Constrained Scrambling in CCG

A case study in Japanese

wren ng thornton

wrnthorn@indiana.edu

Cognitive Science
Indiana University Bloomington

Outline of the Talk

- Introduction to CCG
- Subject and object marking in Japanese
  - Problems with the simple approach
  - Radical Neo-Davidsonian semantics
  - A new semantic calculus
- Constraints on free order
  - Arguments of affective predicates
  - Argument vs. adjunct uses of the dative
- Fine tuning the calculus
  - Scrambling into subordinate clauses
- Future work

Combinatory Categorial Grammar

- Why CCG?

Combinatory Categorial Grammar

- Why CCG?
  - A natural account of coordination with ellipsis
  - Algorithms for on-line and partial parsing
  - Semantics occurs in synchrony with syntax
    - No arbitrary restructuring of complete trees

Combinatory Categorial Grammar

- Why CCG?
  - A natural account of coordination with ellipsis
  - Algorithms for on-line and partial parsing
  - Semantics occurs in synchrony with syntax
    - No arbitrary restructuring of complete trees
- How does it do that?

Combinatory Categorial Grammar

- Why CCG?
  - A natural account of coordination with ellipsis
  - Algorithms for on-line and partial parsing
  - Semantics occurs in synchrony with syntax
    - No arbitrary restructuring of complete trees
- How does it do that?
  - By abandoning traditional notions of constituency

Combinatory Categorial Grammar

- Why CCG?
  - A natural account of coordination with ellipsis
  - Algorithms for on-line and partial parsing
  - Semantics occurs in synchrony with syntax
    - No arbitrary restructuring of complete trees
- How does it do that?
  - By abandoning traditional notions of constituency
- Does it simplify accounts of syntax in non-configurational languages?

Combinatory Categorial Grammar

- Why CCG?
  - A natural account of coordination with ellipsis
  - Algorithms for on-line and partial parsing
  - Semantics occurs in synchrony with syntax
    - No arbitrary restructuring of complete trees
- How does it do that?
  - By abandoning traditional notions of constituency
- Does it simplify accounts of syntax in non-configurational languages?
  - Not as easily as one may hope
  - …but I'll show that it can

CCG: Rich category labels

- Bi-directional functional notation

CCG: Rich category labels

- Bi-directional functional notation
  - "Result on left"
    - $B \swarrow A$ — takes an A on the right, returns a B
    - $B \nwarrow A$ — takes an A on the left, returns a B

CCG: Rich category labels

- Bi-directional functional notation
  - "Result on left"
    - $B \swarrow A$ — takes an A on the right, returns a B
    - $B \nwarrow A$ — takes an A on the left, returns a B
  - "Result on top" (Not used in CCG)
    - $B \swarrow A$ — takes an A on the right, returns a B
    - $A \searrow B$ — takes an A on the left, returns a B

CCG: Rich category labels

- Bi-directional functional notation
  - "Result on left"
    - $B\swarrow A$ — takes an A on the right, returns a B
    - $B\nwarrow A$ — takes an A on the left, returns a B
  - "Result on top" (Not used in CCG)
    - $B\swarrow A$ — takes an A on the right, returns a B
    - $A\searrow B$ — takes an A on the left, returns a B
- Associates to the left
  - $C\swarrow B\swarrow A \equiv (C\swarrow B)\swarrow A$
  - $C\nwarrow B\swarrow A \equiv (C\nwarrow B)\swarrow A$
  - $C\swarrow B\nwarrow A \equiv (C\swarrow B)\nwarrow A$
  - $C\nwarrow B\nwarrow A \equiv (C\nwarrow B)\nwarrow A$

CCG: Rich category labels

- Bi-directional functional notation
  - "Result on left"
    - $B\swarrow A$ — takes an A on the right, returns a B
    - $B\nwarrow A$ — takes an A on the left, returns a B
  - "Result on top" (Not used in CCG)
    - $B\swarrow A$ — takes an A on the right, returns a B
    - $A\searrow B$ — takes an A on the left, returns a B
- Associates to the left
  - $C\swarrow B\swarrow A \equiv (C\swarrow B)\swarrow A$
  - $C\nwarrow B\swarrow A \equiv (C\nwarrow B)\swarrow A$
  - $C\swarrow B\nwarrow A \equiv (C\swarrow B)\nwarrow A$
  - $C\nwarrow B\nwarrow A \equiv (C\nwarrow B)\nwarrow A$
- Higher-order functions are allowed
  - $C\swarrow(B\swarrow A)$ takes a $B\swarrow A$ and returns a C

CCG: Basic Rules

Application $\dfrac{B\swarrow A : f \quad A : a}{B : f\,a}{>}$ $\qquad$ $\dfrac{A : a \quad B\nwarrow A : f}{B : f\,a}{<}$

Composition $\dfrac{C\swarrow B : f \quad B\swarrow A : g}{C\swarrow A : \lambda a.\, f\,(g\,a)}{>\mathbf{B}}$ $\qquad$ $\dfrac{B\nwarrow A : g \quad C\nwarrow B : f}{C\nwarrow A : \lambda a.\, f\,(g\,a)}{<\mathbf{B}}$

- Note for computer scientists
  - Both the colon and the arrows are backwards from the usual notation for typed λ-calculi

