Course Structure

Lecturer: Camilo Thorne

Title: Semantic Complexity of Aggregate Noun Phrases

1. Day 1: Computing meaning
2. Day 2: Formal semantics of aggregation
3. Day 3: Semantic complexity of aggregation
4. Day 4: Distribution of aggregate quantifiers
5. Day 5: Exercises and discussion

1. Computing Meaning

Camilo Thorne[1]

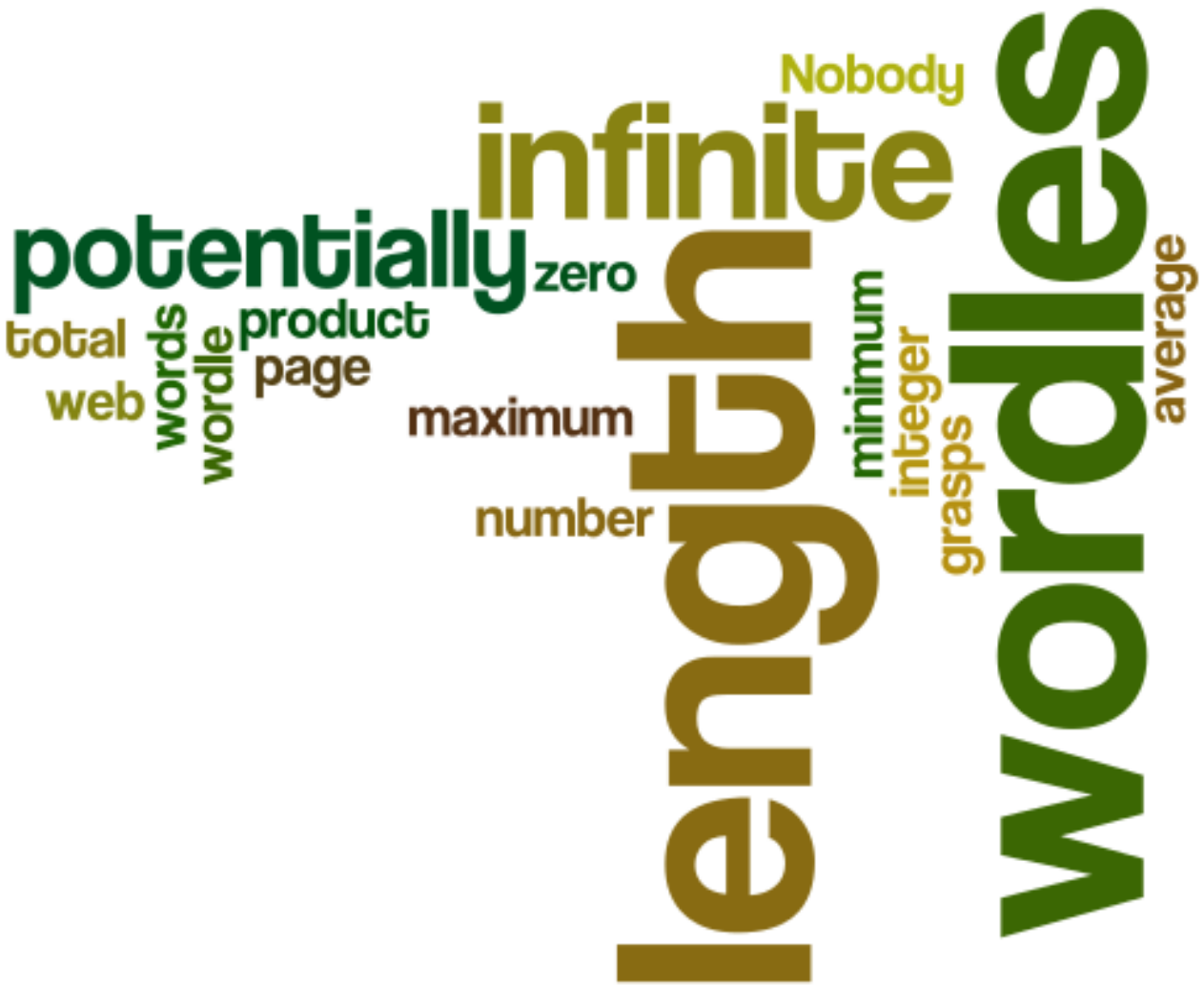
[1]KRDB Research Centre for Knowledge and Data
Free University of Bozen-Bolzano
cthorne@inf.unibz.it
http://www.inf.unibz.it/~cathorne

ESSLLI 2013, Aug 5-9, Düsseldorf

Outline

http://www.inf.unibz.it/~cathorne/agg_esslli

Motivation - Aggregation



Motivation - Aggregation (ctd.)

