The Interaction between Linguistics and Computational Linguistics — A ParGram Perspective

Miriam Butt

University of Konstanz

Celebratory Workshop "20 Years of the Sektion Computerlingustik"
February 26, 2008

The Parallel Grammar (ParGram) Project

Project Structure

- Loose federation of partners/sites.
- No common deliverables, but common interests.
- Linguistic Theory: Lexical-Functional Grammar (LFG)
- Development Platform: XLE, Finite-State Morphology
- Languages: Arabic, Chinese, Danish, English, French, German, Japanese, Malagasy, Norwegian, Turkish, Urdu, Welsh.

Brief Project History

- 1994–1999: English, French, German
  - Solidified grammatical analyses and conventions
  - Developed, expanded and hardened XLE
  - Integrated Finite-State Morphologies
- 1999–2000: Norwegian, Japanese
  - Machine Translation Component, i.e. the XFR rewrite system (Frank 1999)
  - Integration of a Version of Optimality Theory (OT) (Frank et al. 2001)
- 2002: Danish
- 2005: Welsh, Malagasy, Arabic
- 2006: Turkish, Chinese
  - Begin integrating knowledge representation/ontologies

The Parallel in ParGram

- Analyze languages to a degree of abstraction that reflects the common underlying structure (i.e., identity of the subject, the object, the tense, mood, etc.).
- Even at this level, there is usually more than one way to analyze a construction.
    - The same theoretical analysis may have different possible implementations.
    - The implementational possibilities may give rise to new theoretical ideas.
- The ParGram Project decides on common analyses and implementations (via meetings and the feature committee).

The Parallel in ParGram

- Analyses at the level of c(onstituent)-structure are allowed to differ (variance across languages).
- Analyses at f(unctional)-structure are held as parallel as possible across languages (crosslinguistic invariance).
- `[Demo: ``The dog was bitten.'']`
- **Theoretical Advantage:** This models the idea of Universal Grammar.
- **Applicational Advantage:** Machine Translation, Question-Answering is made easier, applications are more easily adapted to new languages.

Grammar Components

Each Grammar Contains:

- Annotated Phrase Structure Rules (S $\longrightarrow$NP VP)
- Lexicon (verb stems and functional elements)
- Finite-State Morphological Analyzer and Tokenizer
- A version of Optimality Theory (OT) — used as a filter to restrict ambiguities and/or parametrize the grammar.
- Interface to statistical preferences.

Interactions between Computational and Theoretical Perspectives

Within ParGram there is a rich culture of discussion and experimentation with differing implementations/architectures.

- Grammar Development experience shows up interactions between analyses and components that remain obscured for the pen-and-paper linguist.
- Discussion of parallel analyses across languages weighs theoretical and computational elegance, adequacy and sheer possibility.

Interactions between Computational and Theoretical Perspectives

Two Examples of Interaction between Computational and Theoretical Perspectives:

1. The Morphology-Syntax Interface
2. The Syntax-Semantics Interface

LFG's Basic Architecture and Design

- Lexical-Functional Grammar (LFG) is a Theory of **Syntax**.

LFG's Basic Architecture and Design

- Lexical-Functional Grammar (LFG) is a Theory of **Syntax**.
- Other modules of grammar are assumed to be independent.

LFG's Basic Architecture and Design

- Lexical-Functional Grammar (LFG) is a Theory of **Syntax**.
- Other modules of grammar are assumed to be independent.
- The other modules are accessed via interfaces.

LFG's Basic Architecture and Design

- Lexical-Functional Grammar (LFG) is a Theory of **Syntax**.
- Other modules of grammar are assumed to be independent.
- The other modules are accessed via interfaces.
  - Phonology — not much done (Butt and King 1998)

LFG's Basic Architecture and Design

- Lexical-Functional Grammar (LFG) is a Theory of **Syntax**.
- Other modules of grammar are assumed to be independent.
- The other modules are accessed via interfaces.
    - Phonology — not much done (Butt and King 1998)
    - Morphology

LFG's Basic Architecture and Design

- Lexical-Functional Grammar (LFG) is a Theory of **Syntax**.
- Other modules of grammar are assumed to be independent.
- The other modules are accessed via interfaces.
  - Phonology — not much done (Butt and King 1998)
  - Morphology
    - grammars use a FST interface based on Karttunen, Kaplan and Kay's ideas.

