Logical form tutorial
http://www-rohan.sdsu.edu/~gawron/semantics

Jean Mark Gawron
San Diego State University, Department of Linguistics

2010-08-19
Overview

1. Introduction
2. Background ideas
   • General principles
   • Predicate Principles
3. Statement logic/predicates
4. Predicate logic
5. A recipe for English-to-Logic translation
6. Logical Form
7. Applying the recipe
8. Ambiguity
9. Embedded sentences
Logical Form

Goal:
A few simple rules to help the beginner get the hang of translating into logic

Problems
There are a LOT of things to cover
The rules can't be complete.
Ambiguity of English
Goal:

- A few simple rules to help the beginner get the hang of translating into logic
Goal:

- A few simple rules to help the beginner get the hang of translating into logic
- Problems
Goal:

- A few simple rules to help the beginner get the hang of translating into logic

Problems

- There are a LOT of things to cover
Goal:

- A few simple rules to help the beginner get the hang of translating into logic

Problems

- There are a LOT of things to cover
- The rules can’t be complete.
Goal:

- A few simple rules to help the beginner get the hang of translating into logic

Problems

- There are a LOT of things to cover
- The rules can’t be complete.
- Ambiguity of English
Outline

1. Introduction

2. Background ideas
   - General principles
   - Predicate Principles

3. Statement logic/predicates

4. Predicate logic

5. A recipe for English-to-Logic translation

6. Logical Form

7. Applying the recipe

8. Ambiguity

9. Embedded sentences
The **logical form** of an English sentence is a **decomposition** of the sentence into **predicates** and **connectives**. The predicates capture the concepts being expressed. The connectives capture how the concepts are related.

1. Almost every noun, verb, and adjectives corresponds to a predicate.
The **logical form** of an English sentence is a **decomposition** of the sentence into **predicates** and **connectives**. The predicates capture the concepts being expressed. The connectives capture how the concepts are related.

1. Almost every noun, verb, and adjectives corresponds to a predicate.
2. Exceptions: No auxiliary verb, including the verb *be* (*is, are, was, being, been*), corresponds to a predicate.
The **logical form** of an English sentence is a **decomposition** of the sentence into **predicates** and **connectives**. The predicates capture the concepts being expressed. The connectives capture how the concepts are related.

1. Almost every noun, verb, and adjectives corresponds to a predicate.
2. Exceptions: No auxiliary verb, including the verb *be* (*is, are, was, being, been*), corresponds to a predicate.
3. Make sure each predicate word is accounted for in your translation.
The **logical form** of an English sentence is a **decomposition** of the sentence into **predicates** and **connectives**. The predicates capture the concepts being expressed. The connectives capture how the concepts are related.

1. Almost every noun, verb, and adjectives corresponds to a predicate.
2. Exceptions: No auxiliary verb, including the verb *be* (*is, are, was, being, been*), corresponds to a predicate.
3. Make sure each predicate word is accounted for in your translation.
4. Use your predicates consistently
The **logical form** of an English sentence is a **decomposition** of the sentence into **predicates** and **connectives**. The predicates capture the concepts being expressed. The connectives capture how the concepts are related.

1. Almost every noun, verb, and adjectives corresponds to a predicate.
2. Exceptions: No auxiliary verb, including the verb *be* (*is, are, was, being,* *been*), corresponds to a predicate.
3. Make sure each predicate word is accounted for in your translation.
4. Use your predicates consistently
   1. Same arity (same number of arguments)
Principles

The logical form of an English sentence is a decomposition of the sentence into predicates and connectives. The predicates capture the concepts being expressed. The connectives capture how the concepts are related.

1. Almost every noun, verb, and adjectives corresponds to a predicate.
2. Exceptions: No auxiliary verb, including the verb be (is, are, was, being, been), corresponds to a predicate.
3. Make sure each predicate word is accounted for in your translation.
4. Use your predicates consistently
   1. Same arity (same number of arguments)
   2. Arguments are in a consistent order
<table>
<thead>
<tr>
<th>Arity</th>
<th>Noun</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-place</td>
<td>man(x)</td>
<td>except relational nouns (husband, father)</td>
</tr>
<tr>
<td>1-place</td>
<td>happy(x)</td>
<td>except relational adjectives (fond+of, angry+at)</td>
</tr>
<tr>
<td>2-place</td>
<td>from(x, Spain)</td>
<td>except sometimes part of verb meaning (rely+on), object of prep is arg2</td>
</tr>
</tbody>
</table>
Connectives

(both) ... and \( \wedge, \& \) \( p \wedge q \)
(Both) John and Bill awakened.
Sue awakened (both) John and Bill.

(either) ... or \( \vee \) \( p \vee q \)
(Either) John or Bill awakened.
Sue awakened John or Bill.

not \( \sim \) \( \sim p \)
John didn't sleep.
It's not the case that John slept.

neither .. nor
Neither Sue nor Mary slept.
Sue neither ran nor swam.

not ... nor
John didn't sleep (and) nor did Sue.

unless
John will win unless he withdraws.

because
Give up!
Connective Principle

Sentential Connective principle

To translate an English sentence using a sentential connective of statement logic, you must find a logically equivalent sentence in which two full sentences are conjoined.

John and Bill awakened.  
$p = \text{John awakened} ; q = \text{Bill awakened}$  
$p \& q$

awaken(j) & awaken(b)

Sue awakened John and Bill.  
$p = \text{Sue awakened John} ; q = \text{Sue awakened Bill}$  
$p \& q$

awaken2(s,j) & awaken2(s,b)
Neither John nor Bill awakened.

John didn’t awaken and Bill didn’t awaken.

\( Q = \text{awaken}; \ p = \text{John Q’ed} ; \ q = \text{Bill Q’ed} \)

\[ \sim p \& \sim q \]

\[ \sim (p \lor q) \]

### Truth table

<table>
<thead>
<tr>
<th>J. Q’ed</th>
<th>B. Q’ed</th>
<th>Neither J. nor B. Q’ed</th>
<th>( \sim p &amp; \sim q )</th>
<th>( \sim (p \lor q) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>
## Verbs

<table>
<thead>
<tr>
<th>Arity</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>intransitive</td>
<td>walk, faint, sleep, fall, ... Ignore tense.</td>
</tr>
<tr>
<td>transitive</td>
<td>hit, kill, kick, eat, ... unpassivize passive sentences (John was loved by Mary → Mary loved John)</td>
</tr>
<tr>
<td>ditransitive</td>
<td>give, send, cost, charge, ...</td>
</tr>
</tbody>
</table>

**Auxiliaries**

syncategorematic

be, do*, have*, may, might, can, could, should, shall, will, would

*: do and have are ambiguous. They are also transitive verbs.
The **arity** of a predicate is the number of arguments it has.

a. John showed Mary the picture.
   \[ \text{show}(j, m, p) \]

b. John showed Mary.
   \[ \text{show}(j, m) \text{ NO NO! Ignore p. 36!} \]

c. \[ \text{show2}(j, m) \]
The **arity** of a linguistic predicate is the number of syntactic arguments it has.