⋮

The average length of words in a wordle is not an integer.

Nobody grasps the maximum length of wordles.

The total length of wordles sucks.

The number of wordles is infinite.

The minimum length of wordles is zero.

The product length of wordles is potentially infinite.

⋮

Motivation - Aggregation (ctd.)

⋮

The average length of words in a wordle is not an integer.

Nobody grasps the maximum length of wordles.

The total length of wordles sucks.

The number of wordles is infinite.

The minimum length of wordles is zero.

The product length of wordles is potentially infinite.

⋮

Computational Semantics

- Seminal work by R. Montague in [Mon70, Mon73]
- The meaning of a natural language expression is given by its truth conditions
- Truth conditions can be described via logic meaning representations (MRs)

Computational Semantics

- Seminal work by R. Montague in [Mon70, Mon73]
- The meaning of a natural language expression is given by its truth conditions
- Truth conditions can be described via logic meaning representations (MRs)

Q1: How can we compute such MRs?

Computational Semantics

- Seminal work by R. Montague in [Mon70, Mon73]
- The meaning of a natural language expression is given by its truth conditions
- Truth conditions can be described via logic meaning representations (MRs)

Q1: How can we compute such MRs?

Q3: How do we model quantification?

First Order Logic - Syntax

- We want to describe meaning via truth conditions
- Can be achieved via first-order logic (FOL) MRs

First Order Logic - Syntax

- We want to describe meaning via truth conditions
- Can be achieved via first-order logic (FOL) MRs

Definition (Terms and Formulas)

First order terms and formulas are defined as follows:

1. every variable x and constant c is a term
2. if P is a predicate of arity n and $t_1, \ldots, t_n$ are terms $P(t_1, \ldots, t_n)$ is a formula
3. if φ, φ' are formulas, so are $\neg\varphi$ and $\varphi \wedge \varphi'$
4. if x is a variable and φ a formula, so is $\exists x\varphi$

First Order Logic - Syntax

- We want to describe meaning via truth conditions
- Can be achieved via first-order logic (FOL) MRs

Definition (Terms and Formulas)

First order terms and formulas are defined as follows:

1. every variable x and constant c is a term
2. if P is a predicate of arity n and $t_1, \ldots, t_n$ are terms $P(t_1, \ldots, t_n)$ is a formula
3. if φ, φ' are formulas, so are $\neg\varphi$ and $\varphi \wedge \varphi'$
4. if x is a variable and φ a formula, so is $\exists x\varphi$

- The free variables $FV(\varphi)$ of φ are those not bound to a quantifier
- $\varphi \vee \varphi' = \neg(\neg\varphi \wedge \neg\varphi')$, $\varphi \Rightarrow \varphi' = \neg\varphi \vee \varphi'$ and $\forall x\varphi = \neg\exists x\neg\varphi$

First Order Logic - Syntax

- We want to describe meaning via truth conditions
- Can be achieved via first-order logic (FOL) MRs

Definition (Terms and Formulas)

First order terms and formulas are defined as follows:

1. every variable x and constant c is a term
2. if P is a predicate of arity n and $t_1, \ldots, t_n$ are terms $P(t_1, \ldots, t_n)$ is a formula
3. if φ, φ' are formulas, so are $\neg\varphi$ and $\varphi \wedge \varphi'$
4. if x is a variable and φ a formula, so is $\exists x\varphi$

- The free variables $FV(\varphi)$ of φ are those not bound to a quantifier
- $\varphi \vee \varphi' = \neg(\neg\varphi \wedge \neg\varphi')$, $\varphi \Rightarrow \varphi' = \neg\varphi \vee \varphi'$ and $\forall x\varphi = \neg\exists x\neg\varphi$

Remark

Formulas and their semantics are defined recursively

First Order Logic - Semantics

Definition (Interpretation)

