Motivation for LFG

- **Lexical** = (not transformational) richly structured lexicon, where relations between, e.g., verbal alternations, are stated
- **Functional** = (not configurational) abstract grammatical functions like subject and object are primitives, i.e., not defined by the phrase structure configurations

LFG in a nutshell

LFG (minimally) distinguishes two kinds of representation:

- **c-structure** (constituent structure): overt linear and hierarchical organization of words into phrases
- **f-structure** (functional structure): abstract functional organization of the sentence, explicitly representing syntactic predicate-argument structure and functional relations

These are two separate levels of representation and formalisms: trees (c-structure) and attribute-value matrices (f-structure).

- Use of both structures allows, e.g., information from different parts of the tree to connect to the same function
- A range of other levels have been proposed, e.g., A-structure and σ-structure.
Part I: C-structure

C-structure corresponds to a fairly traditional notion of phrase structure.

- X-Bar Theory: heads with complements, adjuncts, specifier
- Categories: lexical (N, P, V, A, Adv) and functional (I, C) categories—not universally fixed

Slightly different notions:
- Endocentric category S: has no lexical head (for “internal subject” languages)
- Optionality: all constituent structure positions can be considered optional

Example

Example of c-structure

(1) kogda rodilsja Lermontov?
   when born Lermontov
   ‘When was Lermontov born?’
C-structure rules

- C-structure rules are essentially CFG rules, but:
  - interpreted as node admissibility conditions,
  - i.e., trees must meet c-structure rule descriptions
  - allow for regular expressions (Kleene star, disjunction, optionality, etc.) on the right-hand side
  - can be specified as ID/LP rules

ID/LP Rules

- Rules can be in ID/LP format (Gazdar et al., 1985, cf. also GPSG): ID = immediate dominance, LP = linear precedence
- Examples:
  1. No LP rules:
     a. VP $\rightarrow$ V, NP
     b. VP $\rightarrow$ \{V NP | NP V\}
  2. One LP rule:
     a. VP $\rightarrow$ V, NP $V < NP$
     b. VP $\rightarrow$ V NP
  3. Interacting LP rules:
     a. VP $\rightarrow$ V, NP, PP $V < NP, V < PP$
     b. VP $\rightarrow$ \{V NP PP | V PP NP\}

Part II: F-structure

- F-structure maps more closely to meaning and encodes abstract grammatical relations like subject & object as primitives, i.e. not reducible to tree structure.
- Motivation:
  - Study of grammatical relations predates modern linguistic theory
  - Categories like subject and object are cross-linguistic
  - languages vary less in their f-structure
  - e.g., Keenan-Comrie Hierarchy (for relative clause formation, passivization): SUBJ $>$ DO $>$ IO $>$ OBL $>$ GEN $>$ OCOMP
Grammatical functions

- **Governable functions:** SUBJ, OBJ, OBJ2, COMP, XCOMP, OBL
  - A predicate can govern these functions, i.e., subcategorize for them.
- **Non-governable functions:** ADJ, XADJ
  - ADJ: David devoured a sandwich yesterday.
  - XADJ: Having opened the window, David took a deep breath.
- **Discourse functions:** TOPIC, FOCUS
  - For topic-oriented languages (e.g., Russian)
  - Potentially useful for UDC handling

Governable & non-governable grammatical functions

- **Overview:**
  
<table>
<thead>
<tr>
<th>Governable</th>
<th>Non-governable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unrestricted</td>
<td>Restricted</td>
</tr>
<tr>
<td>SUBJ</td>
<td>OBL</td>
</tr>
<tr>
<td>OBJ</td>
<td>COMP</td>
</tr>
<tr>
<td>OBJ2</td>
<td>XCOMP</td>
</tr>
<tr>
<td>ADJ</td>
<td></td>
</tr>
<tr>
<td>XADJ</td>
<td></td>
</tr>
</tbody>
</table>

- **Semantically restricted = thematic restrictions (θ) can be placed on function**
  - OBLθ: thematically restricted oblique functions (e.g., English to-PPs)
- **Open clausal functions (no internal subject):** XCOMP, XADJ
  - COMP: sentential or closed (nonpredicative) infinitival complement
  - XCOMP: open (predicative) complement with subject externally controlled