LFG's Basic Architecture and Design

- Lexical-Functional Grammar (LFG) is a Theory of **Syntax**.
- Other modules of grammar are assumed to be independent.
- The other modules are accessed via interfaces.
  - Phonology — not much done (Butt and King 1998)
  - Morphology
    - grammars use a FST interface based on Karttunen, Kaplan and Kay's ideas.
    - the morphology interface works extremely well for large-scale grammars (some problems still in derivational morphologhy)

LFG's Basic Architecture and Design

- Lexical-Functional Grammar (LFG) is a Theory of **Syntax**.
- Other modules of grammar are assumed to be independent.
- The other modules are accessed via interfaces.
  - Phonology — not much done (Butt and King 1998)
  - Morphology
    - grammars use a FST interface based on Karttunen, Kaplan and Kay's ideas.
    - the morphology interface works extremely well for large-scale grammars (some problems still in derivational morphologhy)
  - Semantics — most work done on this interface, but only now beginning to work well

The Morphology-Syntax Interface

- LFG was first formulated as a theory of syntax (Kaplan and Bresnan 1982)
- It was deliberately agnostic about the organization of a reasonable theory of morphology.
- Spencer and Sadler (2001) critiqued current notions of the morphology-syntax interface within LFG. They argued that:
  - No reasonable approach to morphology existed within LFG.

The Morphology-Syntax Interface

- LFG was first formulated as a theory of syntax (Kaplan and Bresnan 1982)
- It was deliberately agnostic about the organization of a reasonable theory of morphology.
- Spencer and Sadler (2001) critiqued current notions of the morphology-syntax interface within LFG. They argued that:
  - No reasonable approach to morphology existed within LFG.
  - The only reasonable theoretical approach to morphology is Stump's Realizational Morphology (Paradigm-Function Morphology; Stump 2001).

The Morphology-Syntax Interface

- LFG was first formulated as a theory of syntax (Kaplan and Bresnan 1982)
- It was deliberately agnostic about the organization of a reasonable theory of morphology.
- Spencer and Sadler (2001) critiqued current notions of the morphology-syntax interface within LFG. They argued that:
  - No reasonable approach to morphology existed within LFG.
  - The only reasonable theoretical approach to morphology is Stump's Realizational Morphology (Paradigm-Function Morphology; Stump 2001).
  - Therefore this theory should be adopted within LFG.

The Morphology-Syntax Interface

The claim that LFG contained no workable approach to morphology was astonishing for the ParGram community.

- ParGram integrated Finite-State Morphologies in 1996 (cf. Butt et al. 1999).

`Demo`

The Morphology-Syntax Interface

The claim that LFG contained no workable approach to morphology was astonishing for the ParGram community.

- ParGram integrated Finite-State Morphologies in 1996 (cf. Butt et al. 1999).
- The morphologies are powerful and fast/efficient.

`Demo`

The Morphology-Syntax Interface

The claim that LFG contained no workable approach to morphology was astonishing for the ParGram community.

- ParGram integrated Finite-State Morphologies in 1996 (cf. Butt et al. 1999).
- The morphologies are powerful and fast/efficient.
- They not only deal with every piece of morphology in the language, but also allow for unknown words.

`Demo`

Computational vs. Theoretical

Still: The Computational Linguists were ready to grant that from a theoretical point of view, the ParGram approach to the morphology-syntax interface might not be satisfactory.

But: The closer one looked at Stump's (2001) desiderata, the clearer it became that the finite-state approach had already anticipated and implemented them.

Question: Why weren't theoretical morphologists reading Koskenniemi (1983), Kaplan and Kay (1994), Butt et al. (1999), Beesley and Karttunen (2003), among others?

Obvious Answer: Not considered theoretically relevant.

The Basic Criticism — Morphemes

- Classic LFG treats morphemes as form-content pairs. This has been shown to be untenable in the morphological literature.

(1) -s V_{infl} $(\uparrow \text{TENSE}) = \text{PRES}$
$(\uparrow \text{SUBJ PERS}) = 3$
$(\uparrow \text{SUBJ NUM}) = \text{SG}$
(Börjars, Vincent and Chapman 1997:163)

- Reasons:
  - Non-concatenative morphology (phonological expression, dependence of morphemes one one another, infixation, etc.)
  - Null morphemes
  - Different versions of a morpheme
- (Inflectional) morphology must instead be associated with a set of abstract morphological features (and one should not talk about morphemes).

The Basic Criticsim — Paradigms

- The notion of a *paradigm* finds no formal or theoretical realization within LFG.
- So there is no good way to deal with paradigmatic (ir)regularities (suppletion, periphrastic forms).

