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

Parsing $\neq$ Processing

- A grammar without an efficient parser is useless
  $\Rightarrow$ 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,

  $\vdots$
- 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 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)

The Most Informal Intro to MGs Ever

Minimalist grammars treat syntax like chemistry.

Chemistry	**Syntax**
atoms	words
electrons	features
molecules	sentences

- 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**.

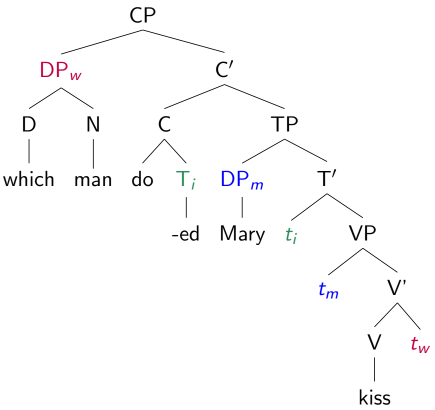
The Most Informal Intro to MGs Ever

Minimalist grammars treat syntax like chemistry.

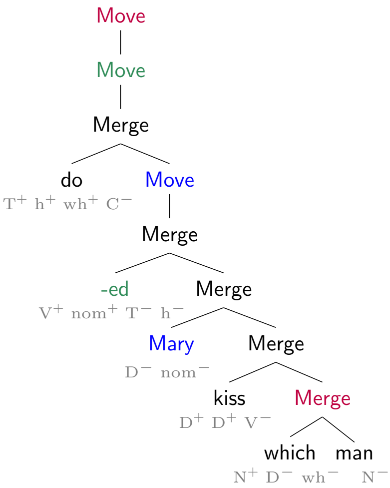
Chemistry	Syntax
atoms	words
electrons	features
molecules	sentences

- 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.

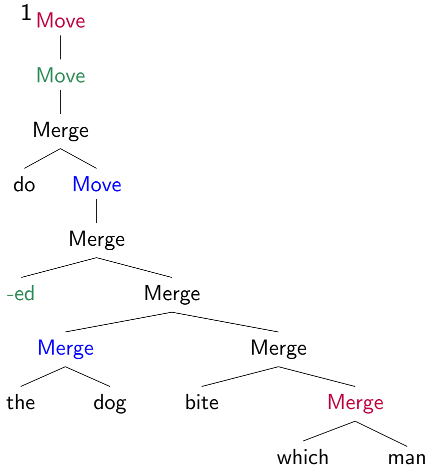
The Top-Down MG Parser

- **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
  $\Rightarrow$ "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

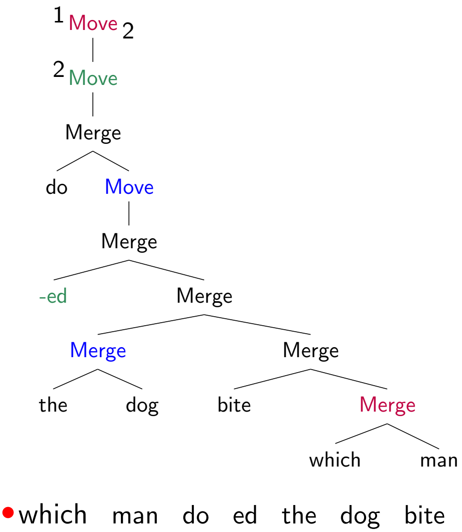


which man do ed the dog bite

Parsing as Node Indexation

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

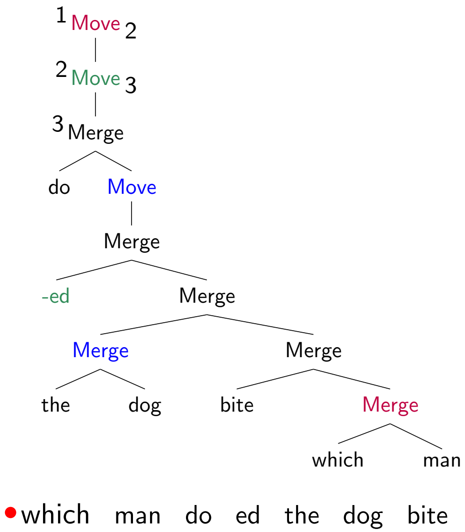
- **Index**
  at which step the node is conjectured
- **Outdex**
  at which step the parser considers the node done