Subcategorization

Subcategorization is done at f-structure

- **Verbs select for grammatical functions**
- **Use the PRED (predicate) feature to specify the semantic form, e.g.,**
  - *yawn:* PRED 'YAWN<subj>'
  - *hit:* PRED 'HIT<subj,OBJ>'
  - *give:* PRED 'GIVE<SUBJ,OBJ,OBJ_THEME>'
  - *eat:* PRED 'EAT<SUBJ,OBJ>'

  Notationally, you may also see, e.g., PRED 'YAWN<↑subj>'
F-structure representation: Simple F-structures

F-structure is a function from attributes to values

- For the proper noun David, pred and num are attributes;
  'DAVID' and sg are the corresponding values

\[(5) \begin{array}{l}
\text{pred} 'DAVID' \\
\text{num} \text{ sg}
\end{array}\]

- F-structures within f-structures: David yawned

\[(6) \begin{array}{l}
\text{pred} 'yawn<subj>' \\
\text{tense} \text{ past} \\
\text{subj}
\end{array}
\begin{array}{l}
\text{pred} 'David' \\
\text{num} \text{ sg}
\end{array}\]

F-structure features

- What sorts of features can be used?
  - Decision is up to the grammar writer.
  - Commonly used features in LFG include ASPECT, PRONTYPE, VFORM, cf. Dalrymple (2006, Table 2)

Note:

- LFG does not have a mechanism to define a set of features or values that must be included in an f-structure.
  - For example, one verb may define VFORM, while another might leave it undefined.
  - This is different from HPSG, as we'll see.

F-structure: Sets

- Values can be sets, in order to handle phenomena with an unbounded number of elements, e.g., adjuncts, coordinates

\[(7) \begin{array}{l}
\text{pred} 'yawn<subj>' \\
\text{tense} \text{ past} \\
\text{subj}
\end{array}
\begin{array}{l}
\text{pred} 'David' \\
\text{num} \text{ sg}
\end{array}
\begin{array}{l}
\text{adj}
\end{array}
\begin{array}{l}
\text{pred} 'quietly' \\
\text{pred} 'yesterday'
\end{array}\]
Introduction

F-structure: Attributes with Common Values

Attributes can share the same values, to describe phenomena such as raising.

(9) David seemed to yawn.

Licensing a string

Phenomena

References
Coherence

- Coherence: All governable grammatical functions present in the f-structure must be mentioned in the argument list of the predicate.
- Like completeness, but in the other direction.
- Example:
  (12) a. * David yawned the sink.
  
  \[
  \begin{array}{c}
  \text{pred}'yawn<subj>' \\
  \text{subj}[\text{pred}'David'] \\
  \text{obj}[\text{pred}'sink'] \\
  \end{array}
  \]

  Definition (Kaplan and Bresnan, 1995, p. 65):
  - An f-structure is locally coherent iff all the governable grammatical functions that it contains are governed by a local predicate.
  - An f-structure is coherent iff it and all its subsidiary f-structures are locally coherent.

Uniqueness (consistency)

- A third condition is often mentioned: **Uniqueness**
  - (Consistency)
    - Every attribute has a single value.
  - This does not need to be explicitly stated or enforced, though.
  - It follows from interpreting attributes as functions (cf. unification).
- Example: Ensuring subject-verb agreement:
  (13) a. * The boys yawns.
  
  \[
  \begin{array}{c}
  \text{pred}'yawn<subj>' \\
  \text{subj}[\text{pred}'boys'] \\
  \text{num sg/pl} \\
  \end{array}
  \]

  Definition:
  - In a given f-structure, a particular attribute may have at most one value.

Constraining f-structures

- We use functional equations (defining equations) on words and phrases to describe acceptable f-structures.
- F-description with a single equation:
  (14) \((g \text{ num}) = \text{ sg}\)
- Different f-structures which satisfy this f-description:
  (15) a. \([\text{num sg}]\)
  b. \([\text{num sg}]\)
Functional Constraints (formally)

(16) \( (fa) = v \) holds iff \( f \) is an f-structure, \( a \) is a symbol, and the pair \( (a, v) \in f \)

\( \Rightarrow \) The f-structure for an utterance is the minimal solution satisfying the constraints introduced by the words and phrase structure of the utterance.

- minimal solution: satisfies all constraints in the f-description and has no additional structure

Constraining equations

- Can also use constraining equations to check the properties of the minimal solution
  - For example, the subj of \( f \) must meet certain conditions: \( (f subj num) = c sg \)
  - Defining equations and constraining equations are similar enough to ignore the distinction in the following.