(2)

	Tense/Aspect	**Voice**	
		Active	**Passive**
(imperfective)	present	laudat	laudātur
	imperfect	laudābat	laudābātur
	future	laudābit	laudābitur
(perfective)	perfect	laudāvit	laudātus est
	pluperfect	laudāverat	laudātus erat
	future perfect	laudāverit	laudātus erit

Third singular of Latin *laudāre* 'to praise'

Realizational Morphology

- In Paradigm-Function Morphology (Stump 2001) the architecture underlying the relationship between the surface string and a given feature bundle can be visualized as a process of *realization*.
- Stump formulates the realization patterns/rules via a notation that is compatible with DATR, a system which uses default inheritance hierarchies non-monotonically.
- Abstracting away from the particular implementation, one can see Stump's formulations as being about the definition of a general realization relation *R*.

(3) **Example:**
$< \{3, \text{Sg}, \text{walk}, \text{Pres}\}, /\text{walks}/ >$

Articulating the ParGram Theory

- A close look at the ParGram architecture shows that it already incorporates the realizational relation argued for by Stump.
- Moreover, it has formal and theoretical advantages over Stump's approach:
  - The architecture is formally clean and better defined: the interface is implemented via formally well-understood finite-state machines in combination with the formally well-defined theory of LFG.
  - The implemented morphologies cover almost **all** of the morphological phenomena of a language, not just fragments.

Articulating the ParGram Theory

- In contrast to Stump, who assumes a simple relation R, the architectural design of LFG presupposes a complex relation *R* which consists of several subrelations (Butt and Kaplan 2002).
- In light of how the architecture of LFG works, it is natural to divide *R* into two parts:
  1. A "Satisfaction" relation *Sat* that holds between an f-structure and an f-description
  2. a "Description" relation *D* that holds between a description and a string (of phonemes or symbols, roughly: a word)

Articulating the ParGram Theory

Sat is acutally just the normal 'Satisfaction" relation that holds between an f-structure and an f-description (cf. Kaplan and Bresnan 1982)

(4)

f-description

$$(f_1 \text{ SUBJ}) = f_2$$
$$(f_2 \text{ SPEC}) = a$$
$$(f_2 \text{ NUM}) = sg$$
$$(f_2 \text{ PRED}) = \textit{'girl'}$$

f-structure

$$\left[\text{SUBJ} \quad \left[\begin{array}{ll} \text{PRED} & \textit{'girl'} \\ \text{SPEC} & \text{a} \\ \text{NUM} & \text{sg} \end{array} \right] \right]$$

Articulating the ParGram Theory

The "Description" relation D holds between a description and a string (of phonemes or symbols, roughly: a word)

- The description is a Boolean combination of primitive constraints.
- Mathematically, the relation D is just a set of ordered pairs: $<$f-description, sequence$>$

- **Example:**

  $< \{ (f_1 \text{ PRED}) = \text{'walk}<(f_1 \text{ SUBJ})>\text{'}, (f_1 \text{ SUBJ PERS})=3, (f_1 \text{ SUBJ NUM}) = \text{sg}, (f_1 \text{ TNS-ASP TENSE}) = \text{pres} \}, /\text{walks}/ >$

Articulating the ParGram Theory

So far, this gives us:

(5) f [Sat ∘ D] w

Butt and Kaplan (2002) further break down the Description relation into two parts.

1. a "Lexical" relation *L* that maps between description-formulas and sets of description-names (D-names)
2. the "Sequence" relation *Seq* which maps between sets of D-names and the sequences of characters/sounds that make up words

Articulating the ParGram Theory

The "Lexical" relation *L* maps between description-formulas and sets of description-names (D-names), atomic symbols that serve to identify those formulas.

(6)

walk:	$(\uparrow \text{PRED}) = \text{'walk}<(\uparrow \text{SUBJ})>\text{'}$
3:	$(\uparrow \text{SUBJ PERS})=3$
Sg:	$(\uparrow \text{SUBJ NUM}) = \text{sg}$
Pres:	$(\uparrow \text{TNS-ASP TENSE}) = \text{pres}$

- The use of D-names (atomic symbols) makes it easy to write down descriptions in a database, retrieve them, and generally keep them organized.
- The choice of the names is arbitrary—-for the sake of convenience we chose mnemonic terms. These have no formal significance.

Articulating the ParGram Theory

- The "Sequence" relation *Seq* maps between sets of D-names and the sequences of characters/sounds that make up words (generally: X^0).

