Introduction to Generalized Quantifier Theory

Jakub Szymanik

Institute for Logic, Language and Computation
University of Amsterdam

EGG 2013
Quantifiers are useful

Everyone knows everyone here.
Quantifiers are useful

*Everyone knows everyone here.*
Quantifiers are useful

Everyone knows everyone here.
Quantifiers are useful

Everyone knows everyone here.
Everyone knows everyone here.
Quantifiers are useful

*Everyone knows everyone here.*
Quantifiers are useful

Everyone knows everyone here.
Quantifiers are useful

*Everyone knows everyone here.*
Quantifiers are useful

*Everyone knows everyone here.*
Quantifiers are useful

*Everyone knows everyone here.*
Quantifiers are useful

*Everyone knows everyone here.*
Quantifiers are useful

*Everyone knows everyone here.*
Quantifiers are useful

*Everyone knows everyone here.*
Quantifiers are useful

*Everyone knows everyone here.*
Quantifiers are useful

Everyone knows everyone here.
Quantifiers are useful

Everyone knows everyone here.
Quantifiers are useful

*Everyone knows everyone here.*
Quantifiers are useful

Everyone knows everyone here.
Quantifiers are useful

*Everyone knows everyone here.*
Quantifiers are useful

Everyone knows everyone here.
Quantifiers are useful

*Everyone knows everyone here.*
Quantifiers are useful

Everyone knows everyone here.
Quantifiers are useful

Everyone knows everyone here.
Quantifiers are useful

*Everyone knows everyone here.*
Westerståhl, Generalized Quantifiers, SEP.

Peters & Westerståhl, Quantifiers in Language and Logic, OUP, 2008.


http://jakubszymanik.com/EGG2013/

http://www.jakubszymanik.com/
Outline

Generalized Quantifiers

Semantic universale

Monotonicity patterns

GQs in logic

Languages and automata

Computing quantifiers

Computational Complexity

Complex GQs

Polyadic quantifiers

Branching Quantifiers

Strong Reciprocity

Collective quantifiers
1. All poets have low self-esteem.
2. Some dean danced nude on the table.
3. At least 3 grad students prepared presentations.
4. An even number of the students saw a ghost.
5. Most of the students think they are smart.
6. Less than half of the students received good marks.
7. Many of the soldiers have not eaten for several days.
8. A few of the conservatives complained about taxes.
Determiners

Definition
Expressions that appear to be descriptions of quantity.

Example
All, not quite all, nearly all, an awful lot, a lot, a comfortable majority, most, many, more than \( n \), less than \( n \), quite a few, quite a lot, several, not a lot, not many, only a few, few, a few, hardly any, one, two, three.
Quantifiers are second-order relations

Observation

If we fix a model $\mathcal{M} = (M, A^M, B^M)$, then we can treat a generalized quantifier as a relation between relations over the universe.

Example

every$[A, B] = 1$ iff $A^M \subseteq B^M$
Quantifiers are second-order relations

Observation

If we fix a model $\mathbb{M} = (M, A^M, B^M)$, then we can treat a generalized quantifier as a relation between relations over the universe.

Example

\[
every[A, B] = 1 \text{ iff } A^M \subseteq B^M
\]

\[
even[A, B] = 1 \text{ iff } \text{card}(A^M \cap B^M) \text{ is even}
\]
Observation

If we fix a model $\mathcal{M} = (M, A^M, B^M)$, then we can treat a generalized quantifier as a relation between relations over the universe.

Example

$\text{every}[A, B] = 1$ iff $A^M \subseteq B^M$

$\text{even}[A, B] = 1$ iff $\text{card}(A^M \cap B^M)$ is even

$\text{most}[A, B] = 1$ iff $\text{card}(A^M \cap B^M) > \text{card}(A^M - B^M)$
Generalized Quantifiers

Definition
A quantifier Q is a way of associating with each set $M$ a function from pairs of subsets of $M$ into $\{0, 1\}$ (False, True).

Example

\[\text{every}_M[A, B] = 1 \text{ iff } A \subseteq B\]
Generalized Quantifiers

Definition
A quantifier $Q$ is a way of associating with each set $M$ a function from pairs of subsets of $M$ into $\{0, 1\}$ (False, True).

Example

$\text{every}_M[A, B] = 1$ iff $A \subseteq B$

$\text{even}_M[A, B] = 1$ iff $\text{card}(A \cap B)$ is even
Definition
A quantifier Q is a way of associating with each set $M$ a function from pairs of subsets of $M$ into $\{0, 1\}$ (False, True).

Example

$$\text{every}_M[A, B] = 1 \text{ iff } A \subseteq B$$

$$\text{even}_M[A, B] = 1 \text{ iff } \text{card}(A \cap B) \text{ is even}$$

$$\text{most}_M[A, B] = 1 \text{ iff } \text{card}(A \cap B) > \text{card}(A - B)$$
If card($M$) = $n$, then there are $2^{2^n}$ GQs.
For $n = 2$ it gives 65,536 possibilities.
Space of GQs

- If card($M$) = $n$, then there are $2^{2^n}$ GQs.
- For $n = 2$ it gives 65,536 possibilities.

Question

*Which of those correspond to simple determiners?*
Outline

Generalized Quantifiers

Semantic universale

Monotonicity patterns

GQs in logic

Languages and automata

Computing quantifiers

Computational Complexity

Complex GQs

Polyadic quantifiers
  Branching Quantifiers
  Strong Reciprocity

Collective quantifiers
Isomorphism closure
(ISOM) If \((M, A, B) \cong (M', A', B')\), then \(Q_M(A, B) \leftrightarrow Q_{M'}(A', B')\)

Topic neutrality
Extensionality

(EXT) If $M \subseteq M'$, then $Q_M(A, B) \iff Q_{M'}(A, B)$
Conservativity

(CONS) $Q_M(A, B) \iff Q_M(A, A \cap B)$
Research questions

- Do all NL determiners satisfy ISOM, EXT and CONS?
- Only?
- If yes, why?
  1. Learnability?
  2. Evolution?

Hunter & Lidz, Conservativity and learnability of determiners, Journal of Semantics, 2010
1. There are blue non-circles . . . Are all the circle blue?
2. There are elephants not being ridden by a girl . . .
   Is every girl riding an elephant?
Gleep and gleeb’

\[ \text{gleeb}_M[A, B] = 1 \text{ iff } A \not\subseteq B \]

\[ \text{gleeb}_M[A, B] = 1 \text{ iff } B \not\subseteq A \]
Picky puppet task

The puppet told me that he likes this card because gleeb girls are on the beach.

The puppet told me that he doesn’t like this card because it’s not true that gleeb girls are on the beach.