Parsing as Node Indexation

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

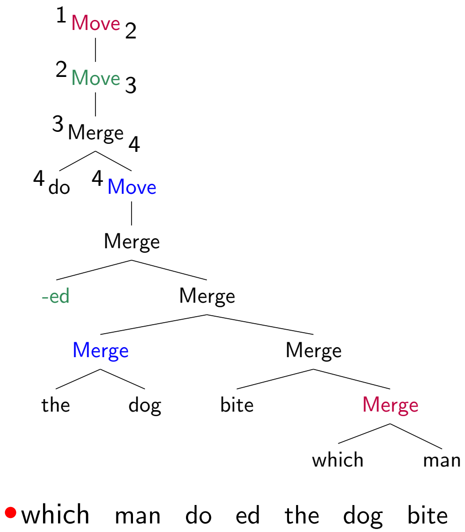
- **Index**
  at which step the node is conjectured
- **Outdex**
  at which step the parser considers the node done



Parsing as Node Indexation

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

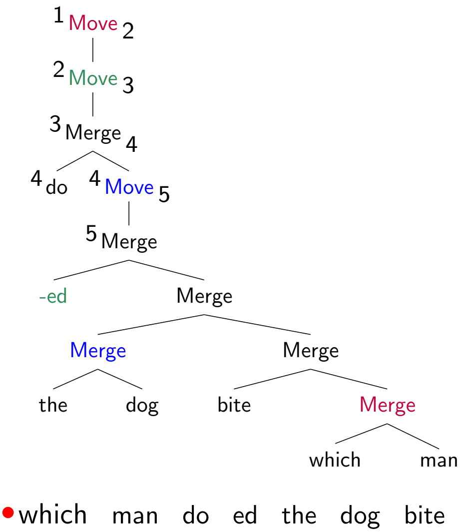
- **Index**
  at which step the node is conjectured
- **Outdex**
  at which step the parser considers the node done



Parsing as Node Indexation

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

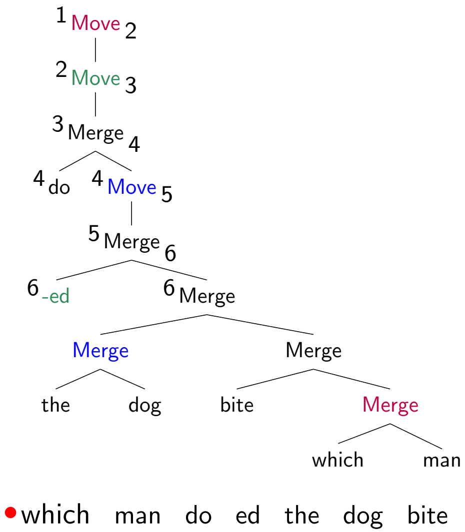
- **Index**
  at which step the node is conjectured
- **Outdex**
  at which step the parser considers the node done



Parsing as Node Indexation

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

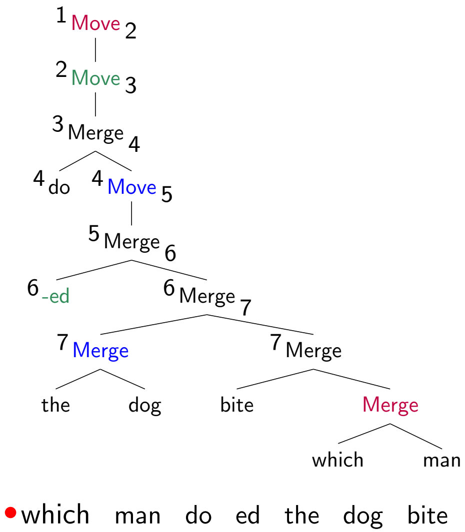
- **Index**
  at which step the node is conjectured
- **Outdex**
  at which step the parser considers the node done