  (7) **Example:**
  $< \{3, \text{Sg}, \text{walk}, \text{Pres}\}, /\text{walks}/ >$

- *Seq* typically relies on phonological rules, morphemic analysis and a paradigmatic organization of affixes and stems.

The morphology-syntax interface is thus realized by a complex relation *R* consisting of a composition of the relations *Sat, L* and *Seq*.

Regular Relations

- A considerable amount of mathematical, linguistic, and computational work has shown that Sequence relations as defined by several different kinds of natural linguistic-rule formalisms belong to the mathematical class of regular relations (e.g., Kaplan and Kay 1994).
- Independently of the rules by which they might be defined (e.g., Two Level rules or ordered rewriting rules), regular relations are just those sequential mappings that can be implemented by Finite-State Transducers (FST), and thus the mappings that they define can be computed very easily, efficiently, and effectively.

The ParGram Architecture

In integrating Finite-State Morphological Analyzers into the ParGram Architecture as shown above, the computational implementation thus:

- anticipated developments in morphological theory (cf. Beesley and Karttunen 2003 for other nice implementations/ideaas)
- provides a formally and theoretically more elegant approach than is available in the theoretical literature

Output of FST Morphological Analyzers
(Beesley and Karttunen 2003)

(8) walks
    1. walk +Verb +Pres +3sg
    2. walk +Noun +Pl

Communication

The Problem is: how can the insights from the computational literature be communicated to the theoretical linguists (computational linguists know they are supposed to read theory) (Kartunen 2003)?

> *Time after time, the computational knights have presented themselves at the Royal Court of Linguistics, rushed up to the Princess of Phonology and Morphology in great exitement to deliver the same message: Dear Princess. I have wonderful news for you: You are not like some of your NP-complete sisters. You are regular. You are rational. You are finite-state. Please marry me. Together we can do great things. And time after time, the put-down response from the Princess has been the same: Not interested. You do not understand Theory. Go away you geek.*

Communication

- With respect to the morphology-syntax interface, the discussions are continuing (and becoming more relevant in other grammar formalisms as well).

Communication

- With respect to the morphology-syntax interface, the discussions are continuing (and becoming more relevant in other grammar formalisms as well).
- Another topic: The Syntax-Semantics Interface

Communication

- With respect to the morphology-syntax interface, the discussions are continuing (and becoming more relevant in other grammar formalisms as well).
- Another topic: The Syntax-Semantics Interface
    - Interactions between

Communication

- With respect to the morphology-syntax interface, the discussions are continuing (and becoming more relevant in other grammar formalisms as well).
- Another topic: The Syntax-Semantics Interface
    - Interactions between
        - Theoretical Formal Semantics

Communication

- With respect to the morphology-syntax interface, the discussions are continuing (and becoming more relevant in other grammar formalisms as well).
- Another topic: The Syntax-Semantics Interface
    - Interactions between
        - Theoretical Formal Semantics
        - The Reality of Computational Implementational (Wide Coverage, Efficient Processing)

Communication

- With respect to the morphology-syntax interface, the discussions are continuing (and becoming more relevant in other grammar formalisms as well).
- Another topic: The Syntax-Semantics Interface
  - Interactions between
    - Theoretical Formal Semantics
    - The Reality of Computational Implementational (Wide Coverage, Efficient Processing)
  - Within LFG — begin of new effort at parallel semantics (ParSem).

History of Semantics within LFG

- First papers in the eighties: Halvorsen, Reyle

History of Semantics within LFG

- First papers in the eighties: Halvorsen, Reyle
- Two different architectural approaches right from the start.

History of Semantics within LFG

- First papers in the eighties: Halvorsen, Reyle
- Two different architectural approaches right from the start.
- Established early on: Two appealing ideas of Montague are not desirable in the context of LFG (Halvorsen 1987, 1988).

History of Semantics within LFG

- First papers in the eighties: Halvorsen, Reyle
- Two different architectural approaches right from the start.
- Established early on: Two appealing ideas of Montague are not desirable in the context of LFG (Halvorsen 1987, 1988).
  - Compositionality of Interpretation (instead go back to Frege's ideas, who does not assume this, cf. Blutner, Hendricks and de Hoop 2003).

