Computational Linguistics: History & Comparison of Formal Grammars

Raffaella Bernardi

KRDB, Free University of Bozen-Bolzano

P.zza Domenicani, Room: 2.28, e-mail: bernardi@inf.unibz.it

Contents

1. Formal Grammars

- We have seen that Formal Grammars play a crucial role in the research on Computational Linguistics.
- We have looked at Context Free Grammars/Phrase Structure Grammars, Categorial Grammar and Categorial Type Logic

But through the years, computational linguists have developed other formal grammars too.

Today, we will look at the most renown ones, at their generative capacity and their complexity. Then we mention some applications.

2. History of Formal Grammars

Important steps in the historical developments of Formal grammar started in the 1950's and can be divided into five phases:

1. Formalization: Away from descriptive linguistics and behavioralism (performance vs. competence) [1950's 1960's]
2. Inclusion of meaning: Compositionality [1970's]
3. Problems with word order: Need of stronger formalisms [1970's 1980's]
4. Grammar meets logic & computation [1990's]
5. Grammar meets statistic [1990's 2000's]

In these phases, theoretical linguists addressed similar issues, but worked them out differently depending on the perspective they took:

- constituency-based or
- dependency-based.

2.1. Constituency-based vs. Dependency-based

Constituency (cf. structural linguists like Bloomfield, Harris, Wells) is a **horizontal** organization principle: it groups together constituents into phrases (larger structures), until the entire sentence is accounted for.

- Terminal and non-terminal (phrasal) nodes.
- Immediate constituency: constituents need to be adjacent (CFPSG).
- But we have seen that meaningful units may not be adjacent –Discontinuous constituency or long-distance dependencies.
- This problem has been tackled by allowing flexible constituency: "phrasal rebracketing"

Dependency is an asymmetrical relation between a head and a dependent, i.e. a **vertical** organization principle.

2.2. Constituency vs. Dependencies

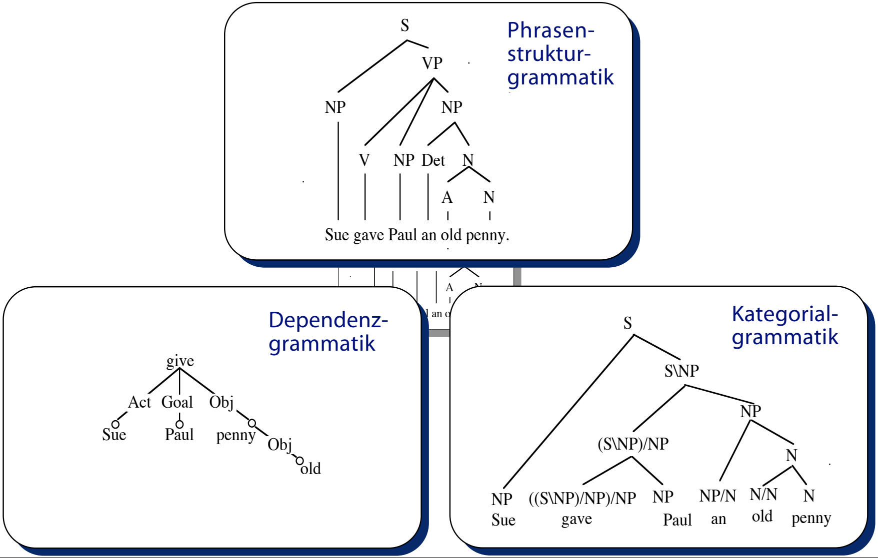
Dependency and constituency describe different dimensions.

1. A phrase-structure tree is closely related to a derivation, whereas a dependency tree rather describes the product of a process of derivation.
2. Usually, given a phrase-structrue tree, we can get very close to a dependency tree by constructing the transitive collapse of headed structures over nonterminals.

Constituency and dependency are not adversaries, they are complementary notions. Using them together we can overcome the problems that each notion has individually.