Parsing as Node Indexation

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

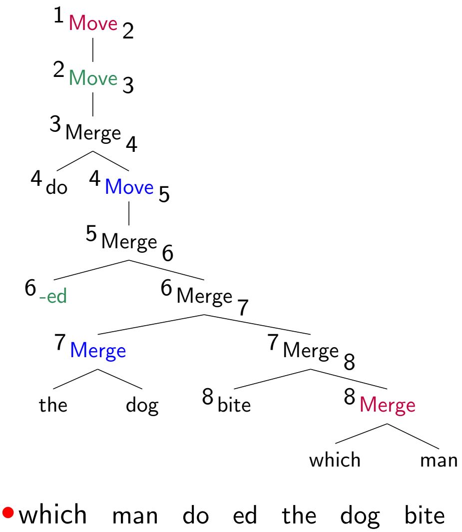
- **Index**
  at which step the node is conjectured
- **Outdex**
  at which step the parser considers the node done



Parsing as Node Indexation

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

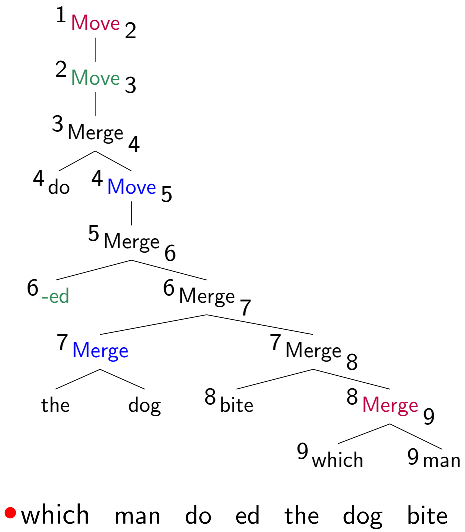
- **Index**
  at which step the node is conjectured
- **Outdex**
  at which step the parser considers the node done



Parsing as Node Indexation

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

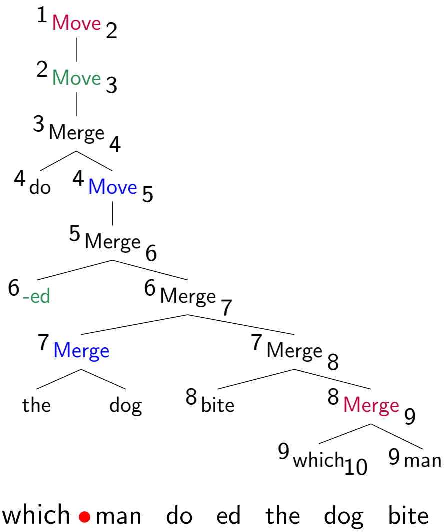
- **Index**
  at which step the node is conjectured
- **Outdex**
  at which step the parser considers the node done



Parsing as Node Indexation

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

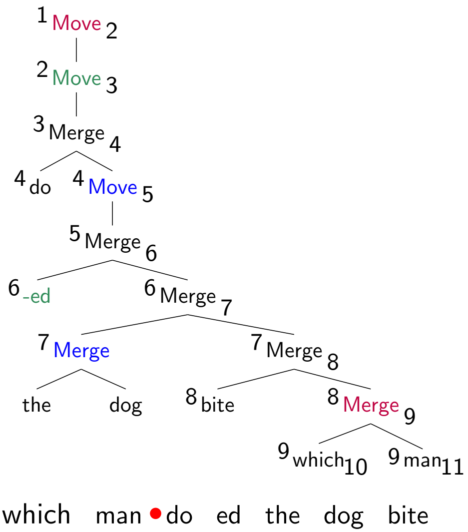
- **Index**
  at which step the node is conjectured
- **Outdex**
  at which step the parser considers the node done



Parsing as Node Indexation

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

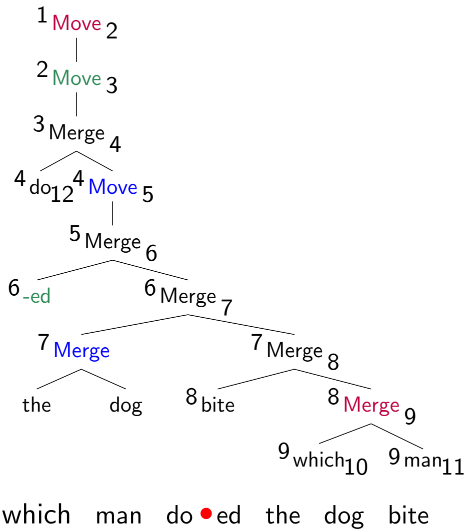
- **Index**
  at which step the node is conjectured
- **Outdex**
  at which step the parser considers the node done