1. If it’s obligatory, it’s an argument.
2. If the same verb shows up with different sets of arguments, use different predicates.
3. Location, Time, and Manner are usually not arguments:
   a. Time  
      John painted the room *yesterday*.  
      \( \text{paint}(j,r) \)
   b. Location  
      John wrote his essay *in the study*,  
      \( \text{write}(j,e) \)
   c. Manner  
      John hid the letter *carefully*.  
      \( \text{hide}(j,l) \)
Predicate Principle examples

John painted the kitchen
John painted in the kitchen.
Da Vinci painted the Mona Lisa

John gave the book to Mary.
John gave Mary the book.
WRONG!
Mary was given the book by John.
Mary was given the book.

paint(j, k)
paint2(j)
give(j,b,m)
give(j,b,m)
give(j,m,b)
give(j,b,m)
give2(m,b)
Predicate Principle examples

John painted the kitchen \( \text{paint}(j, k) \)
John painted in the kitchen. \( \text{paint2}(j) \)
Da Vinci painted the Mona Lisa \( \text{paint32}(d, m) \)

John gave the book to Mary. \( \text{give}(j, b, m) \)
John gave Mary the book. \( \text{give}(j, b, m) \)
WRONG!
Mary was given the book by John. \( \text{give}(j, m, b) \)
Mary was given the book. \( \text{give2}(m, b) \)
Complex Predicates

Sometimes a predicate will be expressed by more than one word.

a. John *signed up for* the class.
   \[\text{sign-up-for}(j,c)\]

b. John *blacked out* in the study.
   \[\text{black-out}(j,e)\]

c. John *called up* Sue
   
   John *called Sue up.*
   \[\text{call-up}(j,s)\]

Frequently such complex predicates are combinations of verbs and prepositions. It’s convenient to use both the verb and preposition in naming such predicates, because it often helps make the meaning clear, and keeps different meanings distinct (*call-up* vs *call-on*)
Connectives: Quantifiers, negation, and sentential

Universals (∀), Existentials (∃), and negation ∼ correspond to appropriate English words, and each quantifier goes with its appropriate sentential connective:

<table>
<thead>
<tr>
<th>English</th>
<th>Symbol</th>
<th>Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>every, all, any</td>
<td>∀</td>
<td>∀ → ∀x dog(x) → bark(x)</td>
</tr>
<tr>
<td>some, a, a certain</td>
<td>∃</td>
<td>∃ &amp; ∃x dog(x) &amp; bark(x)</td>
</tr>
<tr>
<td>not, n’t</td>
<td>∼</td>
<td>∼ ∃ &amp; ∼ ∃x dog(x) &amp; bark(x)</td>
</tr>
<tr>
<td>no</td>
<td>∼ ∃</td>
<td></td>
</tr>
</tbody>
</table>
Ambiguity

(1) a. Every prize was won by some high school kid.
Ambiguity

(2) a. Every prize was won by some high school kid.
   b. For every prize, $x$, there was some high school kid, $y$, such that $y$ won $x$.

   b. $\forall x [\text{prize}(x) \rightarrow \exists y [\text{high-school-kid}(y) \& \text{win}(y, x)]]$

Ambiguity

(3) a. Every prize was won by some high school kid.
   b. For every prize, $x$, there was some high school kid, $y$, such that $y$ won $x$.
   c. Some particular high school kid $y$ won every prize, $x$.

   b. $\forall x [\text{prize}(x) \rightarrow \exists y [\text{high-school-kid}(y) \& \text{win}(y, x)]]$
   c. $\exists y [\text{high-school-kid}(y) \& \forall x [\text{prize}(x) \rightarrow \text{win}(y, x)]]$
Ambiguity

(4) a. Every prize was won by some high school kid.
   b. For every prize, \( x \), there was some high school kid, \( y \), such that \( y \) won \( x \).
   c. Some particular high school kid \( y \) won every prize, \( x \).

\[
\begin{align*}
&b. \quad \forall x [ \text{prize}(x) \rightarrow \exists y [ \text{high-school-kid}(y) \& \text{win}(y, x) ] ] \\
&c. \quad \exists y [ \text{high-school-kid}(y) \& \forall x [ \text{prize}(x) \rightarrow \text{win}(y, x) ] ]
\end{align*}
\]

The two translations share all the same predicates, and even the arguments of the predicates are the same. All that differs is the way the predications are connected.
Translate each Noun phrase (NP) in isolation.

Some examples of noun phrase translations

a kid

∃ x kid(x)

∃ x tall(x) & kid(x)

∃ x kid(x) & from(x, Spain)

∃ x tall(x) & kid(x) & from(x, Spain)

∃ x high-school(x) & kid(x)

Wrong!

∃ x high-school-kid(x)

Right!

∃ x man(x) & like(I)

Wrong!

∃ x man(x) & like(I, x)

Right!
Translate each Noun phrase (NP) in isolation.

Some examples of noun phrase translations

\[ a \; \text{kid}_x \quad \exists \; x \; \text{kid}(x)\]
Noun phrases

Translate each Noun phrase (NP) in isolation.

Some examples of noun phrase translations

\[ \exists x \text{ kid}(x) \]

\[ \exists x \text{ tall kid}(x) \]

\[ \exists x \text{ from Spain} \]

Wrong!

\[ \exists x \text{ high-school kid}(x) \]

Right!

\[ \exists x \text{ man}(x) \]

Wrong!

\[ \exists x \text{ like}(I, x) \]

Right!
Noun phrases

Translate each Noun phrase (NP) in isolation.

Some examples of noun phrase translations

<table>
<thead>
<tr>
<th>English</th>
<th>Logic</th>
</tr>
</thead>
<tbody>
<tr>
<td>a kid&lt;sub&gt;x&lt;/sub&gt;</td>
<td>( \exists x \text{ kid}(x) )</td>
</tr>
<tr>
<td>a tall kid&lt;sub&gt;x&lt;/sub&gt;</td>
<td>( \exists x \text{ tall}(x) &amp; \text{ kid}(x) )</td>
</tr>
</tbody>
</table>
Translate each Noun phrase (NP) in isolation.