<table>
<thead>
<tr>
<th>Condition</th>
<th>Conservative</th>
<th>Nonconservative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cards correctly sorted (out of 5)</td>
<td>mean 4.1 (above chance, ( p &lt; 0.0001 ))</td>
<td>mean 3.1 (not above chance, ( p &gt; 0.2488 ))</td>
</tr>
<tr>
<td>Subjects with “perfect” accuracy</td>
<td>50%</td>
<td>10%</td>
</tr>
</tbody>
</table>
Number triangle representation

(0,0)

(1,0)    (0,1)

(2,0)    (1,1)    (0,2)

(3,0)    (2,1)    (1,2)    (0,3)

(4,0)    (3,1)    (2,2)    (1,3)    (0,4)

...    ...    ...
Number triangle representation
Definition
A monadic generalized quantifier of type (1,1) is a class Q of structures of the form $M = (U, A_1, A_2)$, where $A_1, A_2 \subseteq U$. Additionally, Q is closed under isomorphism.
Examples

\[
\text{every } = \{(M, A, B) \mid A, B \subseteq M \text{ and } A \subseteq B\}.
\]
Examples

every = \{(M, A, B) \mid A, B \subseteq M \text{ and } A \subseteq B\}.

some = \{(M, A, B) \mid A, B \subseteq M \text{ and } A \cap B \neq \emptyset\}.
Examples

every = \{ (M, A, B) \mid A, B \subseteq M \text{ and } A \subseteq B \}.

some = \{ (M, A, B) \mid A, B \subseteq M \text{ and } A \cap B \neq \emptyset \}.

more than k = \{ (M, A, B) \mid A, B \subseteq M \text{ and } \text{card}(A \cap B) > k \}.
Examples

every = \{ (M, A, B) \mid A, B \subseteq M \text{ and } A \subseteq B \}.

some = \{ (M, A, B) \mid A, B \subseteq M \text{ and } A \cap B \neq \emptyset \}.

more than k = \{ (M, A, B) \mid A, B \subseteq M \text{ and } \text{card}(A \cap B) > k \}.

even = \{ (M, A, B) \mid A, B \subseteq M \text{ and } \text{card}(A \cap B) \text{ is even} \}.
Examples

every = \{(M, A, B) \mid A, B \subseteq M \text{ and } A \subseteq B\}.

some = \{(M, A, B) \mid A, B \subseteq M \text{ and } A \cap B \neq \emptyset\}.

more than k = \{(M, A, B) \mid A, B \subseteq M \text{ and } \text{card}(A \cap B) > k\}.

even = \{(M, A, B) \mid A, B \subseteq M \text{ and } \text{card}(A \cap B) \text{ is even}\}.

most = \{(M, A, B) \mid A, B \subseteq M \text{ and } \text{card}(A \cap B) > \text{card}(A - B)\}.
Outline

Generalized Quantifiers

Semantic universale

Monotonicity patterns

GQs in logic

Languages and automata

Computing quantifiers

Computational Complexity

Complex GQs

Polyadic quantifiers
  Branching Quantifiers
  Strong Reciprocity

Collective quantifiers
Monotonicity

\[ \uparrow \text{MON} \quad Q_M[A, B] \text{ and } A \subseteq A' \subseteq M \text{ then } Q_M[A', B]. \]

\[ \downarrow \text{MON} \quad Q_M[A, B] \text{ and } A' \subseteq A \subseteq M \text{ then } Q_M[A', B]. \]

\[ \text{MON} \uparrow \quad Q_M[A, B] \text{ and } B \subseteq B' \subseteq M \text{ then } Q_M[A, B']. \]

\[ \text{MON} \downarrow \quad Q_M[A, B] \text{ and } B' \subseteq B \subseteq M \text{ then } Q_M[A, B']. \]
Inference test

1. Some boy is dirty.
2. Some child is dirty.

1. All child is dirty.
2. All boy is dirty.

1. All boy is muddy.
2. All boy is dirty.

1. No boy is dirty.
2. No boy is muddy.

1. Exactly five children are dirty.
2. Exactly five boys are dirty.
Boolean combinations

1. At least 5 or at most 10 departments can win EU grants. (disjunction)
2. Between 100 and 200 students started in the marathon. (conjunction)
3. Not all students passed. (outer negation)
4. All students did not pass. (inner negation)

Definition

\[(Q \land Q')_M[A, B] \iff Q_M[A, B] \text{ and } Q'_M[A, B] \text{ (conjunction)}\]
\[(Q \lor Q')_M[A, B] \iff Q_M[A, B] \text{ or } Q'_M[A, B] \text{ (disjunction)}\]
\[(-Q)_M[A, B] \iff \text{not } Q_M[A, B] \text{ (complement)}\]
\[(Q\neg)_M[A, B] \iff Q_M[A, M - B] \text{ (post-complement)}\]
Monotonicity interacts with negation

**Theorem**

*Q is MON\uparrow*  
1. *iff ¬Q is MON\downarrow.*  
2. *iff Q\neg is MON\downarrow.*

*Q is ↑MON*  
1. *iff ¬Q is ↓MON.*  
2. *iff Q\neg is ↑MON.*

*Similarly for the downward monotone case.*
Square of opposition

- some, $\neg$ some $=$ no, some $\neg$ $=$ not all, $\neg$ some $\neg$ $=$ all.
- some is $\uparrow$MON$\uparrow$.
- Therefore, no is $\downarrow$MON$\downarrow$, not all is $\uparrow$MON$\downarrow$, and all is $\downarrow$MON$\uparrow$. 
It even helps to find an efficient solution for:
It even helps to find an efficient solution for:

Gerasimczuk & Szymanik, Invariance properties of quantifiers and multiagent information exchange, Proc. of 12th Meeting on Mathematics of Language
Outline

Generalized Quantifiers

Semantic universale

Monotonicity patterns

GQs in logic

Languages and automata

Computing quantifiers

Computational Complexity

Complex GQs

Polyadic quantifiers
  Branching Quantifiers
  Strong Reciprocity

Collective quantifiers
Definition

Let $Q$ be a generalized quantifier and $L$ a logic. We say that the quantifier $Q$ is \textit{definable} in $L$ if there is a sentence $\varphi \in L$ such that for any $M$:

$$M \models \varphi \iff Q_M[A, B].$$
Some GQs, like $\exists^{\leq 3}$, $\exists^{= 3}$, and $\exists^{\geq 3}$, are expressible in FO.

Example

\[
some \ x \ [A(x), B(x)] \iff \exists x [A(x) \land B(x)].
\]
Theorem

The quantifiers ‘there exists (in)finitely many’, most and even are not first-order definable.
Theorem

The quantifiers ‘there exists (in)finitely many’, most and even are not first-order definable.

We can use higher-order logics:
Non-elementary GQs

Theorem
The quantifiers ‘there exists (in)finitely many’, most and even are not first-order definable.

We can use higher-order logics:

Example
In $\mathcal{M} = (M, A^M, B^M)$ the sentence

$$\text{most } x [A(x), B(x)]$$

is true if and only if the following condition holds:

$$\exists f : (A^M - B^M) \rightarrow (A^M \cap B^M)$$

such that $f$ is injective but not surjective.
Theorem (Westerståhl 1998)

In finite models, persistent quantifiers satisfying EXT, ISOM and CONS are FO-definable.
Outline

Generalized Quantifiers
Semantic universale
Monotonicity patterns
GQs in logic
Languages and automata
Computing quantifiers
Computational Complexity
Complex GQs
  Polyadic quantifiers
  Branching Quantifiers
  Strong Reciprocity
  Collective quantifiers
Languages - basic definitions

- **Alphabet** is any non-empty finite set of symbols, e.g., $A = \{a, b\}$ and $B = \{0, 1\}$. 
Languages - basic definitions

- *Alphabet* is any non-empty finite set of symbols, e.g., $A = \{a, b\}$ and $B = \{0, 1\}$.
- A *word (string)* is a finite sequence of symbols from a given alphabet, e.g., “1110001110”.
Languages - basic definitions