Parsing as Node Indexation

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

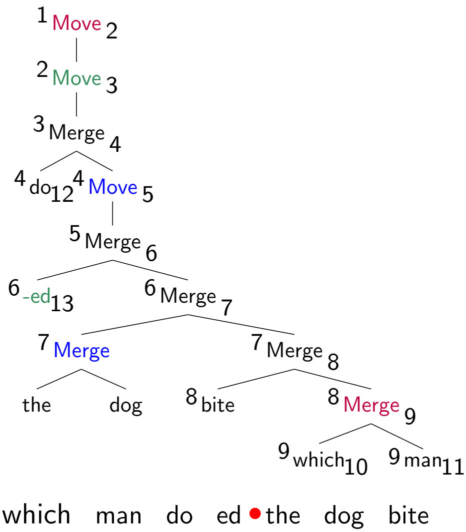
- **Index**
  at which step the node is conjectured
- **Outdex**
  at which step the parser considers the node done



Parsing as Node Indexation

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

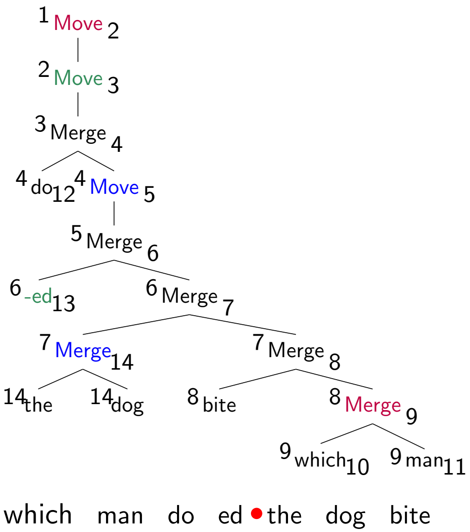
- **Index**
  at which step the node is conjectured
- **Outdex**
  at which step the parser considers the node done



Parsing as Node Indexation

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

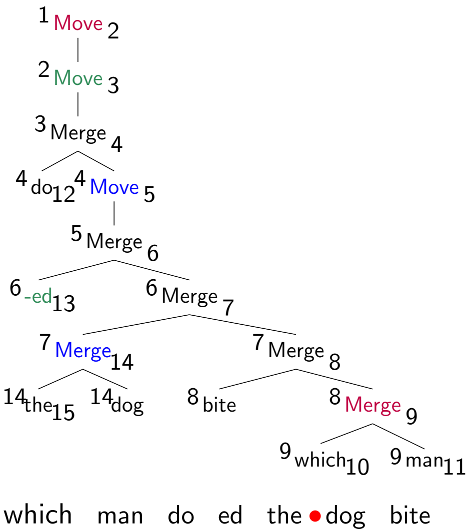
- **Index**
  at which step the node is conjectured
- **Outdex**
  at which step the parser considers the node done



Parsing as Node Indexation

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

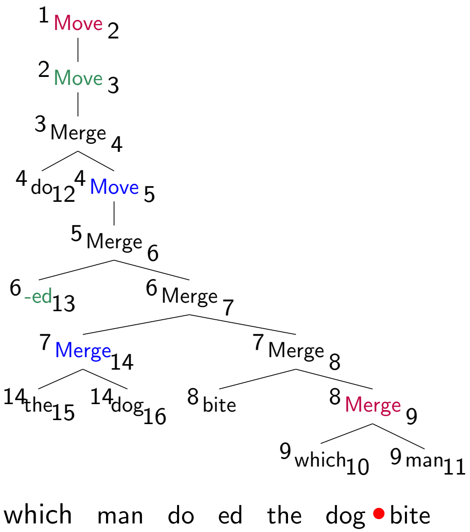
- **Index**
  at which step the node is conjectured
- **Outdex**
  at which step the parser considers the node done



