Syntax, Semantics, Pragmatics: Where do we Find Optimality?

Thomas Graf
tgraf@ucla.edu
tgraf.bol.ucla.edu

University of California, Los Angeles

Universität Göttingen, Göttingen, Germany

December 14, 2010
1. Reference-Set Constraints
2. Linear Tree Transducers — The Shortest Introduction Ever
3. General Results through OT-like Grammars
4. Example 1: Focus Economy
5. Example 2: Merge-over-Move
6. Example 3: Shortest Derivation Principle/Fewest Steps
Many **pragmatic** phenomena are analyzed as requiring comparisons of several alternatives and picking the best one (examples from Blutner and Zeevat 2008)

- Implicatures
  - Conventional implicatures
  - Conversational implicatures
  - Scalar implicatures
  - Exhaustivity implicatures
  - Implicature projection

- Presupposition projection

- Distribution of discourse particles
Optimality in Syntax/Semantics: Reference-Set Constraints

Optimality condition $\approx$ reference-set constraint
$\approx$ transderivational constraint $\approx$ global economy condition $\approx$ interface strategy

An Informal Definition

Given some input tree $t$, a reference-set constraint computes a set of possible output trees for $t$ — called the reference set of $t$ — and picks from said set the optimal output tree according to some economy metric.

Some examples from the literature:
- Rule I (Reinhart 2006)
- Scope Economy (Fox 2000)
- Fewest Steps (Chomsky 1995)
- Merge-over-Move (Chomsky 2000)
- Focus Economy (Reinhart 2006)
Optimality in Syntax/Semantics: Reference-Set Constraints

Optimality condition ≈ reference-set constraint
≈ transderivational constraint ≈ global economy condition ≈ interface strategy

An Informal Definition

Given some input tree $t$, a reference-set constraint computes a set of possible output trees for $t$ — called the reference set of $t$ — and picks from said set the optimal output tree according to some economy metric.

Some examples from the literature:
- Rule I (Reinhart 2006)
- Scope Economy (Fox 2000)
- Fewest Steps (Chomsky 1995)
- Merge-over-Move (Chomsky 2000)
- Focus Economy (Reinhart 2006)
Optimality in Syntax/Semantics: Reference-Set Constraints

Optimality condition $\approx$ reference-set constraint
$\approx$ transderivational constraint $\approx$ global economy condition $\approx$ interface strategy

An Informal Definition

Given some input tree $t$, a reference-set constraint computes a set of possible output trees for $t$ — called the reference set of $t$ — and picks from said set the optimal output tree according to some economy metric.

Some examples from the literature:
- Rule I (Reinhart 2006)
- Scope Economy (Fox 2000)
- Fewest Steps (Chomsky 1995)
- Merge-over-Move (Chomsky 2000)
- Focus Economy (Reinhart 2006)
Example: Focus Economy

(1)  a. \([_{TP} \text{John} \ [_{VP} \text{bought} \ [_{DP} \text{a red car}] for}]\).  
Focus set: \{TP, VP, DP, red car, car\}

b. \([_{TP} \text{John} \ [_{VP} \text{bought} \ [_{DP} \text{a red car}] for}]\).  
Focus set: \{red\}

Focus Projection
Any constituent containing the carrier of sentential main stress may be focused.

Focus Economy Rule
If the main stress has been shifted, a constituent containing its carrier may be focused iff it cannot be focused in the tree with unshifted stress.
Example: Focus Economy

(2) a. \([_{TP} \text{John} \ [_{VP} \text{bought} \ [_{DP} \text{a red car}]]]\).
Focus set: \{_{TP}, \ _{VP}, \ _{DP}, \ _{red\ car}, \ _{car}\}

b. \([_{TP} \text{John} \ [_{VP} \text{bought} \ [_{DP} \text{a red car}]]]\).
Focus set: \{_{red}\}

Focus Projection

Any constituent containing the carrier of sentential main stress may be focused.

Focus Economy Rule

If the main stress has been shifted, a constituent containing its carrier may be focused iff it cannot be focused in the tree with unshifted stress.
Example: Focus Economy, Cont.

