A Refined Notion of Memory Usage for Minimalist Parsing

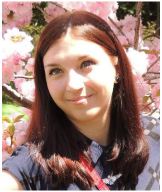
Thomas Graf* Brigitta Fodor James Monette
Gianpaul Rachiele Aunika Warren Chong Zhang

Stony Brook University

*mail@thomasgraf.net http://thomasgraf.net

MOL 14
July 25 2015

The Team



Brigi Fodor

James Monette

Gianpaul Rachielle

Aunika Warren

Chong Zhang

Two Take-Home Messages

The Big Question

Can we give a maximally simple parsing model that derives processing effects purely from memory usage?

Message 1: It's Possible

Yes, a variety of phenomena is explained by

- maximum storage time for parse items, and
- the size of parse items.

Message 2: Proofs, Not Models

The current approach of testing hand-designed models is insufficient. We need a solid mathematical foundation for syntactic processing research!

Two Take-Home Messages

The Big Question

Can we give a maximally simple parsing model that derives processing effects purely from memory usage?

Message 1: It's Possible

Yes, a variety of phenomena is explained by

- maximum storage time for parse items, and
- the size of parse items.

Message 2: Proofs, Not Models

The current approach of testing hand-designed models is insufficient. We need a solid mathematical foundation for syntactic processing research!

Two Take-Home Messages

The Big Question

Can we give a maximally simple parsing model that derives processing effects purely from memory usage?

Message 1: It's Possible

Yes, a variety of phenomena is explained by

- maximum storage time for parse items, and
- the size of parse items.

Message 2: Proofs, Not Models

The current approach of testing hand-designed models is insufficient. We need a solid mathematical foundation for syntactic processing research!

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,

    ⋮
- 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**.

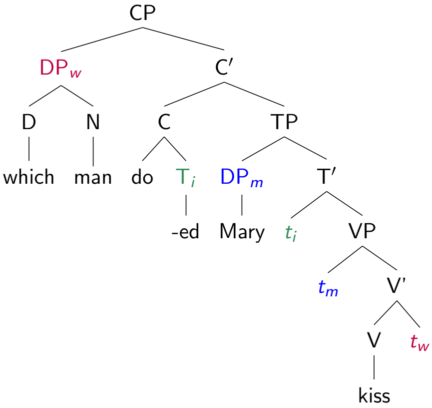
The Most Informal Intro to MGs Ever

Minimalist grammars treat syntax like chemistry.

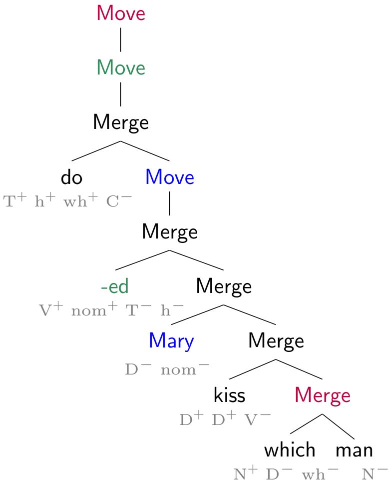
Chemistry	Syntax
atoms	words
electrons	features
molecules	sentences

- 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.

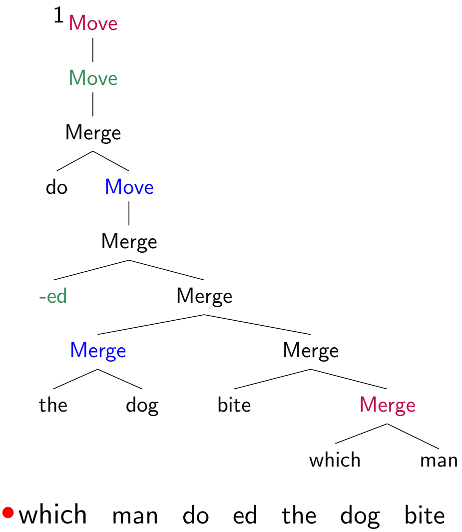
The Top-Down MG Parser

- **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**:

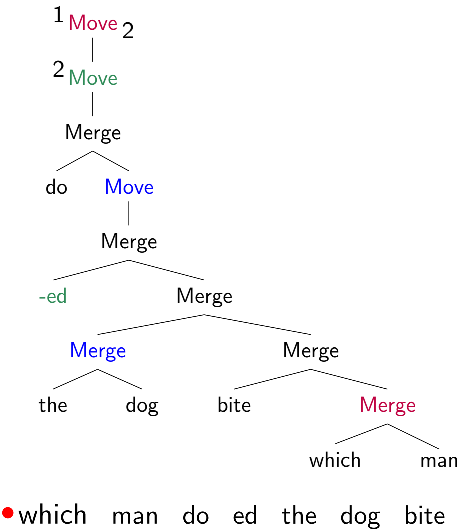
- **Index**
  at which step the node is conjectured
- **Outdex**
  at which step the parser considers the node done



Parsing as Node Indexation

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

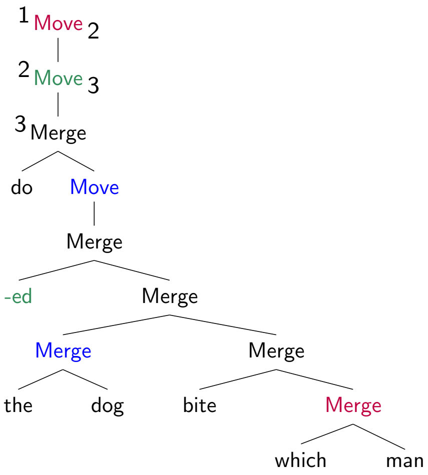
- **Index**
  at which step the node is conjectured
- **Outdex**
  at which step the parser considers the node done



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

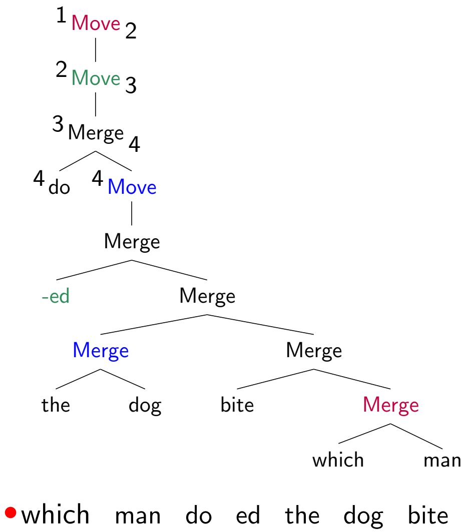


which man do ed the dog bite

Parsing as Node Indexation

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

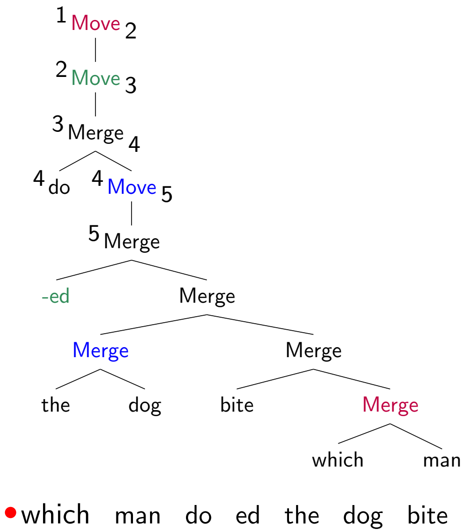
- **Index**
  at which step the node is conjectured
- **Outdex**
  at which step the parser considers the node done



