A derivational approach to phrasal spellout

Gregory M. Kobele

kobele@uchicago.edu
Computation Institute & Department of Linguistics
University of Chicago

BCGL 7, 2012
Brussels
What this talk is about

Hypothesis: “Interface Uniformity”

The interfaces to syntax have the same structure.

The differences between the syntax-semantics and the syntax-morphology interfaces lie only in the objects they are building.

Research Strategy

1. Study the structure of the syntax-semantics interface
2. Reflect these properties onto the syntax-morphology interface

Here we will look at idioms.
In a nutshell

Main Claim:
Morphology is insensitive to derived structure

*Morphology only cares about DS.*
Outline

The Syntax-Semantics Interface
  Direct Compositionality
  What is a derivation?
  Compositional Semantics
  Idioms

The Syntax-Morphology Interface

Conclusion
Basic Assumptions

Direct Compositionality

The syntactic derivation is a recipe for constructing a semantic representation.

Semantic differences must stem from differences in the way expressions are built, not from differences in the surface (or LF) structure.

Historical Antecedents

Generative Semantics

Deep structure is the structure interpreted.

Categorial Grammar

Surface structure [is] no more than the trace of the algorithm that delivers the [...] interpretation (Steedman, 2000)
**Direct Compositionality (in Minimalism)**

### Direct Compositionality
- Each lexical item has a semantic denotation
- Each operation (merge, move) is semantically interpretable;
  - [merge] combines two denotations into a single one
  - [move] maps a single denotation to another one

### Denotations
- Expressions are associated with (bounded size) quantifier stores
- (Bounded size) stores faithfully implement Heim & Kratzer-style LF-interpretation (Kobele; 2006, 2010)
- Can be encoded into the simply typed lambda calculus; (Kobele; 2012)
Direct Compositionality and Derivation Trees

Derivations are the structures interpreted

- We need a precise notion of what a derivation is.

Complete Decomposability

If a derivation $d = M[N]$, then $[[d]] = \lambda x. [[M[x]]] ([[N]])$

\[
\begin{pmatrix}
\begin{pmatrix}
\begin{pmatrix}
\end{pmatrix}
\end{pmatrix}
\end{pmatrix}
= \lambda x.
\begin{pmatrix}
\begin{pmatrix}
\begin{pmatrix}
\end{pmatrix}
\end{pmatrix}
\end{pmatrix}
\left(\begin{pmatrix}
\begin{pmatrix}
\begin{pmatrix}
\end{pmatrix}
\end{pmatrix}
\end{pmatrix}\right)
\]
Representing derivations

A derivation shows how a sentence is built up from the primitives of the grammar.

1. \([DP\ then\ NP\ boy]\) \(\text{merge}(\text{every, boy})\)
2. \([VP\ laugh\ DP\ every\ boy]\) \(\text{merge}(\text{laugh, #1})\)
3. \([IP\ will\ VP\ laugh\ DP\ every\ boy]\) \(\text{merge}(\text{will, #2})\)
4. \([IP[DP\ every\ boy][I\ will\ VP\ laugh\ t]]\) \(\text{move}(\#3)\)

Processes have structure

derivations like the above can be viewed as trees
Representing derivations

move the closest available thing (the DP every boy) to check the features of the current head (will)
The Syntax-Semantics Interface (I)

We typically see:

\[ \sigma(t_1, \ldots, t_n) = \sigma(\sigma(t_1), \ldots, \sigma(t_n)) \]

This conflates two kinds of information:

1. what the meanings of the formatives are
2. that there is an isomorphism between the syntactic structure and the semantic structure

Reformulated:

1. replace each formative \( \phi \) with its meaning \( \boxed{\phi} \)
2. interpret immediate dominance as (uncurried) function application:
   \[ f(t_1, \ldots, t_n) \leadsto f(t_1)(\ldots)(t_n) \]
The way to obtain a meaning from a structure is universal:

\[
\langle \sigma(t_1, \ldots, t_n) \rangle_f = f_{\sigma}(\langle t_1 \rangle_f) \cdots (\langle t_n \rangle_f)
\]

The language particular content of the interface:

a finite list of denotations for all formatives

- **merge** and **move**
- all lexical items

But are **[merge]** and **[move]** really language particular?