Computing the Focus Sets

a) Neutral Stress

b) Shifted Stress
Example: Focus Economy, Cont.

Computing the Focus Sets

a) Neutral Stress

```
TP_S
 /     \
John_W VP_S
 /     \
bought_W DP_S
 /     \
a_W AP_S
 /     \
red_W cars_S
```

b) Shifted Stress

```
TP_S
 /     \
John_W VP_S
 /     \
bought_W DP_S
 /     \
a_W AP_S
 /     \
red_S cars_S
```
Example: Focus Economy, Cont.

Computing the Focus Sets

a) Neutral Stress

\[
\begin{array}{c}
TP_S \\
John_W \\
bought_W \\
a_W \\
red_W \\
ap_S \\
car_S
\end{array}
\]

b) Shifted Stress

\[
\begin{array}{c}
TP_S \\
John_W \\
bought_W \\
a_W \\
red_S \\
ap_S \\
car_W
\end{array}
\]
Example: Focus Economy, Cont.

Computing the Focus Sets

a) Neutral Stress

TP \( \_S \)  
\[ \text{John}_W \rightarrow \text{VP}_S \rightarrow \text{bought}_W \rightarrow \text{DP}_S \rightarrow \text{AP}_S \rightarrow \text{red}_W \rightarrow \text{car}_S \]

b) Shifted Stress

TP \( \_S \)  
\[ \text{John}_W \rightarrow \text{VP}_S \rightarrow \text{bought}_W \rightarrow \text{DP}_S \rightarrow \text{AP}_S \rightarrow \text{red}_S \rightarrow \text{car}_W \]
Example: Focus Economy, Cont.

Computing the Focus Sets

a) Neutral Stress

- TP
  - VP
    - John
    - bought
      - a
      - red
      - car

b) Shifted Stress

- TP
  - VP
    - John
    - bought
      - a
      - red
      - car
Example: Focus Economy, Cont.

Computing the Focus Sets

a) Neutral Stress

- $\text{TP}_S$
- $\text{VP}_S$
- $\text{DP}_S$
- $\text{AP}_S$

b) Shifted Stress

- $\text{TP}_S$
- $\text{VP}_S$
- $\text{DP}_S$
- $\text{AP}_S$
It seems that the same kind of optimality conditions can be found in all three modules:

1. compute set of alternatives
2. pick best option

But if we use linear tree transducers as a model, it turns out that reference-set constraints involve no comparisons. Rather, they are...

- **Insight 1 (theory-internal)**
  a different way of specifying standard well-formedness constraints ⇒ involve no tangible notion of optimality

- **Insight 2 (across theories)**
  connected to unidirectional OT.

Pragmatic optimality conditions, on the other hand, are usually modelled with bidirectional OT ⇒ different from reference-set constraints.
A linear finite-state bottom-up tree transducer
- traverses an input-tree from the leaves towards the root,
- labels it with states $q_i$, and
- transforms it into an output-tree.

It does so using rules of the following kind:
A Simple Example (Part 1)

A Transduction for Restricted wh-Movement, Rules 1–4

1) $\sigma \rightarrow q_i$

2) $\text{what} \rightarrow q_{wh}$

3) $\sigma \rightarrow q_i$

4) $\sigma \rightarrow q_{wh}$
A Simple Example (Part 2)

A Transduction for Restricted wh-Movement, Rule 5

5) \[ \text{TP} \rightarrow q_f \]

\[ q_i \]
\[ q_{wh} \]
\[ \text{DP} \]
\[ \text{subtree 1} \]
\[ \text{T'} \]
\[ \text{subtree 2} \]

\[ \text{CP} \]
\[ \text{what} \]
\[ C' \]
\[ \text{do} \]
\[ \text{TP} \]
\[ \text{DP} \]
\[ \text{subtree 1} \]
\[ \text{T'} \]
\[ \text{subtree 2} \]
A Simple Example (Part 3)

A Transduction for Restricted wh-Movement, Application

```
TP
  / 
DP   T'
  /   /
the men T
  /   /
like what
```
A Simple Example (Part 3)