Some examples of noun phrase translations

- a kid$_{x}$
  - $\exists x$ kid($x$)

- a tall kid$_{x}$
  - $\exists x$ tall($x$) & kid($x$)

- a kid$_{x}$ from Spain
  - $\exists x$ tall($x$) & kid($x$) & from($x$, Spain)

- a high school kid$_{x}$
  - $\exists x$ high-school($x$) & kid($x$)

Wrong!

- a high school kid$_{x}$
  - $\exists x$ high-school-kid($x$)

Right!

- a man$_{x}$
  - $\exists x$ man($x$) & like(I)

Wrong!

- $\exists x$ man($x$) & like(I, $x$)

Right!
Translate each Noun phrase (NP) in isolation.

Some examples of noun phrase translations

- a kid$_x$  :  $\exists x \text{kid}(x)$
- a tall kid$_x$  :  $\exists x \text{tall}(x) \land \text{kid}(x)$
- a kid$_x$ from Spain  :  $\exists x \text{kid}(x) \land \text{from}(x, S)$
Noun phrases

Translate each Noun phrase (NP) in isolation.

Some examples of noun phrase translations

- a kid$_x$  \[ \exists x \text{ kid}(x) \]
- a tall kid$_x$  \[ \exists x \text{ tall}(x) \& \text{ kid}(x) \]
- a kid$_x$ from Spain  \[ \exists x \text{ kid}(x) \& \text{ from}(x, s) \]
- a tall kid$_x$ from Spain

Wrong!

- a high school kid$_x$  \[ \exists x \text{ high-school-kid}(x) \]

Right!
Noun phrases

Translate each Noun phrase (NP) in isolation.

Some examples of noun phrase translations

<table>
<thead>
<tr>
<th>Noun Phrase</th>
<th>Logical Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>a kid$_x$</td>
<td>$\exists x$ kid($x$)</td>
</tr>
<tr>
<td>a tall kid$_x$</td>
<td>$\exists x$ tall($x$) &amp; kid($x$)</td>
</tr>
<tr>
<td>a kid$_x$ from Spain</td>
<td>$\exists x$ kid($x$) &amp; from($x$, $s$)</td>
</tr>
<tr>
<td>a tall kid$_x$ from Spain</td>
<td>$\exists x$ tall($x$) &amp; kid($x$) &amp; from($x$, $s$)</td>
</tr>
</tbody>
</table>
Translate each Noun phrase (NP) in isolation.

Some examples of noun phrase translations

- a kid\(_x\) → \(\exists x \text{ kid}(x)\)
- a tall kid\(_x\) → \(\exists x \text{ tall}(x) & \text{ kid}(x)\)
- a kid\(_x\) from Spain → \(\exists x \text{ kid}(x) & \text{ from}(x, S)\)
- a tall kid\(_x\) from Spain → \(\exists x \text{ tall}(x) & \text{ kid}(x) & \text{ from}(x, S)\)
- a high school kid\(_x\) → \(\exists x \text{ high-school}(x) & \text{ kid}(x)\)
Noun phrases

Translate each Noun phrase (NP) in isolation.

Some examples of noun phrase translations

- a kid\(x\) \(\exists x\) kid\(x\)
- a tall kid\(x\) \(\exists x\) tall\(x\) & kid\(x\)
- a kid\(x\) from Spain \(\exists x\) kid\(x\) & from\(x, s\)
- a tall kid\(x\) from Spain \(\exists x\) tall\(x\) & kid\(x\) & from\(x, s\)
- a high school kid\(x\) \(\exists x\) high-school\(x\) & kid\(x\)  \text{Wrong!}
Noun phrases

Translate each Noun phrase (NP) in isolation.

<table>
<thead>
<tr>
<th>Noun Phrase</th>
<th>Logical Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>a kid_}_x</td>
<td>( \exists x \text{ kid}(x) )</td>
</tr>
<tr>
<td>a tall kid_}_x</td>
<td>( \exists x \text{ tall}(x) &amp; \text{ kid}(x) )</td>
</tr>
<tr>
<td>a kid_}_x from Spain</td>
<td>( \exists x \text{ kid}(x) &amp; \text{ from}(x, s) )</td>
</tr>
<tr>
<td>a tall kid_}_x from Spain</td>
<td>( \exists x \text{ tall}(x) &amp; \text{ kid}(x) &amp; \text{ from}(x, s) )</td>
</tr>
<tr>
<td>a high school kid_}_x</td>
<td>( \exists x \text{ high-school}(x) &amp; \text{ kid}(x) )</td>
</tr>
<tr>
<td>a high school kid_}_x</td>
<td>( \exists x \text{ high-school-kid}(x) )</td>
</tr>
</tbody>
</table>

Wrong!

Wrong!
Noun phrases

Translate each Noun phrase (NP) in isolation.

Some examples of noun phrase translations

<table>
<thead>
<tr>
<th>Noun Phrase</th>
<th>Logical Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>a kid(_x)</td>
<td>(\exists x \text{ kid}(x))</td>
</tr>
<tr>
<td>a tall kid(_x)</td>
<td>(\exists x \text{ tall}(x) &amp; \text{ kid}(x))</td>
</tr>
<tr>
<td>a kid(_x) from Spain</td>
<td>(\exists x \text{ kid}(x) &amp; \text{ from}(x, s))</td>
</tr>
<tr>
<td>a tall kid(_x) from Spain</td>
<td>(\exists x \text{ tall}(x) &amp; \text{ kid}(x) &amp; \text{ from}(x, s))</td>
</tr>
<tr>
<td>a high school kid(_x)</td>
<td>(\exists x \text{ high-school}(x) &amp; \text{ kid}(x)) Wrong!</td>
</tr>
<tr>
<td>a high school kid(_x)</td>
<td>(\exists x \text{ high-school-kid}(x)) Right!</td>
</tr>
</tbody>
</table>
Noun phrases

Translate each Noun phrase (NP) in isolation.

<table>
<thead>
<tr>
<th>Noun phrase</th>
<th>Logical Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>a kid&lt;sub&gt;x&lt;/sub&gt;</td>
<td>( \exists x ) kid(x)</td>
</tr>
<tr>
<td>a tall kid&lt;sub&gt;x&lt;/sub&gt;</td>
<td>( \exists x ) tall(x) &amp; kid(x)</td>
</tr>
<tr>
<td>a kid&lt;sub&gt;x&lt;/sub&gt; from Spain</td>
<td>( \exists x ) kid(x) &amp; from(x, S)</td>
</tr>
<tr>
<td>a tall kid&lt;sub&gt;x&lt;/sub&gt; from Spain</td>
<td>( \exists x ) tall(x) &amp; kid(x) &amp; from(x, S)</td>
</tr>
<tr>
<td>a high school kid&lt;sub&gt;x&lt;/sub&gt;</td>
<td>( \exists x ) high-school(x) &amp; kid(x)</td>
</tr>
<tr>
<td>a high school kid&lt;sub&gt;x&lt;/sub&gt;</td>
<td>( \exists x ) high-school-kid(x)</td>
</tr>
<tr>
<td>a man&lt;sub&gt;x&lt;/sub&gt; I like</td>
<td>( \exists x ) man(x) &amp; like(I, x)</td>
</tr>
</tbody>
</table>

Wrong! Right!
Translate each Noun phrase (NP) in isolation.

