Formal Processing Theory
or
Parsing Without Parsers

Thomas Graf

Stony Brook University
Department of Linguistics
mail@thomasgraf.net
http://thomasgraf.net

First MIT Workshop on Minimalist Parsing
Oct 10 2015
The Take Home Messages

- Formal parsing models of processing are worth pursuing.
- **But**: problem of too many solutions
- Our approach is too fine-grained.
- We need a more general perspective.
- We need
  - abstraction
  - theorems
  - proofs
Outline

1. Why Care About Syntactic Processing?
2. Top-Down Parsing of Minimalist Grammars
3. Memory-Based Processing Predictions
4. Towards a Proof-Based Approach to Processing
   - Embedding Invariance
   - Isolated Embeddings
   - Informal Observations on Other Rankings
   - Movement, oh Movement!
A grammar without an efficient parser is useless
⇒ parsing is an important research area

But syntactic processing is only about the human parser, with all its warts and quirks:
- small working memory,
- no full parallelism or memoization,
- garden paths,
- grammaticality illusions,
- merely local syntactic coherence effects,

From an engineering perspective, the human parser is terribly flawed (neither sound nor complete).

So why should we care about modelling the human parser when CYK, Earley & Co are much more sophisticated?
Why Bother

MG Parsing

Processing

Towards Proofs

Conclusion

Why Syntactic Processing Matters

1. Applications
   - *Performance*
     Despite memory limitations, the human parser outperforms our fastest parsers (better than linear time).
   - *Future applications*
     Once you have a very expressive text generation system, you must ensure that its output is processable.

2. Theory
   - *Inherent interest*
     Every aspect of language is ripe for mathematical inquiry.
   - *Building bridges to other fields*
     We’ve got a great toolkit, let’s show the world what it can do!
   - *Clues about strong generative capacity*
     Processing effects provide **clues about syntactic structure**.
A Recent Attempt to Link Processing and Syntax

- **Stabler (2011, 2013)**
  - top-down parser for full class of Minimalist grammars
  - can handle virtually all analysis in the generative literature

- **Kobele et al. (2012)**
  - memory-usage metric relates parser behavior to processing
  - processing predictions are highly dependent on syntactic analysis (e.g. head VS phrasal movement)
Minimalist grammars treat syntax like chemistry.

<table>
<thead>
<tr>
<th>Chemistry</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>atoms</td>
<td>words</td>
</tr>
<tr>
<td>electrons</td>
<td>features</td>
</tr>
<tr>
<td>molecules</td>
<td>sentences</td>
</tr>
</tbody>
</table>

- Every word is a collection of features.
- Every feature has either positive or negative polarity.
- Features of opposite polarity annihilate each other.
- Feature annihilation drives the structure-building operations **Merge** and **Move**.
Minimalist grammars treat syntax like chemistry.

<table>
<thead>
<tr>
<th>Chemistry</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>atoms</td>
<td>words</td>
</tr>
<tr>
<td>electrons</td>
<td>features</td>
</tr>
<tr>
<td>molecules</td>
<td>sentences</td>
</tr>
</tbody>
</table>

- Every word is a collection of features.
- Every feature has either positive or negative polarity.
- Features of opposite polarity annihilate each other.
- Feature annihilation drives the structure-building operations **Merge** and **Move**.
MGs in Action

Phrase Structure Tree

Derivation Tree
Some Important Properties

- MGs are weakly equivalent to MCFGs and thus mildly context-sensitive. (Harkema 2001; Michaelis 2001)
- But we can decompose them into two finite-state components: (Michaelis et al. 2001; Kobele et al. 2007; Mönnich 2006)
  - a regular language of well-formed derivation trees
  - an MSO-definable mapping from derivations to phrase structure trees
- **Remember:** Every regular tree language can be reencoded as a CFG (with more fine-grained non-terminal labels). (Thatcher 1967)

The Context-Free Backbone of MGs

MGs can be viewed as CFGs with a more complicated mapping from trees to strings.
The Top-Down MG Parser

- **Core Idea**
  - recursive descent parser over context-free derivation trees
    - top-down
    - depth-first
    - left-to-right

- **Essential Modification**
  - linear order in the derivation tree does not correspond to linear order in the string
  - “left-to-right” refers to string order, not tree order

- **Bells and Whistles**
  - parser hooks directly into lexicon and feature calculus
  - beam search weeds out unlikely parses
  - constraints on movement reduce parsing complexity
Parsing as Node Indexation

If one focuses just on how a specific parse tree is assembled, parsing can be represented via **node indexation**:

- **Index**
  - at which step the node is conjectured

- **Outdex**
  - at which step the parser considers the node done

• which man do ed the dog bite
If one focuses just on how a specific parse tree is assembled, parsing can be represented via node indexation:

- **Index**
  at which step the node is conjectured

- **Outdex**
  at which step the parser considers the node done

\[
\begin{align*}
1 & \quad \text{Move} \quad 2 \\
2 & \quad \text{Move} \\
& \quad \text{Merge} \\
& \quad \text{do} \\
& \quad \text{Move} \\
& \quad \text{Merge} \\
& \quad -ed \\
& \quad \text{Merge} \\
& \quad \text{the} \\
& \quad \text{dog} \\
& \quad \text{bite} \\
& \quad \text{Merge} \\
& \quad \text{which} \\
& \quad \text{man}
\end{align*}
\]
If one focuses just on how a specific parse tree is assembled, parsing can be represented via node indexation:

- **Index**
  at which step the node is conjectured

- **Outdex**
  at which step the parser considers the node done

• which man do ed the dog bite
If one focuses just on how a specific parse tree is assembled, parsing can be represented via node indexation:

- **Index**
  at which step the node is conjectured

- **Outdex**
  at which step the parser considers the node done

\[
\begin{align*}
1 & \text{Move} & 2 \\
2 & \text{Move} & 3 \\
3 & \text{Merge} & 4 \\
4 & \text{do} & 4 \text{Move} \\
\text{-ed} & \text{Merge} & \\
\text{Merge} & \text{Merge} \\
\text{the} & \text{dog} & \text{bite} \\
\text{Merge} & \\
\text{which} & \text{man} \\
\end{align*}
\]
If one focuses just on how a specific parse tree is assembled, parsing can be represented via node indexation:

- **Index**
  at which step the node is conjectured

- **Outdex**
  at which step the parser considers the node done

- which man do ed the dog bite
If one focuses just on how a specific parse tree is assembled, parsing can be represented via node indexation:

- **Index**
  at which step the node is conjectured

- **Outdex**
  at which step the parser considers the node done

- `which man do ed the dog bite`
Parsin as Node Indexation

If one focuses just on how a specific parse tree is assembled, parsing can be represented via node indexation:

- **Index**
  - at which step the node is conjectured

- **Outdex**
  - at which step the parser considers the node done

- which man do ed the dog bite
If one focuses just on how a specific parse tree is assembled, parsing can be represented via node indexation:

- **Index**
  at which step the node is conjectured

- **Outdex**
  at which step the parser considers the node done

\[
\text{•which man do ed the dog bite}
\]
If one focuses just on how a specific parse tree is assembled, parsing can be represented via node indexation:

- **Index**
  at which step the node is conjectured

- **Outdex**
  at which step the parser considers the node done

```
• which man do ed the dog bite
```
If one focuses just on how a specific parse tree is assembled, parsing can be represented via node indexation:

- **Index**
  at which step the node is conjectured

- **Outdex**
  at which step the parser considers the node done

which \* man do ed the dog bite
If one focuses just on how a specific parse tree is assembled, parsing can be represented via **node indexation**:

- **Index**
  at which step the node is conjectured

- **Outdex**
  at which step the parser considers the node done

```plaintext
which man do ed the dog bite
```
If one focuses just on how a specific parse tree is assembled, parsing can be represented via node indexation:

- **Index**
  at which step the node is conjectured

- **Outdex**
  at which step the parser considers the node done

which man do•ed the dog bite
If one focuses just on how a specific parse tree is assembled, parsing can be represented via node indexation:

- **Index**
  at which step the node is conjectured

- **Outdex**
  at which step the parser considers the node done

which man do ed • the dog bite
If one focuses just on how a specific parse tree is assembled, parsing can be represented via node indexation:

- **Index**
  at which step the node is conjectured

- **Outdex**
  at which step the parser considers the node done

which man do ed •the dog bite
If one focuses just on how a specific parse tree is assembled, parsing can be represented via **node indexation**:

**Index**
- at which step the node is conjectured

**Outdex**
- at which step the parser considers the node done

**Example**

```
which man do ed the dog bite
```

**Diagram**

```
1 Move 2
  2 Move 3
    3 Merge 4
      4 do 12 Move 5
       5 Merge 6
         6 -ed 13 Merge 7
           7 Merge 8
             8 Merge 9
               9 which 10
                 10 man 11
```

```
If one focuses just on how a specific parse tree is assembled, parsing can be represented via node indexation:

- **Index**
  at which step the node is conjectured

- **Outdex**
  at which step the parser considers the node done

which man do ed the dog • bite
If one focuses just on how a specific parse tree is assembled, parsing can be represented via node indexation:

- **Index**
  - at which step the node is conjectured

- **Outdex**
  - at which step the parser considers the node done