A Transduction for Restricted wh-Movement, Application

```
TP
  /\    /
DP  men T'
  /\  /\  /
the men T  VP
  /\ /\ /\ /
like q_{wh}
    |   |
    t_{wh}
```
A Simple Example (Part 3)

A Transduction for Restricted wh-Movement, Application

```
TP
  /\     /
DP  T'   VP
  /\     /\       /
the men T q_i   q_wh
    /  /
   like t_wh
```
A Simple Example (Part 3)

A Transduction for Restricted wh-Movement, Application

```
[TP]
  [DP]
    [the] [men]
  [T']
    [q_{wh}]
      [VP]
        [like] [t_{wh}]
```

A Simple Example (Part 3)

A Transduction for Restricted wh-Movement, Application

```
                                           TP
                                          /   \
                                         DP   T'
                                        / \  /  \
                                       the men q_i q_{wh}
                                      / \  /  \    \  
                                     T   VP   like t_{wh}
```
A Simple Example (Part 3)

A Transduction for Restricted wh-Movement, Application

```
TP
|-- DP
|   |-- the
|   |-- men

|-- q_{wh}
    |-- T'
        |-- T
            |-- VP
                |-- like
                |-- t_{wh}
```
A Simple Example (Part 3)

A Transduction for Restricted wh-Movement, Application

Diagram:

```
TP
  /\    /
DP  q_{wh} /
  /\      /
the  q_{i} /
    /\  /
   men T
     /\  /
    T'  /
     /
    VP
      /
    like
      /
    t_{wh}
```
A Simple Example (Part 3)

A Transduction for Restricted wh-Movement, Application

```
TP
  /\    /
DP /\  /\  
 q_i /\ q_i
  /\  /\  
the men

q_{wh}

T'
  /\    /
T /\  /\  
  /\  /\  
like t_{wh}

VP
```
A Simple Example (Part 3)

A Transduction for Restricted wh-Movement, Application

![Diagram of transduction for Restricted wh-Movement]

TP
  /   
qi   qwh

DP
  /   
the   men

T
  /   
like

T'
  /   
twh

VP
A Simple Example (Part 3)

A Transduction for Restricted wh-Movement, Application

\[
q_f \quad \text{CP} \\
\text{what} \quad \text{C'} \\
\text{do} \quad \text{TP} \\
\text{DP} \\
\text{the} \quad \text{men} \\
\text{T'} \\
\text{T} \\
\text{VP} \\
\text{like} \quad t_{wh}
\]
Some Important Facts

What is Possible?

- Relabeling nodes
- Deleting subtrees
- Inserting subtrees of bounded size
- Enforcing constraints that define regular tree languages

What is Impossible?

- Copying of arbitrary subtrees
- Switching positions of non-siblings (e.g. specifier and complement)
- Counting past some threshold

Mathematical Properties

- A transducer can be decomposed into a sequence of smaller transducers, *et vice versa*.
- If the input tree language of a transducer is regular, then so is its output language. Regular tree languages are sufficiently powerful for Minimalism (Kobele et al. 2007).
Some Important Facts

What is Possible?
- Relabeling nodes
- Deleting subtrees
- Inserting subtrees of bounded size
- Enforcing constraints that define regular tree languages

What is Impossible?
- Copying of arbitrary subtrees
- Switching positions of non-siblings (e.g. specifier and complement)
- Counting past some threshold

Mathematical Properties
- A transducer can be decomposed into a sequence of smaller transducers, *et vice versa*.
- If the input tree language of a transducer is regular, then so is its output language. Regular tree languages are sufficiently powerful for Minimalism (Kobele et al. 2007).
Some Important Facts

**What is Possible?**
- Relabeling nodes
- Deleting subtrees
- Inserting subtrees of bounded size
- Enforcing constraints that define regular tree languages

**What is Impossible?**
- Copying of arbitrary subtrees
- Switching positions of non-siblings (e.g. specifier and complement)
- Counting past some threshold

**Mathematical Properties**
- A transducer can be decomposed into a sequence of smaller transducers, *et vice versa*.
- If the input tree language of a transducer is regular, then so is its output language. Regular tree languages are sufficiently powerful for Minimalism (Kobele et al. 2007).
Strategy

For a given reference-set constraint $C$, exhibit

- a Minimalist grammar that generates the input language, and
- a sequence of transducers that computes the same mapping from inputs to optimal outputs.