Some examples of noun phrase translations

<table>
<thead>
<tr>
<th>Noun Phrase</th>
<th>Logical Form</th>
<th>Correctness</th>
</tr>
</thead>
<tbody>
<tr>
<td>a kid$_x$</td>
<td>$\exists x$ kid($x$)</td>
<td></td>
</tr>
<tr>
<td>a tall kid$_x$</td>
<td>$\exists x$ tall($x$) &amp; kid($x$)</td>
<td></td>
</tr>
<tr>
<td>a kid$_x$ from Spain</td>
<td>$\exists x$ kid($x$) &amp; from($x$, $S$)</td>
<td></td>
</tr>
<tr>
<td>a tall kid$_x$ from Spain</td>
<td>$\exists x$ tall($x$) &amp; kid($x$) &amp; from($x$, $S$)</td>
<td></td>
</tr>
<tr>
<td>a high school kid$_x$</td>
<td>$\exists x$ high-school($x$) &amp; kid($x$)</td>
<td>Wrong!</td>
</tr>
<tr>
<td>a high school kid$_x$</td>
<td>$\exists x$ high-school-kid($x$)</td>
<td>Right!</td>
</tr>
<tr>
<td>a man$_x$ I like</td>
<td>$\exists x$ man($x$) &amp; like(I)</td>
<td></td>
</tr>
</tbody>
</table>
Translate each Noun phrase (NP) in isolation.

Some examples of noun phrase translations

<table>
<thead>
<tr>
<th>Noun Phrase</th>
<th>Logical Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>a kid_x</td>
<td>$\exists x ; \text{kid}(x)$</td>
</tr>
<tr>
<td>a tall kid_x</td>
<td>$\exists x ; \text{tall}(x) \land \text{kid}(x)$</td>
</tr>
<tr>
<td>a kid_x from Spain</td>
<td>$\exists x ; \text{kid}(x) \land \text{from}(x, S)$</td>
</tr>
<tr>
<td>a tall kid_x from Spain</td>
<td>$\exists x ; \text{tall}(x) \land \text{kid}(x) \land \text{from}(x, S)$</td>
</tr>
<tr>
<td>a high school kid_x</td>
<td>$\exists x ; \text{high-school}(x) \land \text{kid}(x)$</td>
</tr>
<tr>
<td>a high school kid_x</td>
<td>$\exists x ; \text{high-school-kid}(x)$</td>
</tr>
<tr>
<td>a man_x I like</td>
<td>$\exists x ; \text{man}(x) \land \text{like}(I)$</td>
</tr>
<tr>
<td>a man_x I like</td>
<td>$\exists x ; \text{man}(x) \land \text{like}(I, x)$</td>
</tr>
</tbody>
</table>

Wrong!

Wrong!

Right!
Noun phrases

Translate each Noun phrase (NP) in isolation.

Some examples of noun phrase translations

- a kid\(_x\)  \(\exists x\) kid\(_x\)
- a tall kid\(_x\)  \(\exists x\) tall\(_x\) & kid\(_x\)
- a kid\(_x\) from Spain  \(\exists x\) kid\(_x\) & from\(_x\), s
- a tall kid\(_x\) from Spain  \(\exists x\) tall\(_x\) & kid\(_x\) & from\(_x\), s
- a high school kid\(_x\)  \(\exists x\) high-school\(_x\) & kid\(_x\)  \text{Wrong!}
- a high school kid\(_x\)  \(\exists x\) high-school-kid\(_x\)  \text{Right!}
- a man\(_x\) I like  \(\exists x\) man\(_x\) & like\(_I\)  \text{Wrong!}
- a man\(_x\) I like  \(\exists x\) man\(_x\) & like\(_I, x\)  \text{Right!}
Important principle

NP modifiers

A kid_x from Spain
\exists x \ kid(x) \& from(\ , \ s)
Important principle

NP modifiers

\[
A \quad \text{kid}_x \quad \text{from Spain}
\]

\[
\exists x \quad \text{kid}(x) \quad \& \quad \text{from}(x, \ s)
\]

↑

The NP variable always occurs in the translations of its modifiers.
Initial sentence

Some young woman arrived.

Procedure: Simple version

Find each quantified np, and choose a variable for it.

1. Find Quantified np

2. Remove Quantified np

Moved out sentence

Some young woman x arrived.
Procedure: Simple version

Initial sentence
Some young woman arrived.

1. Find each quantified NP, and choose a variable for it.
Procedure: Simple version

Initial sentence
Some young woman arrived.

1. Find each quantified NP, and choose a variable for it.

1. Find Quantified NPs
Some young woman$_x$ arrived.
Procedure: Simple version

Initial sentence
Some young woman arrived.

1. Find each quantified NP, and choose a variable for it.

1. Find Quantified NPs
Some young woman\(_x\) arrived.

2. Move each quantified NP out of the sentence, leaving the variable you chose behind.
Procedure: Simple version

Initial sentence
Some young woman arrived.

1. Find each quantified NP, and choose a variable for it.

1. Find Quantified NPs
Some young woman\(_x\) arrived.

2. Move each quantified NP out of the sentence, leaving the variable you chose behind.

2. Remove Quantified NPs

<table>
<thead>
<tr>
<th>Moved out</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>(_x)</td>
<td>(x) arrived.</td>
</tr>
</tbody>
</table>
Procedure: Simple version

Initial sentence
Some young woman arrived.

1. Find each quantified NP, and choose a variable for it.

2. Move each quantified NP out of the sentence, leaving the variable you chose behind. But keep it around for later!

2. Remove Quantified NPs

<table>
<thead>
<tr>
<th>Moved out</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Some young woman(x)</td>
<td>(x) arrived.</td>
</tr>
</tbody>
</table>
Translate each quantified NP into logic, replacing the head noun with a 1-place predicate whose argument is the NP variable:
Translate each quantified NP into logic, replacing the head noun with a 1-place predicate whose argument is the NP variable:

3. NPs → logic

<table>
<thead>
<tr>
<th>Moved out</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>A young woman</td>
<td>x arrived.</td>
</tr>
</tbody>
</table>
Translate each quantified NP into logic, replacing the head noun with a 1-place predicate whose argument is the NP variable:

3. **NPs → logic**

<table>
<thead>
<tr>
<th>Moved out</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>∃ x woman(x) &amp; young(x)</td>
<td>x arrived.</td>
</tr>
</tbody>
</table>

Turn the sentence into logic, using predicate principles.