- **Alphabet** is any non-empty finite set of symbols, e.g., \( A = \{ a, b \} \) and \( B = \{ 0, 1 \} \).
- A **word (string)** is a finite sequence of symbols from a given alphabet, e.g., "1110001110".
- The **empty word**, \( \varepsilon \), is a sequence without symbols.
Languages - basic definitions

- **Alphabet** is any non-empty finite set of symbols, e.g., $A = \{a, b\}$ and $B = \{0, 1\}$.
- A **word (string)** is a finite sequence of symbols from a given alphabet, e.g., “1110001110”.
- The **empty word**, $\varepsilon$, is a sequence without symbols.
- The **length of a word** is the number of symbols in it.
Languages - basic definitions

- **Alphabet** is any non-empty finite set of symbols, e.g., \( A = \{a, b\} \) and \( B = \{0, 1\} \).
- A **word (string)** is a finite sequence of symbols from a given alphabet, e.g., “1110001110”.
- The **empty word**, \( \varepsilon \), is a sequence without symbols.
- The **length of a word** is the number of symbols in it.
- *The set of all words over alphabet* \( \Gamma \) is denoted by \( \Gamma^* \), e.g., \( \{0, 1\}^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, 000, \ldots\} \).
Alphabet is any non-empty finite set of symbols, e.g., $A = \{a, b\}$ and $B = \{0, 1\}$.

A word (string) is a finite sequence of symbols from a given alphabet, e.g., “1110001110”.

The empty word, $\varepsilon$, is a sequence without symbols.

The length of a word is the number of symbols in it.

The set of all words over alphabet $\Gamma$ is denoted by $\Gamma^*$, e.g., $\{0, 1\}^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, 000, \ldots\}$.

Any set of words, a subset of $\Gamma^*$, will be called a language.
Finite automata

Definition
A *non-deterministic finite automaton* (FA) is a tuple \((A, Q, q_s, F, \delta)\), where:
- \(A\) is an input alphabet;
- \(Q\) is a finite set of states;
- \(q_s \in Q\) is an initial state;
- \(F \subseteq Q\) is a set of accepting states;
- \(\delta : Q \times A \rightarrow \mathcal{P}(Q)\) is a transition function.
Regular languages

Definition
*The language accepted (recognized) by some FA* $H$, $L(H)$, *is the set of all words over the alphabet* $A$ *which are accepted by* $H$.

Definition
*We say that a language* $L \subseteq A^*$ *is regular* if and only if there exists some FA $H$ *such that* $L = L(H)$.
Example 1

Let $A = \{a, b\}$ and consider the language $L_1 = A^*$. 

```
a, b

\rightarrow q_1
```
Example 2

Let $L_2 = \emptyset$
Example 3

\[ L_3 = \{ \varepsilon \} \]
Not every language is regular

\[ L_{ab} = \{ a^n b^n : n \geq 1 \} \]
Definition
A non-deterministic push-down automaton (PDA) is a tuple \((A, \Gamma, \#, Q, q_s, F, \delta)\), where:

- \(A\) is an input alphabet;
- \(\Gamma\) is a stack alphabet;
- \(# \not\in \Gamma\) is a stack initial symbol, empty stack consists only from it;
- \(Q\) is a finite set of states;
- \(q_s \in Q\) is an initial state;
- \(F \subseteq Q\) is a set of accepting states;
- \(\delta: Q \times (A \cup \{\varepsilon\}) \times \Gamma \rightarrow \mathcal{P}(Q \times \Gamma^*)\) is a transition function.
push/pop-off a symbol from the top of the stack
Definition
We say that a language $L \subseteq A^*$ is context-free if and only if there is a PDA $H$ such that $L = L(H)$.
There is a PDA for $L_{ab} = \{a^n b^n : n \geq 1\}$. 
Beyond context-free languages

$L_{abc} = \{a^k b^k c^k : k \geq 1\}$

We will investigate stronger languages in the last lecture.
Chomsky’s Hierarchy

- Regular
- Context-free
- Context-sensitive
- Recursive
General definition
How do we encode models?

This model is uniquely described by $\alpha_M = a_{\bar{A}\bar{B}} a_{\bar{A}B} a_{AB} a_{\bar{A}B} a_{AB}$
Step by step

- Restriction to finite models of the form $M = (U, A, B)$. 

  - List of all elements of the model: $c_1, \ldots, c_5$.
  - Labeling every element with one of the letters: $\bar{A}\bar{B}$, $A\bar{B}$, $\bar{A}B$, $A\bar{A}B$, according to constituents it belongs to.
  - Result: the word $\alpha_M = \bar{A}\bar{B}A\bar{B}\bar{A}\bar{A}B\bar{A}\bar{AB}$. 

  - $\alpha_M$ describes the model in which:
    - $c_1 \in \bar{A}\bar{B}$,
    - $c_2 \in A\bar{B}$,
    - $c_3 \in AB$,
    - $c_4 \in \bar{A}B$,
    - $c_5 \in \bar{A}AB$.

  - The class $Q$ is represented by the set of words describing all elements of the class.
Step by step

- Restriction to **finite models** of the form $M = (U, A, B)$.
- List of all elements of the model: $c_1, \ldots, c_5$. 
Step by step

- Restriction to finite models of the form $M = (U, A, B)$.
- List of all elements of the model: $c_1, \ldots, c_5$.
- Labeling every element with one of the letters: $a_{\bar{A}B}, a_{A\bar{B}}, a_{\bar{A}B}, a_{AB}$, according to constituents it belongs to.
Step by step

- Restriction to finite models of the form $M = (U, A, B)$.
- List of all elements of the model: $c_1, \ldots, c_5$.
- Labeling every element with one of the letters: $a_{\overline{A}B}$, $a_{\overline{A}B}$, $a_{\overline{A}B}$, $a_{AB}$, according to constituents it belongs to.
- Result: the word $\alpha_M = a_{\overline{A}B}a_{\overline{A}B}a_{AB}a_{\overline{A}B}a_{\overline{A}B}$.
Step by step

- Restriction to finite models of the form $M = (U, A, B)$.
- List of all elements of the model: $c_1, \ldots, c_5$.
- Labeling every element with one of the letters: $a_{\overline{AB}}, a_{A\overline{B}}, a_{\overline{A}B}, a_{AB}$, according to constituents it belongs to.
- Result: the word $\alpha_M = a_{\overline{AB}}a_{A\overline{B}}a_{AB}a_{\overline{A}B}a_{\overline{A}B}$.
- $\alpha_M$ describes the model in which:
  - $c_1 \in \overline{A}B$, $c_2 \in A\overline{B}c_3 \in AB$, $c_4 \in \overline{A}B$, $c_5 \in \overline{A}B$. 