Parsing as Node Indexation

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

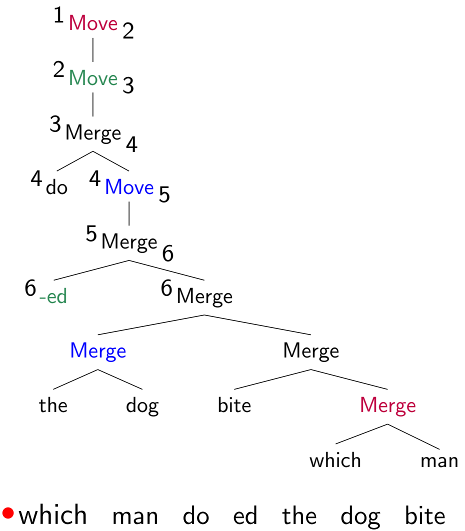
- **Index**
  at which step the node is conjectured
- **Outdex**
  at which step the parser considers the node done



Parsing as Node Indexation

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

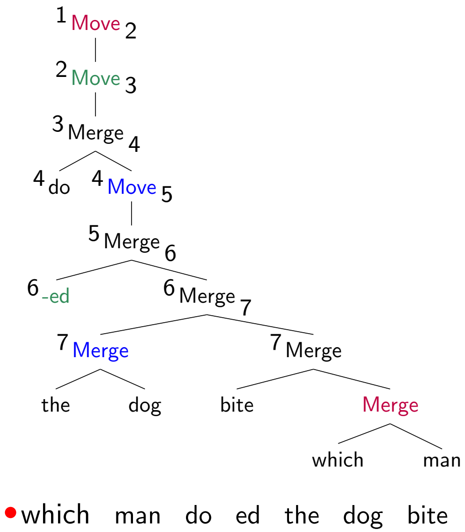
- **Index**
  at which step the node is conjectured
- **Outdex**
  at which step the parser considers the node done



Parsing as Node Indexation

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

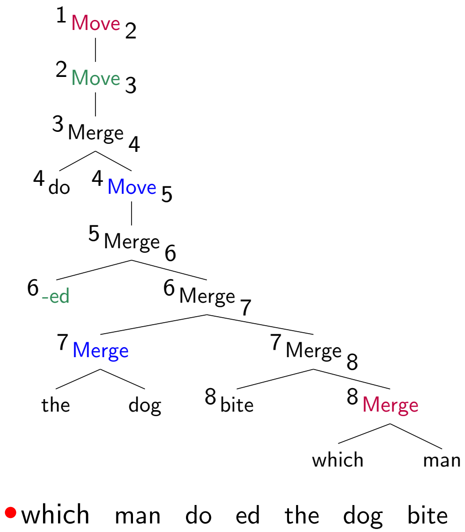
- **Index**
  at which step the node is conjectured
- **Outdex**
  at which step the parser considers the node done



Parsing as Node Indexation

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

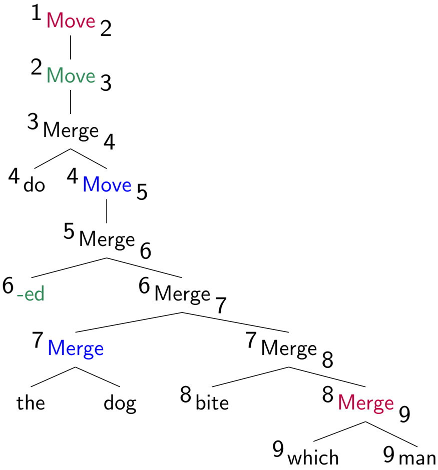
- **Index**
  at which step the node is conjectured
- **Outdex**
  at which step the parser considers the node done