4
Translate each quantified NP into logic, replacing the head noun with a 1-place predicate whose argument is the NP variable:

\[ \exists x \, \text{woman}(x) \land \text{young}(x) \] \[ x \, \text{arrived}. \]

Turn the sentence into logic, using predicate principles.

\[ \exists x \, \text{woman}(x) \land \text{young}(x) \] \[ \text{arrive}(x). \]
Add each NP translation back *one at a time*, using the right sentential connective:
5. **Move NP’s back**

Moved out | Sentence  
---|---
\[ \exists x \text{ woman}(x) \& \text{young}(x) \& \text{arrive}(x) \]
### Find Qtfd NPs

1. Some young woman\(_x\) arrived.

2. 

3. 

4. 

5. 

### Summary

<table>
<thead>
<tr>
<th>Remove Qtfd NPs</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Some young woman(_x) arrived.</td>
</tr>
<tr>
<td>2. (x) arrived.</td>
</tr>
</tbody>
</table>
### Remove Qtfd NPs

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Some young woman(x) arrived.</td>
</tr>
<tr>
<td>2.</td>
<td>Some young woman(x) arrived.</td>
</tr>
<tr>
<td>3.</td>
<td></td>
</tr>
<tr>
<td>4.</td>
<td></td>
</tr>
<tr>
<td>5.</td>
<td></td>
</tr>
</tbody>
</table>
1. Some young woman \( x \) arrived.
2. Some young woman \( x \) arrived.
3. \( \exists x \) young \( (x) \) \& woman \( (x) \) \( x \) arrived.
4. 
5. 
Some young woman \( x \) arrived.

Some young woman \( x \) arrived.

\( \exists x \) young(\( x \)) & woman(\( x \)) \( x \) arrived.

\( \exists x \) young(\( x \)) & woman(\( x \)) arrive(\( x \))
### Move NPs back

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Some young woman(_x) arrived.</td>
</tr>
<tr>
<td>2.</td>
<td>Some young woman(_x) arrived.</td>
</tr>
<tr>
<td>3.</td>
<td>(\exists x) young((x)) &amp; woman((x)) arrived.</td>
</tr>
<tr>
<td>4.</td>
<td>(\exists x) young((x)) &amp; woman((x)) arrived.</td>
</tr>
<tr>
<td>5.</td>
<td>(\exists x) young((x)) &amp; woman((x)) &amp; arrived((x))</td>
</tr>
</tbody>
</table>
1 Introduction

2 Background ideas
   • General principles
   • Predicate Principles

3 Statement logic/predicates

4 Predicate logic

5 A recipe for English-to-Logic translation

6 Logical Form

7 Applying the recipe

8 Ambiguity

9 Embedded sentences
Non ambiguous sentences?

(5) a. Everyone in this room speaks two languages.
   b. Two languages are spoken by everyone in this room.
   c. It is certain that no one will leave.
   d. No one is certain to leave.
What kind of ambiguity?

- S
  - NP_y
    - D: every
    - N: prize
  - S
    - NP_x
      - some high-school-kid
    - S
      - VP: won

- S
  - NP_x
    - D: some
    - N: high-school-kid
  - S
    - NP_y
      - D: every
      - N: prize
    - S
      - VP: won
This is getting weird
Outline

1. Introduction
2. Background ideas
   - General principles
   - Predicate Principles
3. Statement logic/predicates
4. Predicate logic
5. A recipe for English-to-Logic translation
6. Logical Form
7. Applying the recipe
8. Ambiguity
9. Embedded sentences
A different example

Initial sentence
Utopia welcomes every traveler from Spain.

1. Find each quantified \( np \), and choose a variable for it.

2. Move each quantified \( np \) out of the sentence, leaving the variable you chose behind. But keep it around for later!

Moved out Sentence
every traveler from Spain \( x \).

Jean Mark Gawron ( SDSU )
Gawron: Logical Form
2010-08-19 34 / 53
A different example

Initial sentence
Utopia welcomes every traveler from Spain.

1. Find each quantified NP, and choose a variable for it.
A different example

Initial sentence
Utopia welcomes every traveler from Spain.

1. Find each quantified NP, and choose a variable for it.

1. Find Quantified NPs
Utopia welcomes every traveler$_x$ from Spain.
A different example

Initial sentence
Utopia welcomes every traveler from Spain.

1. Find each quantified NP, and choose a variable for it.

2. Remove Quantified NPs
Utopia welcomes every traveler\(x\) from Spain.

2. Move each quantified NP out of the sentence, leaving the variable you chose behind.
A different example

Initial sentence
Utopia welcomes every traveler from Spain.

1. Find each quantified NP, and choose a variable for it.

2. Move each quantified NP out of the sentence, leaving the variable you chose behind.

2. Remove Quantified NPs

Moved out | Sentence
----------|------------
| every traveler\textsubscript{x} from Spain | Utopia welcomes x.
A different example

Initial sentence

Utopia welcomes every traveler from Spain.

1. Find each quantified NP, and choose a variable for it.

1. Find Quantified NPs

Utopia welcomes every traveler_\textcolor{red}{x} from Spain.

2. Move each quantified NP out of the sentence, leaving the variable you chose behind. But keep it around for later!

2. Remove Quantified NPs

<table>
<thead>
<tr>
<th>Moved out</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>every traveler from Spain_\textcolor{red}{x}</td>
<td>Utopia welcomes x.</td>
</tr>
</tbody>
</table>
Translate each quantified NP into logic, replacing the head noun with a 1-place predicate whose argument is the NP variable:
Translate each quantified NP into logic, replacing the head noun with a 1-place predicate whose argument is the NP variable:

3. NPs → logic

<table>
<thead>
<tr>
<th>Moved out</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\forall x \text{traveler}(x) &amp; \text{from}(x, s)$</td>
<td>Utopia welcomes $x$.</td>
</tr>
</tbody>
</table>
3. Translate each quantified NP into logic, replacing the head noun with a 1-place predicate whose argument is the NP variable:

### 3. NPs → logic

<table>
<thead>
<tr>
<th>Moved out</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>∀ x traveler(x) &amp; from(x, s)</td>
<td>Utopia welcomes x.</td>
</tr>
</tbody>
</table>

4. Turn the sentence into logic, using predicate principles.
Translate each quantified NP into logic, replacing the head noun with a 1-place predicate whose argument is the NP variable:

### 3. NPs → logic

<table>
<thead>
<tr>
<th>Moved out</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\forall , x , \text{traveler}(x) , &amp; , \text{from}(x, S)$</td>
<td>Utopia welcomes $x$.</td>
</tr>
</tbody>
</table>

Turn the sentence into logic, using predicate principles:

### 4. Sentence → logic

<table>
<thead>
<tr>
<th>Moved out</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\forall , x , \text{traveler}(x) , &amp; , \text{from}(x, S)$</td>
<td>welcome($U, x$).</td>
</tr>
</tbody>
</table>
Add each NP translation back *one at a time*, using the right sentential connective:
5. Add each NP translation back *one at a time*, using the right sentential connective:

5. **Move NP’s back**

<table>
<thead>
<tr>
<th>Moved out</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>∀ x traveler(x) &amp; from(x, s) → welcome(U, x)</td>
<td></td>
</tr>
</tbody>
</table>
1. Introduction

2. Background ideas
   - General principles
   - Predicate Principles

3. Statement logic/predicates

4. Predicate logic

5. A recipe for English-to-Logic translation

6. Logical Form

7. Applying the recipe

8. Ambiguity

9. Embedded sentences
(6) a. Every prize was won by some high school kid.
(7) a. Every prize was won by some high school kid.
   b. Some high school kid won every prize. (unpassivized form)

Step 0. Unpassivize.
(8) a. Every prize was won by some high school kid.
   b. Some high school kid won every prize. (unpassivized form)

Step 0. Unpassivize.

1. Find each quantified NP, and choose a variable for it.
(9) a. Every prize was won by some high school kid.
    b. Some high school kid won every prize. (unpassivized form)

Step 0. Unpassivize.

1. Find each quantified NP, and choose a variable for it.

1. Find Quantified NPs

Some high school kid$_x$ won every prize$_y$. 
Two qtfd NPs

(10) a. Every prize was won by some high school kid.
    b. Some high school kid won every prize. (unpassivized form)

Step 0. Unpassivize.

1. Find each quantified NP, and choose a variable for it.

2. Move each quantified NP out of the sentence, leaving the variable you chose behind.
(11) a. Every prize was won by some high school kid.
    b. Some high school kid won every prize. (unpassivized form)

Step 0. Unpassivize.

1. Find each quantified NP, and choose a variable for it.

Some high school kid\(x\) won every prize\(y\).

2. Move each quantified NP out of the sentence, leaving the variable you chose behind.

<table>
<thead>
<tr>
<th>Moved out</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Some high school kid(x)</td>
<td>(x) won (y).</td>
</tr>
<tr>
<td>every prize(y)</td>
<td></td>
</tr>
</tbody>
</table>
Translate each quantified \( NP \) into logic, replacing the head noun with a 1-place predicate whose argument is the \( NP \) variable:
Translate each quantified NP into logic, replacing the head noun with a 1-place predicate whose argument is the NP variable:

<table>
<thead>
<tr>
<th>Moved out</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\exists x$ high-school-kid($x$)</td>
<td>$x$ won $y$.</td>
</tr>
<tr>
<td>$\forall y$ prize($y$)</td>
<td></td>
</tr>
</tbody>
</table>
3. Translate each quantified NP into logic, replacing the head noun with a 1-place predicate whose argument is the NP variable:

<table>
<thead>
<tr>
<th>Moved out</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>∃x high-school-kid(x)</td>
<td>x won y.</td>
</tr>
<tr>
<td>∀y prize(y)</td>
<td></td>
</tr>
</tbody>
</table>

4. Turn the sentence into logic, using predicate principles.
Continuing procedure

3. Translate each quantified NP into logic, replacing the head noun with a 1-place predicate whose argument is the NP variable:

<table>
<thead>
<tr>
<th>Moved out</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\exists x \text{ high-school-kid}(x)$</td>
<td>$x$ won $y$.</td>
</tr>
<tr>
<td>$\forall y \text{ prize}(y)$</td>
<td></td>
</tr>
</tbody>
</table>

4. Turn the sentence into logic, using predicate principles:

<table>
<thead>
<tr>
<th>Moved out</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\exists x \text{ high-school-kid}(x)$</td>
<td>$\text{win}(x, y)$</td>
</tr>
<tr>
<td>$\forall y \text{ prize}(y)$</td>
<td></td>
</tr>
</tbody>
</table>
Add each NP translation back *one at a time*, using the right sentential connective:
Add each NP translation back one at a time, using the right sentential connective:

5a. Move first NP back, using right connective ( & ).

<table>
<thead>
<tr>
<th>Moved out</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \forall y \text{prize}(y) )</td>
<td>( \exists x \text{high-school-kid}(x) &amp; \text{win}(x, y) )</td>
</tr>
</tbody>
</table>
Add each NP translation back *one at a time*, using the right sentential connective:

### 5a. Move first NP back, using right connective ( & ).

<table>
<thead>
<tr>
<th>Moved out</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\forall y \text{prize}(y)$</td>
<td>$\exists x \text{high-school-kid}(x) &amp; \text{win}(x, y)$</td>
</tr>
</tbody>
</table>

### 5b. Move second NP back, using right connective ( → ).

<table>
<thead>
<tr>
<th>Moved out</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\forall y \text{prize}(y)$</td>
<td>$\forall y \text{prize}(y) \rightarrow (\exists x \text{high-school-kid}(x) &amp; \text{win}(x, y))$</td>
</tr>
</tbody>
</table>
Hold on! We’ve only got ONE of the two readings!

(12) a. For every prize, $x$, there was some high school kid, $y$, such that $y$ won $x$.

\[
\forall x [\text{prize}(x) \rightarrow \exists y [\text{high-school-kid}(y) \& \text{win}(y, x)] ]
\]

b.
Hold on! We’ve only got ONE of the two readings!

(13) a. For every prize, $x$, there was some high school kid, $y$, such that $y$ won $x$.

b. Some particular high school kid, $y$, won every prize, $x$.

a. $\forall x[\text{prize}(x) \rightarrow \exists y[\text{high-school-kid}(y) \& \text{win}(y, x)]]$

b. $\exists y[\text{high-school-kid}(y) \& \forall x[\text{prize}(x) \rightarrow \text{win}(y, x)]]$
Add each NP translation back *one at a time*, using the right sentential connective:
Add each \texttt{NP} translation back \textit{one at a time}, using the right sentential connective:

\begin{itemize}
  \item \textbf{5a. Move second \texttt{NP} back, using right connective (→).}
\end{itemize}

<table>
<thead>
<tr>
<th>Moved out</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\exists x \text{ high-school-kid}(x)$</td>
<td>$\forall y \text{ prize}(y) \rightarrow \text{win}(x, y)$</td>
</tr>
</tbody>
</table>
Add each NP translation back *one at a time*, using the right sentential connective:

5a. Move second NP back, using right connective ($\rightarrow$).