- Due to the mathematical properties of transducers, the output language is no more complex than the input language
- Hence it can be generated by some Minimalist grammar
- Hence $C$ is equivalent to some “constraint” that does not involve reference-set computation.
Strategy

For a given reference-set constraint $C$, exhibit

- a Minimalist grammar that generates the input language, and
- a sequence of transducers that computes the same mapping from inputs to optimal outputs.

Due to the mathematical properties of transducers, the output language is no more complex than the input language.

Hence it can be generated by some Minimalist grammar.

Hence $C$ is equivalent to some “constraint” that does not involve reference-set computation.

But why should this work for arbitrary reference-set constraints?
It seems natural to model reference-set constraints via OT. 

**Reference-Set Constraints as OT Grammars**

- Use $\text{GEN}$ to compute the reference-sets.
- Use a sequence of constraints to filter out the suboptimal candidates.

**A Major Complaint**

Without further restrictions, OT grammars can generate any kind of (tree) language

⇒ they don’t tell us anything about reference-set constraints.

Fortunately, there is a weaker alternative...
It seems natural to model reference-set constraints via OT.

Reference-Set Constraints as OT Grammars

- Use $\text{GEN}$ to compute the reference-sets.
- Use a sequence of constraints to filter out the suboptimal candidates.

A Major Complaint

Without further restrictions, OT grammars can generate any kind of (tree) language
⇒ they don’t tell us anything about reference-set constraints.

Fortunately, there is a weaker alternative...
It seems natural to model reference-set constraints via OT.

Reference-Set Constraints as OT Grammars

- Use $G_{\text{EN}}$ to compute the reference-sets.
- Use a sequence of constraints to filter out the suboptimal candidates.

A Major Complaint

Without further restrictions, OT grammars can generate any kind of (tree) language
$\Rightarrow$ they don’t tell us anything about reference-set constraints.

Fortunately, there is a weaker alternative...
Optimality Systems: A Restricted Version of OT

Optimality Systems (OSs; Frank and Satta 1998)

- A variant of OT that keeps just the bare skeleton.
  - All constraints only consider the output, never the input.
  - No correspondence theory
  - No output-output correspondence
  - No sympathy constraints

There are mathematical conditions that ensure that an OS can be implemented by a tree transducer.

A Minor Quibble

\textsc{Gen} is too “flat” for faithful models of reference-set computation, it does not directly represent reference-sets and their algebraic properties.
Optimality Systems: A Restricted Version of OT

Optimality Systems (OSs; Frank and Satta 1998)

A variant of OT that keeps just the bare skeleton.

- All constraints only consider the output, never the input.
- No correspondence theory
- No output-output correspondence
- No sympathy constraints

There are mathematical conditions that ensure that an OS can be implemented by a tree transducer.

A Minor Quibble

\textsc{Gen} is too “flat” for faithful models of reference-set computation, it does not directly represent reference-sets and their algebraic properties.
Depiction of a Controlled OS

Reference Types

Reference Sets

Evaluation

Output Language
Almost all constraints in the literature exhibit one of the two configurations below.

What do the two have in common?
Output Joint Preservation

If two reference sets overlap, then so do the reference types that are mapped to them.

Theorem (Frank and Satta 1998; Wartena 2000; Jäger 2002)

A controlled OS can be implemented as a transducer if

- the OS is output-joint preserving, and
- the input language is regular, and
- GEN and all constraints can be implemented as transducers.

Time to check this for specific reference-set constraints!
Output Joint Preservation

If two reference sets overlap, then so do the reference types that are mapped to them.

Theorem (Frank and Satta 1998; Wartena 2000; Jäger 2002)

A controlled OS can be implemented as a transducer if

- the OS is output-joint preserving, and
- the input language is regular, and
- \text{GEN} and all constraints can be implemented as transducers.

Time to check this for specific reference-set constraints!
Output Joint Preservation

If two reference sets overlap, then so do the reference types that are mapped to them.