```
which man do ed the dog bite
```
Relating Parsing and Processing

- **General Approach** (Kobele et al. 2012; Graf and Marcinek 2014; Graf et al. 2015)
  - pick competing syntactic analyses
  - pick metric to relate parsing behavior to processing difficulty
  - see which analysis gets it right

- **Simplifying Assumption**
  - consider only parser’s behavior for correct parse
  - factors out problem of finding correct parse

- **Appeal**
  - maximally simple
  - MGs allow for explicit, linguistically sophisticated analyses
  - fully specified parsing model with precise predictions
Notions of Memory Usage

All metrics studied so far build on **memory usage**.
(cf. Gibson 1998)

- **Tenure** how long a parse item (≈ node) \( p \) is stored
  \[ \text{outdex}(p) - \text{index}(p) \]

- **Payload** how many parse items were stored during the parse
  \[ | \{ p \mid \text{outdex}(p) - \text{index}(p) > 2 \} | \]

- **Gap** size of parse items \( \approx \) distance of movement

```
6 Merge 7
  7 Merge 14
  14 the 15 14 dog 16
  8 bite 17

7 Merge 8
  8 Merge 9
  9 which 10 9 man 11
```
Notions of Memory Usage

All metrics studied so far build on **memory usage**.
(cf. Gibson 1998)

- **Tenure** how long a parse item (≈ node) \( p \) is stored
  
  \[ \text{outdex}(p) - \text{index}(p) \]

- **Payload** how many parse items were stored during the parse
  
  \[ | \{ p | \text{outdex}(p) - \text{index}(p) > 2 \} | \]

- **Gap** size of parse items ≈ distance of movement
Notions of Memory Usage

All metrics studied so far build on memory usage. (cf. Gibson 1998)

- **Tenure** how long a parse item (≈ node) \( p \) is stored
  \[ \text{outdex}(p) - \text{index}(p) \]

- **Payload** how many parse items were stored during the parse
  \[ | \{ p \mid \text{outdex}(p) - \text{index}(p) > 2 \} | \]

- **Gap** size of parse items ≈ distance of movement
Memory-Based Metrics of Processing Difficulty

**Max**  highest tenure in parse
\[ \max(\{ t \mid t \text{ is the tenure of some node } n \}) \]

**Max^R**  vector of tenure for all nodes, in decreasing order

**Box**  payload of parse
| \{ n \mid n \text{ is a node with tenure } > 2 \} |

**Sum**  summed tenure of payload
\[ \sum_{n \text{ has tenure } > 2} \text{tenure-of}(n) \]
Example Values for Each Metric

Max 9
Max^R ⟨9, 8, 7, 7, 2, ...⟩
Box 4
Sum 31
Processing Phenomena: Embedding

- Left embedding is easy
  
  (1) John’s father’s cousin’s house’s roof collapsed.

- Center embedding is hard, right embedding is easy
  
  (2) a. The cheese that the mouse that the cat chased ate was rotten.
  
          b. The cheese was rotten that the mouse ate that the cat chased.

- Crossing dependencies are easier than nested dependencies.
  
  (3) a. that John Mary Peter swim teach let.  (German)
  
          b. that John Mary Peter let teach swim.  (Dutch)
A relative clause inside a sentential clause is easy.

(4) The fact that the employee who the manager hired stole office supplies worried the executive.

A sentential clause inside a relative clause is hard.

(5) The executive who the fact that the employee stole office supplies worried hired the manager.
Subject relative clauses (SRCs) are easier than object relative clauses (ORCs).

(6) a. The reporter who _ attacked the senator admitted the error.

b. The reporter who the senator attacked _ admitted the error.
RCs in East Asian

RCs **precede the modified noun** in Chinese, Japanese, Korean. SRC is still preferred over ORC.

(7) **Chinese**

a. __ attacked the senator who **reporter** admitted the error.

b. the senator attacked __ who **reporter** admitted the error.

In addition, Korean and Japanese also have SOV order.

(8) **Korean**

a. __ the senator attacked who **reporter** admitted the error.

b. the senator __ attacked who **reporter** admitted the error.
Overview of Findings

Methodology

1. take derivations for sentences with processing contrast
2. compute indices and outdices
3. compute value according to chosen metric
4. easier sentence should have lower value

<table>
<thead>
<tr>
<th></th>
<th>Max</th>
<th>Max^R</th>
<th>Sum</th>
<th>Box</th>
</tr>
</thead>
<tbody>
<tr>
<td>Center/Right</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Center/Crossing</td>
<td>✓</td>
<td>✓</td>