An interpretation is a pair $\mathcal{I} = (\Delta, \cdot^{\mathcal{I}})$ that:

1. maps constants c to points $c^{\mathcal{I}} \in \Delta$
2. maps predicates P to subsets $P^{\mathcal{I}} \subseteq \Delta^k$

First Order Logic - Semantics

Definition (Interpretation)

An interpretation is a pair $\mathcal{I} = (\Delta, \cdot^{\mathcal{I}})$ that:

1. maps constants c to points $c^{\mathcal{I}} \in \Delta$
2. maps predicates P to subsets $P^{\mathcal{I}} \subseteq \Delta^k$

- An assignment is a function $\gamma : FV(\varphi) \to \Delta$

First Order Logic - Semantics

Definition (Interpretation)

An interpretation is a pair $\mathcal{I} = (\Delta, \cdot^{\mathcal{I}})$ that:

1. maps constants c to points $c^{\mathcal{I}} \in \Delta$
2. maps predicates P to subsets $P^{\mathcal{I}} \subseteq \Delta^k$

- An assignment is a function $\gamma : FV(\varphi) \to \Delta$

Definition (Satisfaction)

The satisfaction relation among $\mathcal{I}$ and φ w.r.t. γ is defined as follows:

1. $\mathcal{I}, \gamma \models P(c_1, \ldots, c_n)$ iff $(c_1^{\mathcal{I}}, \ldots, c_n^{\mathcal{I}}) \in P^{\mathcal{I}}$
2. $\mathcal{I}, \gamma \models P(x_1, \ldots, x_n)$ iff $(\gamma(t_1), \ldots, \gamma(t_n)) \in P^{\mathcal{I}}$
3. $\mathcal{I}, \gamma \models \neg\varphi$ iff $\mathcal{I}, \gamma \not\models \varphi$
4. $\mathcal{I}, \gamma \models \varphi \wedge \varphi'$ iff $\mathcal{I}, \gamma \models \varphi$ and $\mathcal{I}, \gamma \models \varphi'$
5. $\mathcal{I}, \gamma \models \exists x\varphi$ iff exists $d \in \Delta$ s.t. $\mathcal{I}, \gamma[x := d] \models \varphi$

First Order Logic - Semantics

Definition (Interpretation)

An interpretation is a pair $\mathcal{I} = (\Delta, \cdot^{\mathcal{I}})$ that:

1. maps constants c to points $c^{\mathcal{I}} \in \Delta$
2. maps predicates P to subsets $P^{\mathcal{I}} \subseteq \Delta^k$

- An assignment is a function $\gamma : FV(\varphi) \to \Delta$

Definition (Satisfaction)

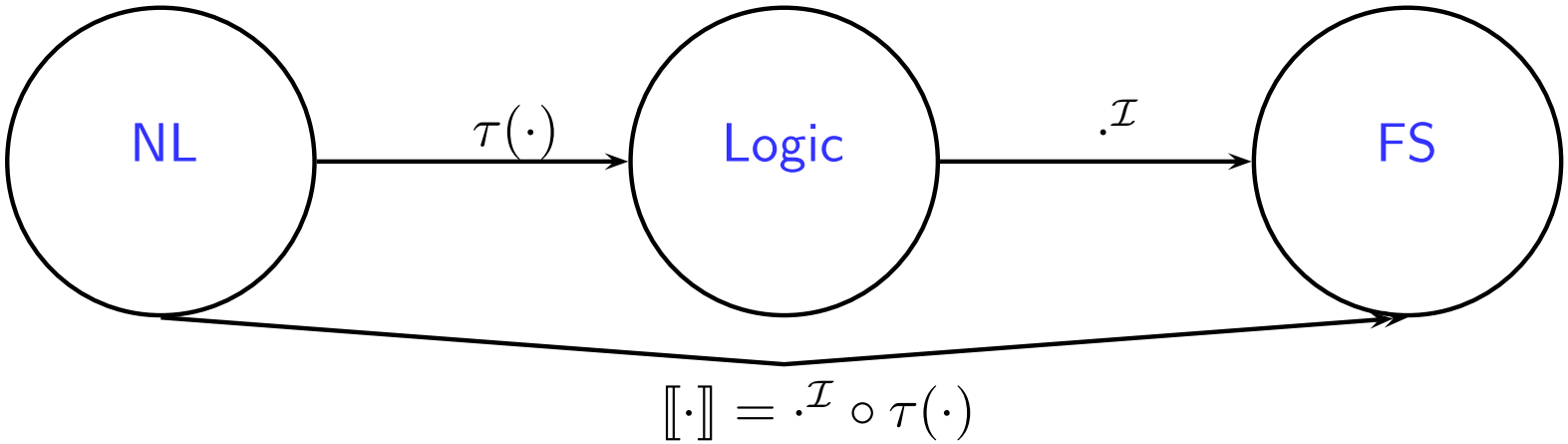
The satisfaction relation among $\mathcal{I}$ and φ w.r.t. γ is defined as follows:

1. $\mathcal{I}, \gamma \models P(c_1, \ldots, c_n)$ iff $(c_1^{\mathcal{I}}, \ldots, c_n^{\mathcal{I}}) \in P^{\mathcal{I}}$
2. $\mathcal{I}, \gamma \models P(x_1, \ldots, x_n)$ iff $(\gamma(t_1), \ldots, \gamma(t_n)) \in P^{\mathcal{I}}$
3. $\mathcal{I}, \gamma \models \neg\varphi$ iff $\mathcal{I}, \gamma \not\models \varphi$
4. $\mathcal{I}, \gamma \models \varphi \wedge \varphi'$ iff $\mathcal{I}, \gamma \models \varphi$ and $\mathcal{I}, \gamma \models \varphi'$
5. $\mathcal{I}, \gamma \models \exists x\varphi$ iff exists $d \in \Delta$ s.t. $\mathcal{I}, \gamma[x := d] \models \varphi$

- If $\mathcal{I}$ satisfies φ w.r.t. all γs, then $\mathcal{I}$ is a model of φ

Principle of Compositionality (C)

(C) The meaning of a complex utterance is a function of the meaning of its syntactic constituents



$[\![\cdot]\!] = \cdot^{\mathcal{I}} \circ \tau(\cdot)$ is an homomorphism from natural language (NL) to formal semantics (FS) via formal logic

FOL with lambdas (λ-FOL) [BB03]

(FS)		(NL)
$[\![\text{sleeps}]\!]([\![\text{Mary}]\!])$	$\leftrightharpoons$	Mary sleeps
$\Updownarrow$		$\Updownarrow$
sleep(Mary)	$\leftrightharpoons$	$\lambda x.\textit{sleep}(x)(\text{Mary})$
(Fo)		(λ-Fo)

We go from NL to FS via λ-Fo and Fo!

ξ	$\rightarrow$	$v \mid c \mid P \mid \alpha$	(terms)
ϵ	$\rightarrow$	ξ	(basic expressions)
ϵ	$\rightarrow$	$\lambda\xi.\epsilon$	(abstractions)
ϵ	$\rightarrow$	$\epsilon_1(\epsilon_2)$	(applications)

Generalized Quantifiers [BC80]

LF	$\tau(\text{every}) = \lambda P.\lambda Q.\forall x(P(x) \Rightarrow Q(x))$
Sem	$[\![\text{every}]\!] = \{(A, B) \subseteq \Delta \times \Delta \mid A \subseteq B\}$
LF	$\tau(\text{some}) = \lambda P.\lambda Q.\exists x(P(x) \wedge Q(x))$
Sem	$[\![\text{some}]\!] = \{(A, B) \subseteq \Delta \times \Delta \mid A \cap B \neq \emptyset\}$

(every and some)

Generalized Quantifiers [BC80]

LF	$\tau(\text{every}) = \lambda P.\lambda Q.\forall x(P(x) \Rightarrow Q(x))$
Sem	$[\![\text{every}]\!] = \{(A, B) \subseteq \Delta \times \Delta \mid A \subseteq B\}$
LF	$\tau(\text{some}) = \lambda P.\lambda Q.\exists x(P(x) \wedge Q(x))$
Sem	$[\![\text{some}]\!] = \{(A, B) \subseteq \Delta \times \Delta \mid A \cap B \neq \emptyset\}$

(every and some)

- Generalized quantifiers are realized in NL by **Det**s and **NP**s
- Seminal work by J. Barwise and R. Cooper in [BC80]
- They state relations that hold over properties in a model