History of Semantics within LFG

- First papers in the eighties: Halvorsen, Reyle
- Two different architectural approaches right from the start.
- Established early on: Two appealing ideas of Montague are not desirable in the context of LFG (Halvorsen 1987, 1988).
    - Compositionality of Interpretation (instead go back to Frege's ideas, who does not assume this, cf. Blutner, Hendricks and de Hoop 2003).
    - Completeness of Interpretation Process

History of Semantics within LFG

- First papers in the eighties: Halvorsen, Reyle
- Two different architectural approaches right from the start.
- Established early on: Two appealing ideas of Montague are not desirable in the context of LFG (Halvorsen 1987, 1988).
  - Compositionality of Interpretation (instead go back to Frege's ideas, who does not assume this, cf. Blutner, Hendricks and de Hoop 2003).
  - Completeness of Interpretation Process
- So LFG's approach to semantics is quite different/freer than that seen in GB/MP.

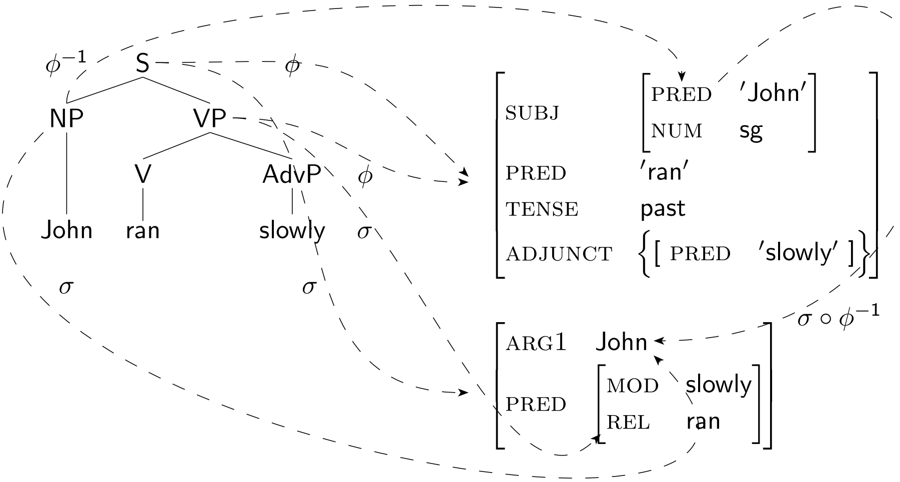
History of Semantics within LFG

Two Initial Appraoches:

- *Description by Analysis* (Halvorsen 1983, Reyle 1987)
  F-structures are mapped/transferred into semantic representations
- *Co-Description* (Halvorsen and Kaplan 1988)
  F-structures and S(emantic)-structures are both projected from the c-structure.

The co-description approach makes the most use of LFG's *projection architecture*.

Co-Description (Halverson and Kaplan 1988)



Scope

- Adverbial scope facts seemed to rule in favor of codescription.

Scope

- Adverbial scope facts seemed to rule in favor of codescription.
- Scope facts are encoded at c-structure, not f-structure

  an alleged former racketeer (Andrews and Manning 1999)

Scope

- Adverbial scope facts seemed to rule in favor of codescription.
- Scope facts are encoded at c-structure, not f-structure

  an alleged former racketeer (Andrews and Manning 1999)

- So it seemed more elegant/practical to project s-structure directly from the c-structure (and not refer to scope facts via inverse functions over the f-structure).

Distribution of Information

- Andrews and Manning (1999) see the problem as a difference of alignment in heads and propose an integrated f-structure (a bit like HPSG).