The class $Q$ is represented by the set of words describing all elements of the class.
Step by step

- Restriction to finite models of the form $M = (U, A, B)$.
- List of all elements of the model: $c_1, \ldots, c_5$.
- Labeling every element with one of the letters: $a_{\bar{A}B}$, $a_{A\bar{B}}$, $a_{\bar{A}B}$, $a_{AB}$, according to constituents it belongs to.
- Result: the word $\alpha_M = a_{\bar{A}B}a_{A\bar{B}}a_{AB}a_{\bar{A}B}a_{\bar{A}B}$.
- $\alpha_M$ describes the model in which: $c_1 \in \bar{A}\bar{B}$, $c_2 \in A\bar{B}$, $c_3 \in AB$, $c_4 \in \bar{A}B$, $c_5 \in \bar{A}B$.
- The class $Q$ is represented by the set of words describing all elements of the class.
Aristotelian quantifiers

“all”, “some”, “no”, and “not all”

\[ \Gamma - \{ a_{\bar{A}B} \} \]

Finite automaton recognizing \( L_{\text{All}} \)

\[ L_{\text{All}} = \{ \alpha \in \Gamma^* : \#a_{\bar{A}B}(\alpha) = 0 \} \]
Cardinal quantifiers

E.g. “more than 2”, “less than 7”, and “between 8 and 11”

\[ \Gamma - \{ a_{AB} \} \]

Finite automaton recognizing \( L_{\text{More than two}} \)

\[ L_{\text{More than two}} = \{ \alpha \in \Gamma^* : \#a_{AB}(\alpha) > 2 \} \]
Parity quantifiers

E.g. “an even number”, “an odd number”

\[ \Gamma - \{ a_{AB} \} \]

Finite automaton recognizing \( L_{\text{Even}} \)

\[ L_{\text{Even}} = \{ \alpha \in \Gamma^* : \#a_{AB}(\alpha) \text{ is even} \} \]
Proportional quantifiers

- E.g. “most”, “less than half”.
- Most $A$ are $B$ iff $\text{card}(A \cap B) > \text{card}(A - B)$.
- $L_{\text{Most}} = \{ \alpha \in \Gamma^* : \#a_{AB}(\alpha) > \#a_{\overline{A}B}(\alpha) \}$.
- There is no finite automaton recognizing this language.
- We need internal memory.
- A push-down automata will do.
Summing up

<table>
<thead>
<tr>
<th>Definability</th>
<th>Examples</th>
<th>Recognized by</th>
</tr>
</thead>
<tbody>
<tr>
<td>FO</td>
<td>“all” “at least 3”</td>
<td>acyclic FA</td>
</tr>
<tr>
<td>FO($D_n$)</td>
<td>“an even number”</td>
<td>FA</td>
</tr>
<tr>
<td>PrA</td>
<td>“most”, “less than half”</td>
<td>PDA</td>
</tr>
</tbody>
</table>

Quantifiers, definability, and complexity of automata

Mostowski, Computational semantics for monadic quantifiers, 1998.
Does it say anything about processing?

Question

Do minimal automata predict differences in verification?
Quantifiers
Logic

Quantifiers
Quantifiers

Logic

Psycholinguistics
Quantifiers
A simple study

More than half of the cars are yellow.
Neurobehavioral studies

Differences in brain activity.

- All quantifiers are associated with numerosity: recruit right inferior parietal cortex.
- Only higher-order activate working-memory capacity: recruit right dorsolateral prefrontal cortex.

McMillan et al., Neural basis for generalized quantifiers comprehension, Neuropsychologia, 2005

Szymanik, A Note on some neuroimaging study of natural language quantifiers comprehension, Neuropsychologia, 2007
Experiment with schizophrenic patients

- Compare performance of:
  - Healthy subjects.
  - Patients with schizophrenia.
    - Known WM deficits.
RT data
Accuracy data

Zajenkowski et al., A computational approach to quantifiers as an explanation for some language impairments in schizophrenia, Journal of Communication Disorders, 2011.
Teasing apart WM from executive control

- Executive control within Attention Networks Test (ANT)
  - resolution of conflict between expectation, stimulus, and response
  - = incongruent flanking – congruent flanking

<table>
<thead>
<tr>
<th>Neutral</th>
<th>Congruent</th>
<th>Incongruent</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
+ Intelligence

- Quantifier verification + Sternberg’s STM + ANT
- Raven’s Advanced Progressive Matrices Test (APM)
  1. test of fluid intelligence
  2. find a missing one
Quantifiers, WM, and intelligence

- All quantifier correlated with STM.
- Only proportional quantifiers correlated with ANT.
- APM correlated best with proportional quantifiers.
- APM attenuated ANT.

Zajenkowski and Szymanik. Most intelligent people are accurate and some fast people are intelligent, Intelligence 2013
Research questions

1. Build computational cognitive model
   - visual aspects
   - working memory model
   - pragmatics
   - etc.
Distribution is skewed towards quantifiers of low complexity

Outline

Generalized Quantifiers

Semantic universale

Monotonicity patterns

GQs in logic

Languages and automata

Computing quantifiers

Computational Complexity

Complex GQs

Polyadic quantifiers

Branching Quantifiers

Strong Reciprocity

Collective quantifiers
Model of Computation
Question

What amount of resources TM needs to solve a task?
Computational Complexity Theory

Question
What amount of resources TM needs to solve a task?

Theorem (determinism vs. non-determinism)
If there is a non-deterministic Turing machine $N$ recognizing a language $L$, then there exists a deterministic Turing machine $M$ for language $L$. 

Question
The simulation takes $O(f(n))$. Can we do it significantly faster?
Question
What amount of resources TM needs to solve a task?

Theorem (determinism vs. non-determinism)
If there is a non-deterministic Turing machine N recognizing a language L, then there exists a deterministic Turing machine M for language L.

Question
The simulation takes $O(c^{f(n)})$. Can we do it significantly faster?
Time Complexity

Let $f : \omega \rightarrow \omega$. 
Let $f : \omega \rightarrow \omega$.

**Definition**
TIME($f$) is the class of languages (problems) which can be recognized by a deterministic Turing machine in time bounded by $f$ with respect to the length of the input.
Time Complexity

Let $f : \omega \rightarrow \omega$.

**Definition**
TIME($f$) is the class of languages (problems) which can be recognized by a deterministic Turing machine in time bounded by $f$ with respect to the length of the input.

**Definition**
NTIME($f$), is the class of languages $L$ for which there exists a non-deterministic Turing machine $M$ such that for every $x \in L$ all branches in the computation tree of $M$ on $x$ are bounded by $f(n)$ and moreover $M$ decides $L$.
Complexity Classes P and NP

Definition

- $\text{PTIME} = \bigcup_{k \in \omega} \text{TIME}(n^k)$
- $\text{NPTIME} = \bigcup_{k \in \omega} \text{NTIME}(n^k)$
Complexity Classes P and NP

Definition

- $\text{PTIME} = \bigcup_{k \in \omega} \text{TIME}(n^k)$
- $\text{NPTIME} = \bigcup_{k \in \omega} \text{NTIME}(n^k)$

Question (Millenium Problem)

$P=NP$?
Definition
We say that a function $f : A \rightarrow A$ is a *polynomial time computable function* iff there exists a deterministic Turing machine computing $f(w)$ for every $w \in A$ in polynomial time.

Definition
A problem $L$ is polynomial reducible to a problem $L'$ if there is a polynomial time computable function such that $w \in L \iff f(w) \in L'$.

Definition
A language $L$ is NP-complete if $L \in \text{NP}$ and every language in NP is reducible to $L$. 

(In)tractability
(In)tractability

Definition
We say that a function $f : A \rightarrow A$ is a *polynomial time computable function* iff there exists a deterministic Turing machine computing $f(w)$ for every $w \in A$ in polynomial time.