Generalized Quantifiers [BC80]

LF	$\tau(\text{every}) = \lambda P.\lambda Q.\forall x(P(x) \Rightarrow Q(x))$
Sem	$[\![\text{every}]\!] = \{(A, B) \subseteq \Delta \times \Delta \mid A \subseteq B\}$
LF	$\tau(\text{some}) = \lambda P.\lambda Q.\exists x(P(x) \wedge Q(x))$
Sem	$[\![\text{some}]\!] = \{(A, B) \subseteq \Delta \times \Delta \mid A \cap B \neq \emptyset\}$

(every and some)

- Generalized quantifiers are realized in NL by **Det**s and **NP**s
- Seminal work by J. Barwise and R. Cooper in [BC80]
- They state relations that hold over properties in a model

Definition (Generalized Quantifier)

Given $\mathcal{I}$, a generalized quantifier Q of type $(k_1, \ldots, k_n)$ over $R_1, \ldots, R_n$ is a relation of tuples $(R_1, \ldots, R_n)$ s.t., for $1 \leq i \leq k$, $R_i \subseteq \Delta^{k_i}$.

Generalized Quantifiers (ctd.)

LF $\tau(\text{most}) = \lambda P.\lambda Q.\mathbf{most}(P, Q)$

Sem $[\![\text{most}]\!] = \{(A, B) \subseteq \Delta \times \Delta \mid \#(A \cap B) \geq \#(A \setminus B)\}$

LF $\tau(\text{more than } n/k) = \lambda P.\lambda Q. \geq n/k(P, Q)$

Sem $[\![\text{more than } n/k \text{ of}]\!] = \{(A, B) \subseteq \Delta \times \Delta \mid \#(A \cap B) \geq n/k \cdot \#(A)\}$

LF $\tau(\text{at most } k) = \lambda P.\lambda Q.\exists_{\leq k} x(P(x) \wedge Q(x))$

Sem $[\![\text{at most } k]\!] = \{(A, B) \subseteq \Delta \times \Delta \mid \#(A \cap B) \leq k\}$

LF $\tau(\text{few}) = \lambda P.\lambda Q.\mathbf{few}(P, Q)$

Sem $[\![\text{few}]\!] = \{(A, B) \subseteq \Delta \times \Delta \mid \#(A \setminus B) \geq \#(A \cap B)\}$

Semantically Augmented Grammars [JM09]

Definition (Augmented Grammar)

In semantically augmented grammars to each rule $C_1 \rightarrow C_2 C_3$ (resp. $C_1 \rightarrow C_2$) semantic action $\tau(C_1) = \tau(C_2)(\tau(C_3))$ (resp. $\tau(C_1) = \tau(C_2)$) is associated

Semantically Augmented Grammars [JM09]

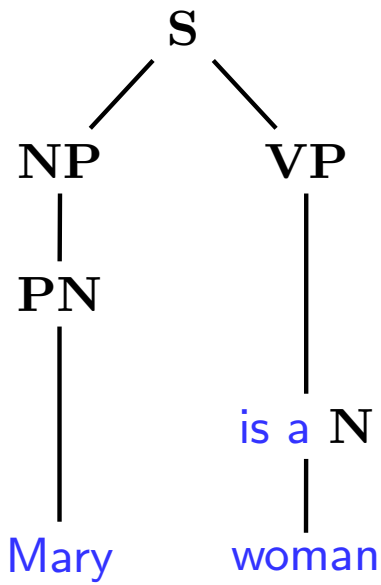
Definition (Augmented Grammar)

In semantically augmented grammars to each rule $C_1 \rightarrow C_2 C_3$ (resp. $C_1 \rightarrow C_2$) semantic action $\tau(C_1) = \tau(C_2)(\tau(C_3))$ (resp. $\tau(C_1) = \tau(C_2)$) is associated

