like Latin, Russian, Basque, Finnish, etc.), precompilation of forms does not look like such a good idea, but precompilation of morphological processes could reduce on-line computations to very simple (deterministic) processes.

such a way that many of these sources of complexity could combine additively instead of multiplicatively.

Lexical objects and syntactic constraints

A first kind of partition, already implicit in traditional conceptions of language, is the partition of linguistic entities into *lexical objects* (stored in the lexicon) and *grammatical constraints* (represented in the grammar). In practical terms, since lexical objects are, for the most part, precompiled, this suggests a partition of parsing, for example, in two distinct phases.

First, lexical *initialization* (retrieving all stored lexical information for all the segments of a string $\omega = \omega_1\ \omega_2 \ldots \omega_m$

$$lex(\omega) = lex(\omega_1)\ lex(\omega_2) \ldots lex(\omega_m)$$

and then *parsing* proper (applying grammatical constraints to find an analysis for ω).

$$parse(lex(\omega_1)\ lex(\omega_2) \ldots lex(\omega_m)) = \sigma$$

Many parsing algorithms (including bottom-up filtering (Blache, 1990, Blache & Morin, 1990)) use such a partition.[26] Such a partition is useful inasmuch as lexical information is stored, and not computed. Even if a lot of ambiguous or disjunctive information is retrieved in this phase, it does not involve any computations. Therefore, trying to disambiguate at this point would simply reduce the size of the $lex(\omega_i)$'s, potentially removing information which will have to be recovered at a later point. It is just not worth the effort.[27]

---

[26] Lexical initialization could even be done in parallel, since lexical access for a form is completely independent of lexical access for the other forms. Cf. Sabot (1988) for the details of such a proposal.

[27] On the other hand, in some cases like speech recognition, or for languages like Chinese (where words are not separated in writing) or like Basque or Finnish (where morphological computations are needed), it might be the case that the lexical initialization part should itself be further decomposed. Cf. Gan (1994) for an interesting integrated model of word segmentation in written Chinese, where, instead of predefined partitions like the ones we are suggesting, partitions are defined on-line, by taking into account the "computational temperature of the system". When computational temperature is

Phrase structure constraints and functional constraints

Another type of partition, which is implicit in work inspired by LFG (Bresnan, 1983 ed.), or by GPSG (Gazdar et al., 1985), but much less so in HPSG (Pollard & Sag, 1994), is the one between phrase structure and functional constraints.

Maxwell & Kaplan (1993) discuss the interface between these two types of constraints, which have very different computational properties, CFG-phrase structure parsing being polynomial in the size of the input string, while known general constraint satisfaction algorithms are exponential in the size of the constraint system. They show that simple composition or simple interleaving (on-line pruning of phrasal edges not satisfying a set of functional constraints in an active chart) are both exponential in the worst case, while non interleaved pruning (caching the constraint solutions on each edge) is polynomial but involves a lot of copying overhead. What they suggest instead is *factored extraction*: extracting a concise set of functional constraints from the active chart and passing them to a constraint solver. First, a chart is built, based only on the context-free backbone grammar. Then a set of constraints is recursively extracted (starting at the root node) and combined conjunctively (except for ambiguous nodes, where they are combined disjunctively) and reduced using various classical techniques.

But what makes the strategy particularly interesting is that it uses specific linguistic knowledge in the reduction phase. Since heads and their projections share all constraints (this is itself a constraint, expressed in LFG by the equation $\uparrow = \downarrow$), head constraints are substituted for their projections. Therefore, in the case of ambiguous constituents with the same head, the disjunction can be reduced to only the constraints

coming from the effective differences.[28] This is a special case of what we call *propagation constraints* below.

Further partitions of PS-constraints

Immediate dominance and linear precedence constraints

Phrase structure constraints can themselves be further partitioned. A natural dividing line is between ID and LP constraints. Again, ID-rules or schema could be used directly to parse input and LP constraints to filter ungrammatical combinations. A variant of this general strategy, bottom-up filtering (Blache, 1990, Blache & Morin, 1990), precompiles LP relations in exclusion tables that act as prefilters on ID-rules.