<td>≈</td>
<td>≈</td>
</tr>
<tr>
<td>Left embedding</td>
<td>×</td>
<td>×</td>
<td>◯</td>
<td>◯</td>
</tr>
<tr>
<td>SC/RC vs RC/SC</td>
<td>≈</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SRC vs ORC (Eng)</td>
<td>≈</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SRC vs ORC (Asian)</td>
<td>≈</td>
<td>×</td>
<td>◯</td>
<td>◯</td>
</tr>
</tbody>
</table>
Overview of Findings

**Methodology**

1. take derivations for sentences with processing contrast
2. compute indices and outdices
3. compute value according to chosen metric
4. easier sentence should have lower value

<table>
<thead>
<tr>
<th></th>
<th>Max</th>
<th>Max$^R$</th>
<th>Sum</th>
<th>Box</th>
</tr>
</thead>
<tbody>
<tr>
<td>Center/Right</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Center/Crossing</td>
<td>✓</td>
<td>✓</td>
<td>≃</td>
<td>≃</td>
</tr>
<tr>
<td>Left embedding</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>≃</td>
</tr>
<tr>
<td>SC/RC vs RC/SC</td>
<td>≃</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SRC vs ORC (Eng)</td>
<td>≃</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SRC vs ORC (Asian)</td>
<td>≃</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
</tbody>
</table>
## Predictions for East Asian RC-Processing

<table>
<thead>
<tr>
<th></th>
<th>Promotion</th>
<th>Wh-Movement</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>all</td>
<td>lex.</td>
</tr>
<tr>
<td><strong>Korean</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Max</td>
<td>tie</td>
<td>tie</td>
</tr>
<tr>
<td>Max^R</td>
<td>ORC</td>
<td>ORC</td>
</tr>
<tr>
<td>Sum</td>
<td>ORC</td>
<td>ORC</td>
</tr>
<tr>
<td>Box</td>
<td>tie</td>
<td>ORC</td>
</tr>
<tr>
<td><strong>Chinese</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Max</td>
<td>tie</td>
<td>tie</td>
</tr>
<tr>
<td>Max^R</td>
<td>ORC</td>
<td>ORC</td>
</tr>
<tr>
<td>Sum</td>
<td>SRC</td>
<td>ORC</td>
</tr>
<tr>
<td>Box</td>
<td>SRC</td>
<td>SRC</td>
</tr>
</tbody>
</table>
Why Modelling is not Enough

Parameters of the modelling approach...
1. Syntactic analysis
2. Parser/Node Indexation algorithm
3. Processing difficulty metric

... and a swath of problems
- infinitely many choices for each parameter
- complex and unpredictable interaction
- solution underspecified by evidence

Solution

What we need are the standard tools of mathematical linguistics:
- precisely defined yet general properties,
- proofs instead of simulations,
- theorems about infinite classes of parsers/metrics
Why Modelling is not Enough

Parameters of the modelling approach...

1. Syntactic analysis
2. Parser/Node Indexation algorithm
3. Processing difficulty metric

... and a swath of problems

- infinitely many choices for each parameter
- complex and unpredictable interaction
- solution underspecified by evidence

Solution

What we need are the standard tools of mathematical linguistics:

- precisely defined yet general properties,
- proofs instead of simulations,
- theorems about infinite classes of parsers/metrics
**Metric Property 1: Embedding Invariance**

A metric $M$ is embedding invariant iff

$$a \leq_M c \iff b \leq_M c$$

**Psycholinguistic Motivation**

Many contrasts are independent of the containing clause:

- SC/RC vs RC/SC
- SRC vs ORC
- Center embedding vs right embedding
- Nested vs crossing dependencies
Metric Property 1: Embedding Invariance

A metric $M$ is embedding invariant iff

\[ a <_M M = M \implies b <_M M = M. \]

Psycholinguistic Motivation

Many contrasts are independent of the containing clause:

- SC/RC vs RC/SC
- SRC vs ORC
- Center embedding vs right embedding
- Nested vs crossing dependencies
Shape-Blind

Definition

Two subtrees are

- **feature-equivalent** iff their list of unchecked features is identical.

- **$M$-equivalent** with respect to metric $M$ iff $M$ assigns them the same value.

Definition (Shape-Blind)

A metric $M$ is **shape-blind** iff it holds that

$$a \overset{=} \M b \quad \text{if } \quad a \text{ and } c \text{ are feature-equivalent and } M\text{-equivalent}.$$
Embedding Invariance Implies Shape-Blindness

Theorem

A metric $M$ is embedding invariant only if it is shape-blind.

Lemma

Max and Gap are not shape-blind.

Proof.

- Max: size of left subtree determines tenure of its right sibling