<table>
<thead>
<tr>
<th>Moved out</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\exists x$ high-school-kid($x$)</td>
<td>$\forall y$ prize($y$) $\rightarrow$ win($x$, $y$)</td>
</tr>
</tbody>
</table>

5b. Move first NP back, using right connective ($\&$).

<table>
<thead>
<tr>
<th>Moved out</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\exists x$ high-school-kid($x$) $&amp;$ ($\forall y$ prize($y$) $\rightarrow$ win($x$, $y$))</td>
<td></td>
</tr>
</tbody>
</table>
Consequences

What we did

We were able to capture the ambiguity by allowing the quantified $n_p$s to recombine with the main sentence translation in either order.

A new kind of ambiguity

1. Not lexical ambiguity
2. Not syntactic ambiguity.
3. What is it?
Consequences

What we did

We were able to capture the ambiguity by allowing the quantified \( n_p \)'s to recombine with the main sentence translation in either order.

A new kind of ambiguity

1. Not lexical ambiguity
2. Not syntactic ambiguity.
3. What is it?

Jean Mark Gawron (SDSU)
What we did

We were able to capture the ambiguity by allowing the quantified NPs to recombine with the main sentence translation in either order.
Consequences

What we did

We were able to capture the ambiguity by allowing the quantified NPs to recombine with the main sentence translation in either order.
Consequences

What we did

We were able to capture the ambiguity by allowing the quantified NPs to recombine with the main sentence translation in either order.

A new kind of ambiguity

Not lexical ambiguity

Not syntactic ambiguity.

What is it?
What we did
We were able to capture the ambiguity by allowing the quantified NPs to recombine with the main sentence translation in either order.

A new kind of ambiguity
1. Not lexical ambiguity
Consequences

What we did
We were able to capture the ambiguity by allowing the quantified NPs to recombine with the main sentence translation in either order.

A new kind of ambiguity
1. Not lexical ambiguity
2. Not syntactic ambiguity.
Consequences

What we did

We were able to capture the ambiguity by allowing the quantified NPs to recombine with the main sentence translation in either order.

A new kind of ambiguity

1. Not lexical ambiguity
2. Not syntactic ambiguity.
3. What is it?
Separating NP meanings from sentences

1. Unpassivize the sentence, if necessary.
2. Find all Qtfd np s and choose variable for each.
3. Remove Qtfd np s, leaving behind their variables.
4. Translate each np into logic.
5. Translate the main s into logic.
6. Move the np s back. Putting the np s back in different orders will capture different readings.
Separating NP meanings from sentences

- Unpassivize the sentence, if necessary.
Separating NP meanings from sentences

- Unpassivize the sentence, if necessary.
- Find all Qtfd NPs and choose variable for each.
Separating NP meanings from sentences

- Unpassivize the sentence, if necessary.
- Find all Qtfd NPs and choose variable for each.
- Remove Qtfd NPs, leaving behind their variables.
Separating NP meanings from sentences

- Unpassivize the sentence, if necessary.
- Find all Qtfd NPs and choose variable for each.
- Remove Qtfd NPs, leaving behind their variables.
- Translate each NP into logic.
Separating NP meanings from sentences

- Unpassivize the sentence, if necessary.
- Find all Qtfd NPs and choose variable for each.
- Remove Qtfd NPs, leaving behind their variables.
- Translate each NP into logic.
- Translate the main s into logic.
Separating NP meanings from sentences

- Unpassivize the sentence, if necessary.
- Find all Qtfd NPs and choose variable for each.
- Remove Qtfd NPs, leaving behind their variables.
- Translate each NP into logic.
- Translate the main s into logic.
- Move the NPs back. Putting the NPs back in different orders will capture different readings.
Outline

1. Introduction
2. Background ideas
   - General principles
   - Predicate Principles
3. Statement logic/predicates
4. Predicate logic
5. A recipe for English-to-Logic translation
6. Logical Form
7. Applying the recipe
8. Ambiguity
9. Embedded sentences
Relative clause

Replace pronouns with their antecedents.

(14) a. Maxine sent every letter John had written to her to Ruth.
Replace pronouns with their antecedents.

(15) a. Maxine sent every letter John had written to her to Ruth.
    b. Maxine sent every letter John had written to Maxine to Ruth.
Relative clause

Replace pronouns with their antecedents.

(16) a. Maxine sent every letter John had written to her to Ruth.
   b. Maxine sent every letter John had written to Maxine to Ruth.

Find each quantified NP, and choose a variable for it.
Replace pronouns with their antecedents.

(17) a. Maxine sent every letter John had written to her to Ruth.
    b. Maxine sent every letter John had written to Maxine to Ruth.

Find each quantified NP, and choose a variable for it.

1. Find Quantified NPs

Maxine sent every letter_\(x\) John had written to Maxine to Ruth.
Replace pronouns with their antecedents.

(18) a. Maxine sent every letter John had written to her to Ruth.
    b. Maxine sent every letter John had written to Maxine to Ruth.

1. Find each quantified NP, and choose a variable for it.

1. Find Quantified NPs

Maxine sent every letter$_x$ John had written to Maxine to Ruth.

2. Move each quantified NP out of the sentence, leaving the variable you chose behind.
Relative clause

Replace pronouns with their antecedents.

(19) a. Maxine sent every letter John had written to her to Ruth.
    b. Maxine sent every letter John had written to Maxine to Ruth.

1. Find each quantified NP, and choose a variable for it.

1. **Find Quantified NPs**

Maxine sent every letter\(x\) John had written to Maxine to Ruth.

2. Move each quantified NP out of the sentence, leaving the variable you chose behind.

2. **Remove Quantified NPs**

<table>
<thead>
<tr>
<th>Moved out</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>every letter(x) John had written to Maxine</td>
<td>Maxine sent (x) to Ruth.</td>
</tr>
</tbody>
</table>
Translate each quantified NP into logic, replacing the head noun with a 1-place predicate whose argument is the NP variable:
3. Translate each quantified NP into logic, replacing the head noun with a 1-place predicate whose argument is the NP variable:

### 3. NPs $\rightarrow$ logic

<table>
<thead>
<tr>
<th>Moved out</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\forall x \text{ letter}(x) &amp; \text{ write}(J, x, M)$</td>
<td>Maxine sent $x$ to Ruth.</td>
</tr>
</tbody>
</table>
Translate each quantified NP into logic, replacing the head noun with a 1-place predicate whose argument is the NP variable:

### 3. NPs → logic

<table>
<thead>
<tr>
<th>Moved out</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \forall x ) letter( (x) ) &amp; write( (J, x, M) )</td>
<td>Maxine sent ( x ) to Ruth.</td>
</tr>
</tbody>
</table>

Turn the sentence into logic, using predicate principles.
Translate each quantified NP into logic, replacing the head noun with a 1-place predicate whose argument is the NP variable:

3. NPs → logic

\[
\forall x \text{ letter}(x) \& \text{write}(J, x, M) \quad \text{Sentence: Maxine sent } x \text{ to Ruth.}
\]

Turn the sentence into logic, using predicate principles.