Decomposition, adjunction and coordination constraints

It is a well-known fact that adjunction (and coordination which is just a particular kind of adjunction with tighter constraints) enormously complicate the search space of a parser. It might be interesting to separate the straight decomposition rules with lexical heads from both of these types. *D-rules* constrain the obligatory unification of minor category phrasal projections with permissible (and accessible) major category phrasal projections and the attachment of subcategorized complements. *A-rules* and *C-rules* constrain adjunction and coordination of lexical or phrasal categories. We can then have a partition of the *parse* function where we first do strict decomposition :

$$decomposition(lex(\omega_1)\ lex(\omega_2)\ \ldots\ lex(\omega_m)) \quad = \quad \sigma'$$

---

high, anything goes, so to speak, and low-level constraints are applied, more or less at random. When computational temperature cools down and some structures have crystallized, only high-level constraints are applicable, if this does not work, temperature goes up again and so on, so forth.

[28] They also discuss the necessity of moving some functional constraints into the context-free part in order for factored extraction to be efficient, since their strategy is very sensitive to the specific form or the grammar used, as demonstrated in their experiments. Propagation constraints are more general (and, hopefully, robust) in that respect.

and then apply *adjunction* and *coordination* in an interleaved manner, but only if needed.[29]

$$adjunction\text{-}coordination(\sigma') = \sigma$$

In other words, only *D-rules* are always active (and their applicability is bound by the number and nature of lexical forms $lex(\omega_i)$ in the representation to be parsed. *A-rules* and *C-rules* are only activated when no more *D-rules* are applicable and there are still unattached constituents. *C-rules* also need the presence of specific markers. In that way, *adjunction-coordination* never interferes with *decomposition* and the composition of *decomposition* and *adjunction-coordination* is additive, not multiplicative.

An interesting feature of D-rules is that they only need to refer to coarse- or medium-grained grammatical information : parts of speech, projection level, functional constraints (*SUBCAT*, *SPEC-OF*, *ADJUNCT-OF*, etc.) and all this information is directly accessible in lexical entries and does not have to be computed.

Furthermore, once we adopt the hypothesis of *universal projection of lexical information* and *bounded projection*, it becomes possible, to strictly bound the number n of possible nodes in a parse tree given a sequence of i lexical forms ($n < 3i$) ($n \leq 2i$ for decomposition nodes and $n \leq i-1$ nodes for adjunction-coordination nodes). Of course, this presupposes that global dependencies are never expressed through empty categories.

Propagation constraints and coherence constraints

Fine-grained grammatical information is treated only in propagation and coherence constraints.

---

[29] Adjunction is needed only if a constituent is intrinsically an adjunct (e.g. a clitic, a sentential Comp, etc.) or is left unattached by decomposition (e.g. a non selected PP, an appositive NP, etc.). Coordination is needed only if a conjunction is detected (some constituent must also have been left unattached, since this is a special case of adjunction).

Propagation constraints apply to local trees. They guarantee that some types of information are propagated from daughters to mother (and vice-versa) in a local tree (but never between siblings). Given type abstraction over objects and grammatical information, they have the following form:

$$\kappa(\delta(\tau)) \quad = \quad \kappa(\delta'(\tau))$$

where τ is a local tree,

δ, δ' are abstract types of nodes (e.g. MOTHER, DAUGHTER, FILLER-DAUGHTER, etc.) and

κ is an abstract category type (a path in more traditional terminology).[30]

For example:

$$H(M(\tau)) \quad = \quad H(HD(\tau))$$

The head (H) value of the mother (M) is identical with the head value of the head (HD).

$$MINOR(M(\tau)) \quad \geq \quad MINOR(LEXD(\tau))$$

The MINOR value of the mother is an extension of the MINOR value of the lexical daughters (LEXD).

Coherence constraints, on the other hand, guarantee that every node in the final product is coherently labeled. *Coherence constraints* correspond more or less to FCRs in GPSG. They have the following form.

$$\alpha \quad \supset \quad \beta$$

where α and β are elementary constraints on categories (disjunctive and negative combinations are excluded, but conjunctive and doubly implicative combinations are allowed, since they are deterministic).

For example:[31]

[LEVEL : phrasal] $\supset$

---

[30] In our approach, paths are invisible. They are named by abstract types. So, changes in the representation do not affect access to proper values.

[31] These rules are part of our description of quantified NP's in Chinese (Morin & Ren, 1992).

([CLASS : α] ⇔ [QUANT : β])

A (phrasal) object is classified if and only if it is quantified.

It should be noted that *propagation constraints* and *coherence constraints* not need not take into account the origin of the relevant grammatical information.[32] Constraints can thus be applied blindly and locally again reducing non determinism. Furthermore, coherence constraints never instantiate anything, they just check their input and filter it out if they are not satisfied (unless we allow constraint relaxation). There is no free instanciation, any value appearing in a structure is entirely constrained, either by lexical or by grammatical constraints.[33]

Conclusion

In this paper, we have discussed some notions of complexity and some sources of effective complexity in natural language processing. We tried to show that, once certain hypotheses are adopted, sources of determinism in natural languages become apparent, and it is possible to use these results to partition the space of grammatical and lexical constraints in such a way as to guarantee efficient parsing.