Parsing as Node Indexation

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

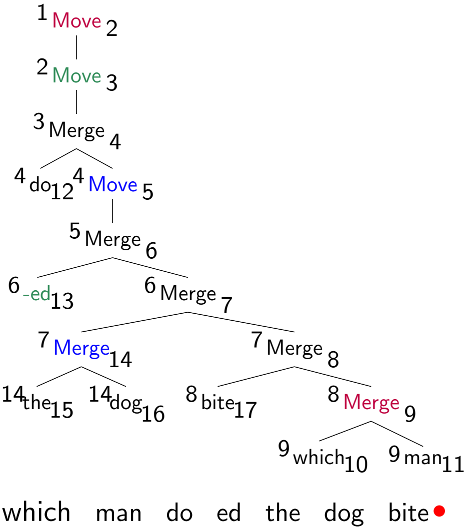
- **Index**
  at which step the node is conjectured
- **Outdex**
  at which step the parser considers the node done



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



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

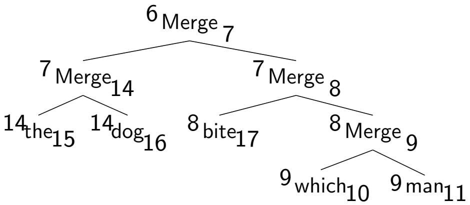
Notions of Memory Usage

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

Tenure how long a parse item ($\approx$ node) p is stored
$\mathit{outdex}(p) - \mathit{index}(p)$

Payload how many parse items were stored during the parse
$|\{p \mid \mathit{outdex}(p) - \mathit{index}(p) > 2\}|$

Gap size of parse items $\approx$ distance of movement



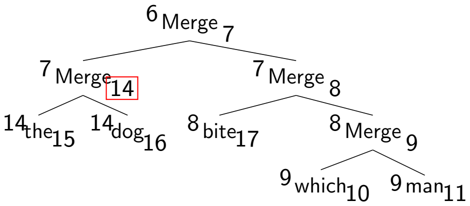
Notions of Memory Usage

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

Tenure how long a parse item (≈ node) p is stored
$outdex(p) - index(p)$

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

Gap size of parse items ≈ distance of movement



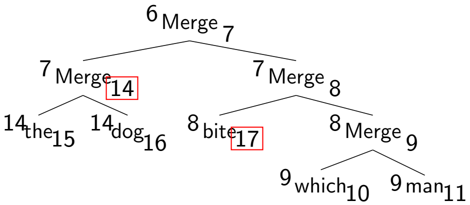
Notions of Memory Usage

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

Tenure how long a parse item ($\approx$ node) p is stored
$outdex(p) - index(p)$

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

Gap size of parse items $\approx$ distance of movement



Memory-Based Metrics of Processing Difficulty

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

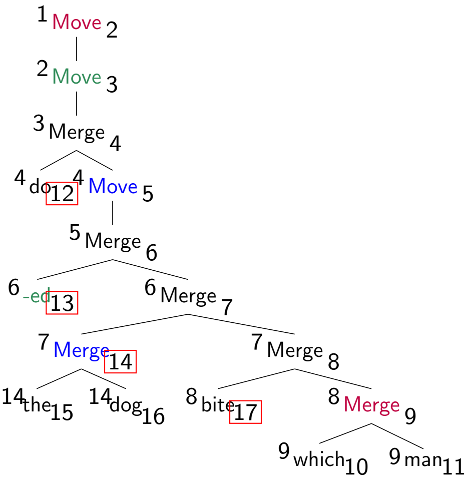
Max$^{\text{R}}$ vector of tenure for all nodes, in decreasing order

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$^{\mathbf{R}}$	$\langle 9, 8, 7, 7, 2, \ldots \rangle$
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)

Sentential Clauses and Relative Clauses

- 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 and Object Relative Clauses in English

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

	Max	Max^R	Sum	Box
Center/Right	✓	✓	✓	✓
Center/Crossing	✓	✓	≈	≈
Left embedding	×	×	×	≈
SC/RC vs RC/SC	≈	✓	✓	✓
SRC vs ORC (Eng)	≈	✓	✓	✓
SRC vs ORC (Asian)	≈	×	×	×