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

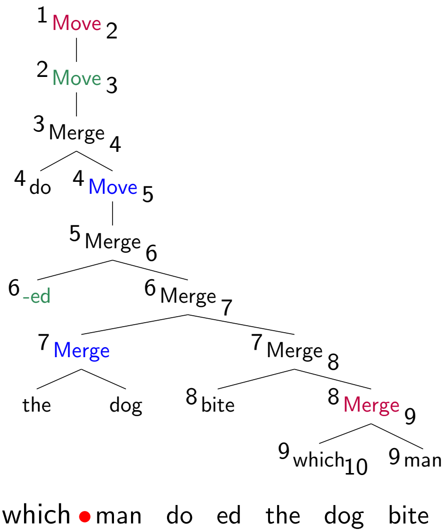


• which man do ed the dog bite

Parsing as Node Indexation

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

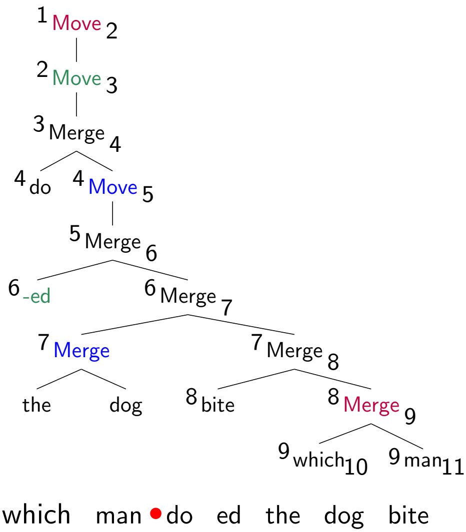
- **Index**
  at which step the node is conjectured
- **Outdex**
  at which step the parser considers the node done



Parsing as Node Indexation

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

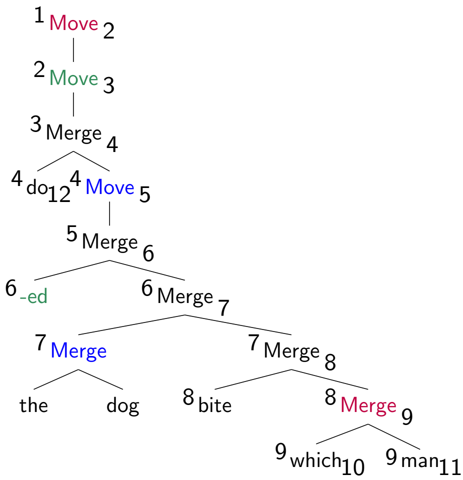
- **Index**
  at which step the node is conjectured
- **Outdex**
  at which step the parser considers the node done



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

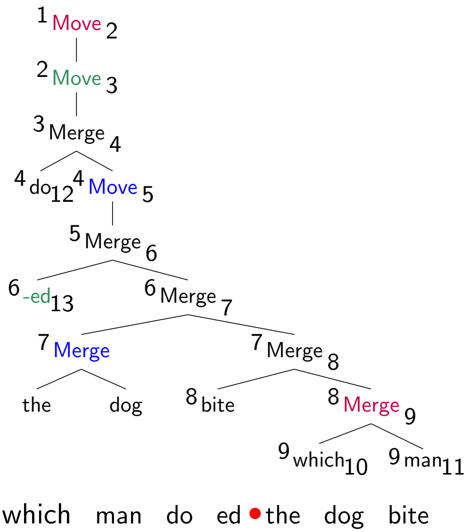


which man do ● ed the dog bite

Parsing as Node Indexation

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

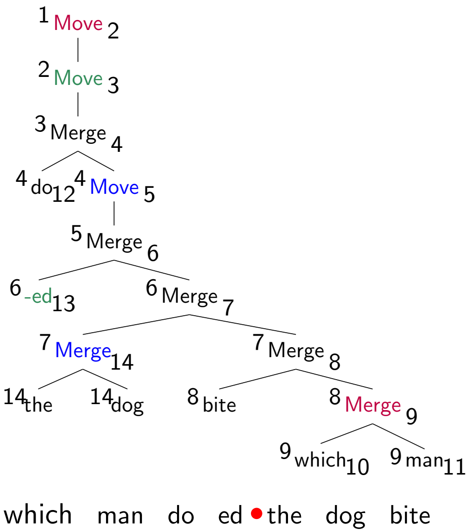
- **Index**
  at which step the node is conjectured
- **Outdex**
  at which step the parser considers the node done