Application $\dfrac{f : A \to B \quad \$ \quad a : A}{f\,a : B}$

Composition $\dfrac{f : B \to C \quad \circ \quad g : A \to B}{\lambda a.\, f\,(g\,a) : A \to C}$

CCG: Some More Rules

Crossing Composition

$$\frac{C\swarrow B : f \quad B\nwarrow A : g}{C\nwarrow A : \lambda a.\, f\,(g\,a)}\,{>}\mathbf{Bx} \qquad \frac{B\swarrow A : g \quad C\nwarrow B : f}{C\swarrow A : \lambda a.\, f\,(g\,a)}\,{<}\mathbf{Bx}$$

Substitution

$$\frac{C\swarrow B\swarrow A : f \quad B\swarrow A : g}{C\swarrow A : \lambda a.\, f\,a\,(g\,a)}\,{>}\mathbf{S} \qquad \frac{B\nwarrow A : g \quad C\nwarrow B\nwarrow A : f}{C\nwarrow A : \lambda a.\, f\,a\,(g\,a)}\,{<}\mathbf{S}$$

Crossing Substitution

$$\frac{C\swarrow B\nwarrow A : f \quad B\nwarrow A : g}{C\nwarrow A : \lambda a.\, f\,a\,(g\,a)}\,{>}\mathbf{Sx} \qquad \frac{B\swarrow A : g \quad C\nwarrow B\swarrow A : f}{C\swarrow A : \lambda a.\, f\,a\,(g\,a)}\,{<}\mathbf{Sx}$$

Type-raising

$$\frac{A : a}{B\swarrow(B\nwarrow A) : \lambda f.\, f\,a}\,{>}\mathbf{T} \qquad \frac{A : a}{B\nwarrow(B\swarrow A) : \lambda f.\, f\,a}\,{<}\mathbf{T}$$

- Make up your own!

CCG: Some More Rules

Crossing Composition

$$\frac{C/B : f \quad B\backslash A : g}{C\backslash A : \lambda a.\, f\,(g\,a)}\,{>}\mathbf{Bx} \qquad \frac{B/A : g \quad C\backslash B : f}{C/A : \lambda a.\, f\,(g\,a)}\,{<}\mathbf{Bx}$$

Substitution

$$\frac{C/B/A : f \quad B/A : g}{C/A : \lambda a.\, f\,a\,(g\,a)}\,{>}\mathbf{S} \qquad \frac{B\backslash A : g \quad C\backslash B\backslash A : f}{C\backslash A : \lambda a.\, f\,a\,(g\,a)}\,{<}\mathbf{S}$$

Crossing Substitution

$$\frac{C/B\backslash A : f \quad B\backslash A : g}{C\backslash A : \lambda a.\, f\,a\,(g\,a)}\,{>}\mathbf{Sx} \qquad \frac{B/A : g \quad C\backslash B/A : f}{C/A : \lambda a.\, f\,a\,(g\,a)}\,{<}\mathbf{Sx}$$

Type-raising

$$\frac{A : a}{B/(B\backslash A) : \lambda f.\, f\,a}\,{>}\mathbf{T} \qquad \frac{A : a}{B\backslash(B/A) : \lambda f.\, f\,a}\,{<}\mathbf{T}$$

- Make up your own! …but be careful

CCG: Some More Rules

Crossing Composition

$$\frac{C/B : f \quad B\backslash A : g}{C\backslash A : \lambda a.\, f\,(g\,a)}\,{>}\mathbf{Bx} \qquad \frac{B/A : g \quad C\backslash B : f}{C/A : \lambda a.\, f\,(g\,a)}\,{<}\mathbf{Bx}$$

Substitution

$$\frac{C/B/A : f \quad B/A : g}{C/A : \lambda a.\, f\,a\,(g\,a)}\,{>}\mathbf{S} \qquad \frac{B\backslash A : g \quad C\backslash B\backslash A : f}{C\backslash A : \lambda a.\, f\,a\,(g\,a)}\,{<}\mathbf{S}$$

Crossing Substitution

$$\frac{C/B\backslash A : f \quad B\backslash A : g}{C\backslash A : \lambda a.\, f\,a\,(g\,a)}\,{>}\mathbf{Sx} \qquad \frac{B/A : g \quad C\backslash B/A : f}{C/A : \lambda a.\, f\,a\,(g\,a)}\,{<}\mathbf{Sx}$$

Type-raising

$$\frac{A : a}{B/(B\backslash A) : \lambda f.\, f\,a}\,{>}\mathbf{T} \qquad \frac{A : a}{B\backslash(B/A) : \lambda f.\, f\,a}\,{<}\mathbf{T}$$

- Make up your own! …but be careful
- **Bx**, **S**, and **Sx** are not theorems in Lambek calculus

CCG: Some More Rules

Crossing Composition

$$\frac{C/B : f \quad B\backslash A : g}{C\backslash A : \lambda a.\, f\,(g\,a)}\,{>}\mathbf{Bx} \qquad \frac{B/A : g \quad C\backslash B : f}{C/A : \lambda a.\, f\,(g\,a)}\,{<}\mathbf{Bx}$$

Substitution

$$\frac{C/B/A : f \quad B/A : g}{C/A : \lambda a.\, f\,a\,(g\,a)}\,{>}\mathbf{S} \qquad \frac{B\backslash A : g \quad C\backslash B\backslash A : f}{C\backslash A : \lambda a.\, f\,a\,(g\,a)}\,{<}\mathbf{S}$$