Definition
A problem $L$ is polynomial reducible to a problem $L'$ if there is a polynomial time computable function such that

$$w \in L \iff f(w) \in L'.$$
(In)tractability

Definition
We say that a function \( f : A \rightarrow A \) is a \textit{polynomial time computable function} iff there exit a deterministic Turing machine computing \( f(w) \) for every \( w \in A \) in polynomial time.

Definition
A problem \( L \) is polynomial reducible to a problem \( L' \) if there is a polynomial time computable function such that

\[
    w \in L \iff f(w) \in L'.
\]

Definition
A language \( L \) is \( NP \)-complete if \( L \in NP \) and every language in \( NP \) is reducible to \( L \).
Quantifiers in Finite Models

- Finite models can be encoded as strings.
- GQs as classes of such finite strings are languages.
Quantifiers in Finite Models

- Finite models can be encoded as strings.
- GQs as classes of such finite strings are languages.

**Definition**

By the *complexity of a quantifier* $Q$ we mean the computational complexity of the corresponding class of finite models.

**Question**

$M \in Q$? (equivalently $M \models Q$?)
Outline

Generalized Quantifiers

Semantic universale

Monotonicity patterns

GQs in logic

Languages and automata

Computing quantifiers

Computational Complexity

Complex GQs

Polyadic quantifiers

Branching Quantifiers

Strong Reciprocity

Collective quantifiers
Outline

Generalized Quantifiers

Semantic universale

Monotonicity patterns

GQs in logic

Languages and automata

Computing quantifiers

Computational Complexity

Complex GQs

Polyadic quantifiers

Branching Quantifiers

Strong Reciprocity

Collective quantifiers
Multi-quantifier sentences

1. Most villagers and most townsmen hate each other.
2. Three PMs referred to each other indirectly
   \[ Q[A, B, R] \text{ or } Q[A, R] \]
Lindström quantifiers

Definition
Let \( t = (n_1, \ldots, n_k) \) be a \( k \)-tuple of positive integers. A *generalized quantifier* of type \( t \) is a class \( Q \) of models of a vocabulary \( \tau_t = \{ R_1, \ldots, R_k \} \), such that \( R_i \) is \( n_i \)-ary for \( 1 \leq i \leq k \), and \( Q \) is closed under isomorphisms.
Lindström quantifiers

**Definition**
Let \( t = (n_1, \ldots, n_k) \) be a \( k \)-tuple of positive integers. A *generalized quantifier* of type \( t \) is a class \( Q \) of models of a vocabulary \( \tau_t = \{ R_1, \ldots, R_k \} \), such that \( R_i \) is \( n_i \)-ary for \( 1 \leq i \leq k \), and \( Q \) is closed under isomorphisms.

**Definition**
If in the above definition for all \( i: n_i = 1 \), then we say that a quantifier is *monadic*, otherwise we call it *polyadic*. 
Lindström quantifiers

Definition
Let $t = (n_1, \ldots, n_k)$ be a $k$-tuple of positive integers. A generalized quantifier of type $t$ is a class $Q$ of models of a vocabulary $\tau_t = \{R_1, \ldots, R_k\}$, such that $R_i$ is $n_i$-ary for $1 \leq i \leq k$, and $Q$ is closed under isomorphisms.

Definition
If in the above definition for all $i$: $n_i = 1$, then we say that a quantifier is monadic, otherwise we call it polyadic.

\[
W = \{(M, R) \mid R \subseteq M^2 \text{ and } R \text{ is a well-order}\}.
\]
\[
\text{Ram} = \{(M, A, R) \mid A \subseteq M, R \subseteq M^2 \text{ and } \forall a, b \in A \ R(a, b)\}.
\]
Coding

Definition
Let \( \tau = \{ R_1, \ldots, R_k \} \) be a relational vocabulary and \( \mathcal{M} \) a \( \tau \)-model of the following form:
\( \mathcal{M} = (U, R^M_1, \ldots, R^M_k) \), where \( U = \{1, \ldots, n\} \) is the universe of model \( \mathcal{M} \) and \( R^M_i \subseteq U^{n_i} \) is an \( n_i \)-ary relation over \( U \), for \( 1 \leq i \leq k \). We define a binary encoding for \( \tau \)-models. The code for \( \mathcal{M} \) is a word over \( \{0, 1, \#\} \) of length \( O((\text{card}(U))^c) \), where \( c \) is the maximal arity of the predicates in \( \tau \) (or \( c = 1 \) if there are no predicates).
The code has the following form:
\[
\tilde{n}\#\tilde{R}^M_1\#\ldots\#\tilde{R}^M_n,
\]
where:

- \( \tilde{n} \) is the part coding the universe of the model and consists of \( n \) 1s.
- \( \tilde{R}^M_i \) — the code for the \( n_i \)-ary relation \( R^M_i \) — is an \( n_i \)-bit string whose \( j \)-th bit is 1 iff the \( j \)-th tuple in \( U^{n_i} \) (ordered lexicographically) is in \( R^M_i \).
- \( \# \) is a separating symbol.
Consider vocabulary $\sigma = \{P, R\}$, where $P$ is a unary predicate and $R$ a binary relation. Take the $\sigma$-model $\mathcal{M} = (M, P^M, R^M)$, where the universe $M = \{1, 2, 3\}$, the unary relation $P^M \subseteq M$ is equal to $\{2\}$ and the binary relation $R^M \subseteq M^2$ consists of the pairs $(2, 2)$ and $(3, 2)$.

- $\tilde{n}$ consists of three 1s as there are three elements in $M$.
- $\tilde{P}^M$ is the string of length three with 1s in places corresponding to the elements from $M$ belonging to $P^M$. Hence $\tilde{P}^M = 010$ as $P^M = \{2\}$.
- $\tilde{R}^M$ is obtained by writing down all $3^2 = 9$ binary strings of elements from $M$ in lexicographical order and substituting 1 in places corresponding to the pairs belonging to $R^M$ and 0 in all other places. As a result $\tilde{R}^M = 000010010$.

Adding all together the code for $\mathcal{M}$ is $111\#010\#000010010$. 
1. Most logicians criticized some papers.
2. It(most, some)[Logicians, Papers, Criticized].

**Definition**
Let $Q$ and $Q'$ be generalized quantifiers of type $(1, 1)$. Let $A, B$ be subsets of the universe and $R$ a binary relation over the universe. Suppressing the universe, we will define the *iteration* operator as follows:

$$\text{It}(Q, Q')[A, B, R] \iff Q[A, \{a \mid Q'[B, R(a)]\}],$$

where $R(a) = \{b \mid R(a, b)\}$. 
Most girls and most boys hate each other.
Theorem (Steinert-Threlkeld & Icard)

Let $Q$ and $Q'$ be computable by DFA (PDA), then $\text{It}(Q, Q')$ is also DFA (PDA) computable.
Eighty professors taught sixty courses at ESSLLI’08.