3. DG & CFPSG & CG

Three Traditions



3.1. Combining Constituency and Dependencies

In 1975, Joshi et ali introduced a grammatical formalism called Tree-Adjoining Grammars (TAGs), which are tree-generating systems. The application of TAGs to natural language is known as LTAGs.

- New way of thinking of domain of dependencies
- Localization of dependencies : elementary structures of a formalisms over which dependencies such as agreement, subcategorization and filler-gap relation can be specified.

4. TAG & CFG

CFG:

```
S --> NP VP        NP --> Harry          ADV --> passionately
VP --> V NP        NP --> peanuts
VP --> VP ADV      V --> likes
```

TAG:

```
a1   S                 a2    NP             a3 NP
   /   \                     |                 |
 NP|     VP                  peanuts           Harry
       /    \
     V     NP |
     |
   likes
```

4.1. TAG rules

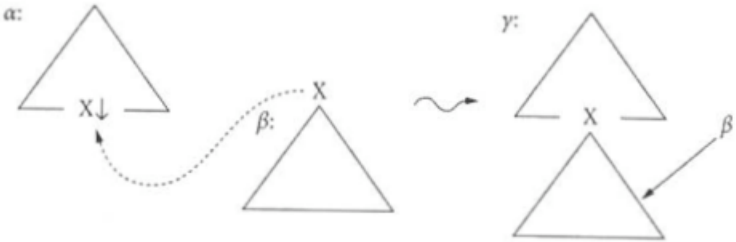


Fig. 26.2 Substitution

4.2. Example

Try to apply the substitution rules to the entries given above:

```
a1   S                   a2    NP              a3 NP
    /   \                      |                  |
 NP|      VP                   peanuts            Harry
        /    \
        V    NP |
        |
     likes
```

What does this rule correspond to in CG?

Do you think this rule is going to be enough?

4.3. Example

"Harry thinks Bill likes John"

what's the entry for "thinks"?

```
          S
        /   \
     NP|      VP
             /  \
            V    S|
            |
          think
```

And what about the sentence "Who does Harry think Bill likes?"

4.4. Example

To account for gaps, new elementary trees are assigned to e.g. TV:

```
            S
          /   \
   NP(wh)|     S
             /   \
          NP|     VP
                 /  \
                V    NP|
                |    |
            likes empty
```

4.5. Adjunction

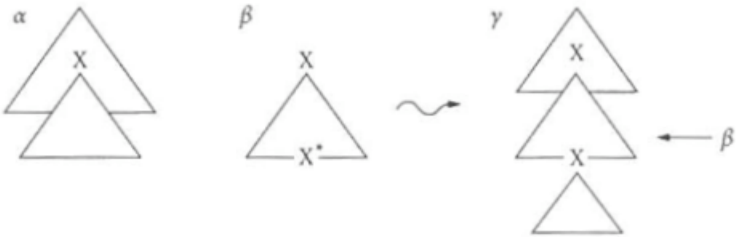
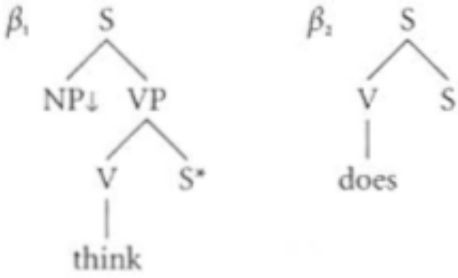


Fig. 26.5 Adjoining

The lexical entries "does" and "think" carry the special marker:



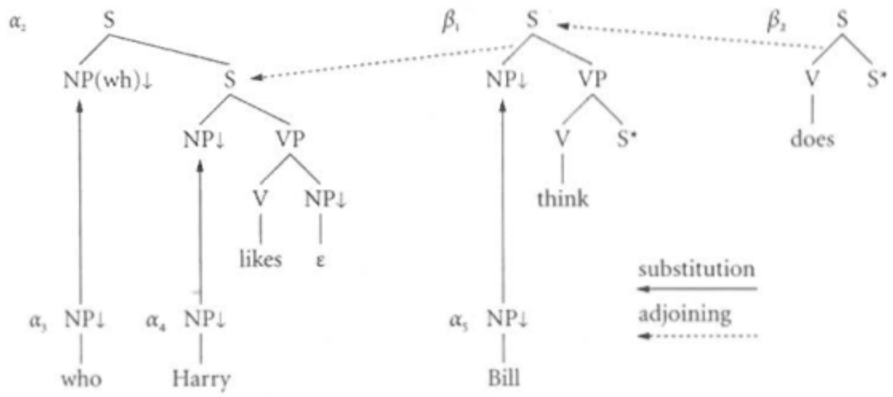


Fig. 26.9 LTAG derivation for *who does Bill think Harry likes*

Again, do you see any corresponds between TAG and CTL/CG?

5. Generative Power and Complexity of FGs

Recall, every (formal) grammar generates a unique language. However, one language can be generated by several different (formal) grammars.

Formal grammars differ with respect to their **generative power**:

One grammar is of a greater generative power than another if it can recognize a language that the other cannot recognize.

Two grammars are said to be

- **weakly** equivalent if they generate the same string language.
- **strongly** equivalent if they generate both the same string language and the same tree language.

5.1. DG, CG, CTL, CCG, and TAG

- DG: Gross (1964)(p.49) claimed that the dependency languages are **exactly** the context-free languages. This claim turned out to be a mistake, and now there is new interested in DG. (Used in QA)
- CG: Chomsky (1963) conjectured that **Lambek calculi** were also **context-free**. This conjectured was proved by Pentus and Buszkowski in 1997.
- TAG and CCG: have been proved to be Mildly Context Free.
- CTL has been proved to be Mildly Sensitive (Moot), or Context Sensitive (Moot) or Turing Complete (Carpenter), accordingly to the structural rules allowed.
- LG has been proved to be Mildly Context Free. (Moot 2008)

5.2. FG and applications

Wide coverage: Syntax-Semantics interface... with all the "compromise" needed to go wide.

- Steedman (and Szabolcsi): theory of CCG.
- Julia Hockenmaier: CCG Bank
- Curran, Clark, Bos: softwares http://svn.ask.it.usyd.edu.au/trac/candc/wiki

E.g. Used in QA and Textual Entailment. Could be useful for many applications!

6. HCI via Natural Language

In the '50 Machine Translation work pointed out serious problems in trying to deal with unrestricted, extended text in open domains. This led researchers in the '60 and early '70 to focus on question-answering dialogues in restricted domain.

Attention shifted from developing NL systems to solving individual language-related problems, e.g., to develop faster, and more efficient parsers.

Now, researchers are back to deal with unrestricted extended text and dialogues.

1. NLDB
2. Dialogue Systems,
3. QA
4. IQA

All of them aim at assisting users to access data from some source. Today we speak of NLDB, next time of Dialogue Systems and IQA.

7. Natural Language Interfaces to Data Bases

NLDB refers to systems that allow the user to access information stored in a database by typing requests in some natural language. Its history (see Androutsopoulos for more details):

’60/’70 they were built having a particular DB in mind. No interest in portability issues. E.g., LUNAR

late ’70 Dialogues; large DB; semantic grammars (domain dependent - no portable). E.g. LADDER

early ’80 From English into Prolog evaluated against Prolog DB. Eg., CHAT-80

mid ’80 popular research area. Research focused on portability issues. E.g. TEAM

’90 NLIDBs did not gain the expected commercial acceptance. Alternative solutions were successful (graphical or form-based interface). Decrease in the nubmer of papers on the topic.

7.1. Advantages & Disadvantages

- Advantages:
  - NLDB should be easier to use. But: currently only limited subsets of NL. Hence, training is needed.
  - It supports anaphoric and elliptic expressions.
- Disadvantages:
  - The NL coverage is not clear to the user. False positive expectation and False negative expectation
  - It is not clear to the user whether the rejected question is outside the system's linguistic coverage or the system's conceptual coverage. Need of diagnostic messages.
  - User assume intelligence of the NLIDBs.
  - NL is verbose and ambiguous.
  - Tedious configuration.