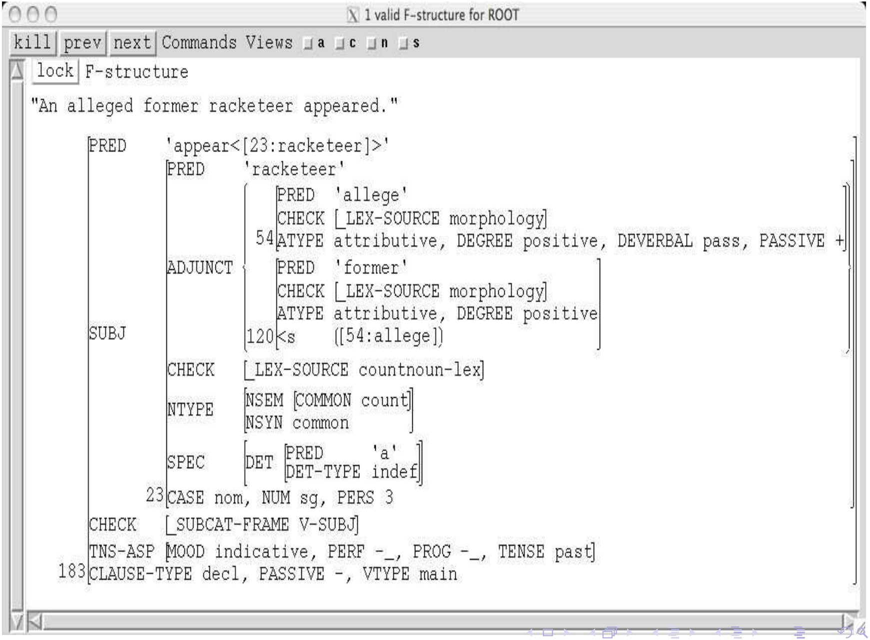
Distribution of Information

- Andrews and Manning (1999) see the problem as a difference of alignment in heads and propose an integrated f-structure (a bit like HPSG).
- Development within the ParGram project: encode scope facts as a diacritic at f-structure ($<$**s**).

Distribution of Information

- Andrews and Manning (1999) see the problem as a difference of alignment in heads and propose an integrated f-structure (a bit like HPSG).
- Development within the ParGram project: encode scope facts as a diacritic at f-structure ($<$**s**).
- This opens the door again for a direct relationship between f-structure and s-structure.

Scope (XLE/ParGram)



```
1 valid F-structure for ROOT
kill  prev  next  Commands  Views  a  c  n  s
lock  F-structure

"An alleged former racketeer appeared."

      PRED      'appear<[23:racketeer]>'
                PRED     'racketeer'
                            PRED   'allege'
                            CHECK  [LEX-SOURCE morphology]
                         54 ATYPE  attributive, DEGREE positive, DEVERBAL pass, PASSIVE +
                ADJUNCT  {  PRED   'former'
                            CHECK  [LEX-SOURCE morphology]
                            ATYPE  attributive, DEGREE positive
      SUBJ              120 <s     ([54:allege])
                CHECK    [LEX-SOURCE countnoun-lex]
                NTYPE    NSEM [COMMON count]
                         NSYN common
                SPEC     DET  PRED      'a'
                              DET-TYPE  indef
             23 CASE nom, NUM sg, PERS 3
      CHECK     [SUBCAT-FRAME V-SUBJ]
      TNS-ASP   [MOOD indicative, PERF -_, PROG -_, TENSE past]
  183 CLAUSE-TYPE decl, PASSIVE -, VTYPE main
```

Flat Semantics

- The experience within Verbmobil showed that a *flat semantics* was computationally the most useful.

Flat Semantics

- The experience within Verbmobil showed that a *flat semantics* was computationally the most useful.
  - Hole Semantics, a Version of Underspecified Discourse Representations (UDRS)

Flat Semantics

- The experience within Verbmobil showed that a *flat semantics* was computationally the most useful.
  - Hole Semantics, a Version of Underspecified Discourse Representations (UDRS)
  - Minimal Recursion Semantics (MRS)

Flat Semantics

- The experience within Verbmobil showed that a *flat semantics* was computationally the most useful.
  - Hole Semantics, a Version of Underspecified Discourse Representations (UDRS)
  - Minimal Recursion Semantics (MRS)
- van Genabith and Crouch (1996, 1997, 1999) show that f-structures are equivalent to quasi-logical forms (QLFs) and that they can be easily translated into UDRSs.

Flat Semantics

- The experience within Verbmobil showed that a *flat semantics* was computationally the most useful.
  - Hole Semantics, a Version of Underspecified Discourse Representations (UDRS)
  - Minimal Recursion Semantics (MRS)
- van Genabith and Crouch (1996, 1997, 1999) show that f-structures are equivalent to quasi-logical forms (QLFs) and that they can be easily translated into UDRSs.
- **Conclusion:** F-structures seem like the best starting point for deep semantic construction.

Current Developments

- **Goal:** Develop a Question-Answering System

Current Developments

- **Goal:** Develop a Question-Answering System
- For this one needs less of strategies dealing with quantifier scope, more of world knowledge and lexical semantics.