(No.)
Alternative Representations (‘Mirror Theory’)

MOVE

MERGE

will

MERGE

laugh

MERGE

every

boy
Alternative Representations (‘Mirror Theory’)
Alternative Representations (‘Mirror Theory’)

Diagram:
- "will"
- "laugh"
- "MERGE"
- "every"
- "boy"
Alternative Representations (‘Mirror Theory’)
The language particular content of the interface:

- a finite list $\Lambda$ of denotations for
  - all lexical items

A derivation is interpreted at the interface by replacing all of its formatives (lexical items) with their associated denotations.
Interpreting chunks

MOVE

MERGE

will

MERGE

MOVE

John

MERGE

v

MERGE

kick

MERGE

the

bucket
Interpreting chunks
Interpreting chunks

John will kick the bucket
Interpreting chunks

```
will
v
kick
merge
the
bucket
John
```
Interpreting chunks

will

v

kick

the

bucket

John

Kobele (U of C)  A derivational approach to phrasal spellout  BCGL 7, 2012  Brussels  14 / 38
Interpreting chunks

John will kick the bucket
Syntax-Semantics Interface (IV)

The language particular content of the interface:

- a finite list $\Lambda$ of denotations for
  - all derivation chunks

A derivation is interpreted at the interface by replacing all of its chunks with their associated denotations.
SyntSem Idioms

Syntax-Semantics Interface (IV)

The language particular content of the interface:

- A finite list $\Lambda$ of denotations for all derivation chunks

A derivation is interpreted at the interface by replacing all of its chunks with their associated denotations.

This is a contradiction!
Syntax-Semantics Interface (IV)

The language particular content of the interface:

- A finite list $\Lambda$ of denotations for some derivation chunks

A derivation is interpreted at the interface by replacing all of its chunks with their associated denotations.
The language particular content of the interface:

A finite list $\Lambda$ of denotations for

- **some** derivation chunks

A derivation is interpreted at the interface by replacing all of its chunks with their associated denotations.

Nondeterministic!
Historical Antecedents

Fraser:  Idioms are identical to non-idioms at DS

Koopman & Sportiche:  “If X is the minimal constituent containing all the idiomatic material, the head of X is part of the idiom.”

Jackendoff:  Idioms are triples $\langle Phon, Syn, Sem \rangle$ of structured entities.

O'Grady:  “An idiom’s component parts must form a chain.”

The main difference:
We are looking at the derivation
Historical Antecedents

Fraser: Idioms are identical to non-idioms at DS

Koopman & Sportiche: “If X is the minimal constituent containing all the idiomatic material, the head of X is part of the idiom.”

Jackendoff: Idioms are triples \(\langle \text{Phon}, \text{Syn}, \text{Sem}\rangle\) of structured entities.

O’Grady: “An idiom’s component parts must form a chain.”

The main difference:
We are looking at the derivation

A sequence of heads such that each is the head of a selected argument of another
Last Remarks

No derived idioms

Example
Raising verb and its derived subject cannot form an idiom.

Derivational patterns
Outline

The Syntax-Semantics Interface

The Syntax-Morphology Interface
  Suppletion
  Linearity
  Interpretation

Conclusion
Interface Uniformity

Interpretation

Interfaces are lists of associations between derivation pieces and things

Syntax-Semantics

- things are lambda terms over semantic domain
- ‘chunks’ of derivations correspond to *idioms/constructions*

Syntax-Morphology

- things are ???
- what do ‘chunks’ correspond to?
**Interface Uniformity**

**Interpretation**

Interfaces are lists of associations between derivation pieces and things.

**Syntax-Semantics**

- Things are lambda terms over semantic domain.
- ‘Chunks’ of derivations correspond to idioms/constructions.

**Syntax-Morphology**

- Things are ???
- What do ‘chunks’ correspond to?