7.2. Experiments

- Training of the interfaces (graphical, SQL, NL). Then ask queries most of which are similar to the ones used in the training period.

  Results: NLIDBs seem to be better in queries where data from many tables have to be combined and in queries that were not similar to the ones the users had encountered during the training period.

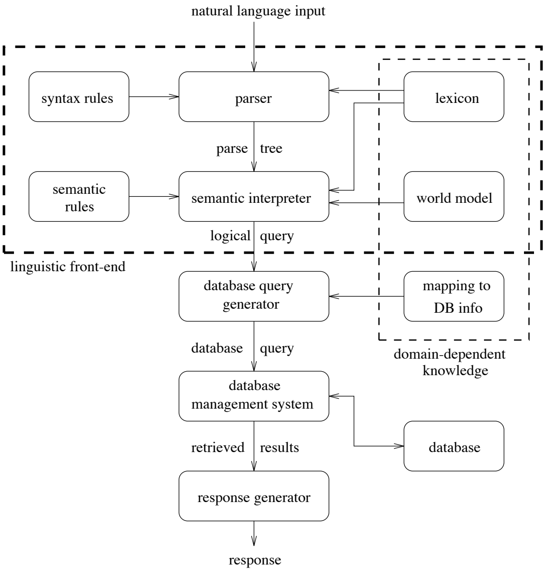
- NL is an effective method of interaction for casual users with a good knowledge of the DB, who perform question-answering tasks in a restricted domain.

- Another approach: Wizard of Oz experiment.

7.3. Linguistic problems

- Quantifier scoping. Ambiguous, ad hoc solutions (e.g., choose only one reading as possible, give different weights to QPs.)
- Conjunction and Disjunction: Sometime conjunctions in NL are actually interpreted as disjunction. E.g., List all applicants who live in California and Arizona. There are also cases of ambiguous use of "and", e.g., Which minority and female applicants know Fortran?"
- Nominal compound problem: E.g., "research department" vs. "research system". In the first case, the department carries out the research, in the second the system is used in research.
- Anaphora: Use of pronouns and possessive determiners or noun phrases to denote entities mentioned in the discourse. Solution: keep list of all entities, use the most recent one as link to the anaphora. Use of world knowledge.
- Elliptical sentences. E.g., U1: "Who is the manager of the largest department?" U2: "The smallest department?" Need of discourse model.

7.4. Sample Architecture



7.5. Which approach

Advantages of the last approach: modularity of the architecture

- the linguistic front-end is independent of the underlying DBMS
- domain knowledge is separated from the rest of the front end
- reasoning modules can be added between the semantic interpreter and the DB query generator.

7.6. Response generation

Failure Explain cause of failure to retrieve answer.

False Presupposition The system should report the false presupposition about the DB world.

Literal answers some time a literal answer would be "yes/no" but it won't be an acceptable answer. Cooperative answers can help. Sometime important to reason about the user's goal.

Misunderstandings translate the SQL query back to NL, (paraphrase modules)

7.7. Restricted NL input

Currently systems use limited subsets of NL.

Limitation user doesn't know which is this subset. Has to rephrase the question, does not know which questions could be handled.

Long term aim to broad the linguistic coverage.

Alternative approach deliberately and explicitly restrict the set of NL the user is allowed to input (controlled natural language.)

- **syntactic pattern**
- **menu-based**
- **ontology-driven** See Paolo Dongilli's work.
- **complexity of NL fragments** See Ian Pratt (will be here in one month) and Camilo Thorne works

7.8. Online Demos

Examples of today NLDBs:

- ACE: http://attempto.ifi.unizh.ch/site/tools/
- Geo http://www.cs.utexas.edu/users/ml/geo-demo.html
- PENG: http://www.ics.mq.edu.au/~peng/PengEditor.html
- PRECISE

7.9. Attempto Controlled English (ACE)

- Lexicon: limited set of type of words: e.g.

  "ACE verbs are in 3rd person singular or plural, in indicative mood, and in simple present tense. Both indicative and passive verbs can be used but passive constructions must include a prepositional phrase, e.g. '... by ... '."