Crossing Substitution

$$\frac{C/B\backslash A : f \quad B\backslash A : g}{C\backslash A : \lambda a.\, f\,a\,(g\,a)}\,{>}\mathbf{Sx} \qquad \frac{B/A : g \quad C\backslash B/A : f}{C/A : \lambda a.\, f\,a\,(g\,a)}\,{<}\mathbf{Sx}$$

Type-raising

$$\frac{A : a}{B/(B\backslash A) : \lambda f.\, f\,a}\,{>}\mathbf{T} \qquad \frac{A : a}{B\backslash(B/A) : \lambda f.\, f\,a}\,{<}\mathbf{T}$$

- Make up your own! …but be careful
- **Bx**, **S**, and **Sx** are not theorems in Lambek calculus
- **T** are not syntax directed

CCG: Some More Rules

Crossing Composition

$$\frac{C/B : f \quad B\backslash A : g}{C\backslash A : \lambda a.\, f\,(g\,a)}\,{>}\mathbf{Bx} \qquad \frac{B/A : g \quad C\backslash B : f}{C/A : \lambda a.\, f\,(g\,a)}\,{<}\mathbf{Bx}$$

Substitution

$$\frac{C/B/A : f \quad B/A : g}{C/A : \lambda a.\, f\,a\,(g\,a)}\,{>}\mathbf{S} \qquad \frac{B\backslash A : g \quad C\backslash B\backslash A : f}{C\backslash A : \lambda a.\, f\,a\,(g\,a)}\,{<}\mathbf{S}$$

Crossing Substitution

$$\frac{C/B\backslash A : f \quad B\backslash A : g}{C\backslash A : \lambda a.\, f\,a\,(g\,a)}\,{>}\mathbf{Sx} \qquad \frac{B/A : g \quad C\backslash B/A : f}{C/A : \lambda a.\, f\,a\,(g\,a)}\,{<}\mathbf{Sx}$$

Type-raising

$$\frac{A : a}{B/(B\backslash A) : \lambda f.\, f\,a}\,{>}\mathbf{T} \qquad \frac{A : a}{B\backslash(B/A) : \lambda f.\, f\,a}\,{<}\mathbf{T}$$

- Make up your own! …but be careful
- **Bx**, **S**, and **Sx** are not theorems in Lambek calculus
- **T** are not syntax directed
- It's easy to *lose* formal power

Outline of the Talk

- Introduction to CCG
- Subject and object marking in Japanese
  - Problems with the simple approach
  - Radical Neo-Davidsonian semantics
  - A new semantic calculus
- Constraints on free order
  - Arguments of affective predicates
  - Argument vs. adjunct uses of the dative
- Fine tuning the calculus
  - Scrambling into subordinate clauses
- Future work

Subject and Object Particles

- Japanese is a head-final agglutinative language
- Case marking for primary arguments
  - Intransitive predicates
    - *ga* — for the single argument
  - Transitive affective predicates
    - *ga* — primary affect
    - *ga* — secondary affect, too
  - Transitive operational predicates
    - *ga* — subject/agent
    - *wo* — direct object/patient

First Approximation

- *Tarou -ga* *hon -wo* *yon-da*
  — SUBJ book ACC read-PERF

  ‘Taro read the book.’

- $$\dfrac{\dfrac{\dfrac{\dfrac{\textit{Tarou}}{n} \quad \dfrac{\textit{ga}}{np_{subj} \nwarrow n}}{np_{subj}} < \dfrac{\dfrac{\dfrac{\textit{hon}}{n} \quad \dfrac{\textit{wo}}{np_{acc} \nwarrow n}}{np_{acc}} < \dfrac{\textit{yonda}}{s \nwarrow np_{subj} \nwarrow np_{acc}}}{s \nwarrow np_{subj}} <}{s} <$$

- Looks fine to me

First Approximation: Danger Ahead

hon	*-wo*	*Tarou*	*-ga*	*yon-da*
book	ACC	—	SUBJ	read-PERF

  ‘Taro read the book.’

- Derivation:

  $$\dfrac{\dfrac{\dfrac{\textit{hon}}{n} \quad \dfrac{\textit{wo}}{np_{\text{acc}} \nwarrow n}}{np_{\text{acc}}}< \quad \dfrac{\dfrac{\dfrac{\dfrac{\textit{Tarou}}{n} \quad \dfrac{\textit{ga}}{np_{\text{subj}} \nwarrow n}}{np_{\text{subj}}}<}{s \swarrow (s \nwarrow np_{\text{subj}})}>\mathbf{T} \quad \dfrac{\textit{yonda}}{(s \nwarrow np_{\text{subj}}) \nwarrow np_{\text{acc}}}}{s \nwarrow np_{\text{acc}}}>\mathbf{Bx}}{s}<$$

- Requires both **T** and **Bx** to handle free order,
  Which is dangerously close to losing formal power!
- This ordering is non-default, but it isn’t marked,
  Is the syntactic complexity cognitively justified?