References

Aho, A. & J. Ullman (1972-1973) *The Theory of Parsing, Translation and Compiling, vol.1: Parsing; vol. 2: Compiling* . Englewood-Cliff, NJ: Prentice-Hall.

Barton, E. (1985) "On the complexity of ID/LP parsing," *Computational Linguistics*, 11, 4: 205-218.

Barton, E. (1986) "Computational complexity in two-level morphology," *ACL-24*: 53-59.

Barton, E. et al. (1987) *Computational Complexity and Natural Language*, Cambridge, Mass.: MIT Press.

Berwick, R. (1982) "Computational complexity and lexical-functional grammar," *AJCL*, 8, 3-4: 97-109.

Berwick, R. (1984) "Strong generative capacity, weak generative capacity, and modern linguistic theories," *Computational Linguistics*, 10: 189-202.

---

[32] This is the *chaptalization* hypothesis (Morin & Ren, 1992).

[33] All the problems related to the complexity of free instanciation are thus eliminated a priori. Cf. Ristad (1986a, b, c, d, 1990b), Barton et al. (1987, ch. 8) and Jutras (1990) for different analyses of the complexity of free instanciation and related linguistic and computational problems.

Boekee, D. E. et al. (1982) "On complexity and syntactic information," IEEE Transactions SMC-12: 71-79.

Blache, Ph. (1990) *L'analyse syntaxique dans le cadre des grammaires syntagmatiques généralisées: Interprétation et stratégies*, doctoral dissertation, Université d'Aix-Marseille II.

Blache, Ph. & J.Y. Morin (1990) "Bottom-up filtering, a parsing strategy for GPSG," *COLING-90*, 2: 19-23.

Bresnan, J. (1983 ed.) *The Mental Representation of Grammatical Relations*, Cambridge: MIT Press.

Carpenter R. (1992) *The Logic of Typed Feature Structures*, Cambridge University Press.

Chaitin, G. J. (1987) *Algorithmic Information Theory*, Cambridge: Cambridge University Press.

Gan, Kok Wee (1994) *Integrating Word Boundary Disambiguation with Sentence Understanding*, Ph. D. dissertation, National University of Singapore.

Haas, Andrew (1989) "A parsing algorithm for unification grammar," *Computational Linguistics*, 15, 4: 219-232.

Jutras, J.-M. (1990) *L'instanciation en grammaire syntagmatique généralisée*, M.A. thesis, Université de Montréal.

Kaplan, R. & J. Bresnan (1983) "Lexical-functional grammar: A formal system for grammatical representation," *in* Bresnan (1983 ed.).

Kasper, R. (1987) *Feature Structures: A Logical Theory with Applications to Language Analysis*, Ph. D. dissertation, Univ. of Michigan.

Kasper, R. & W. Rounds (1986) "A logical semantics for feature structures," *ACL-24*: 257-266.

Kolmogorov, A. (1965) "Three approaches to the quantitative definition of information," *Problems of Information Transmission*, 1: 1-7 (translated from Russian).

Kornai, A. (1994) "The generative power of feature geometry," *Annals of Mathematics and Artificial Intelligence*.

Kornai, A. & G. Pullum (1990) "The X-bar theory of syntax", *Language*, 66, 1.

Koskenniemi, K. (1983) *Two-Level Morphology: A General Computational Model for Word-Form Recognition and Production*, doctoral dissertation, University of Helsinki.

Koskenniemi, K. & K. Church (1988) "Complexity, two-level morphology and Finnish", *COLING-88*.

Li, Ming & Paul Vitányi (1993) *An Introduction to Kolmogorov Complexity and its Applications*, Berlin, New-York: Springer-Verlag.

Manaster-Ramer, A. (1994) "Review of Ristad, E. S. (1993) *The Language Complexity Game*," *Computational Linguistics*, 21, 1: 124-131.

Marr, D. (1980) *Vision*, San Francisco: W.H. Freeman.

Maxwell, J. & R. Kaplan (1993) "The interface between phrasal and functional constraints," *Computational Linguistics*, 19, 4: 571-590.

McCarthy, J. (1981) A prosodic theory of nonconcatenative morphology, *Linguistic Inquiry*, 12: 373-418.