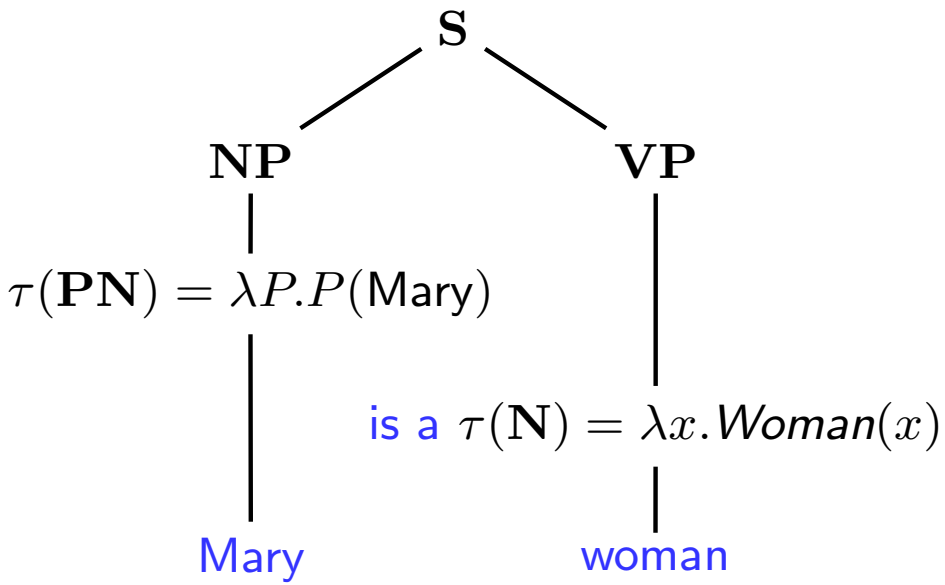
Syntax Rules	Semantics $(= \tau(\cdot))$
$\mathbf{S} \rightarrow \mathbf{NP\ VP}$	$\tau(\mathbf{S}) = \tau(\mathbf{NP})(\tau(\mathbf{VP}))$
$\mathbf{VP} \rightarrow$ is a $\mathbf{N}$	$\tau(\mathbf{VP}) = \tau(\mathbf{N})$
$\mathbf{VP} \rightarrow$ is not a $\mathbf{N}$	$\tau(\mathbf{VP}) = \neg\tau(\mathbf{N})$
$\mathbf{NP} \rightarrow \mathbf{PN}$	$\tau(\mathbf{NP}) = \tau(\mathbf{PN})$
$\mathbf{NP} \rightarrow \mathbf{Det\ N}$	$\tau(\mathbf{NP}) = \tau(\mathbf{Det})(\tau(\mathbf{N}))$

Lexicon	Semantics $(= \tau(\cdot))$
$\mathbf{N} \rightarrow$ woman	$\tau(\mathbf{N}) = \lambda x.\mathit{Woman}(x)$
$\mathbf{N} \rightarrow$ man	$\tau(\mathbf{N}) = \lambda x.\mathit{Man}(x)$
$\mathbf{PN} \rightarrow$ Mary	$\tau(\mathbf{PN}) = \lambda P.P(\text{Mary})$
$\mathbf{Det} \rightarrow$ every	$\tau(\mathbf{Det}) = \lambda P.\lambda Q.\forall x(P(x) \Rightarrow Q(x))$
$\mathbf{Det} \rightarrow$ some	$\tau(\mathbf{Det}) = \lambda P.\lambda Q.\exists x(P(x) \wedge Q(x))$
$\mathbf{Det} \rightarrow$ no	$\tau(\mathbf{Det}) = \lambda P.\lambda Q.\forall x(P(x) \Rightarrow \neg Q(x))$