First Approximation: Postmortem

- Even more problems ahead
  - All six orders possible for ditransitives
    - Not to mention adjuncts, etc.
  - Almost any argument can be dropped
    - CCG does not permit functional categories at the root of a derivation
- So what went wrong?
  - No reason to think the particles are at fault
  - It seems to be a problem with the predicate
  - Even though derivations are *bi-directional* they're still inherently *linear*
- But CCG is a theory of syntax *and semantics*
  Could semantics help?

Neo-Davidsonian Semantics

- Reify “eventualities” as objects whose properties can be inspected
- Instead of building up structured terms…
  - $read(Tarou, book)$
- …describe events by an intersection of constraints
  - $\exists e \in \mathscr{E}.$
    $event(reading, e) \wedge subj(Tarou, e) \wedge obj(book, e)$
- Radical versions end up eliding the distinction between arguments and adjuncts
  - Everything is treated as an adjunct
  - This can be problematic semantically
    - …but what about syntactically?

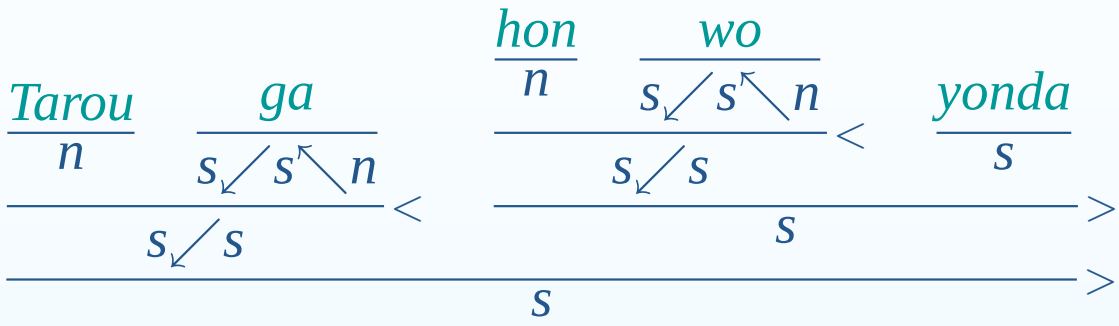
Why *not* make everything an adjunct?

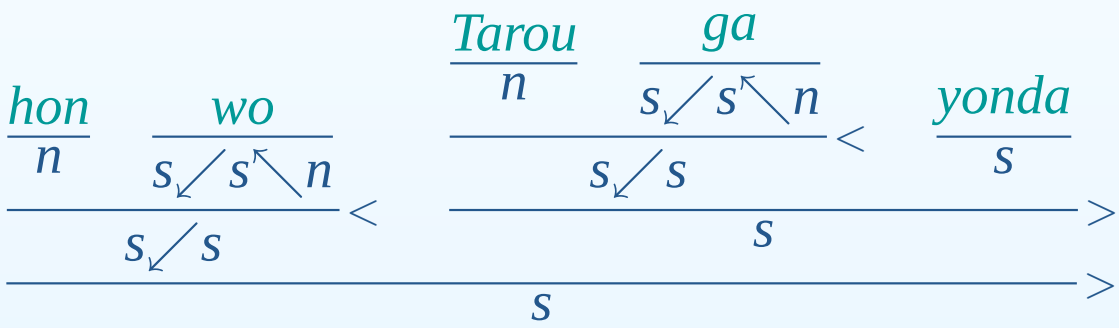
- $\dfrac{\dfrac{\dfrac{\textit{Tarou}}{n} \quad \dfrac{\textit{ga}}{s\swarrow s\nwarrow n}}{s\swarrow s} < \quad \dfrac{\dfrac{\textit{hon}}{n} \quad \dfrac{\textit{wo}}{s\swarrow s\nwarrow n}}{s\swarrow s} < \dfrac{\textit{yonda}}{s}}{s} >$

- $\dfrac{\dfrac{\dfrac{\textit{hon}}{n} \quad \dfrac{\textit{wo}}{s\swarrow s\nwarrow n}}{s\swarrow s} < \quad \dfrac{\dfrac{\textit{Tarou}}{n} \quad \dfrac{\textit{ga}}{s\swarrow s\nwarrow n}}{s\swarrow s} < \dfrac{\textit{yonda}}{s}}{s} >$

- Same derivation for all argument orderings
- Optional arguments are handled seamlessly
- Easy as pie.

Why *not* make everything an adjunct?

- 

- 

- Same derivation for all argument orderings
- Optional arguments are handled seamlessly
- Easy as pie.
- . . .but we haven't eliminated the problem, only moved it from syntax into semantics

A New Semantic Calculus

- If the problem is only having one dimension for application, then what if we add extra dimensions?
- Label value expressions with a dimension
  - $[e]_A$
- Label every abstraction with an argument dimension
  - $(\lambda_A a.e)$
- Define application only for matching dimensions
  - $(\lambda_A a.e_1)\,[e_2]_B \rightsquigarrow_\beta \{a \mapsto e_2\}e_1$ iff $A = B$
- Introduce a reordering rule
  - $(\lambda_A a.\lambda_B b.e) \rightsquigarrow_\chi (\lambda_B b.\lambda_A a.e)$ iff $A \neq B$
  - Could be eliminated if we used a non-linear representation for terms