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

	Max	Max^R	Sum	Box
Center/Right	✓	✓	✓	✓
Center/Crossing	✓	✓	≈	≈
Left embedding	×	×	×	≈
SC/RC vs RC/SC	≈	✓	✓	✓
SRC vs ORC (Eng)	≈	✓	✓	✓
SRC vs ORC (Asian)	≈	×	×	×

Predictions for East Asian RC-Processing

Korean

	Promotion			Wh-Movement		
	all	lex.	pron.	all	lex.	pron.
Max	tie	tie	tie	tie	tie	tie
Max$^{\mathbf{R}}$	ORC	ORC	ORC	ORC	ORC	ORC
Sum	ORC	ORC	ORC	ORC	ORC	ORC
Box	tie	ORC	ORC	ORC	ORC	ORC

Chinese

	Promotion			Wh-Movement		
	all	lex.	pron.	all	lex.	pron.
Max	tie	tie	tie	tie	tie	tie
Max$^{\mathbf{R}}$	ORC	ORC	ORC	ORC	ORC	ORC
Sum	SRC	ORC	ORC	tie	ORC	ORC
Box	SRC	SRC	tie	SRC	tie	ORC

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

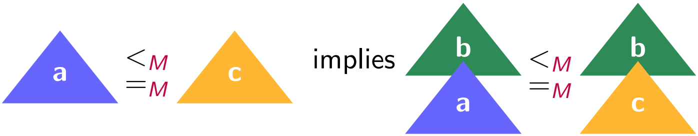
Solution

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



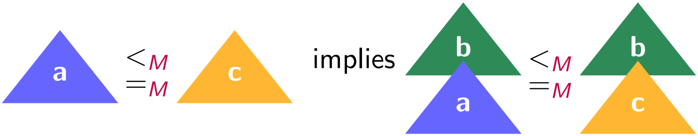
Psycholinguistic Motivation

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



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

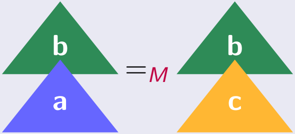
Definition

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



if a and c are feature-equivalent and M-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.*

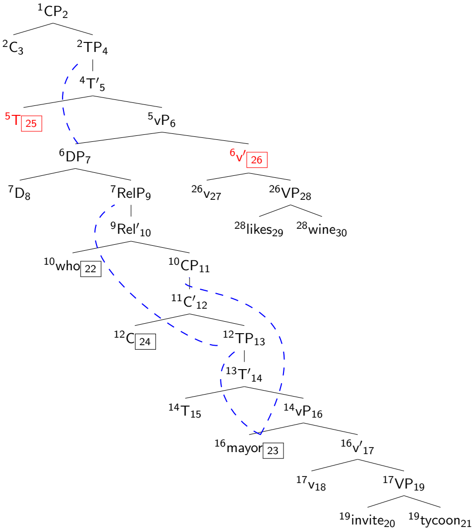
Proof.

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

Corollary

Max *and* **Gap** *are not embedding invariant.*

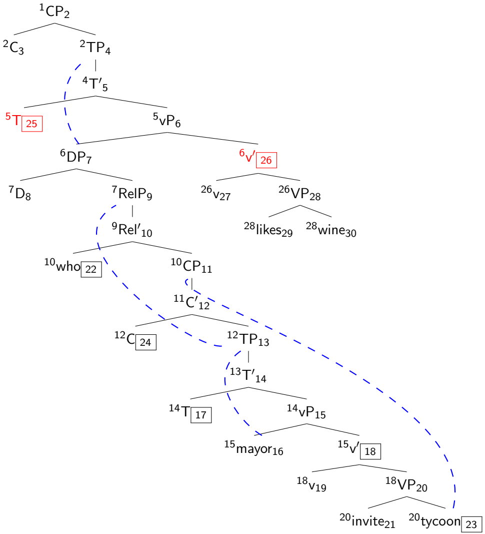
Explaining the Failure of **Max** for Chinese SRC/ORC



Intuition

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

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?

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?