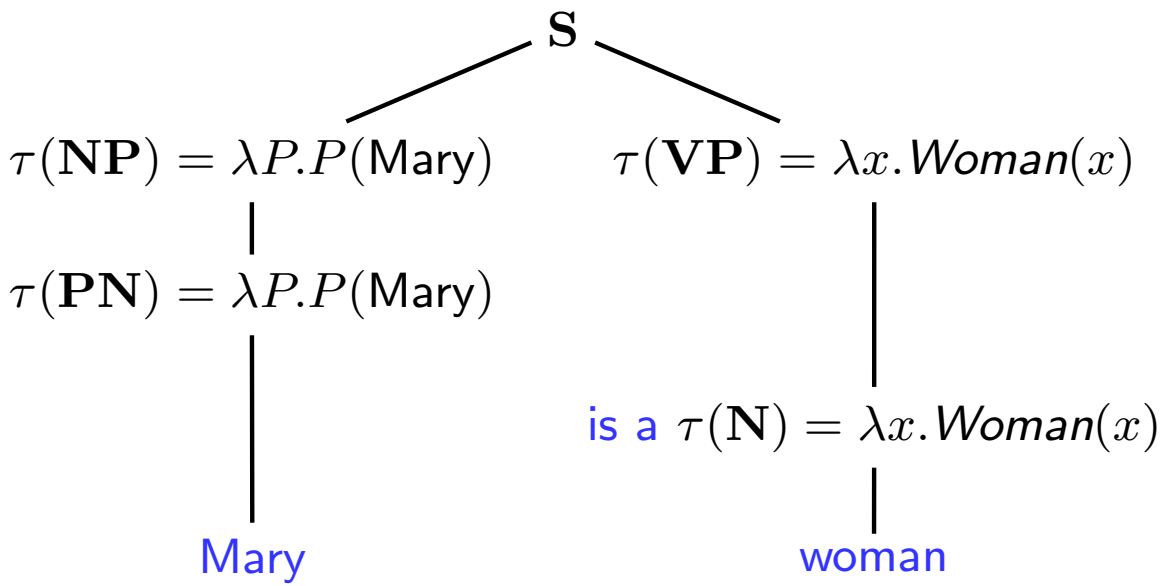
Semantic Interpretation [JM09]



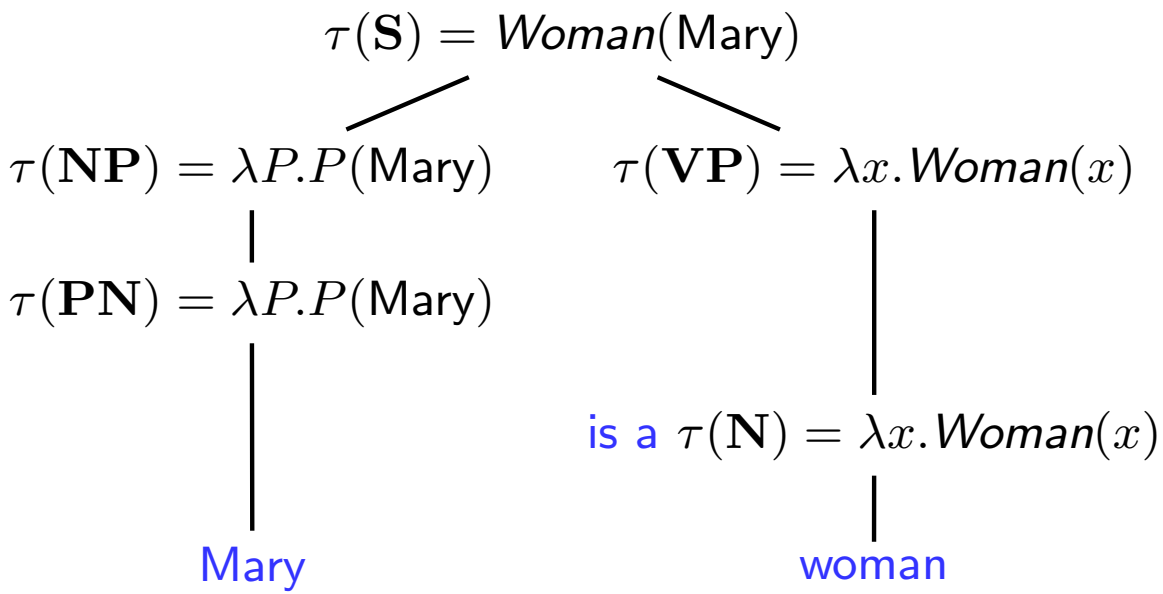
Semantic Interpretation [JM09]