- Grammar: limited set of constructions.

  "Sentence are a concatentation of a NP with a VP. It is possible to create well-formed-sentence with a single NP prefixed by the fixed phrase "there is/are". Composite declarative sentences are recursively built from simpler sentence using the predefined constructors: coordination, negation, global quantification, if-then subordination."

7.10. Ambiguity

The sentences of ACE are handled by the parsers and receive always only one MR, even in case they could be ambiguous.

E.g., relative clauses always attach to the most recent noun.

Every man owns a dog$_d$ that$_d$ likes a cat$_c$ that$_c$ likes a mouse and that$_c$ eats a bone.

They also deal with anaphora resolution –the use Discourse Representation Structures (DRS).

They generate paraphrases of the sentence to make sure the system and the user agree in the assigned interpretation.

Paraphrases is becoming a hot topic for HCI via NL.

8. Complexity of NL fragments

The FOL meaning representation of the entailment above is:

$$\{\forall x(man(x) \rightarrow mortal(x)), man(socrates)\} \models mortal(socrates)$$

Pratt has proved that `COP` is **PTIME**

Fragment	Decision class for satisfiability
`COP+TV+DTV`	**PTIME**
`COP+REL`	**NP-Complete**
`COP+REL+TV`	**EXPTIME-Complete**
`COP+REL+TV+DTV`	**NEXPTIME-Complete**
`COP+REL+TV+RA`	**NEXPTIME-Complete**
`COP+REL+TV+GA`	undecidable

`REL` relative pronoun.
`RA` restricted anaphora, pronouns take their closest allowed antecedents.
`GA` general anaphora.

8.1. "Which" from the ontology perspective

Which fragment? Our proposal is to merge Pratt's approach with the research mentioned above and use, as controlled language for accessing ontologies, those fragments with a **desirable computational complexity**.

- Description Logics (DLs) are the logics that provide the formal underpinning to ontologies and the Semantic Web.
- They are a decidable fragment of FOL, and experience has shown that they have the right expressivity required by the most commonly used formalisms for conceptual modeling, e.g. `UML` class diagrams and entity-relationship schemas.
- DL-lite is the maximal DL that has the ability to efficiently and effectively manage very large data repositories by relying on industrial-strength relational database management systems (`RDBMS`). Moreover, DL-lite can still capture the essential features of both `UML` class diagrams and `ER` schemas.
- Hence, we use DL-lite as the starting point to answer the **which** part of our question, viz. to pinpoint the most suitable fragment to add **specifications** in the ontology.

8.3. User: specification and queries

We consider the case where the ABox is actually stored in a database, and hence managed by a `DBMS`.

Given a DL-lite TBox $\mathcal{T}$ and a DB (storing the ABox), a user can be interested in:

1. adding new specifications to the TBox,
2. adding new facts to the DB, or
3. querying the DB.

9. English lite

The constraints expressed in the `TBox` are universals. They are of the form $Cl \sqsubseteq Cr$ that translates into FOL as $\forall x.Cl(x) \rightarrow Cr(x)$ and in natural language as

(a) [Every $\underbrace{\texttt{NOUN}}_{Cl}$] $\underbrace{\texttt{VERB\_PHRASE}}_{Cr}$

(b) [[Everyone $\underbrace{\text{[who } \texttt{VERB\_PHRASE}]}_{Cl}$] $\underbrace{\texttt{VERB\_PHRASE}}_{Cr}$]

Hence, the determiner "every" and the quantifier "everyone" play a crucial role in determining the linguistic structures that belong to the natural language fragment corresponding to a DL-lite TBox.

We have to zoom into the `NOUN` and `VERB_PHRASE` constituents.

In other words, we spell out how the Cl and Cr of DL-lite can be expressed in English.

10. Questions

- Can we be satisfied?
- Can we do more, and define a grammar that recognizes "all and only" linguistic structures whose meaning representation is in DL-lite?
- But how can we define the "all"?
- Would a user be happy in using a Controlled Natural Language?
- How far is this CNL from the sentences that a user would naturally use to access Information Systems?
- Would we ever be able to bridge this gap?