4. Sentence → logic

\[
\forall x \text{ letter}(x) \& \text{write}(J, x, M) \quad \text{Sentence: send}(M, x, R).
\]
5. Add each NP translation back *one at a time*, using the right sentential connective:
5 Add each **NP** translation back *one at a time*, using the right sentential connective:

5. Move **NP**’s back

Moved out Sentence
\[
\forall x \ (\text{letter}(x) \land \text{write}(J, x, M)) \rightarrow \text{send}(M, x, R).
\]
A complicated NP

Every letter\(_x\) John had sent to Maxine
\(\forall x \text{ letter}(x) \& \text{write}(J, x, M)\)

1. Recognize this Noun phrase contains a SENTENCE (relative clause).
   every letter  \([S \text{ John had sent to Maxine }]\)
   SUBJ    VERB    PP
A complicated NP

Every letter_x John had sent to Maxine
∀ x letter(x) & write(J, x, M)

1 Recognize this Noun phrase contains a SENTENCE (relative clause).
   every letter [s John had sent to Maxine]
   SUBJ VERB PP

That sentence says something about x

? John had sent to Maxine.
   every letter x [John had sent x to Maxine]
   send(J, x, M)
A complicated NP

Every letterᵧ John had sent to Maxine
∀ x letter(x) & write(J, x, M)

1. Recognize this Noun phrase contains a SENTENCE (relative clause).
   every letter  [s John had sent to Maxine ]
   SUBJ VERB PP

That sentence says something about x

? John had sent to Maxine.

∀ x [John had sent x to Maxine]
send(J, x, M)

2. Find where x goes, and translate the sentence on its own; add the translation to the translation of the NP:
   ∀x letter(x) & send(J, x, M)
Separating NP meanings from sentences

1. Unpassivize the sentence, if necessary.
2. Remove pronouns, if necessary.
3. Find all Qtfd np s and choose variable for each.
4. Remove Qtfd np s, leaving behind their variables.
5. Translate each np into logic, making sure you understand how each modifier relates to the np variable.
6. Translate the main s into logic.
7. Move the np s back. Putting the np s back in different orders will capture different readings.
Separating NP meanings from sentences

- Unpassivize the sentence, if necessary.
Separating NP meanings from sentences

- Unpassivize the sentence, if necessary.
- Remove pronouns, if necessary.
Separating NP meanings from sentences

- Unpassivize the sentence, if necessary.
- Remove pronouns, if necessary.
- Find all Qtfd NPs and choose variable for each.

Remove Qtfd NPs, leaving behind their variables.
Translate each NP into logic, making sure you understand how each modifier relates to the NP variable.
Translate the main s into logic.
Move the NPs back. Putting the NPs back in different orders will capture different readings.
Separating NP meanings from sentences

- Unpassivize the sentence, if necessary.
- Remove pronouns, if necessary.
- Find all Qtfd NPs and choose variable for each.
- Remove Qtfd NPs, leaving behind their variables.
Separating NP meanings from sentences

- Unpassivize the sentence, if necessary.
- Remove pronouns, if necessary.
- Find all Qtfd NPs and choose variable for each.
- Remove Qtfd NPs, leaving behind their variables.
- Translate each NP into logic, making sure you understand how each modifier relates to the NP variable.
Separating NP meanings from sentences

- Unpassivize the sentence, if necessary.
- **Remove pronouns, if necessary.**
- Find all Qtfd NPs and choose variable for each.
- Remove Qtfd NPs, leaving behind their variables.
- Translate each NP into logic, ensuring you understand how each modifier relates to the NP variable.
- Translate the main s into logic.
Separating NP meanings from sentences

- Unpassivize the sentence, if necessary.
- **Remove pronouns, if necessary.**
- Find all Qtfd NPs and choose variable for each.
- Remove Qtfd NPs, leaving behind their variables.
- Translate each NP into logic, making sure you understand how each modifier relates to the NP variable.
- Translate the main S into logic.
- Move the NPs back. Putting the NPs back in different orders will capture different readings.
(20)  a. A young woman arrived.
    b. Utopia welcomes every traveler from Spain.
    c. Every prize was won by some high school student.
    d. Maxine sent every letter John had written to her to Ruth.
Other translations

1. No spider plants dance.

2. There is a Santa Claus. [Paraphrase as] A Santa Claus exists

3. There's no business like show business. [Treat show business as a name; treat there is as before, paraphrase:] No business like show business exists, treat like as a preposition
Other translations

1. No spider plants dance.

\[ \sim \exists x \text{ spider-plant}(x) \& \text{dance}(x) \]

2. There is a Santa Claus. [Paraphrase this as A Santa Claus exists]
1. No spider plants dance.

   \[ \sim \exists x \text{spider-plant}(x) \& \text{dance}(x) \]

2. There is a Santa Claus. [Paraphrase this as *A Santa Claus exists*]

   \[ \exists x \text{Santa Claus}(x) \& \text{exists}(x) \]

3. There’s no business like show business. [Treat *show business* as a name; treat *there is* as before, paraphrase: *No business like show business exists*, treat *like* as a preposition]

   \[ \sim \exists x \text{business}(x) \& \text{like}(x, \text{sb}) \& \text{exists}(x) \]
1. No spider plants dance.

\[
\sim \exists x \text{ spider-plant}(x) \& \text{dance}(x)
\]

2. There is a Santa Claus. [Paraphrase this as *A Santa Claus exists*]

\[
\exists x \text{ Santa Claus}(x) \& \text{exists}(x)
\]

3. There’s no business like show business. [Treat *show business* as a name; treat *there is* as before, paraphrase: *No business like show business exists*, treat *like* as a preposition]

\[
\sim \exists \text{ business}(x) \& \text{like}(x, \text{SB}) \& \text{exists}(x)
\]
Conjunction can often be treated by producing a paraphrase with two conjoined sentences:

Grammar A generates all and only well-formed formula. [paraphrase this as *Grammar A generates all well-formed formula and Grammar A generates only well-formed formula.*; translate each conjoined sentence on its own.]