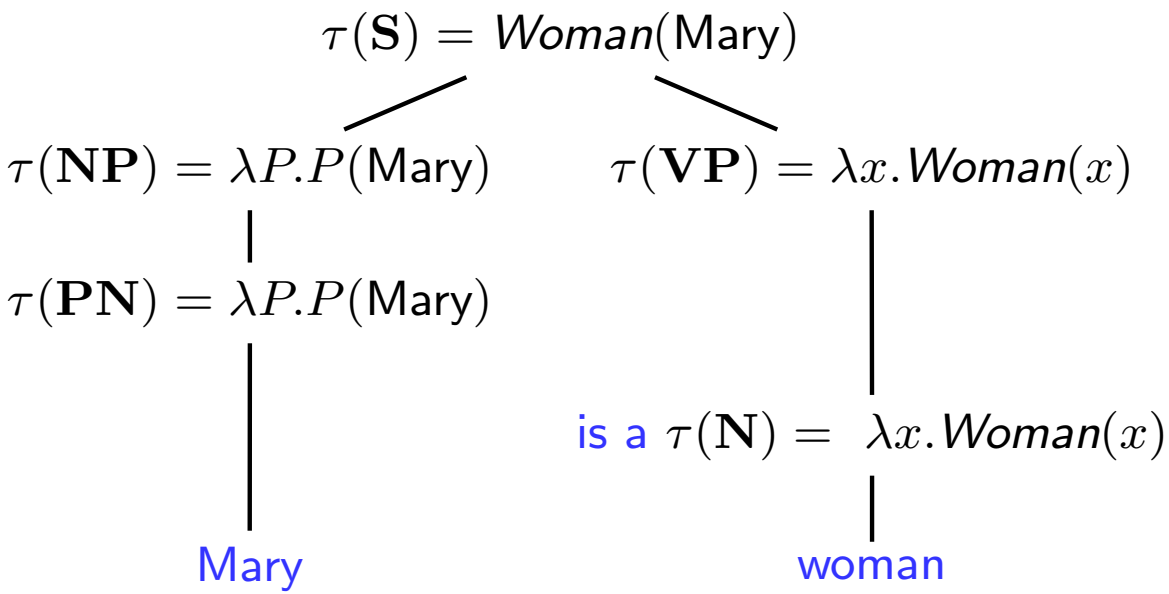
Semantic Interpretation [JM09]



Semantic Interpretation [JM09]



Semantic Interpretation [JM09]



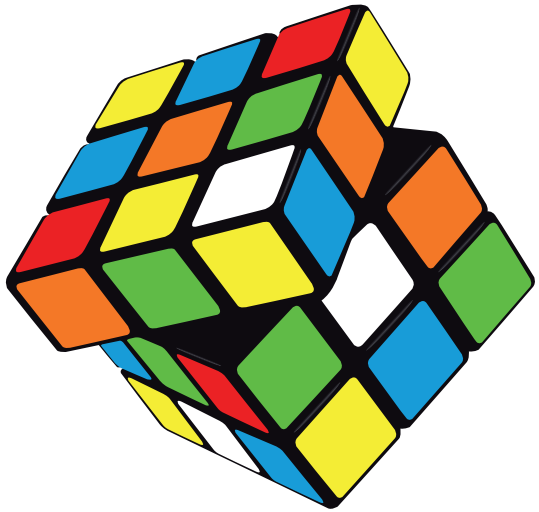
Remark

Computing $\tau(\cdot)$ is "easy" for context-free fragments of, e.g., English:

1. parsing: polynomial time in the size of the input utterance
2. interpreting: linear in the size of the parse tree

Summary

1. Provided an overview of formal semantics
2. Formal semantics provides a uniform computational model for natural language semantics
3. Key tool to understand semantic complexity
4. Formal semantics has also received attention by philosophers [Dav84]
5. Introduced the notion of generalized quantifier
6. Generalized quantifiers provide the denotation of English noun phrases and determiners [BC80]



Thank you :-)

References I


Johan Bos and Patrick Blackburn.
Computational semantics.
Theoria, 18(1):27–45, 2003.


John Barwise and Robin Cooper.
Generalized quantifiers and natural language.
Linguistics and Philosophy, 4(2):159–219, 1980.


Donald Davidson.
Essays on Truth and Interpretation.
Oxford University Press, 1984.


Daniel Jurafsky and James Martin.
Speech and Language Processing.
Prentice Hall, 2nd edition, 2009.


Richard Montague.
Universal grammar.
Theoria, 36(3):373–398, 1970.

References II


Richard Montague.
The proper treatment of quantification in ordinary english.
In *Approaches to Natural Language: Proceedings of the 1970 Stanford Workshop on Grammar and Semantics*, 1973.