Functional constraints example

Lexical constraints:
- **John**
  - \( (g\ pred) = 'John' \)
  - \( (g\ num) = sg \)

- **runs**
  - \( (f\ pred) = 'run<subj>' \)
  - \( (f\ subj case) = nom \)
  - \( (f\ subj num) = sg \)

Phrasal constraints (more on this later):
- \( (f\ subj) = g \)
Combining lexical and phrasal constraints, we have:

- \((f\ subj = g)\)
- \((g\ pred) = 'John'\)
- \((g\ num) = SG\)
- \((f\ pred) = 'RUN<subj>''\)
- \((g\ case) = NOM\)
- \((g\ num) = SG\)

**Minimal solution:**

\[
\begin{array}{c}
(f) \\
\begin{array}{c}
(g) \\
\end{array}
\begin{array}{c}
(pred) = 'RUN<subj>'' \\
\end{array}
\begin{array}{c}
(subj) \\
\end{array}
\begin{array}{c}
(g) \\
\begin{array}{c}
(pred) = 'John' \\
\end{array}
\begin{array}{c}
(case) = NOM \\
\end{array}
\begin{array}{c}
(num) = SG \\
\end{array}
\end{array}
\end{array}
\]

We want more ways to define the set of acceptable f-structures

- Disjunction
- Negation
- Existential Constraints
- Optionality

**Disjunction:** different options can be used to satisfy an f-description

(17) I met/have met him.

- Lexical entry for **met**:
  - \((f\ pred) = 'MEET<OBJ>''\)
  - \[\{(f\ tense) = PAST \mid (f\ form) = PASTPART\}\]
**Negation**: an f-description is specified that cannot be true

(18) a. I know whether/if David yawned.
    b. You have to justify whether/*if your journey is really necessary.

⇒ if is not allowed with justify (unlike know)

- justify  V  (f comp compform) ≠ IF

**Existential Constraints**: an f-structure must have some attribute, but the value of that attribute is unconstrained.

(19) a. The man who yawns/yawned/will yawn.

⇒ In a relative clause, yawn must be tensed, but which tense is not important

- Relative clause constraint is simply: (f tense)
- Can also specify negative existential constraints, e.g., ¬ (f tense)

**Optionality**: an f-description may (or may not) be satisfied

(20) a. Juan compró un reloj a Pedro.
    Juan bought a watch PREP Pedro
    b. Juan le compró un reloj.
    Juan him bought a watch
    c. Juan le compró un reloj a Pedro.
    Juan him bought a watch PREP Pedro
    'Juan bought a watch for Pedro.'

By specifying the semantic information contributed by le as optional, both Pedro and le can appear in the same sentence.

- Pedro  N  (f pred) = 'Pedro'
- le  Pro  ((f pred) = 'pro')

**Note**: Which objects can be doubled or dropped in Spanish is complex and depends on the semantic role (e.g., the above case is for benefactive/experiencer objects).
Part III: How a string is licensed

- A context-free c-structure grammar licenses the c-structure of a string.
- The grammar is augmented with functional equations, which map the c-structure to an f-structure representation.
- A function $\phi$ (phi) maps the c-structure to the f-structure of a sentence.
  - It is a function, so each c-structure is related to only one f-structure (but not necessarily vice versa, i.e., it can be a many-to-one mapping).

\[
\begin{array}{c}
\text{yawned} \\
\phi(V) \\
\text{[pred 'yawn<subj>']}
\end{array}
\]

The Head Convention

Multiple c-structures can map onto the same f-structure → this allows nodes to inherit properties from their head.

\[
\begin{array}{c}
\text{yawned} \\
\phi(V) = \phi(V') = \phi(V) \\
\text{[pred 'yawn<subj>']}
\end{array}
\]

F-structure/C-structure Regularities

Can have set mappings for particular positions, e.g., the specifier of IP in English maps to subj, the same position in Russian maps to topic, and in Bulgarian to focus.

\[
\begin{array}{c}
\text{David} \\
\phi(NP) \\
\text{[pred 'David']}
\end{array}
\]

\[
\begin{array}{c}
\text{yawned} \\
\phi(VP) \\
\phi(V') \\
\phi(V)
\end{array}
\]

\[
\begin{array}{c}
\text{yawned} \\
\phi(IP) \\
\text{[pred 'yawn<subj>']}
\end{array}
\]

\[
\begin{array}{c}
\text{yawned} \\
\phi(N) \\
\text{[pred 'David']}
\end{array}
\]
Notation

A way to specify this constraint on the specifier of IP is the following:

(21) $\text{IP} \rightarrow \text{XP} \quad I^\prime$

\[ (\text{SUBJ}) = \downarrow \quad \uparrow = \downarrow \]

- This says: The value of SUBJ for XP’s mother is equal to XP’s f-structure.
- IP and I’ have the same f-structure.