Parsing as Node Indexation

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

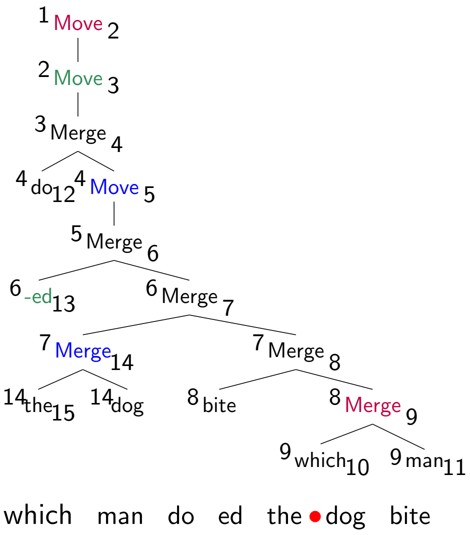
- **Index**
  at which step the node is conjectured
- **Outdex**
  at which step the parser considers the node done



Parsing as Node Indexation

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

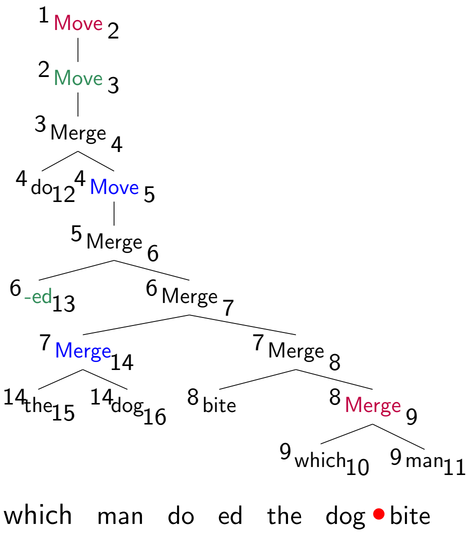
- **Index**
  at which step the node is conjectured
- **Outdex**
  at which step the parser considers the node done



Parsing as Node Indexation

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

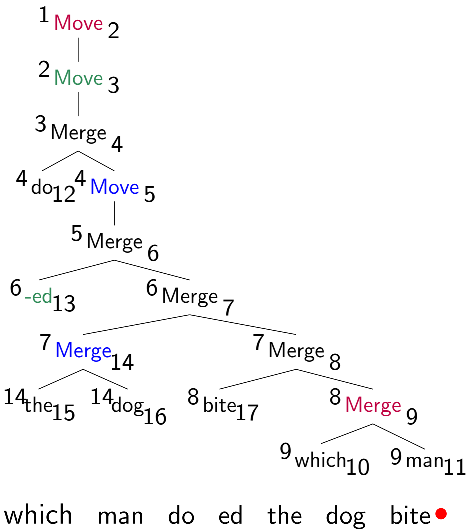
- **Index**
  at which step the node is conjectured
- **Outdex**
  at which step the parser considers the node done



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



Defining the Node Indexation Scheme

- $index(root) = 1$
- $index(n) = outdex(\text{mother of } n)$
- if
  - $m \neq n$,
  - $index(m) = index(n) = i$,
  - n reflexively dominates a node that is not string preceded by any node reflexively dominated by m,

  then $outdex(n) = index(n) + 1$
- otherwise, $outdex(n) = \max(i + 1, j + 1)$, where $j \geq 0$ is greatest among the outdices of all nodes that
  - string precede n and
  - are not reflexively dominated by n.