Theorem (Frank and Satta 1998; Wartena 2000; Jäger 2002)

A controlled OS can be implemented as a transducer if

- the OS is output-joint preserving, and
- the input language is regular, and
- $\text{GEN}$ and all constraints can be implemented as transducers.

Time to check this for specific reference-set constraints!
Output Joint Preservation

If two reference sets overlap, then so do the reference types that are mapped to them.

Theorem (Frank and Satta 1998; Wartena 2000; Jäger 2002)

A controlled OS can be implemented as a transducer if

- the OS is output-joint preserving, and
- the input language is regular, and
- $\text{GEN}$ and all constraints can be implemented as transducers.

Time to check this for specific reference-set constraints!
Output Joint Preservation

If two reference sets overlap, then so do the reference types that are mapped to them.

Theorem (Frank and Satta 1998; Wartena 2000; Jäger 2002)

A controlled OS can be implemented as a transducer if

- the OS is output-joint preserving, and
- the input language is regular, and
- $\text{GEN}$ and all constraints can be implemented as transducers.

Time to check this for specific reference-set constraints!
Focus Economy Rule (Reminder)

If the main stress has been shifted, a constituent containing its carrier may be focused iff it cannot be focused in the tree with unshifted stress.

Computing the Focus Sets

a) Neutral Stress

b) Shifted Stress

Example 1: Focus Economy
Transducer Model: GEN

Step 1 & 2: GEN

- Non-deterministically relabel input with S/W-subscripts.
- Non-deterministically focus some node along the “stress path”.

Transducing an Input into a Stress-Annotated Output with Focus

```
TP

John

bought

a

red

VP

bought

a

red

DP

bought

a

red

AP

bought

a

red

car
```
### Transducer Model: GEN

**Step 1 & 2: GEN**

- Non-deterministically relabel input with S/W-subscripts.
- Non-deterministically focus some node along the “stress path”.

### Transducing an Input into a Stress-Annotated Output with Focus

```
TP_S
   └── VP_S
       └── DP_S
           └── AP_S
               └── red_S
                   └── car_W
   └── John_W
       └── bought_W
           └── a_W
```

John bought a red car
Transducer Model: \textbf{GEN}

**Step 1 & 2: \textbf{GEN}**

- Non-deterministically relabel input with S/W-subscripts.
- Non-deterministically focus some node along the “stress path”.

**Transducing an Input into a Stress-Annotated Output with Focus**

![Diagram](attachment:image.png)
Focus Economy requires reference to the neutral stress pattern. We allow this by **implicitly representing the neutral stress within the same tree**!

**Strategy**

- Define two paths `StressPath` and `NeutralPath`.
- `StressPath` represents the path of the current stress.
- `NeutralPath` represents the path of the neutral stress.
- Add a constraint that requires focus to be in the stress path, but unless `StressPath` and `NeutralPath` pick out the same nodes, focus may not be in `NeutralPath`. 
Example of $\phi$

**StressPath** and **NeutralPath**

```
TP_s
  /\    /\  
John_w VP_s  
  /\    /\  
bought_w DP_s
    /\  
a_w AP_s
      /\  
red_s car_w
```
Example of $\phi$

**StressPath and NeutralPath**

- $\text{TP}_S$
- $\text{VP}_S$
- $\text{DP}_S$
- $\text{AP}_S$
- $\text{red}_S$
- $\text{car}_W$
- $\text{bought}_W$
- $\text{John}_W$
Example of $\phi$
Merge-over-Move (MOM)

If two convergent derivations $d$ and $d'$ are built from the same lexical items and identical up to step $n$, at which point $d$ continues with Merge and $d'$ with Move, filter out $d'$.