McCarthy, J. & A. Prince (1990) Foot and word in prosodic morphology,: the Arabic broken plural, *Natural Language and Linguistic Theory*, 8: 209-283.

Mel'čuk, I. et al. (1984-...) *Dictionnaire explicatif et combinatoire du français contemporain*, Montréal, Paris : Presses de l'Université de Montréal and Presses du CNRS, 3 volumes published.

Morin, J.-Y. (1985) "Théorie syntaxique et théorie du parsage: quelques réflexions," *Revue québécoise de linguistique*, 14, 2: 1-40.

Morin, J.-Y. (1989) "Particules et parsage universel," *in* Weydt, H. (1989 ed.) *Sprechen mit Partikeln.*, Berlin: De Gruyter, pp. 713-728.

Morin, J.-Y. & X. Ren (1992) "Classifier Features in Chinese: A GPSG Approach", *ICCL-1*.

Nagata, M. (1991) "An empirical study on rule granularity and unification interleaving, toward an efficient unification-based parsing system," *COLING-92*: 177-183.

Perrault, R. (1984) "On the mathematical properties of linguistic theories," *Computational Linguistics*, 10: 165-176.

Pinker, S. (1984) *Language Learnability and Language Development*, Cambridge: Harvard University Press.

Pollard, C.J. & I. Sag (1994) *Head-Driven Phrase Structure Grammar*, Chicago: University of Chicago Press.

Ristad, E. S. (1986a) "Computational complexity of current GPSG theory, *ACL-24*: 30-39.

Ristad, E. S. (1986b) "Defining natural language grammars in GPSG, *ACL-24*: 40-44.

Ristad, E. S. (1986c) "Sources of complexity in GPSG theory, *Theoretical Linguistics*, 13, 1-2: 105-124.

Ristad, E. S. (1986d) *Complexity of linguistic models: a computational analysis and reconstruction of generalized phrase structure grammar*. S.M. thesis, MIT.

Ristad, E. S. (1990a) "Computational structure of generative phonology and its relation to language comprehension," *ACL-28*: 235-242.

Ristad, E. S. (1990b) "Computational structure of GPSG models," *Linguistics and Philosophy*, 13, 5: 523-590.

Ristad, E. S. (1994) "Complexity of morpheme acquisition," *in* Ristad (1994 ed.: 185-198).

Ristad, E. S. (1993) *The Language Complexity Game*, Cambridge, Mass.: MIT Press.

Ristad, E. S. (1994 ed.) *Language Computations*, DIMACS, vol. 17, American Mathematical Society..

Ristad, E. S. & R. Berwick (1989) "Computational consequences of agreement and ambiguity in natural language," *Journal of Mathematical Psychology*, 33: 379-396.

Rounds, W. (1973) "A grammatical characterization of the exponential time languages," *Proceedings of the 11th Annual Symposium on Foundations of Computer Science*. New-York: IEEE Computer Society, p. 135-143.

Rounds, W. (1987) "Review of Barton, E., R. Berwick et E. S. Ristad (1987),*Computational Complexity and Natural Language*," *Computational Linguistics*, 13, 3-4: 354-356.

Rounds, W. (1988) "LFP: A logic for linguistic descriptions and an analysis of its complexity," *Computational Linguistics*, 14, 4: 1-9.

Rounds, W. (1991) "The relevance of computational complexity theory to natural language processing," *in* Sells, P., S. M. Shieber & T. Wasow (1991 ed.: 9-29).

Rounds, W. et al. (1986) "Finding natural languages a home in formal language theory, *in* Manaster-Ramer (1986 ed.) *Mathematics of Language*. New-York: John Benjamins.

Sabot, G. (1988) *The Paralation Model, Architecture-Independent Parallel Programming*, Cambridge: MIT Press.

Sells, P. et al. (1991 ed.) *Foundational Issues in Natural Language Processing*, Cambridge: MIT Press.

Shieber, S. M.(1983) "Direct parsing of ID/LP grammars," *Linguistics and Philosophy*, 7, 2: 135-154.

Shieber, S. M. (1985) "Using restriction to extend parsing algorithms for complex feature-based formalisms," *ACL-23*: 145-152.

Shieber, S. M. (1992) *Constraint-based grammar formalisms*, Cambridge, Mass.: MIT Press.

Solomonoff, R. (1964) "A formal theory of inductive inference," *Information and Control*, 7: 1-22 et 224-254.

Watanabe, O. (1992 ed.) *Kolmogorov Complexity and Computational Complexity*, Berlin, New-York: Springer-Verlag.