Quantifiers & Cognition
Lecture 3:
Quantifiers and Counting

Jakub Szymanik
Outline

Languages and automata

Quantifiers are classes of models

Quantifier automata

Complexity and reaction time

Complexity and working memory

Outlook
Outline

Languages and automata

Quantifiers are classes of models

Quantifier automata

Complexity and reaction time

Complexity and working memory

Outlook
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., $\{0, 1\}^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, 000, \ldots\}$. 
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., $\{0, 1\}^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, 000, \ldots\}$.
- 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 $L_1 = A^*$. 

\[ q_1 \hspace{1cm} a, b \]
Example 2

Let $L_2 = \emptyset$
Example 3

$L_3 = \{ \varepsilon \}$
Not every language is regular

\[ L_{ab} = \{ a^n b^n : n \geq 1 \} \]
Push down automata

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;
- $\delta : Q \times (A \cup \{\epsilon\}) \times \Gamma \rightarrow \mathcal{P}(Q \times \Gamma^*)$ is a transition function.
push/pop-off a symbol from the top of the stack
Context-free languages

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.
Outline

Languages and automata

Quantifiers are classes of models

Quantifier automata

Complexity and reaction time

Complexity and working memory

Outlook
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, i.e. if $M$ and $M'$ are isomorphic, then

$$(M \in Q \iff M' \in Q).$$
Lindström definition

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, i.e. if $M$ and $M'$ are isomorphic, then

$$(M \in Q \iff M' \in Q).$$

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.
Simple quantifier sentences

- Every poet has low self-esteem.
- Some dean danced nude on the table.
- At least 3 grad students prepared presentations.
- An even number of the students saw a ghost.
- Most of the students think they are smart.
- Less than half of the students received good marks.
Monadic quantifiers of type (1, 1)

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

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

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

\[ \text{some} = \{ (M, A, B) \mid A, B \subseteq M \text{ and } A \cap B \neq \emptyset \}. \]
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)\}.
2 concepts of GQ are equivalent

\[(M, A_1, A_2) \in Q \iff Q_M(A_1, A_2), \text{ where } A_i \subseteq M, \ i = 1, 2.\]

Example

\[(M, A_1, A_2) \in \text{most} \iff \text{most}_M(A_1, A_2).\]
2 concepts of GQ are equivalent

\[(M, A_1, A_2) \in Q \iff Q_M(A_1, A_2), \text{ where } A_i \subseteq M, \ i = 1, 2.\]

Example

\[(M, A_1, A_2) \in \text{most} \iff \text{most}_M(A_1, A_2).\]

Corollary

*The two definitions of generalized quantifiers are equivalent.*
Outline

Languages and automata

Quantifiers are classes of models

Quantifier automata

Complexity and reaction time

Complexity and working memory

Outlook
How do we encode models

This model is uniquely described by $\alpha_M = a_{\bar{A}\bar{B}}a_{\bar{A}B}a_{AB}a_{\bar{A}B}a_{\bar{A}B}$
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: $\bar{a}A\bar{B}, a\bar{A}B, a\bar{A}B, a\bar{A}B, a\bar{A}B$, according to constituents it belongs to.

- Result: the word $\alpha_M = \bar{a}A\bar{B}a\bar{A}Ba\bar{A}Ba\bar{A}B$. 

- $\alpha_M$ 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.
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: $ar{A}, A, \bar{B}, AB, \bar{AB}$, according to constituents it belongs to.

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

- 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: $c_1, \ldots, c_5$.
- Labeling every element with one of the letters: $a_{\bar{A}\bar{B}}$, $a_{A\bar{B}}$, $a_{\bar{A}B}$, $a_{AB}$, 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: $a_{\bar{A}\bar{B}}$, $a_{A\bar{B}}$, $a_{\bar{A}B}$, $a_{AB}$, according to constituents it belongs to.
- Result: the word $\alpha_M = a_{\bar{A}\bar{B}}a_{A\bar{B}}a_{AB}a_{\bar{A}B}a_{\bar{A}B}$.
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: $a_{\overline{AB}}, a_{AB}, a_{\overline{AB}}, a_{\overline{AB}}$, according to constituents it belongs to.
- Result: the word $\alpha_M = a_{\overline{AB}} a_{AB} a_{AB} a_{\overline{AB}} a_{\overline{AB}}$.
- $\alpha_M$ describes the model in which:
  $c_1 \in \overline{AB}$, $c_2 \in AB$, $c_3 \in AB$, $c_4 \in \overline{AB}$, $c_5 \in \overline{AB}$. 
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: $\bar{a}_{A\bar{B}}, \bar{a}_{AB}, \bar{a}_{AB}, a_{AB}$, according to constituents it belongs to.
- Result: the word $\alpha_M = \bar{a}_{A\bar{B}} a_{AB} \bar{a}_{AB} a_{AB} \bar{a}_{AB} a_{AB}$.
- $\alpha_M$ describes the model in which: $c_1 \in \bar{A\bar{B}}, c_2 \in A\bar{B} c_3 \in AB, c_4 \in \bar{A}\bar{B}, c_5 \in \bar{A}\bar{B}$.
- The class Q is represented by the set of words describing all elements of the class.
Definition
The class $K_Q$ of finite models of the form $(M, A_1, \ldots, A_n)$ can be represented by the set of nonempty words $L_Q$ over the alphabet $A = \{a_1, \ldots, a_{2^n}\}$ such that: $\alpha \in L_Q$ if and only if there are $(U, A_1, \ldots, A_n) \in K_Q$ and linear ordering $U = \{c_1, \ldots, c_k\}$, such that $\text{length}(\alpha) = k$ and $i$-th character of $\alpha$ is $a_j$ exactly when $c_i \in S_1 \cap \ldots \cap S_n$, where:

$$S_i = \begin{cases} A_i & \text{if integer part of } \frac{j}{2^i} \text{ is odd} \\ U - A_i & \text{otherwise.} \end{cases}$$
Aristotelian quantifiers