Annotated Phrase Structure Rules

(22) $V' \rightarrow V \quad \text{NP}$
\[ \uparrow = \downarrow \quad (\text{OBJ}) = \downarrow \]

(23) $\text{VP} \rightarrow V \quad \text{NP} \quad \text{NP}$
\[ \uparrow = \downarrow \quad (\text{OBJ}) = \downarrow \quad (\text{OBJ}2) = \downarrow \]

(24) $\text{VP} \rightarrow V \quad \text{NP} \quad \text{PP}$
\[ \uparrow = \downarrow \quad (\text{OBJ}) = \downarrow \quad (\text{OBJ}2) = \downarrow \quad (\text{PFORM}) = $to$

Lexical Entries

Can use the same notation to express lexical entries

(25) a. $\text{yawned} \quad V \quad (\text{TPRED}) = '\text{wawn<subj>}'$
\[ (\text{TENSE}) = \text{past} \]

b. $\text{David} \quad N \quad (\text{TPRED}) = '\text{David}'$

The complete setup is best illustrated with an example.
Example grammar: C-structure rules with annotations
(based on Kaplan and Bresnan, 1995, pp. 40ff)

(26) a. $S \rightarrow NP \quad VP$
    $\quad \uparrow subj = \downarrow$

b. $NP \rightarrow Det \quad N$
    $\quad \uparrow = \downarrow \quad \uparrow = \downarrow$

c. $VP \rightarrow V \quad NP \quad NP$
    $\quad \uparrow = \downarrow \quad \{\uparrow obj\} = \downarrow \quad \{\uparrow obj2\} = \downarrow$

Example grammar: Lexicon

(27) a. $a \quad Det \quad \{\uparrow spec\} = a$
    $\quad \{\uparrow num\} = sg$

b. $girl \quad N \quad \{\uparrow num\} = sg$
    $\quad \{\uparrow pred\} = 'girl'$

c. $handed \quad V \quad \{\uparrow tense\} = past$
    $\quad \{\uparrow pred\} = 'hand' \quad \{\uparrow subj\}, \{\uparrow obj\}, \{\uparrow obj2\} >$

d. $the \quad Det \quad \{\uparrow spec\} = the$

e. $baby \quad N \quad \{\uparrow num\} = sg$
    $\quad \{\uparrow pred\} = 'baby'$

f. $toy \quad N \quad \{\uparrow num\} = sg$
    $\quad \{\uparrow pred\} = 'toy'$

A sentence licensed by the example grammar

```
\[\begin{array}{l}
(S) f/S \\
  \{\uparrow subj\} = f/N \\
  \{\uparrow obj\} = f/N \\
  \{\uparrow num\} = sg \\
  \{\uparrow pred\} = past \\
  \{\uparrow tense\} = past \quad 'hand' \quad \{\uparrow subj\} \quad \{\uparrow obj\} \\
  \{\uparrow obj2\} > \\
  (NP) f/N \\
  \{\uparrow num\} = sg \\
  \{\uparrow pred\} = 'girl' \\
  (Det) the \\
  \{\uparrow spec\} = the \\
  \{\uparrow num\} = sg \\
  \{\uparrow pred\} = past \quad 'hand' \quad \{\uparrow subj\}, \{\uparrow obj\}, \{\uparrow obj2\} > \\
  (V) handed \\
  \{\uparrow subj\} = \downarrow \\
  \{\uparrow obj\} = \downarrow \\
  \{\uparrow obj2\} > \\
  (N) girl \\
  \{\uparrow spec\} = a \\
  \{\uparrow num\} = sg \\
  \{\uparrow pred\} = 'girl' \\
  (Det) a \\
  \{\uparrow spec\} = a \\
  \{\uparrow num\} = sg \\
  \{\uparrow pred\} = 'toy' \\
  \{\uparrow tense\} = past \\
  \{\uparrow pred\} = past \quad 'hand' \quad \{\uparrow subj\}, \{\uparrow obj\}, \{\uparrow obj2\} > \\
  \{\uparrow obj\} = \downarrow \\
  \{\uparrow obj2\} = \downarrow \\
  \end{array}\]
```
**Example**