**Definition**

\[ \text{Cum}(Q, Q’)[A, B, R] \iff \text{It}(Q, \text{some})[A, B, R] \land \text{It}(Q’, \text{some})[B, A, R^{-1}] \]
Most girls and most boys hate each other.
Possibly branching sentences

1. Most villagers and most townsmen hate each other.
2. One third of villagers and half of townsmen hate each other.
3. 5 villagers and 7 townsmen hate each other.
Branching reading

- Most girls and most boys hate each other.

\[
\exists A \exists A' [\text{most}(G, A) \land \text{most}(B, A') \land \forall x \in A \forall y \in A' H(x, y)].
\]
Most girls and most boys hate each other.
Definition

Let \( Q \) and \( Q' \) be both \( \text{MON}^\uparrow \) quantifiers of type \((1, 1)\). Define the branching of quantifier symbols \( Q \) and \( Q' \) as the type \((1, 1, 2)\) quantifier symbol \( \text{Br}(Q, Q') \). A structure \( M = (M, A, B, R) \in \text{Br}(Q, Q') \) if the following holds:

\[
\exists X \subseteq A \ \exists Y \subseteq B[(X, A) \in Q \land (Y, B) \in Q' \land X \times Y \subseteq R].
\]
Branching readings are intractable

**Theorem**

*Proportional branching sentences are NP-complete.*
Two-way quantification

\[ \text{lt}(Q_1, Q_2) \land \text{lt}(Q_2, Q_1) \]
Two-way quantification

\[ \text{It}(Q_1, Q_2) \land \text{It}(Q_2, Q_1) \]
Two-way quantification

\[ \text{It}(Q_1, Q_2) \land \text{It}(Q_2, Q_1) \]

Subjects are happy to accept such interpretation.

Gerasimczuk & Szymanik, Branching Quantification vs. Two-way Quantification, Journal of Semantics, 2009
Potentially strong reciprocal sentences

1. Andi, Jarmo and Jakub laughed at one another.
2. 15 men are hitting one another.
3. Most of the PMs refer to each other.
Most of the PMs refer to each other.
Strong reciprocal lift

**Definition**
Let \( Q \) be a right monotone increasing quantifier of type \((1, 1)\). We define:

\[
\text{Ram}_{s}(Q)[A, R] \iff \exists X \subseteq A[Q(A, X) \\
\land \forall x, y \in X(x \neq y \implies R(x, y))].
\]
Most Boston pitchers sat alongside each other.
Intermediate reciprocal lift

**Definition**

\[
\text{Ram}_1(Q)[A, R] \iff \exists X \subseteq A[Q(A, X)] \\
\wedge \forall x, y \in X (x \neq y \implies \exists \text{ sequence } z_1, \ldots, z_\ell \in X \text{ such that} \\
(z_1 = x \wedge R(z_1, z_2) \wedge \ldots \wedge R(z_{\ell-1}, z_\ell) \wedge z_\ell = y).
\]
Some pirates were staring at each other in surprise.
Weak reciprocal lift

Definition

\[ \text{Ram}_W(Q)[A, R] \iff \exists X \subseteq A[Q(A, X) \land \forall x \in X \exists y \in X(x \neq y \land R(x, y))] \]
Hypothesis

Reading associated with the reciprocal in a given sentence is the strongest available reading which is consistent with relevant information supplied by the context.
Strong Meaning Hypothesis

Hypothesis

*Reading associated with the reciprocal in a given sentence is the strongest available reading which is consistent with relevant information supplied by the context.*

Example

1. The children followed each other into the church.
2. The children followed each other around the Maypole.

Dalrymple et al., Reciprocal Expressions and the Concept of Reciprocity. Linguistics and Philosophy, 1998.

Research question

Draw:
1. All/Most of the dots are connected to each other.
Research question

Draw:

1. All/Most of the dots are connected to each other.

   - Against SMH:
     - Ambiguous between strong and intermediate.
Research question

Draw:

1. All/Most of the dots are connected to each other.

   ▶ Against SMH:
      ▶ Ambiguous between strong and intermediate.
      ▶ In line with complexity: fewer strong pictures for ‘most’.

Bott et al., Interpreting Tractable versus Intractable Reciprocal Sentences, Proceedings of the International Conference on Computational Semantics, 2011.

Outline

Generalized Quantifiers

Semantic universale

Monotonicity patterns

GQs in logic

Languages and automata

Computing quantifiers

Computational Complexity

Complex GQs

Polyadic quantifiers

Branching Quantifiers

Strong Reciprocity

Collective quantifiers
Collectivity

(1.) All the Knights but King Arthur met in secret.
(2.) Most climbers are friends.
(3.) John and Mary love each other.
(4.) The samurai were twelve in number.
(5.) Many girls gathered.
(6.) Soldiers surrounded the Alamo.
(7.) Tikitu and Samson lifted the table.
Let's start with examples

(1.) Five people lifted the table.
Let's start with examples

(1.) Five people lifted the table.
(1'.) \( \exists^5 x [\text{People}(x) \land \text{Lift}(x)] \).
Let’s start with examples

(1.) Five people lifted the table.

(1’. ) $\exists^=5 x[\text{People}(x) \land \text{Lift}(x)]$.

(1”.) $\exists X [X \subseteq \text{People} \land \text{Card}(X) = 5 \land \text{Lift}(X)]$. 

(2.) Some students played poker together.

(2’. ) $\exists x [\text{Students}(x) \land \text{Play}(x)]$. 

(2”.) $\exists X [X \subseteq \text{Students} \land \text{Play}(X)]$. 

Let’s start with examples

(1.) Five people lifted the table.
(1’.)
\[ \exists^5 x [\text{People}(x) \land \text{Lift}(x)]. \]
(1”). \[ \exists X [X \subseteq \text{People} \land \text{Card}(X) = 5 \land \text{Lift}(X)]. \]

(2.) Some students played poker together.
Let’s start with examples

(1.) Five people lifted the table.
(1’.) \( \exists^{=5} x [\text{People}(x) \land \text{Lift}(x)] \).
(1’’. \( \exists X [X \subseteq \text{People} \land \text{Card}(X) = 5 \land \text{Lift}(X)] \).

(2.) Some students played poker together.
(2’. \( \exists X [X \subseteq \text{Students} \land \text{Play}(X)] \).
Existential modifier

Definition (van der Does 1992)
Fix a universe of discourse $U$ and take any $X \subseteq U$ and $Y \subseteq \mathcal{P}(U)$. Define the existential lift $Q^{EM}$ of a quantifier $Q$ in the following way:

$$Q^{EM}(X, Y) \text{ is true } \iff \exists Z \subseteq X[Q(X, Z) \land Z \in Y].$$
Existential modifier

Definition (van der Does 1992)
Fix a universe of discourse $U$ and take any $X \subseteq U$ and $Y \subseteq \mathcal{P}(U)$. Define the existential lift $Q^{EM}$ of a quantifier $Q$ in the following way:

$$Q^{EM}(X, Y) \text{ is true} \iff \exists Z \subseteq X [Q(X, Z) \land Z \in Y].$$

$$((et)((et)t)) \sim ((et)((et)t))$$
Van Benthem problem

Observation

\((\cdot)^{EM}\) works only for right monotone increasing quantifiers.
Van Benthem problem

Observation

\((\cdot)^E_M\) works only for right monotone increasing quantifiers.

(1.) No students met yesterday at the coffee shop.
Van Benthem problem

Observation
\((\cdot)^{EM}\) works only for right monotone increasing quantifiers.

(1.) No students met yesterday at the coffee shop.
— \(\downarrow\text{MON}\downarrow\sim\uparrow\text{MON}\uparrow\)
Observation
\((\cdot)^{EM}\) works only for right monotone increasing quantifiers.