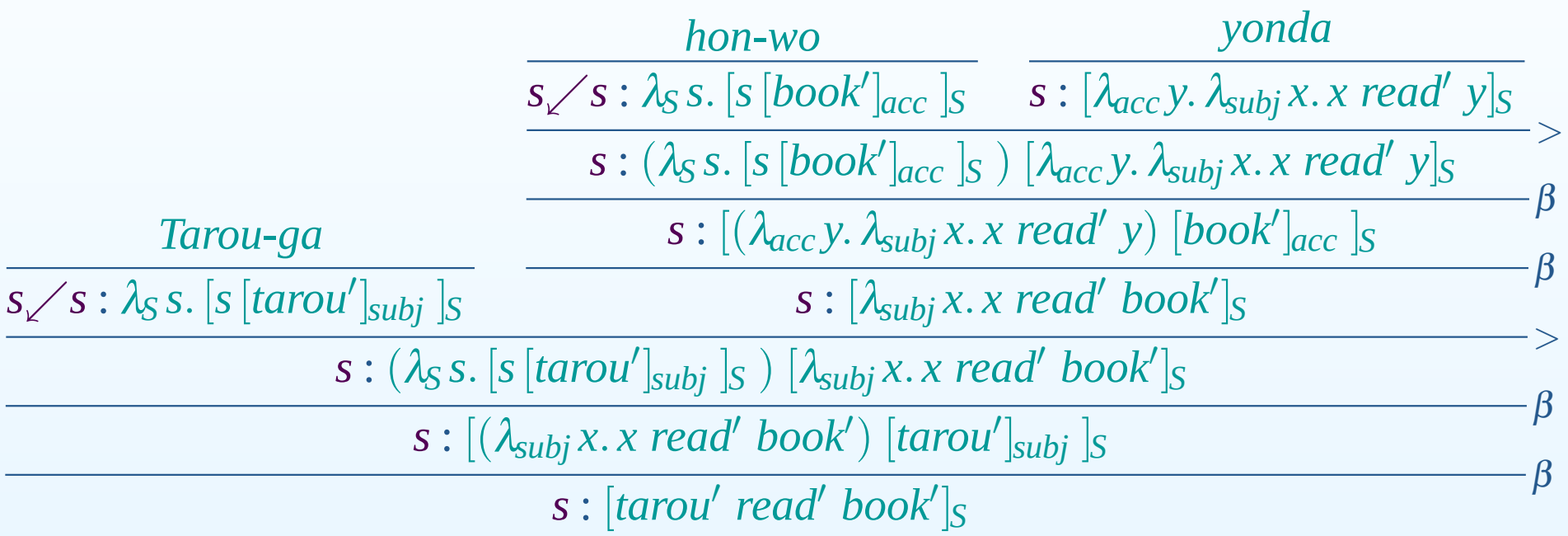
The New Calculus in Action

- Some preliminary derivations

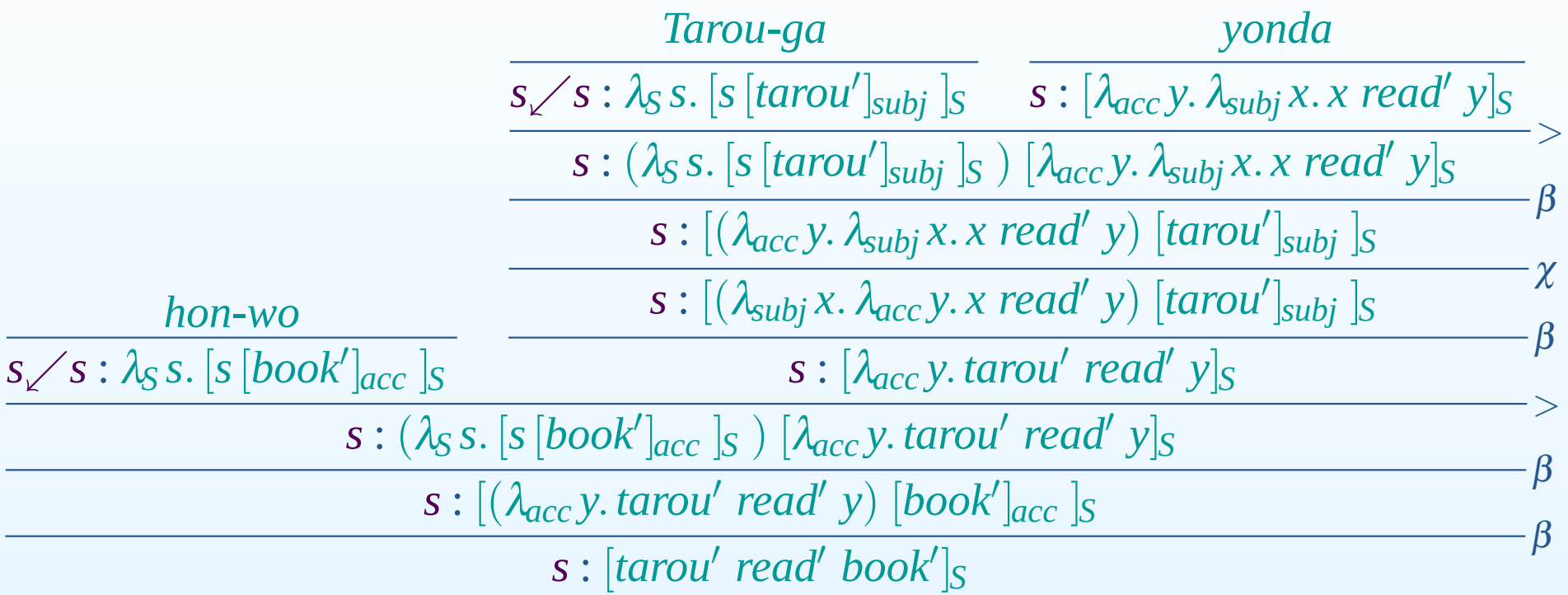
$$\dfrac{\dfrac{hon}{n : [book']_N} \quad \dfrac{wo}{s_{\swarrow} s^{\nwarrow} n : \lambda_N n.\, \lambda_S s.\, [s\, [n]_{acc}\,]_S}}{s_{\swarrow} s : (\lambda_N n.\, \lambda_S s.\, [s\, [n]_{acc}\,]_S\,)\, [book']_N}\, < \quad \dfrac{}{s_{\swarrow} s : \lambda_S s.\, [s\, [book']_{acc}\,]_S}\, \beta$$