(3)  

a. There seems $t_{\text{there}}$ to be a man in the garden.

b. * There seems a man to be $t_{a\text{man}}$ in the garden.

c. A man seems $t_{a\text{man}}$ to be $t_{a\text{man}}$ in the garden.
Example

```
M

C O

M seems M

M there M

M to be M

M a M

M man M

M in M

M the M

M garden
```

Derivation Trees of (3a) and (3b)
There seems to be a man in the garden.
- Fuse the two derivations into one **underspecified derivation**.
  - Remove all features but the category feature.
  - Inside TP: Replace O or Merger of *there* by new label O/there.
Transducer Model: $\text{GEN}$ (Step 2)

- Turn O/there back into O or Merge of there.
  - Use a transducer with states $q_*$, $q_O$ and $q_C$.
  - In state $q_*$, the transducer non-deterministically rewrites O/there as O or Merge of there.
  - If the transducer rewrites O/there as O, it switches into state $q_0$.
  - In state $q_0$, every occurrence of O/there is rewritten just as O.
  - The transducer switches out of $q_0$ only if it encounters a CP (indicated by state $q_C$; cf. structured numerations).

- Re instantiate the features.
Example 1

```
M
  /\  
C   O/there
     /\    
    M     
      /\      
  seems O/there
     /\        
    q*         
      /\      
    M       
        /\    
   to be a man in the garden
```
Transducer Model: Examples of Step 2

Example 1

[M]
[C]
O/there

seems

q*

[M]
there

[M]
to be a man in the garden
Example 1

There seems to be a man in the garden.
Example 1

There seems to be a man in the garden.
Example 1

To be a man in the garden

there

seems

M

M

M

M

O

C

qC

q0
Example 1

\[ q_* \]

\[ M \]

\[ C \quad O \]

\[ M \]

seems

\[ M \]

there

\[ M \]

to be a man in the garden
Transducer Model: Examples of Step 2

Example 2

M

C  O/there

seems  O/there

q*

M

to be a man in the garden
Example 2

\[ \text{C} \quad \text{O}/\text{there} \quad \text{M} \]

seems

\[ q_0 \quad \text{O} \quad \text{M} \]

to be a man in the garden

There seems to be a man in the garden.
Example 2

M

C

O/there

M

q*

seems

q0

O

M

to be a man in the garden
Example 2

M

C

O/there

q0

M

seems

O

M

to be a man in the garden
Example 2

M

\( q_C \)

C

O

M

seems

O

M

to be a man in the garden

\( q_0 \)
Example 2

\[ q^* \]

\[ M \]

\[ C \]
\[ O \]

seems

\[ O \]

\[ M \]

\[ M \]

\[ \text{to be a man in the garden} \]
The output candidates for both (4a) and (4b) are now (5a)–(5b).

(4)  
  a.  There seems $t_{\text{there}}$ to be a man in the garden.  
  b.  * There seems a man to be $t_{\text{a\ man}}$ in the garden.

(5)  
  a.  * There seems there to be a man in the garden.  
  b.  There seems $t_{\text{there}}$ to be a man in the garden.  
  c.  A man seems $t_{\text{a\ man}}$ to be $t_{\text{a\ man}}$ in the garden.

- We may extend the mapping such that (5c) is also assigned this reference set.
- (5a) still has to be ruled out.
The only constraint is the input language itself! By turning it into a transducer and composing it with \( \text{GEN} \), we remove all instances of overgeneration and filter out the illicit MOM violators.
Shortest Derivation Principle (SDP)

Given convergent derivations $d_1, \ldots, d_n$ over the same lexical items, pick the one(s) with the fewest instances of Move.

Why do we find the following contrast?

(6) a. Who$_i$ did John take $[\text{DP}_j \text{ a picture of } t_i]$?
   b. *Who$_i$ was $[\text{DP}_j \text{ a picture of } t_i]$ taken $t_j$ by John?
Derivations for (6b)

Two derivations are possible for (6b).

CED violation in (7c)

(7)  
  a. \([VP \text{ taken } [DP_j \text{ a picture of who}_i] \text{ by John}]\)
  b. \([TP [DP_j \text{ a picture of who}_i] T [VP \text{ taken } t_j \text{ by John}]]\)
  c. \([CP \text{ who}_i \text{ was } [TP [DP_j \text{ a picture of } t_i] T [VP \text{ taken } t_j \text{ by John}]]]\)

Derivation (8) is longer than (7)!