Current Developments

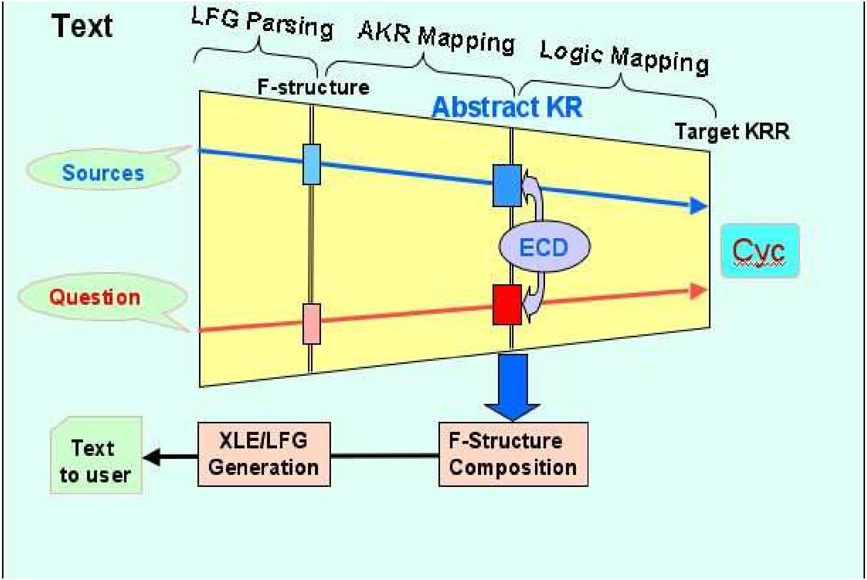
- **Goal:** Develop a Question-Answering System
- For this one needs less of strategies dealing with quantifier scope, more of world knowledge and lexical semantics.
- Crouch and King: The *Unified Lexicon* plus Abstract Knowledge Representation (AKR)
- The Unified Lexicon consists of combined lexical semantic information from:
  - The ParGram English grammar
  - Cyc (not releasable)
  - VerbNet (based on Levin's verb classes)
  - WordNet

The Unified (Semantic) Lexicon — Verbs

Crouch and King (2005)

- 9,835 different verb stems
- 46,000 verb entries (indexed by subcat/sense)
- 24,000 with VerbNet information
- 2,800 with Cyc information

Going Full Circle: Text-AKR-Text



(ECD = Entailment and Contradiction Detection)

AKR Architecture

- The current architecture eschews codescription.

AKR Architecture

- The current architecture eschews codescription.
- Instead, f-structures are rewritten to abstract knowledge representations by a set of powerful rewrite rules.

AKR Architecture

- The current architecture eschews codescription.
- Instead, f-structures are rewritten to abstract knowledge representations by a set of powerful rewrite rules.
- The rewrite rules were first developed for machine translation (Kay, Frank, Crouch, Kaplan).

AKR Architecture

- The current architecture eschews codescription.
- Instead, f-structures are rewritten to abstract knowledge representations by a set of powerful rewrite rules.
- The rewrite rules were first developed for machine translation (Kay, Frank, Crouch, Kaplan).
- Crouch (2006) and Crouch and King (2006) describe the on-going current effort.

AKR Architecture

- The current architecture eschews codescription.
- Instead, f-structures are rewritten to abstract knowledge representations by a set of powerful rewrite rules.
- The rewrite rules were first developed for machine translation (Kay, Frank, Crouch, Kaplan).
- Crouch (2006) and Crouch and King (2006) describe the on-going current effort.
- The rewrite rules are destructive, but can map back from AKR to f-structures to allow for generation from semantics (interlingua!).

Semantic Canonicalizations

- Canonicalizing syntax:
  - Depassivize: *A cake was eaten.*
  - Null pronouns:
    Going to the store, John took the bus.
    To open it, John broke the seal.
- Semantic rewrites
  - Coordination: *Mary and Jane hop. vs. Mary or Jane hops.*
  - Deverbal nouns:
    The Romans destroyed the city long ago.
    = *The Romans' destruction of the city long ago.*

Flattened contexts

- Syntactic embedded structures may correspond to *contexts*
  Mary knows that John left.
- Other elements may introduce contexts
  Mary did not leave.
  Mary certainly left.
  Mary certainly did not leave.

Concepts and Roles

- Derive roles from VerbNet
  - Thematic roles abstract from syntactic realization
  - Tie specific verb meaning to specific syntactic realization
  - Use heuristics to extend coverage for missing verbs or frames
- Derive concepts from WordNet
  - Hard wire certain concept (proper nouns, pronouns)
  - Future work on word sense disambiguation

From Semantics to AKR

- Determine (un)instantiability
  Mary managed to leave.
  Mary failed to leave.
  John knows that Mary left.
  John believes that Mary left.
- Temporal relations

Syntax of Rewrite Rules

Rule form

Input terms	$\longrightarrow$	Output terms	(obligatory)
Input terms	$?\!\longrightarrow$	Output terms	(optional)

Patterns

Consume term if matched:	Term
Test on term without consumption:	+Term
Test that term is missing:	-Term