A suppletive form is a chunk.
Suppletion
Suppletion

\[
\text{Pst} \rightarrow \text{merge} \rightarrow \text{go} \rightarrow \text{John}
\]

\[ \Rightarrow \text{"went"} \]

\[ \Rightarrow \text{"John"} \]
Suppletion

\[
\begin{array}{c}
Pst \\
\text{go} \\
\text{John}
\end{array}
\Rightarrow
\begin{array}{c}
\text{Pst} \\
\text{go} \\
\text{John}
\end{array} \rightsquigarrow \text{“went”}
\]

\[
\begin{array}{c}
\text{Pst} \\
\text{go} \\
\text{John}
\end{array} \rightsquigarrow \text{“John”}
\]
Suppletion

\[ \text{Pst} \xrightarrow{\text{go}} \text{John} \]

\[ \text{Interface} \]

\[ \text{Pst} \xrightarrow{\text{go}} \text{“went”} \]

\[ \text{John} \xrightarrow{\text{}} \text{“John”} \]
Suppletion

“went”
| “John”

\[ \text{Pst} \]
\[ \text{go} \] \[ \rightsquigarrow \] “went”

\[ \text{John} \] \[ \rightsquigarrow \] “John”
Suppletion

Interface

<table>
<thead>
<tr>
<th>Pst</th>
<th>“went”</th>
</tr>
</thead>
<tbody>
<tr>
<td>go</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>John</th>
<th>“John”</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fut</td>
<td>“will”</td>
</tr>
<tr>
<td>go</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Neg</th>
<th>“not”</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fut</td>
<td>“won’t”</td>
</tr>
<tr>
<td>Neg</td>
<td></td>
</tr>
<tr>
<td>go</td>
<td>“go”</td>
</tr>
</tbody>
</table>

Kobele (U of C)
Suppletion

"won’t"
"go"
"John"

Interface

\[
\begin{align*}
\text{Pst} & \rightarrow \text{“went”} \\
\text{go} & \rightarrow \text{“John”} \\
\text{John} & \rightarrow \text{“will”} \\
\text{Neg} & \rightarrow \text{“not”} \\
\text{Fut} & \rightarrow \text{“won’t”} \\
\end{align*}
\]
Suppletion

```
"won't"
  "go"
  "John"

"will"
  "not"
  "go"
  "John"
```

**Interface**

```
Pst  \(\mapsto\)  "went"

John  \(\mapsto\)  "John"

Fut  \(\mapsto\)  "will"

Neg  \(\mapsto\)  "not"

Fut  \(\mapsto\)  "won't"

Neg  \(\mapsto\)  "go"

go  \(\mapsto\)  "go"
```
Prediction

If ‘morphological idiom’ = suppletion, then

Clitics cannot trigger suppletion

(Unless we hack the features in an otherwise unmotivated way)
‘Lowering’ vs ‘Local Dislocation’

Embick & Noyer

- Jane is even prettier than Kim.
- Jane is even more naturally pretty than Kim.
- *Jane is even naturally prettier than Kim.

Evidence for sensitivity to order/adjacency (?)
‘Lowering’ vs ‘Local Dislocation’

Embick & Noyer

- Jane is even prettier than Kim.
- Jane is even more naturally pretty than Kim.
- *Jane is even naturally prettier than Kim.

Evidence for sensitivity to order/adjacency (?)

```
- re
  |  - re
  adj  |  pretty
  /  \  
naturally  pretty
```
Differing Predictions


- Jane is even prettier than Kim.
- Jane is even more naturally pretty than Kim.

1. Jane is even prettier naturally than Kim.
2. Jane is even more pretty naturally than Kim.

Embick & Noyer

1 is good

GK

1 is bad
Interpretation of morphological objects

How is the following interpreted?

\[
\begin{align*}
\text{Fut} & \quad \leadsto \quad \text{“won’t”} \\
\text{Neg} & \quad \leadsto \quad \text{“won’t”}
\end{align*}
\]

i.e. what does “won’t” mean?

A simple answer:

It means the same thing that the following does, where \( \alpha \) and \( \beta \) are the feature bundles of Fut and Neg, respectively.

\[
\begin{array}{c}
\text{MOVE} \\
\text{MERGE} \\
\text{won’t} :: \alpha \\
\epsilon :: \beta
\end{array}
\]
Interpretation (II)

This looks familiar!

- Chomsky’s *Strict Lexicalism*
- Brody’s *Mirror Theoretic Spellout*

\[
\text{A MW is pronounced in its highest strong position}
\]

Formal Simulability Relations:

\[\text{Chomsky} < \text{Brody} < \text{GK}\]
Where’s Morphology?