10.1. Strategies

- Analyze corpus of questions to DB

  We have looked at `Geo880` (set of 880 queries to a US geography).

  Most of the queries were conjunctive queries, but the one involving: (i) aggregation functions (highest, most, longest etc.), and (ii) counting (how many, higher, etc) but the latter could be handled in some restricted form.

- Built a grammar able to recognize only `CQs` while building their meaning representation.

- Try experiments to test user satisfiability to enter specifications in the ontology and query a DB.

- Study the literature on Text Simplification for e.g. people with aphasia. Aim: to re-write users' questions into simplified and suitable ones.

- "All" sentences in DL-lite . . . still a mystery.

11. Fun: Appelscript

```
tell application "Address Book"
set shortDate to short date string of date "16.04.08"
repeat with thePerson in (every person)
set theDate to modification date of thePerson
if short date string of theDate = shortDate then
add thePerson to group "telefonino"
end if
end repeat
save addressbook
end tell
```

12. Conclusion

13.1. Chomsky's Syntactic Structure

The preface of "Syntactic Structures" emphasizes the **heuristic role of formalization** in clarifying linguistic analyses, supporting empirical testing and falsification:

> " ... The search for rigorous formulation in linguistics has a much more serious **motivation** than mere concern for logical niceties or the desire to purify well-established methods of linguistic analysis. Precisely constructed models for linguistic structure can play an important role, both negative and positive, in the **process of discovery itself**. By pushing a precise but inadequate formulation to an unacceptable conclusion, we can often expose the exact source of this inadequacy and, consequently, **gain a deeper understanding of the linguistic data**. More positively, a formalized theory may **automatically provide solutions** for many problems other than those for which it was explicitly designed. Obscure and intuition-bound notions can neither lead to absurd conclusions nor provide new and correct ones, and hence they fail to be useful in two important respects."

13.2. Generative grammar

A context-free component, generating "kernel sentences", and a transformation component (cf. Harris (1957)). Two used approaches have been

- (A) Generate a (finite) set of **elementary sentences**, and use transformations to broaden it to the class of representations of all sentences for a language.
- (B) Generate a (finite) set of representations of **all sentences** of a language, and then use transformations to arrive at surface forms.

Variant (B) lead to stratificational grammar,

- Stratificational grammar, cf. e.g. (Hays, 1964; Lamb, 1966).
- Chomsky's (1965) "Aspects of the Theory of Syntax" adopts (B), and would later develop into "Government & Binding theory" (Chomsky, 1981), cf. (Haegeman, 1991; Higginbotham, 1997)

13.3. Early non transformational approach

The landscape of formal grammar was not covered solely by generative (transformational) approaches.

- Bar-Hillel focused primarily on categorial grammar (Bar-Hillel, 1953), elaborating Ajdukiewicz's (1935) syntactic calculus, though provided with his **algebraic linguistics** (Bar-Hillel, 1964) a notion that was intended to cover a broader range of approaches to formal description of grammar.
- Lambek (1958; 1961) similarly focused on categorial grammar, though of a more logical (**proof-theoretical**) kind than Bar-Hillel's.

14. Meaning entered the scene

Chomsky was, in general, **sceptical of efforts to formalize semantics**. Interpretative semantics or the autonomy of syntax: Syntax can be studied without reference to semantics (cf. also Jackendoff).

Criticism on both transformational and non-transformational approaches:

- Transformations do not correspond to syntactic relations, relying too much on linear order.
- Similarly, Curry (1961; 1963) criticized Lambek for the focus on order (directionality).

14.1. Different ongoing efforts

- Developing a notion of (meaningful) logical form, to which a syntactic structure could be mapped using transformations. Efforts either stayed close to a constituency-based notion of structure, like in generative semantics (Fodor, Katz), or were dependency-based (Sgall et al, particularly Panevová (1974; 1975); Fillmore (1968)). Cf. also work by Starosta, Bach, Karttunen.
- Montague's formalization of semantics – though Montague and the semanticists in linguistics were unaware of one another, cf. (Partee, 1997)

14.2. Montague and the development of formal semantics

The foundational work by Frege, Carnap, and Tarski had led to a rise in work on modal logic, tense logic, and the analysis of **philosophically interesting issues in natural language**. Philosophers like Kripke and Hintikka added model theory.