$$\dfrac{\dfrac{Tarou}{n : [tarou']_N} \quad \dfrac{ga}{s_{\swarrow} s^{\nwarrow} n : \lambda_N n.\, \lambda_S s.\, [s\, [n]_{subj}\,]_S}}{s_{\swarrow} s : (\lambda_N n.\, \lambda_S s.\, [s\, [n]_{subj}\,]_S\,)\, [tarou']_N}\, < \quad \dfrac{}{s_{\swarrow} s : \lambda_S s.\, [s\, [tarou']_{subj}\,]_S}\, \beta$$

The New Calculus in Action



The New Calculus in Action



- The only difference is one use of χ-reordering

A Little Bit of Metatheory

- What linguistic predictions does it make?

A Little Bit of Metatheory

- What linguistic predictions does it make?
  - Typology
    - Inverse relation between presence of ordering restrictions vs. case marking

A Little Bit of Metatheory

- What linguistic predictions does it make?
  - Typology
    - Inverse relation between presence of ordering restrictions vs. case marking
  - Morpho-Syntax
    - Phrases & clauses are fundamentally different
    - Free order isn't entirely free

A Little Bit of Metatheory

- What linguistic predictions does it make?
  - Typology
    - Inverse relation between presence of ordering restrictions vs. case marking
  - Morpho-Syntax
    - Phrases & clauses are fundamentally different
    - Free order isn't entirely free
  - Morpho-Semantics
    - Case is not "meaningless" to semantics

A Little Bit of Metatheory

- What linguistic predictions does it make?
  - Typology
    - Inverse relation between presence of ordering restrictions vs. case marking
  - Morpho-Syntax
    - Phrases & clauses are fundamentally different
    - Free order isn't entirely free
  - Morpho-Semantics
    - Case is not "meaningless" to semantics
- What is its formal power?

A Little Bit of Metatheory

- What linguistic predictions does it make?
  - Typology
    - Inverse relation between presence of ordering restrictions vs. case marking
  - Morpho-Syntax
    - Phrases & clauses are fundamentally different
    - Free order isn't entirely free
  - Morpho-Semantics
    - Case is not "meaningless" to semantics
- What is its formal power?
  - We can embed the untyped λ-calculus
    e.g. $(\lambda x.\, x\, x)\ (\lambda x.\, x\, x) \Longrightarrow (\lambda_X\, x.\, x\, [x]_X\ )\ [\lambda_X\, x.\, x\, [x]_X\ ]_X$

A Little Bit of Metatheory

- What linguistic predictions does it make?
  - Typology
    - Inverse relation between presence of ordering restrictions vs. case marking
  - Morpho-Syntax
    - Phrases & clauses are fundamentally different
    - Free order isn't entirely free
  - Morpho-Semantics
    - Case is not "meaningless" to semantics
- What is its formal power?
  - We can embed the untyped λ-calculus
    e.g. $(\lambda x.\, x\, x)\ (\lambda x.\, x\, x) \Longrightarrow (\lambda_X\, x.\, x\, [x]_X\ )\ [\lambda_X\, x.\, x\, [x]_X\ ]_X$
  - But only because we took it as our basis

A Little Bit of Metatheory

- What linguistic predictions does it make?
  - Typology
    - Inverse relation between presence of ordering restrictions vs. case marking
  - Morpho-Syntax
    - Phrases & clauses are fundamentally different
    - Free order isn't entirely free
  - Morpho-Semantics
    - Case is not "meaningless" to semantics
- What is its formal power?
  - We can embed the untyped λ-calculus
    e.g. $(\lambda x.\, x\, x)\ (\lambda x.\, x\, x) \Longrightarrow (\lambda_X\, x.\, x\, [x]_X\ )\ [\lambda_X\, x.\, x\, [x]_X\ ]_X$
  - But only because we took it as our basis
  - Dimensions are orthogonal to types

Outline of the Talk

- Introduction to CCG
- Subject and object marking in Japanese
  - Problems with the simple approach
  - Radical Neo-Davidsonian semantics
  - A new semantic calculus
- Constraints on free order
  - Arguments of affective predicates
  - Argument vs. adjunct uses of the dative
- Fine tuning the calculus
  - Scrambling into subordinate clauses
- Future work