The resulting f-structure for the example sentence:

```
    Σj  f[
        subj  f[
            spec a
            num sg
            pred 'girl'
        ]
        tense past
        pred 'hand < (^subj), (^obj), (^obj2) >'
        obj  f[
            spec the
            num sg
            pred 'baby'
        ]
        obj2 f[
            spec a
            num sg
            pred 'toy'
        ]
    ]
```

**Computational issues**

Processing c-structure by itself is essentially equivalent to processing CFGs, which is very efficient.

How does one account for f-structures?

- Can be interleaved, which requires sophisticated algorithms to do this efficiently.
- Can post-process c-structures with f-structure constraints.

It has been shown that if an f-structure is acyclic, the set of strings it corresponds to are equivalent to a context-free language.

- This can help constrain both parsing and generation.

**Part IV: Phenomena**

- Head mobility
- Passives
- Unbounded dependency phenomena (Extraction)
- Nonfinite Constructions
Head mobility

Phenomenon: The position of a main verb w.r.t. adverbs depending upon the presence of an auxiliary

Solution: use different categories

(28) a. *mange* T (↑pred) = 'eat<(↑subj), (↑obj)' (↑tense) = present

b. *mangé* T (↑pred) = 'eat<(↑subj), (↑obj)'

c. *ai* T (↑tense) = present

---

Head mobility: main verb as T

```
TP
  NP
    T
      VP
        V
          AdvP
            V
              NP
                AdvP
                  V
                    NP
                      AdvP
                        V
                          NP
                            AdvP
                              V
                                NP
```

Head mobility: main verb as V

```
TP
  NP
    T
      VP
        V
          AdvP
            V
              NP
                AdvP
                  V
                    NP
                      AdvP
                        V
                          NP
```

Passives

Passives are handled entirely via the lexicon
- Lexical rules (or the like) relate one lexical entry to another, e.g., a + en rule:

(29) a. \( k\text{iss} \) \( V \) \((↑\text{pred}) = 'k\text{iss}<↑\text{subj}), \( (↑\text{obj}) \)' 

b. \((↑\text{subj}) \Rightarrow \emptyset, \( (↑\text{obj}) \Rightarrow (↑\text{subj}) \)

c. \( k\text{iss} \) \( V \) \((↑\text{pred}) = 'k\text{iss}<\emptyset (↑\text{subj})>\)'

Extraction: Functional uncertainty

Extraction is handled in LFG by functional uncertainty:
- a functional equation sets up a relation between some initial, extracted object with an OBJ grammatical function later in the sentence.

(30) \( CP \rightarrow XP \quad C' \)
\((↑\text{topic}) = \downarrow \quad (↑\text{topic}) = (↑\text{comp}^* \text{obj}) \quad ↑ = ↓ \)

This says that the topic element is equated with an OBJ found under some path of comp functions.

Extraction example

(31) What do you think Chris bought?

(32) The principle of completeness ensures that bought has a realized object, and the functional equation fills it in.
(33) What do you think Chris hoped David bought?

(34)

(35) John tries to smile.
(36) John seems to smile.

- Equi: Verb embedding infinitive assigns two thematic roles.

(37) \( \text{tries} \ V \ (↑\text{pred}) = '\text{try}<↑\text{subj}, \text{xcomp}>' \)

(\(↑\text{subj}) = (↑\text{xcomp subj})

- Raising: Verb embedding infinitive assigns one thematic role.

(38) \( \text{seems} \ V \ (↑\text{pred}) = '\text{seem}<↑\text{xcomp}>' \)

(\(↑\text{subj}) = (↑\text{xcomp subj})

Nonfinite constructions (cont.)

Depending on the interface to semantics employed, sometimes the functions not assigned to a semantic role are notated after the angled brackets<>:

(39) \( \text{seems} \ V \ (↑\text{subj}) = '\text{seem}<↑\text{xcomp}>' \)

(\(↑\text{subj}) = (↑\text{xcomp subj})
Introduction

C-structure

F-structure

Licensing a string

Phenomena

References

Nonfinite constructions