These developments went hand-in-hand with the **logical syntax** tradition (Peirce, Morris, Carnap), distinguishing syntax (well-formedness), from semantics (interpretation), and pragmatics (use).

Though the division was inspired by language, **few linguists attempted to apply the logician's tools in linguistics as such**.

This changed with **Montague**.

> "I reject the contention that an important theoretical difference exists between formal and natural languages." (Montague, 1974)(p.188)

A compositional approach, using a "rule-by-rule" translation (Bach) of a syntactic structure into a first-order, intensional logic. This differed substantially from transformational approaches (generative or interpretative semantics).

15. The trouble with Word Order

Traditional phrase-structure grammar (Bloomfield) is context-free (CFPSG) and therefore it's unable to account for **long-distance dependency** without an extra apparatus.

Chomsky (1957) therefore added transformations on top of a CFPSG.

But, there are both linguistic **problems with transformations** (no corresponding linguistic concept), and formal problems, (Peters & Ritchie (1971; 1973)).

Moreover, Chomsky's arguments against CFPSG (incapable of generalization, mathematical proof concerning string languages) were shown to be awed (e.g. by Gazdar, Pullum –as we have seen last time).

Finally, studies in **nonconfigurational** languages (e.g. Australian) starting in the 1970's gave rise to a more relational view on structure, in contrast to the configurationality of English.

These problems led to the development of new, non-transformational grammar frameworks like Relational Grammar and Arc Pair Grammar, Lexicalized Formal Grammar (LFG), Generalized Phrase Structure Grammars (GPSG). And further frameworks influenced by the latter, as Head-Driven Phrase Structure Grammar (HPSG).

15.1. Heads entered the scene

Heads (asymmetric relations) start entering the scene, in various guises.

Studies in nonconfigurational languages revealed that **relations** rather than phrases are typologically significant for the expression of meaning (cf. also (Bresnan, 2001)); Relational Grammar, Arc Pair Grammar, LFG.

The developments in GPSG, LFG, and Arc Pair Grammar showed the **feasibility of a nontransformational perspective**, employing a relational perspective to obtain better generalizations.

15.2. Heads enter the scene: Categorial Grammar

Related work was going on in categorial grammar (functional rather phrasal structure): (Venneman, 1977) binding dependency and functional structure – again, combining vertical and horizontal organization.

In general though, categorial grammar tried to deal with flexible word order by introducing means of composition that were more powerful than application:

- Bach's wrap operations (1984).
- Ades and Steedman's combinatorial rules (1982) (also Jacobson, Szabolcsi, and later Hoffman (1995), Steedman (1996; 2000)).
- Moortgat's generalized connectives (cf. (1988), also work by Oehrle, Morrill, Van Benthem).
- Only towards the end of the 1980's, early 1990's is dependency again explicitly introduced into categorial grammar: (Steedman, 1985; Hepple, 1990; Pickering, 1991; Moortgat and Morrill, 1991; Barry and Pickering, 1992; Moortgat and Oehrle, 1994).

15.3. Combining Constituency and Dependencies

In 1975, Joshi et ali introduced a grammatical formalism called Tree-Adjoining Grammars (TAGs), which are tree-generating systems. The application of TAGs to natural language is known as LTAGs.

- New way of thinking of domain of dependencies
- Localization of dependencies : elementary structures of a formalisms over which dependencies such as agreement, subcategorization and filler-gap relation can be specified.

15.4. Disadvantages of Transformational Grammars

Chomsky's Extended Standard Theory has been criticized from various perspectives.