(1.) No students met yesterday at the coffee shop.
   \[ \downarrow \text{MON} \downarrow \sim \uparrow \text{MON} \uparrow \]

(2.) No left-wing students met yesterday at the coffee shop.
(3.) No students met yesterday at the “Che” coffee shop.
(1.) Exactly 5 students drank a whole keg of beer together.
(1.) Exactly 5 students drank a whole keg of beer together.

(1'.) $\exists^{=5} EM$ [Student, Drink-a-whole-keg-of-beer].
(1.) Exactly 5 students drank a whole keg of beer together.

(1’) \((\exists^5)^{EM} [\text{Student, Drink-a-whole-keg-of-beer}].\)

(1’’.) \(\exists A \subseteq \text{Student}[\text{card}(A) = 5 \land \text{Drink-a-whole-keg-of-beer}(A)]\)
Definition (van der Does 1992)

Let $U$ be a universe, $X \subseteq U$, $Y \subseteq \mathcal{P}(U)$, and $Q$ a type $(1, 1)$ quantifier. We define the neutral modifier:

$$Q^N[X, Y] \text{ is true} \iff Q[X, \bigcup(Y \cap \mathcal{P}(X))].$$
Fact (Ben-Avi and Winter 2003)

Let $Q$ be a distributive determiner. If $Q$ belongs to one of the classes $\uparrow\text{MON}\uparrow$, $\downarrow\text{MON}\downarrow$, $\text{MON}\uparrow$, $\text{MON}\downarrow$, then the collective determiner $Q^N$ belongs to the same class. Moreover, if $Q$ is conservative and $\sim\text{MON}$ (MON$\sim$), then $Q^N$ is also $\sim\text{MON}$ (MON$\sim$).
What about split groups?

(1.) Exactly 5 students drank a whole keg of beer together.
What about split groups?

(1.) Exactly 5 students drank a whole keg of beer together.
(1’. ) \( (\exists = 5)^N \) [Student, Drink-a-whole-keg-of-beer].
What about split groups?

(1.) Exactly 5 students drank a whole keg of beer together.

(1'.) \( (\exists = 5)^N[\text{Student, Drink-a-whole-keg-of-beer}] \).

\[
\text{card}\left( \{ x \mid \exists A \subseteq \text{Student}[x \in A \land \text{Drink-a-whole-keg-of-beer}(A)] \} \right) = 5.
\]
Definition (Winter 2001)
For all $X, Y \subseteq \mathcal{P}(U)$ we have that

$$Q^{\text{dfit}}(X, Y) \text{ is true} \iff Q[\cup X, \cup (X \cap Y)] \land [X \cap Y = \emptyset \lor \exists W \in X \cap Y \land Q(\cup X, W)].$$
Determiner fitting

**Definition (Winter 2001)**

For all \( X, Y \subseteq \mathcal{P}(U) \) we have that

\[
Q^{\text{dfit}}(X, Y) \text{ is true} \iff \\
Q[\bigcup X, \bigcup (X \cap Y)] \land [X \cap Y = \emptyset \lor \exists W \in X \cap Y \land Q(\bigcup X, W)].
\]

\[
(((et)((et)t)) \leadsto (((et)t)(((et)t)t))
\]
Dfit works

1. Exactly 5 students drank a whole keg of beer together.
Dfit works

(1.) Exactly 5 students drank a whole keg of beer together.

(1'.) $\exists^{=5}_{dfit}[\text{Student, Drink-a-whole-keg-of-beer}]$. 
Exactly 5 students drank a whole keg of beer together.

\((\exists^{=5})^{\text{dfit}}[\text{Student, Drink-a-whole-keg-of-beer}]\).

\[
\text{card}\left(\{x \in A | A \subseteq \text{Student} \land \text{Drink-a-whole-keg-of-beer}(A)\}\right) = 5 \\
\land \exists W \subseteq \text{Student}[\text{Drink-a-whole-keg-of-beer}(W) \land \text{card}(W) = 5].
\]
It really works...

<table>
<thead>
<tr>
<th>Monotonicity of Q</th>
<th>Monotonicity of $Q^{dfit}$</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>↑MON↑</td>
<td>↑MON↑</td>
<td>Some</td>
</tr>
<tr>
<td>↓MON↓</td>
<td>↓MON↓</td>
<td>Less than five</td>
</tr>
<tr>
<td>↓MON↑</td>
<td>~MON↑</td>
<td>All</td>
</tr>
<tr>
<td>↑MON↓</td>
<td>~MON↓</td>
<td>Not all</td>
</tr>
<tr>
<td><del>MON</del></td>
<td><del>MON</del></td>
<td>Exactly five</td>
</tr>
<tr>
<td>~MON↓</td>
<td>~MON↓</td>
<td>Not all and less than five</td>
</tr>
<tr>
<td>~MON↑</td>
<td>~MON↑</td>
<td>Most</td>
</tr>
<tr>
<td>↓MON~</td>
<td><del>MON</del></td>
<td>All or less than five</td>
</tr>
<tr>
<td>↑MON~</td>
<td><del>MON</del></td>
<td>Some but not all</td>
</tr>
</tbody>
</table>

Table: Monotonicity under the determiner fitting operator; cf. (Ben-Avi and Winter 2003).
...but violates invariance properties

Definition
A distributive determiner of type (1, 1) is conservative if and only if the following holds for all $M$ and all $A, B \subseteq M$:

$$Q_M[A, B] \iff Q_M[A, A \cap B].$$
...but violates invariance properties

**Definition**
A distributive determiner of type $(1, 1)$ is conservative if and only if the following holds for all $M$ and all $A, B \subseteq M$:

\[ Q_M[A, B] \iff Q_M[A, A \cap B]. \]

**Fact**
*For every $Q$ the quantifiers $Q^{EM}$, $Q^N$, and $Q^{dfit}$ are not CONS.*
...and not only because of technicalities

Definition
We say that a collective determiner $Q$ of type $((et)(((et)t)t))$ satisfies *collective conservativity* iff the following holds for all $M$ and all $A, B \subseteq M$:

$$Q_M[A, B] \iff Q_M[A, \mathcal{P}(A) \cap B].$$
Definition
We say that a collective determiner $Q$ of type $(((et)((et)t)t))$ satisfies collective conservativity iff the following holds for all $M$ and all $A, B \subseteq M$:

$$Q_M[A, B] \iff Q_M[A, \mathcal{P}(A) \cap B].$$

Fact
For every $Q$ the collective quantifiers $Q^{EM}$, $Q^N$, and $Q^{dfit}$ satisfy collective conservativity.
Invariance properties are forced

- Conservativity incorporated into the lifts.
- We need less arbitrary approach.
Second-order GQs

\[ \exists^2 = \{ (M, P) \mid P \subseteq \mathcal{P}(M) & P \neq \emptyset \} \].

\[ \text{EVEN} = \{ (M, P) \mid P \subseteq \mathcal{P}(M) & \text{card}(P) \text{ is even} \} \].

\[ \text{EVEN}' = \{ (M, P) \mid P \subseteq \mathcal{P}(M) & \forall X \in P(\text{card}(X) \text{ is even}) \} \].