Ordered Rule application

- Rule1 applied in all possible ways to Input to produce Output1
- Rule2 applied in all possible ways to Output1 to produce Output2

Entailment and Contradiction Detection (ECD)

```
AKR: A little girl disappeared.

context(t),
instantiable(disappear4,  t)
instantiable(girl3,  t)
temporalRel(startsAfter, Now, disappear4)
role(Theme,  disappear4,  girl3)
role(cardinality_restriction, girl3, sg)


subconcept(disappear4,
           [[422658], ..., [220927]])
subconcept(girl3,
           [[9979060...1740],
            [9934281...9771976...1740],
            ..., [9979646...1740]])
```

→

```
AKR: A child vanished

context(t),
instantiable(vanish2,  t)
instantiable(child1,  t)
temporalRel(startsAfter, Now, vanish2)
role(Theme, vanish2, child1)
role(cardinality_restriction, child1, sg)


subconcept(vanish2,
           [[422658], ..., [2136731]])
subconcept(child1,
           [[9771320, ...1740],
            [9771976, ...1740],
            ..., [9772490, ..., 1740]])
```

(**Note:** The numbers in square brackets are pointers to WordNet concepts.)

Text-AKR-Text

Demo of the System

Summary

- The original theoretical ideas on semantics within LFG are not being consistently pursued anymore.
- Rather, from the “bottom-up” (need driven) a new method for modeling semantic information is being developed.
- But this new method is informed by fundamental insights from formal semantic analyses.

Summary

- Interactions between Computational and Theoretical Linguistics are fruitful.
- There should be greater awareness of problems and solutions
  - across theories
  - across implementations
  - across theories and implementations

References I

Beesley, Kenneth and Lauri Karttunen. 2003. *Finite State Morphology*. CSLI Publications.

Butt, Miriam and Ronald M. Kaplan. 2002. The morphology-syntax interface in LFG. In *On-Line Proceedings of the LFG02 Conference*. Athens.

Butt, Miriam, Tracy King, Maria-Eugenia Niño, and Frédérique Segond. 1999. *A Grammar Writer's Cookbook*. CSLI Publications.

Butt, Miriam and Tracy Holloway King. 1998. Interfacing phonology with LFG. In M. Butt and T. King, eds., *Proceedings of the LFG98 Conference*. http://csli-publications.stanford.edu/LFG/.

Crouch, Richard. 2006. Packed rewriting for mapping text to semantics and kr. In M. Butt, M. Dalrymple, and T. King, eds., *Intelligent Linguistic Architectures: Variations on Themes by Ronald M. Kaplan*. CSLI Publications.

Crouch, Richard and Tracy Holloway King. 2005. Unifying lexical resources. In *Proceedings of the Interdisciplinary Workshop on the Identification and Representation of Verb Features and Verb Classes*. Saarbrücken.

Crouch, Richard and Tracy Holloway King. 2006. Semantics via f-structure rewriting. In *On-Line Proceedings of the LFG06 Conference*. To Appear.

Frank, Anette. 1999. From parallel grammar development towards machine translation. In *Proceedings of MT Summit VII*, pages 134–142.

Frank, Anette, Tracy King, Jonas Kuhn, and John T. Maxwell III. 2001. Optimality Theory style constraint ranking in large-scale LFG grammars. In P. Sells, ed., *Formal and Empirical Issues in Optimality Theory*, pages 367–397. CSLI Publications.

Kaplan, Ron and Joan Bresnan. 1982. Lexical-Functional Grammar: A formal system for grammatical representation. In J. Bresnan, ed., *The Mental Representation of Grammatical Relations*, pages 173–281. Cambridge, MA: MIT Press.

References II

Kaplan, Ronald M. and Martin Kay. 1994. Regular models of phonological rule systems. *Computational Linguistics* 20(3):331–378.

Kartunen, Lauri. 2003. Computing with realizational morphology. In *Computational Linguistics: Lecture notes in computer science*. Berlin: Springer Verlag.

Koskenniemi, Kimmo. 1983. Two-level morphology. Publication 11, University of Helsinki, Department of General Linguistics, Helsinki, Finland.

Spencer, Andrew and Louisa Sadler. 2001. Syntax as an exponent of morphological features. In G. Booij, ed., *Yearbook of Morphology*. Dordrecht: Kluwer Academic Publishers.

Stump, Gregory T. 2001. *Inflectional Morphology: A Theory of Paradigm Structure*. Cambridge: Cambridge University Press.