Relating Parsing and Processing

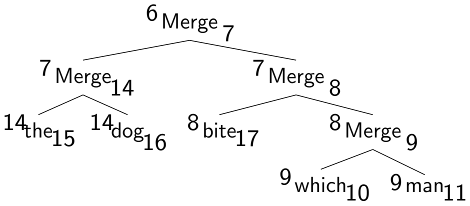
- **General Approach**
  (Kobele et al. 2012; Graf and Marcinek 2014)
  - pick competing syntactic analyses
  - pick metric to relate parsing behavior to processing difficulty
  - see which analysis gets it right
- **Simplifying Assumption**
  - consider only parser's behavior for correct parse
  - factors out problem of finding correct parse
- **Appeal**
  - maximally simple
  - MGs allow for explicit, linguistically sophisticated analyses
  - fully specified parsing model with precise predictions

Two Notions of Memory Usage

All metrics in Kobele et al. (2012) and Graf and Marcinek (2014) 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
$|\{p \mid outdex(p) - index(p) > 2\}|$

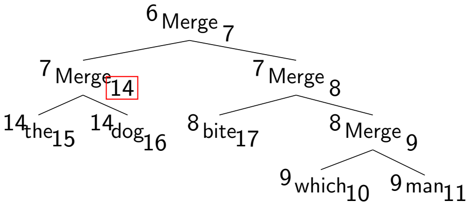


Two Notions of Memory Usage

All metrics in Kobele et al. (2012) and Graf and Marcinek (2014) 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
$|\{p \mid outdex(p) - index(p) > 2\}|$

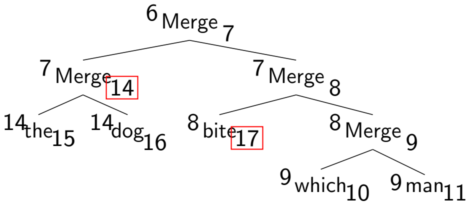


Two Notions of Memory Usage

All metrics in Kobele et al. (2012) and Graf and Marcinek (2014) 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
$|\{p \mid outdex(p) - index(p) > 2\}|$



Memory-Based Metrics of Processing Difficulty

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

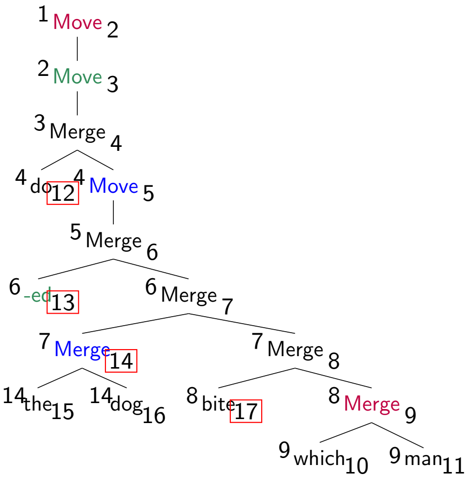
Max^{R} vector of tenure for all nodes, in decreasing order

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

Overview of Previous 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	≈	✓	✓	✓

Overview of Previous 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	≈	✓	✓	✓

RCs in East Asian

RCs **precede the modified noun** in Chinese, Japanese, Korean.

(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.

What's the Problem?

- The changes in word order do not affect the SRC advantage.
- Previous processing models (memory-based, surprisal) incorrectly predicted ORC advantage.
- **Recent Success**
  Yun et al. (2014) using MGs and surprisal

Question

Can our much simpler approach derive the SRC advantage?

Jiwon Yun

John Hale

What's the Problem?

- The changes in word order do not affect the SRC advantage.
- Previous processing models (memory-based, surprisal) incorrectly predicted ORC advantage.
- **Recent Success**
  Yun et al. (2014) using MGs and surprisal

Question

Can our much simpler approach derive the SRC advantage?

Jiwon Yun

John Hale

Two Movement Analyses of RCs

- **Promotion Analysis**
  noun starts in gap position and moves out of RC

  (9) a. t_i attacked the senator who reporter$_i$
  b. the senator attacked t_i who reporter$_i$

- **Wh-Movement Analysis**
  relative pronoun starts in gap position and moves into Spec,RC

  (10) a. t_i attacked the senator who$_i$ reporter
  b. the senator attacked t_i who$_i$ reporter

- Both analyses require additional movement steps to get the right word order.