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)

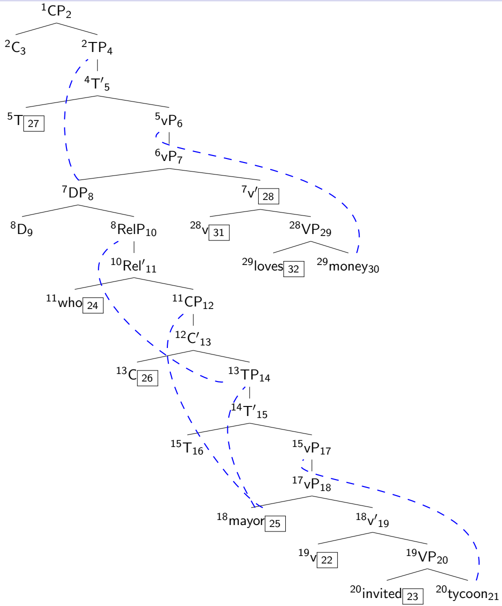
A metric M is **surface-oriented** iff it holds for all trees a and c that

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

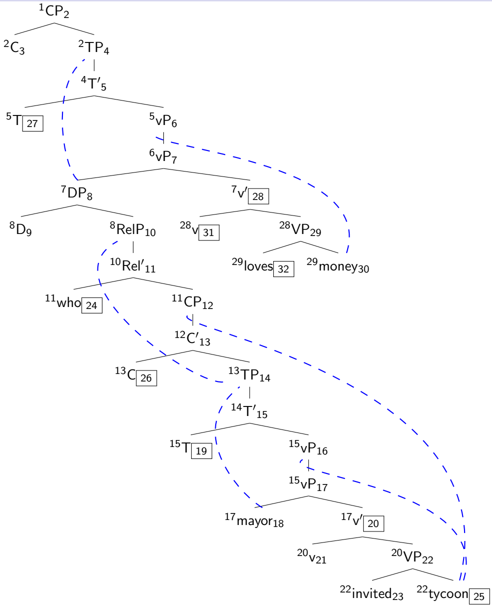
Theorem

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.

References I

Gibson, Edward. 1998. Linguistic complexity: Locality of syntactic dependencies. *Cognition* 68:1–76.

Graf, Thomas, Brigitta Fodor, James Monette, Gianpaul Rachiele, Aunika Warren, and Chong Zhang. 2015. A refined notion of memory usage for minimalist parsing. In *Proceedings of the 14th Meeting on the Mathematics of Language (MoL 2015)*, 1–14. Chicago, USA: Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/W15-2301`.

Graf, Thomas, and Bradley Marcinek. 2014. Evaluating evaluation metrics for minimalist parsing. In *Proceedings of the 2014 ACL Workshop on Cognitive Modeling and Computational Linguistics*, 28–36.

Harkema, Henk. 2001. A characterization of minimalist languages. In *Logical aspects of computational linguistics (LACL'01)*, ed. Philippe de Groote, Glyn Morrill, and Christian Retoré, volume 2099 of *Lecture Notes in Artificial Intelligence*, 193–211. Berlin: Springer.

Kobele, Gregory M., Sabrina Gerth, and John T. Hale. 2012. Memory resource allocation in top-down minimalist parsing. In *Proceedings of Formal Grammar 2012*.

Kobele, Gregory M., Christian Retoré, and Sylvain Salvati. 2007. An automata-theoretic approach to minimalism. In *Model Theoretic Syntax at 10*, ed. James Rogers and Stephan Kepser, 71–80.

References II

Michaelis, Jens. 2001. Transforming linear context-free rewriting systems into minimalist grammars. *Lecture Notes in Artificial Intelligence* 2099:228–244.

Michaelis, Jens, Uwe Mönnich, and Frank Morawietz. 2001. On minimalist attribute grammars and macro tree transducers. In *Linguistic form and its computation*, ed. Christian Rohrer, Antje Roßdeutscher, and Hans Kamp, 287–326. Stanford: CSLI.

Mönnich, Uwe. 2006. Grammar morphisms. Ms. University of Tübingen.

Stabler, Edward P. 2011. Top-down recognizers for MCFGs and MGs. In *Proceedings of the 2011 Workshop on Cognitive Modeling and Computational Linguistics*. To appear.

Stabler, Edward P. 2013. Bayesian, minimalist, incremental syntactic analysis. *Topics in Cognitive Science* 5:611–633.

Thatcher, James W. 1967. Characterizing derivation trees for context-free grammars through a generalization of finite automata theory. *Journal of Computer and System Sciences* 1:317–322.