\[ \text{MOST} = \{ (M, P, S) \mid P, S \subseteq \mathcal{P}(M) & \text{card}(P \cap S) > \text{card}(P - S) \} \].
Second-order GQs

\[ \exists^2 = \{(M, P) \mid P \subseteq \mathcal{P}(M) \& P \neq \emptyset \} \]

\[ \text{EVEN} = \{(M, P) \mid P \subseteq \mathcal{P}(M) \& \text{card}(P) \text{ is even}\} \]

\[ \text{EVEN}' = \{(M, P) \mid P \subseteq \mathcal{P}(M) \& \forall X \in P(\text{card}(X) \text{ is even})\} \]

\[ \text{MOST} = \{(M, P, S) \mid P, S \subseteq \mathcal{P}(M) \& \text{card}(P \cap S) > \text{card}(P - S)\} \]

Observation

*SOGQs do not decide invariance properties!*
Second-order GQs

\[ \exists^2 = \{(M, P) \mid P \subseteq \mathcal{P}(M) \land P \neq \emptyset\} \]

\[ \text{EVEN} = \{(M, P) \mid P \subseteq \mathcal{P}(M) \land \text{card}(P) \text{ is even}\} \]

\[ \text{EVEN'} = \{(M, P) \mid P \subseteq \mathcal{P}(M) \land \forall X \in P(\text{card}(X) \text{ is even})\} \]

\[ \text{MOST} = \{(M, P, S) \mid P, S \subseteq \mathcal{P}(M) \land \text{card}(P \cap S) > \text{card}(P - S)\} \]

Observation

SOGQs do not decide invariance properties!

Question

How invariance properties interact with definability?
Warning!

Do not confuse:

▶ FO GQs (Lindström) with FO-definable quantifiers. E.g. most is FO GQs but is not FO-definable.

▶ SO GQs with SO-definable quantifiers. E.g. most is SO GQs but probably not SO-definable.
Warning!

Do not confuse:
- FO GQs (Lindström) with FO-definable quantifiers
  E.g. most is FO GQs but is not FO-definable.
Warning!

Do not confuse:

- **FO GQs (Lindström)** with FO-definable quantifiers
  E.g. most is FO GQs but is not FO-definable.
- **SO GQs** with SO-definable quantifiers
  E.g. MSOT is SO GQs but probably not SO-definable.
GQs are not enough

Theorem (Kontinen 2002)

*The extension $\mathcal{L}^*$ of first-order logic by all Lindström quantifiers cannot define the monadic second-order existential quantifier.*
GQs are not enough

**Theorem (Kontinen 2002)**

The extension $\mathcal{L}^*$ of first-order logic by all Lindström quantifiers cannot define the monadic second-order existential quantifier.

**Corollary**

Lindström quantifiers alone are not adequate for formalizing all natural language quantification.
GQs are not enough

Theorem (Kontinen 2002)

The extension $\mathcal{L}^*$ of first-order logic by all Lindström quantifiers cannot define the monadic second-order existential quantifier.

Corollary

Lindström quantifiers alone are not adequate for formalizing all natural language quantification.

Example

Some students gathered to play poker.
For example . . .

Definition
Denote by some $^{EM}$:

$$\{(M, P, G) \mid P \subseteq M; \ G \subseteq \mathcal{P}(M) : \exists Y \subseteq P (Y \neq \emptyset \ \& \ P \in G)\}.$$
For example . . .

**Definition**
Denote by some $EM^E$:

$$\{(M, P, G) \mid P \subseteq M; \ G \subseteq \mathcal{P}(M) : \exists Y \subseteq P(Y \neq \emptyset \& P \in G)\}.$$  

(3.) Some students played poker together.
For example . . .

Definition
Denote by some $^{EM}$:

$$\{(M, P, G) \mid P \subseteq M; G \subseteq \mathcal{P}(M) : \exists Y \subseteq P (Y \neq \emptyset \& P \in G)\}.$$  

(3.) Some students played poker together.
(3'). some $^{EM} x, X[\text{Student}(x), \text{Play}(X)]$. 
Another example . . .

**Definition**

We take five$^E_M$ to be the second-order quantifier denoting:

\[ \{(M, P, G) \mid P \subseteq M; \ G \subseteq \mathcal{P}(M) : \exists Y \subseteq P(\text{card}(Y) = 5 \ & \ P \in G)\} . \]
Another example . . .

**Definition**

We take \( \text{five}^{EM} \) to be the second-order quantifier denoting:

\[
\{(M, P, G) \mid P \subseteq M; \ G \subseteq \mathcal{P}(M) : \ \exists Y \subseteq P (\text{card}(Y) = 5 \ \& \ P \in G)\}.
\]

(4.) Five people lifted the table.
Definition
We take \(\text{five}^{EM}\) to be the second-order quantifier denoting:

\[
\{(M, P, G) \mid P \subseteq M; \ G \subseteq \mathcal{P}(M) : \exists Y \subseteq P(\text{card}(Y) = 5 \& P \in G)\}.
\]

(4.) Five people lifted the table.
(4′.) \(\text{five}^{EM}x, X[\text{Student}(x), \text{Lift}(X)]\).
Theorem

Let $Q$ be a Lindström quantifier definable in SO. Then the collective quantifiers $Q^{EM}$, $Q^N$, and $Q^{dfit}$ are definable in SO.
SO-definable GQs are closed on lifts

**Theorem**

Let $Q$ be a Lindström quantifier definable in $SO$. Then the collective quantifiers $Q_{EM}^{E}$, $Q^{N}$, and $Q^{dfit}$ are definable in $SO$.

**Proof.**

Let us consider the case of $Q_{EM}^{E}$. Let $\psi(x)$ and $\phi(Y)$ be formulas. We want to express $Q_{EM}^{E} x, Y(\psi(x), \phi(Y))$ in second-order logic. By the assumption, the quantifier $Q$ can be defined by some sentence $\theta \in SO[\{P_1, P_2\}]$. We can now use the following formula:

$$\exists Z(\forall x(Z(x) \rightarrow \psi(x)) \land (\theta(P_1/\psi(x), P_2/Z) \land \phi(Y/Z))).$$
And this is the case for all SO-definable lifts

**Theorem**

Let us assume that the lift $(\cdot)^*$ and a Lindström quantifier $Q$ are both definable in second-order logic. Then the collective quantifier $Q^*$ is also definable in second-order logic.
Some collectives are not definable in SO

Theorem (Kontinen and Szymanik 2012)

The quantifier MOST is not definable in second-order logic.

Proof.
By translating into FO(+, x) over cardinalities $2^n$ and using Ajtai’s 1983 results.
Consequences

Corollary

*The type-shifting strategy is not general enough to cover all collective quantification in natural language.*
What is the right ontology for semantics?

- $L^*$ and SO doesn’t capture natural language?
What is the right ontology for semantics?

- $\mathcal{L}^*$ and SO doesn’t capture natural language?
- Are many-sorted (algebraic) models more plausible?
  - Type-shifting is too complex;
  - In principle this question is psychologically testable.
Hypothesis

Our everyday language is semantically bounded by the properties expressible in the existential fragment of second-order logic.
Does SOGQ “MOST” belong to everyday language?
  - Everyday language doesn’t realize prop. coll. qua.
  - No need to extend the higher-order approach to prop. qua.
Does SOGQ “MOST” belong to everyday language?
  ▶ Everyday language doesn’t realize prop. coll. qua.
  ▶ No need to extend the higher-order approach to prop. qua.

**Question**

*Have we just encountered an example where complexity restricts the expressibility of everyday language as suggested by $\Sigma^1_1$-thesis?*