Overview of RC-Processing Predictions

		Promotion			Wh-Movement		
		all	lex.	pron.	all	lex.	pron.
Korean	**Max**	tie	tie	tie	tie	tie	tie
	Max^R	ORC	ORC	ORC	ORC	ORC	ORC
	Sum	ORC	ORC	ORC	ORC	ORC	ORC
	Box	tie	ORC	ORC	ORC	ORC	ORC

		Promotion			Wh-Movement		
		all	lex.	pron.	all	lex.	pron.
Chinese	**Max**	tie	tie	tie	tie	tie	tie
	Max^R	ORC	ORC	ORC	ORC	ORC	ORC
	Sum	SRC	ORC	ORC	tie	ORC	ORC
	Box	SRC	SRC	tie	SRC	tie	ORC

An Ancillary Metric

- None of the current metrics derive SRC advantage for Korean.
- **But:** We have plenty of ties with **Max**.
  Can we turn those into SRC preferences?

  Gap "prefer short movement paths" (cf. O'Grady 2011)
  $\sum_{p \text{ a mover}} \mathit{index}(p) - \mathit{index}(\text{final landing site of } p)$

- If **Gap** is used to resolve ties, **Max** consistently favors SRCs.
- But what motivates **Gap**?

Gap Approximates Parse Item Size

- There are at least three aspects of memory usage:

Memory Usage	Parsing Concept
time item is in memory	tenure
number of items in memory	payload
amount of memory consumed by item	?

- Each node in the derivation corresponds to a parse item with two components
  - remaining features of the current head,
  - list of movers that still need to be found.
- The more movers an item contains, the more bits are required for its encoding.
- The longer a movement path, the more items have to contain the mover.
- Short movement paths minimize memory usage.

Current Lay of the Land

- **Max** + **Gap** captures SRC preference.
- The joint metric also makes the right predictions for
  - Center/Right
  - Center/Crossing
  - SC/RC vs RC/SC
  - SRC vs ORC in English
- **Downside:** choice of syntactic analysis immaterial

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

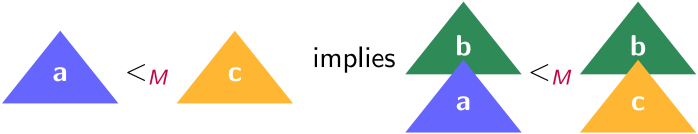
Solution

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

A Property of **Max**

A metric M is **embedding invariant** iff



Theorem

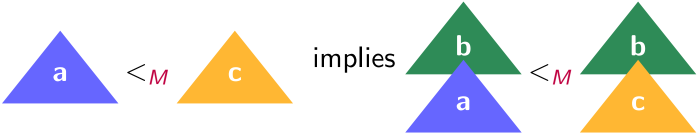
Max *is not embedding invariant.*

Proof.

- Suppose ${}^i r_j$ has a left sibling l with m nodes (and no movement takes place).
- Then $j \geq i + m$.
- *Reason*: the parser introduces r and l in the same step but explores r only after l is completed. □

A Property of **Max**

A metric M is **embedding invariant** iff



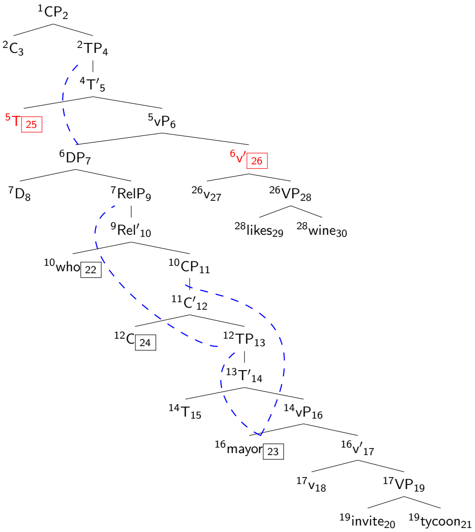
Theorem

Max *is not embedding invariant.*

Proof.