Arguments of affective predicates

- Primary affect always precedes secondary affect
- *sensei -ga* *gakusei -ga* *wakar-ana-i*
  teacher SUBJ student SUBJ understand-NEG-IMPF
  'The teacher doesn't understand the student'

Arguments of affective predicates

- Primary affect always precedes secondary affect
- *sensei -ga* *gakusei -ga* *wakar-ana-i*
  teacher SUBJ student SUBJ understand-NEG-IMPF

  'The teacher doesn't understand the student'
- Since the dimensions are the same, the new calculus degenerates into linear ordering

Arguments of affective predicates

- Primary affect always precedes secondary affect
- *sensei -ga* *gakusei -ga* *wakar-ana-i*
  teacher SUBJ student SUBJ understand-NEG-IMPF

  'The teacher doesn't understand the student'
- Since the dimensions are the same, the new calculus degenerates into linear ordering
  - Supports constrained-free-order hypothesis

Arguments of affective predicates

- Primary affect always precedes secondary affect
- *sensei -ga* *gakusei -ga* *wakar-ana-i*
  teacher SUBJ student SUBJ understand-NEG-IMPF

  'The teacher doesn't understand the student'
- Since the dimensions are the same, the new calculus degenerates into linear ordering
  - Supports constrained-free-order hypothesis
- θ-rôle assignment for affective predicates mirrors default argument order for operational predicates

Arguments of affective predicates

- Primary affect always precedes secondary affect
- *sensei -ga* *gakusei -ga* *wakar-ana-i*
  teacher SUBJ student SUBJ understand-NEG-IMPF

  'The teacher doesn't understand the student'
- Since the dimensions are the same, the new calculus degenerates into linear ordering
  - Supports constrained-free-order hypothesis
- θ-rôle assignment for affective predicates mirrors default argument order for operational predicates
  - Independent support that this semantic form is on the right track

Arguments of affective predicates

- Primary affect always precedes secondary affect
- *sensei -ga* *gakusei -ga* *wakar-ana-i*
  teacher SUBJ student SUBJ understand-NEG-IMPF

  'The teacher doesn't understand the student'
- Since the dimensions are the same, the new calculus degenerates into linear ordering
  - Supports constrained-free-order hypothesis
- θ-rôle assignment for affective predicates mirrors default argument order for operational predicates
  - Independent support that this semantic form is on the right track
- Overall, supports the case-vs.-order hypothesis?

Argument vs. Adjunct Datives

- There are many uses of the particle *ni*
  - I'll ignore most of them today

Argument vs. Adjunct Datives

- There are many uses of the particle *ni*
  - I'll ignore most of them today
- Adjunct datives always precede argument datives
- *tugi -no kaigi wa getuyoubi -ni doyoubi -ni*
  next GEN meeting TOP monday DAT saturday DAT
  kime-masi-ta
  decide_on-DIST-PERF

  'As for the next meeting, monday we decided on (it being) saturday.'

Argument vs. Adjunct Datives

- There are many uses of the particle *ni*
  - I'll ignore most of them today
- Adjunct datives always precede argument datives
- *tugi -no kaigi wa getuyoubi -ni doyoubi -ni*
  next GEN meeting TOP monday DAT saturday DAT
  kime-masi-ta
  decide_on-DIST-PERF

  'As for the next meeting, monday we decided on (it being) saturday.'
- Again, same dimensions for dative $\rightsquigarrow$ linear ordering

Argument vs. Adjunct Datives

- There are many uses of the particle *ni*
  - I'll ignore most of them today
- Adjunct datives always precede argument datives
- *tugi -no kaigi wa getuyoubi -ni doyoubi -ni*
  next GEN meeting TOP monday DAT saturday DAT
  kime-masi-ta
  decide_on-DIST-PERF

  'As for the next meeting, monday we decided on (it being) saturday.'
- Again, same dimensions for dative $\rightsquigarrow$ linear ordering
- Moving adjuncts to the "outside" matches our semantic intuitions about arguments vs. adjuncts

Outline of the Talk

- Introduction to CCG
- Subject and object marking in Japanese
  - Problems with the simple approach
  - Radical Neo-Davidsonian semantics
  - A new semantic calculus
- Constraints on free order
  - Arguments of affective predicates
  - Argument vs. adjunct uses of the dative
- Fine tuning the calculus
  - Scrambling into subordinate clauses
- Future work

Scrambling into Subordinate Clauses

- Some (but not all) speakers accept sentences like
- [*sono hon* *-wo* ⟨ *Hanako -ga* ⟩ *Tarou -ga* *kat-ta*] *-to*
  that book ACC — SUBJ — SUBJ buy-PERF COMP
  omot-te iru
  think.PROG

  ‘Hanako thinks that Taro bought that book.’