- Gap: movement paths can differ in length

Corollary

Max and Gap are not embedding invariant.
Embedding Invariance Implies Shape-Blindness

Theorem

A metric $M$ is embedding invariant only if it is shape-blind.

Lemma

Max and Gap are not shape-blind.

Proof.

- **Max**: size of left subtree determines tenure of its right sibling
- **Gap**: movement paths can differ in length

Corollary

Max and Gap are not embedding invariant.
Explaining the Failure of \textbf{Max} for Chinese SRC/ORC

**Intuition**
Embedding the DPs in their clauses causes high tenure. This outweighs all SRC/ORC differences.
Explaining the Failure of Max for Chinese SRC/ORC

Intuition
Embedding the DPs in their clauses causes high tenure. This outweighs all SRC/ORC differences.
Isolated Embeddings

Definition (Isolation)
A subtree is isolated iff the only unchecked feature is the category feature of its root.

Theorem
Every “reasonable” shape-blind metric is embedding invariant for isolated subtrees.
Other Rankings are Embedding Invariant

**Theorem**

**Box, Gap, and Sum are invariant under isolated embeddings.**

**Proof.**

- An isolated embedding of $a$ into $b$ only adds a constant number $n$ of tenure nodes, where $n$ depends only on $b$.
- This guarantees that the value of a derivation under the respective metric is only increased by a constant amount that is a function of $n$ and the choice of metric.

- The East Asian RC cases can be analyzed as isolated embeddings of distinct DPs into the same matrix clause.
- So why do most of these metrics fail nonetheless?
Other Rankings are Embedding Invariant

**Theorem**

*Box, Gap, and Sum are invariant under isolated embeddings.*

**Proof.**

- An isolated embedding of a into b only adds a constant number $n$ of tenure nodes, where $n$ depends only on b.

- This guarantees that the value of a derivation under the respective metric is only increased by a constant amount that is a function of $n$ and the choice of metric.

- The East Asian RC cases can be analyzed as isolated embeddings of distinct DPs into the same matrix clause.

- So why do most of these metrics fail nonetheless?
Theorem

**Box, Gap, and Sum are invariant under isolated embeddings.**

Proof.

- An isolated embedding of a into b only adds a constant number $n$ of tenure nodes, where $n$ depends only on b.
- This guarantees that the value of a derivation under the respective metric is only increased by a constant amount that is a function of $n$ and the choice of metric.

- The East Asian RC cases can be analyzed as isolated embeddings of distinct DPs into the same matrix clause.
- So why do most of these metrics fail nonetheless?
The Role of Movement

**Definition (Move Power)**

The **Move power** of a derivation is the number of precedence relations that are altered by Move.

**Definition (Surface orientation)**

A metric $M$ is **surface-oriented** iff it holds for all trees $a$ and $c$ that

- if $a$ and $c$ are identical modulo Move, and
- the Move power of $a$ is less than the Move power of $c$, then
  - $M(a) \leq M(c)$.

**Theorem**

Max, Box, and Sum are surface oriented. Gap is not.
The Role of Movement

Definition (Move Power)
The Move power of a derivation is the number of precedence relations that are altered by Move.

Definition (Surface orientation)
A metric $M$ is surface-oriented iff it holds for all trees $a$ and $c$ that
- if $a$ and $c$ are identical modulo Move, and
- the Move power of $a$ is less than the Move power of $c$, then $M(a) \leq M(c)$.

Theorem
Max, Box, and Sum are surface oriented. Gap is not.
ORC Preference in Korean
ORC Preference in Korean
Which Properties do we Want?

- **Embedding Invariance**  
  mostly yes, but some apparent exceptions

- **Isolated Embedding Invariance**  
  yes

- **Surface-Oriented**  
  mostly no?
The Bigger Picture

- Modeling provides important clues, but it is not enough.
- Modeling cannot provide a formal theory of what properties an adequate processing metric need to satisfy.
- We need to think in terms of more abstract and general properties like embedding invariance.
- We may never find a unique solution to the processing problem due to insufficient evidence, but we can try to characterize the (infinite?) class of viable solutions.