- Suppose ${}^{i}\mathrm{r}_{j}$ has a left sibling l with m nodes (and no movement takes place).
- Then $j \geq i + m$.
- *Reason*: the parser introduces r and l in the same step but explores r only after l is completed. $\square$

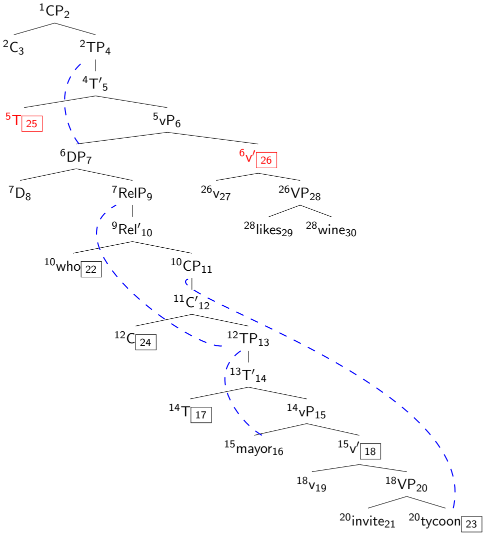
Explaining the Failure of **Max**



Intuition

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

Explaining the Failure of **Max**



Intuition

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

Other Rankings are Embedding Invariant

Theorem

Box, **Gap**, **Max**$^{\mathbf{R}}$, *and* **Sum** *are invariant under 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 RC cases can be analyzed as 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**, **Max**$^{\mathbf{R}}$, *and* **Sum** *are invariant under 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 RC cases can be analyzed as 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**, **Max**$^{\mathbf{R}}$, *and* **Sum** *are invariant under 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 RC cases can be analyzed as embeddings of distinct DPs into the same matrix clause.
- So why do most of these metrics fail nonetheless?

Failure in Chinese (Intuition)

- Subject is structurally prominent (Spec,vP) and thus conjectured early.
- In the SRC, the subject moves to the right and thus cannot be completed until the whole vP has been completed $\Rightarrow$ high tenure on subject
- In contrast, rightward movement of the object comes for free since the object is the rightmost part of the vP.

Intuition

ORC is preferred to SRC because object extraction does not delay the processing of other material.

Failure in Korean (Intuition)

- Movements that invert string order with respect to derivational order cause tenure.
- Objects are by default post-verbal in the derivation tree.
- But Korean is SOV, so objects are preverbal $\Rightarrow$ tenure
- In ORCs, the object ends up in a postverbal position again $\Rightarrow$ no tenure

Intuition

ORC is preferred to SRC because object extraction undoes the standard penalties accrued by the Korean SOV order.

Conclusion I: The Narrow View

- Syntax and processing can be related in an explicit fashion.
- Linking hypothesis via metrics of memory usage:
  - time an item stays in memory
  - how many items are kept in memory
  - size of items in memory
- **Max** + **Gap** covers a wide range of phenomena.
- Next step: look at Basque
  - consistent ORC preference reported
  - ergative language ⇒ different structure?

Conclusion II: The Bigger Picture

- Modeling by itself is not enough.
- The current approach 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, 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.

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

References II

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.

O'Grady, William. 2011. Relative clauses. processing and acquisition. In *Processing, typology, and function*, ed. Evan Kidd, 13–38. Amsterdam: John Benjamins.

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.

Yun, Jiwon, Zhong Chen, Tim Hunter, John Whitman, and John Hale. 2014. Uncertainty in processing relative clauses across East Asian languages. *Journal of East Asian Linguistics* 1–36.