Answer: Inside the Interface

Regularities in ‘⇝’:

- Paradigms
- DM operations
- Templates
Outline

The Syntax-Semantics Interface

The Syntax-Morphology Interface

Conclusion
Conclusion

Interface Uniformity

Interfaces interpret syntactic structure in the same way

Derivationalism

Derived structure is interface irrelevant

- Relations between theories (Mirror Theory, Strict Lexicalism)
- Location of morphology (at interface - spelling out ‘⇝’)

Questions

- Is the interface sensitive to derived structure/order?
- Are the differences between morphological and semantic interfaces best viewed in terms of which chunks are ‘idioms’?
- Can we formulate a learning theory which operates by breaking big interface chunks into smaller ones?
Outline

Inherited vs Inherent Features

Constraints on ‘Idioms’

Minimalist Grammars
Inherited vs Inherent Features

Inherited Features

He$_{3s,nom}$ is$_{pres,3s}$ happy.

Pres
be
happy
pro$_{3s}$

The fundamental claim:

Only deep configurations matter for morphology

Analytical Possibilities

Checking: There are no inherited features.

Valuation: Interface objects are functions with inherited features as arguments.
Inherited vs Inherent Features

Inherited Features

He$_{3s,nom}$ is$_{pres,3s}$ happy.

The fundamental claim:

Only deep configurations matter for morphology

Analytical Possibilities

Checking: There are no inherited features.

Pres$_{3s}$
be
happy
pro$_{3s,nom}$
Inherited vs Inherent Features

**Inherited Features**

He$_{3s,nom}$ is$_{pres,3s}$ happy.

The fundamental claim:
Only deep configurations matter for morphology

Analytical Possibilities

Checking: There are no inherited features.
Valuation: Interface objects are functions with inherited features as arguments
Inherited Features and Valuation

A closer look at Pres

feature matrix: $=v +k s$

Kobele (U of C)  A derivational approach to phrasal spellout  BCGL 7, 2012  Brussels  32 / 38
Inherited vs Inherent Features

**Inherited Features and Valuation**

A closer look at $\text{Pres}$

feature matrix: $=v +k s$

A closer look at (this) $+k$

- assigns *nominative* case
- inherits *person* and *number* features

$\leadsto +k\left[\begin{array}{c}
\text{CASE} : \text{nom} \\
\text{PERSON} : \alpha \\
\text{NUMBER} : \beta
\end{array}\right]$
Inherited vs Inherent Features

Inherited Features and Valuation

A closer look at Pres

feature matrix: \( =v +k\ s \)

A closer look at (this) \(+k\)

- assigns \textit{nominative} case
- inherits \textit{person} and \textit{number} features

\[ \sim \rightarrow +k \begin{bmatrix}
\text{CASE} & : \text{nom} \\
\text{PERSON} & : \alpha \\
\text{NUMBER} & : \beta
\end{bmatrix} \]

\[
\text{Pres} \begin{array}{c}
\sim \rightarrow \lambda \alpha, \beta, x : \langle v, -k[\text{PER} : \alpha, \text{NUM} : \beta] \rangle . \text{match} \ \alpha, \beta \ \text{with} \\
| 3, s \rightarrow "is" (x) \\
| 1, s \rightarrow "am" (x) \\
| -, - \rightarrow "are" (x)
\end{array}
\]
Outline

Inherited vs Inherent Features

Constraints on ‘Idioms’

Minimalist Grammars
Constraints on ‘Idioms’

Constraints on Chunks at the Interfaces

**Semantics**
- Chunks are arbitrary
- Chunks can contain specifiers or complements
- Chunks are independent

**Morphology**
- Chunks are linear (i.e. non-branching treelets)
- Chunks go down complements (not specifiers)
- Chunks are uniform:
  
  Many chunks are present which differ just in the particular choice of content morphemes.

  (For many choices of V, T-v-V is an idiom)
Outline

Inherited vs Inherent Features

Constraints on ‘Idioms’

Minimalist Grammars
To specify a grammar, we need to specify two things:

1. The features
   (which features we will use in our grammar)
2. The lexicon
   (which syntactic feature sequences are assigned to which words)
Merge

\[
\begin{align*}
= x.\gamma + x\delta & \Rightarrow \gamma \delta < \\
\neq x.\gamma + x\delta & \Rightarrow \delta \gamma >
\end{align*}
\]
Minimalist Grammars

Move

\[ +x\gamma \quad \Rightarrow \quad \delta \quad \gamma \quad t \]

\[ -x\delta \]
Move

\[ +x\gamma \quad \Rightarrow \quad \delta \quad > \quad \gamma \quad t \]

No other possible mover