- **Semantics**: Generative Semantics vs. interpretative semantics. Connections with the formal models developed from the work of Montague and with the proponents of Categorial Grammar. This line influences the development of Phrase Structure grammars such as GPSG and HPSG.
- **Psycholinguistics**: experiments did not show that the transformational model is a plausible one. It doesn't seem to be able to represent the competence of speakers. Within an unconstrained grammar, the analysis of a sentence becomes an indecidable problem. Development of more constrained theories of grammar such as LFG.
- **Representation**: Inadequacy of tree representation for non configurational languages: use of graphs or of features.
- **Implementation**: transformational grammars cannot be easily integrated in computational systems for the analysis of natural language.

16. Grammars meet Logic & ...

Logics to specify a grammar framework as a mathematical system:

- Feature logics: HPSG, cf. (King, 1989; Pollard and Sag, 1993; Richter et al., 1999)
- Categorial Type Logics (Kurtonina, 1995; Moortgat, 1997)

Logics to interpret linguistically realized meaning:

- Montague semantics: used in early LFG, GPSG, Montague Grammar, Categorial Type Logic, TAG (Synchronous LTAG)
- Modal logic: used in dependency grammar frameworks, e.g. (Broeker, 1997; Kruijff , 2001).
- Linear logic: used in contemporary LFG, (Crouch and van Genabith, 1998).

17. .. Computation

Computation of linguistic structures

- Unification (constraint-based reasoning): LFG, HPSG, categorial grammar (UCG), dependency grammar (UDG, DUG, TDG)
- "Parsing as deduction": CTL
- Optimality theory: robust constraint-solving, e.g. LFG

17.1. Unification

The development of Unification Grammars has strongly been influenced by the:

- use of tools developed in Logics and in AI;
- the progress made in the area of Natural Language Processing;
- Development of Logic Programming: Prolog.

    1. Declarative character: grammar is not a set of rules, but a set of constraints that a sequence needs to satisfy in order for it to be a grammatical phrase.
    2. Constraints do not need to be ordered.

Transformational grammars are inadequate if faced with implementation problems. Derivations proceed from deep structures while automatic sentence analysis requires the inverse process.

17.2. Importance of Unification Grammars

Unification grammars or constraint based grammars represent the new syntactic models of the 80's.

Four models which are representative of this trend are: Lexical Functional Grammar (LFG) (Bresnan 1982), Generalized Phrase Structure Grammar (GPSG) (Gazdar et al. 1985), Head-driven Phrase Structure Grammar (HPSG) (Pollard Sag 1987, 1994), Tree Adjoining Grammar (TAG) (Joshi et al. 1991).

They are grammar models which try to find an explicit division of labour among the **lexicon, syntax and semantics**.

They are **based on logical models** which are well studied and for which programming techniques have been developed.

They represent an adequate compromise between linguistic expressivity and possibility of implementation.

17.3. Common aspects

These four models have some common properties:

1. surface based;
2. use of complex features to develop syntactic descriptions;
3. definition of general principles of grammaticality;
4. integration of lexicon, syntax and semantics.

Rewrite rules are not interepreted in a procedural way, but as a description of well formed syntactic structures.

17.4. Approaches to rewrite rules

Lexical rules replace certain transformations The various models have different approaches with respect to rewrite rules:

- in LFG there is a parallel level of syntactic representation which is called functional structure;
- in HPSG rewrite rules are replaced by typed feature structures which undergo general principles with respect to the distribution of features;
- in TAG rewrite rules are replaced by elementary trees which are directly associated with the lexical items and combined among each other by specific operations among which is unification

18. Criticisms to Formal Grammars

General criticism:

- formal grammars have the advantage of being suitable for implementation, but are not interesting from a cognitive point of view.
- They make a sharp division between grammatical and ungrammatical sentences while this division is normally less sharp.
- It is possible to establish degrees of violation of grammatical principles, which constitute the basis for psycholinguistic hypotheses (Fodor 1983).

In the various models certain principles are formulated to define which features are appropriate, how they cooccur and how they propagate.

The formulation of principles make these models more plausible from a cognitive point of view.

19. Grammars meeting Statistics

If you are interested read:

> Lillian Lee “I’am sorry Dave, I am afraid I can’t do that: Linguistics, Statistics and NLP circa 2001*”