(8)  
  a. \([VP \text{ taken } [DP_j \text{ a picture of who}_i] \text{ by John}]\)
  b. \([VP \text{ who}_i \text{ taken } [DP_j \text{ a picture of } t_i] \text{ by John}]\)
  c. \([TP [DP_j \text{ a picture of } t_i] T [VP \text{ who}_i \text{ taken } t_j \text{ by John}]]\)
  d. \([CP \text{ who}_i \text{ was } [TP [DP_j \text{ a picture of } t_i] T [VP \text{ taken } t_j \text{ by John}]]]\)
Derivation Tree of (7)

```
was O
     | M
  O
  | M
T
  M
  | M
M
  | M
taken by
  M
  | DP
a picture of who
```
Derivation Tree of (8)

```
was
\( O \)
\( M \)
\( O \)
\( T \)
\( O \)
\( M \)
\( M \)
\( M \)
taken
\( DP \)
\( a \ picture \ of \ who \)
\( by \)
\( John \)
```
Underspecified Derivation Tree of (7) and (8)
Strategy

- Compute reference-set by
  1. mapping to underspecified derivation (i.e. remove Move-nodes)
  2. arbitrarily adding Move-nodes to underspecified derivation
  3. discarding all derivation trees that aren’t in the input language (i.e. the junk)

- Filter out the suboptimal derivation trees (those that can be obtained from others by adding Move-nodes)
  1. Let $R$ be the transduction that maps a derivation tree to the trees in its reference-set and $+O$ the transduction defined by adding Move-nodes
  2. The range of the composition of $R$ with $+O$ is the set of derivation trees that can be obtained from some tree in the range of $R$ by adding Move-nodes, i.e. the suboptimal outputs.
  3. Thus, the relative complement of the range of $R$ and the range of the composition of $R$ with $+O$ is the set $S$ of optimal outputs. Composing $R$ with the diagonal over $S$ maps every tree to its optimal outputs.
Architecture of SDP
Scope Economy ≠ Semantic SDP

**Scope Economy**

QR is licit only if it induces a change in meaning.

**Scope Economy (Rephrased)**

Given convergent derivations $d_1, \ldots, d_n$ that are identical modulo QR and have identical meaning, prefer the one with the fewest instances of Move.

- Checking semantic identity is hard.
- Even if we ignore semantics, Scope Economy needs more power than the SDP because the number of QR-able phrases per CP is not finitely bounded!
- We can move to a more powerful type of transducer that still preserves regularity, but we lose closure under composition $\Rightarrow$ Scope Economy structurally more demanding than SDP.
A Rule of Thumb

A reference-set constraint is likely to be computable by a transducer if

- one can find a structure that encodes the commonalities of all the competitors, and
- neither the underspecification step nor the recovery step require insertion of material of unbounded size, and
- the economy metric can be implemented as
  - a well-formedness constraint on underspecified structures, or
  - a specific restriction on the recovery step, or
  - a transducer that turns optimal candidates into suboptimal ones.
Advantages of Reference-Set Constraints

- **Modularity**
  Constraint only depends on input language, not on mechanisms that generate it

- **Succinctness**
  Non-reference-set correspondent may fail to make the restriction explicit or be much more complicated; reference-set constraint may subsume very different constraints, depending on input grammar

- **More Tweakable Parameters**
  Reference-set constraint gives us at least four parametrizable components: reference types, reference sets, the map between the two, and the economy metric.

- **Reaching out**
  Connections to OT, sTAG and others may allow us to incorporate results from these frameworks
Tree transducers were proposed as a model for reference-set constraints.

OSs offer a bird’s eye view on them (Insight 2).

Most requirements for an OS to be efficiently computable are fulfilled by reference-set constraints; in particular, their corresponding OSs are output joint preserving.

The only problematic areas are $\text{GEN}$ and the OS constraints.

The underspecification-and-filtration strategy offers a general solution.
Conclusion (Part 2)

- **Syntax**
  optimality conditions can be modelled by transducers
  ⇒ no optimality considerations involved

- **Semantics**
  - Incorporating semantic information is difficult.
  - Even on a purely structural level, more powerful transducers are necessary (cf. Scope Economy).

- **Pragmatics**
  assumed to require at least bidirectional optimization,
  whereas transducers correspond to unidirectional optimality
  ⇒ optimality in pragmatics fundamentally different
References I