Scrambling into Subordinate Clauses

- Some (but not all) speakers accept sentences like
- [*sono hon -wo* ⟨ *Hanako -ga* ⟩ *Tarou -ga kat-ta*] *-to*
  that book ACC — SUBJ — SUBJ buy-PERF COMP
  omot-te iru
  think.PROG

  ‘Hanako thinks that Taro bought that book.’
- Where the matrix subject must occur before the subordinate subject

Scrambling into Subordinate Clauses

- Some (but not all) speakers accept sentences like
- [*sono hon* *-wo* ⟨ *Hanako -ga* ⟩ *Tarou -ga* *kat-ta*] *-to*
  that book ACC — SUBJ — SUBJ buy-PERF COMP
  omot-te iru
  think.PROG

  ‘Hanako thinks that Taro bought that book.’
- Where the matrix subject must occur before the subordinate subject
- But our grammar won’t accept it
  - *Hanako-ga* blocks *sono hon-wo* from being passed to *katta*
  - We can only analyze *sono hon-wo* as an argument/adjunct to *omotte iru*

Fine tuning the calculus

- Instead of reordering abstractions…
  - $(\lambda_A a.\, \lambda_B b.\, e) \rightsquigarrow_\chi (\lambda_B b.\, \lambda_A a.\, e)$ iff $A \neq B$

Fine tuning the calculus

- Instead of reordering abstractions…
  - $(\lambda_A a.\, \lambda_B b.\, e) \rightsquigarrow_\chi (\lambda_B b.\, \lambda_A a.\, e)$ iff $A \neq B$
- …we could have chosen to reorder applications
  - $e_1\, [e_2]_A\ [e_3]_B \rightsquigarrow_\xi e_1\, [e_3]_B\ [e_2]_A$ iff $A \neq B$

Fine tuning the calculus

- Instead of reordering abstractions…
  - $(\lambda_A a.\, \lambda_B b.\, e) \rightsquigarrow_\chi (\lambda_B b.\, \lambda_A a.\, e)$ iff $A \neq B$
- …we could have chosen to reorder applications
  - $e_1\, [e_2]_A\ [e_3]_B \rightsquigarrow_\xi e_1\, [e_3]_B\ [e_2]_A$ iff $A \neq B$
- They only differ in their treatment of
  - scrambling into subordinate clauses
  - filling in optional arguments from context

Fine tuning the calculus

- Instead of reordering abstractions…
  - $(\lambda_A a.\, \lambda_B b.\, e) \rightsquigarrow_\chi (\lambda_B b.\, \lambda_A a.\, e)$ iff $A \neq B$
- …we could have chosen to reorder applications
  - $e_1\, [e_2]_A\ [e_3]_B \rightsquigarrow_\xi e_1\, [e_3]_B\ [e_2]_A$ iff $A \neq B$
- They only differ in their treatment of
  - scrambling into subordinate clauses
  - filling in optional arguments from context
- Both rules seem reasonable, how can we choose?

Fine tuning the calculus

- Instead of reordering abstractions…
  - $(\lambda_A a.\, \lambda_B b.\, e) \rightsquigarrow_\chi (\lambda_B b.\, \lambda_A a.\, e)$ iff $A \neq B$
- …we could have chosen to reorder applications
  - $e_1\, [e_2]_A\ [e_3]_B \rightsquigarrow_\xi e_1\, [e_3]_B\ [e_2]_A$ iff $A \neq B$
- They only differ in their treatment of
  - scrambling into subordinate clauses
  - filling in optional arguments from context
- Both rules seem reasonable, how can we choose?
  - Native speakers disagree on grammaticality

Fine tuning the calculus

- Instead of reordering abstractions…
  - $(\lambda_A a.\, \lambda_B b.\, e) \rightsquigarrow_\chi (\lambda_B b.\, \lambda_A a.\, e)$ iff $A \neq B$
- …we could have chosen to reorder applications
  - $e_1\, [e_2]_A\ [e_3]_B \rightsquigarrow_\xi e_1\, [e_3]_B\ [e_2]_A$ iff $A \neq B$
- They only differ in their treatment of
  - scrambling into subordinate clauses
  - filling in optional arguments from context
- Both rules seem reasonable, how can we choose?
  - Native speakers disagree on grammaticality
  - Perhaps they've chosen differently?

Fine tuning the calculus

- Instead of reordering abstractions…
  - $(\lambda_A a.\, \lambda_B b.\, e) \rightsquigarrow_\chi (\lambda_B b.\, \lambda_A a.\, e)$ iff $A \neq B$
- …we could have chosen to reorder applications
  - $e_1\, [e_2]_A\, [e_3]_B \rightsquigarrow_\xi e_1\, [e_3]_B\, [e_2]_A$ iff $A \neq B$
- They only differ in their treatment of
  - scrambling into subordinate clauses
  - filling in optional arguments from context
- Both rules seem reasonable, how can we choose?
  - Native speakers disagree on grammaticality
  - Perhaps they've chosen differently?
  - This could be seen as another validated prediction

Outline of the Talk

- Introduction to CCG
- Subject and object marking in Japanese
  - Problems with the simple approach
  - Radical Neo-Davidsonian semantics
  - A new semantic calculus
- Constraints on free order
  - Arguments of affective predicates
  - Argument vs. adjunct uses of the dative
- Fine tuning the calculus
  - Scrambling into subordinate clauses
- Future work

Future Work

- Capturing the continuative and verbal structures
  - Complete treatment of datives for *na*-nouns vs. *no*-nouns
  - Verbal stems in purpose/manner constructions
  - Verbal patterns in honorific/humble *keigo*
- More metatheory
  - Prove properties about formal power, safety, etc.
  - Explore connections with other logics and category theory
    - Dimensions are similar to channels in π-calculus
    - Un/boxing is related to monads, staged compilation, and partial evaluation

~fin.