“all”, “some”, “no”, and “not all”

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

\[[q_0] \xrightarrow{a_{\bar{A}B}} [q_1] \]

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”

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

Finite automaton recognizing \( \mathcal{L}_{\text{More than two}} \)

\[
\mathcal{L}_{\text{More than two}} = \{ \alpha \in \Gamma^* : \#a_{AB}(\alpha) > 2 \} 
\]
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 $\text{card}(A \cap B) > \text{card}(A - B)$.
- $L_{\text{Most}} = \{ \alpha \in \Gamma^* : \# a_{AB}(\alpha) > \# a_{AB}(\alpha) \}$.
- There is no finite automaton recognizing this language.
- We need internal memory.
- A push-down automata will do.
Correspondence

Question

*What does it mean that a class of monadic quantifiers is recognized by a class of devices?*
Question
What does it mean that a class of monadic quantifiers is recognized by a class of devices?

Definition
Let $D$ be a class of recognizing devices, $\Omega$ a class of monadic quantifiers. We say that $D$ accepts $\Omega$ if and only if for every monadic quantifier $Q$:

$$Q \in \Omega \iff \text{there is device } A \in D (A \text{ accepts } L_Q).$$
Relevant results: acyclic FA and FA

Theorem (J. van Benthem)

Monadic quantifier $Q$ is first–order definable iff $L_Q$ is accepted by an acyclic finite automaton.
Relevant results: acyclic FA and FA

Theorem (J. van Benthem)

*Monadic quantifier $Q$ is first-order definable iff $L_Q$ is accepted by an *acyclic* finite automaton.*

Theorem (M. Mostowski)

*Monadic quantifier $Q$ is definable in the divisibility logic iff $L_Q$ is accepted by a finite automaton.*
Relevant results: acyclic FA and FA

Theorem (J. van Benthem)

Monadic quantifier $Q$ is first–order definable iff $L_Q$ is accepted by an acyclic finite automaton.

Theorem (M. Mostowski)

Monadic quantifier $Q$ is definable in the divisibility logic iff $L_Q$ is accepted by a finite automaton.

FA do not use any kind of working memory device.
Odds of “even”

- “Even” and “odd” are non-FO.
- They can be however recognized by FA.
- But opposite to FO quantifiers you need FA with cycle.
- Difference between FA and acyclic FA.
Relevant results: quantifiers and PDAs

Theorem (J. van Benthem)
Quantifier $Q$ of type (1) is semilinear iff $L_Q$ is accepted by push–down automaton.
Relevant results: quantifiers and PDAs

Theorem (J. van Benthem)

Quantifier $Q$ of type (1) is semilinear iff $L_Q$ is accepted by push–down automaton.

PDA use stack which is simple working memory device.
## 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?*
Question

Do minimal automata predict differences in verification?

We’ll try to convince you that the answer is positive!
Outline

Languages and automata

Quantifiers are classes of models

Quantifier automata

Complexity and reaction time

Complexity and working memory

Outlook
Predictions

- RT will increase along with the computational resources.
Predictions

- RT will increase along with the computational resources.
- Aristotelian qua. < parity qua. < proportional qua.
Predictions

- RT will increase along with the computational resources.
- Aristotelian qua. < parity qua. < proportional qua.
- Aristotelian qua. < cardinal qua. of high rank.
Participants

- 40 native Polish-speaking adults (21 female).
- Volunteers: undergraduates from the University of Warsaw.
- The mean age: 21.42 years (SD = 3.22).
- Each participant tested individually.
80 grammatically simple propositions in Polish, like:

1. Some cars are red.
2. More than 7 cars are blue.
3. An even number of cars is yellow.
4. Less than half of the cars are black.
Materials continued

More than half of the cars are yellow.

An example of a stimulus used in the first study
Procedure

- 8 different quantifiers divided into four groups.

- "all" and "some" (acyclic 2-state FA);
- "odd" and "even" (2-state FA);
- "less than 8" and "more than 7" (FA);
- "less than half" and "more than half" (PDA).

Each quantifier was presented in 10 trials.
The sentence true in the picture in half of the trials.
Quantity of target items near the criterion of validation.
Practice session followed by the experimental session.
Each quantifier problem was given one 15.5 s event.
Subjects were asked to decide the truth-value.
Procedure

- 8 different quantifiers divided into four groups.
  - “all” and “some” (acyclic 2-state FA);
  - “odd” and “even” (2-state FA);
  - “less than 8” and “more than 7” (FA);
  - “less than half” and “more than half” (PDA).

Each quantifier was presented in 10 trials. The sentence true in the picture in half of the trials. Quantity of target items near the criterion of validation. Practice session followed by the experimental session. Each quantifier problem was given one 15.5 s event. Subjects were asked to decide the truth-value.
Procedure

- 8 different quantifiers divided into four groups.
  - “all” and “some” (acyclic 2-state FA);
  - “odd” and “even” (2-state FA);
Procedure

- 8 different quantifiers divided into four groups.
  - “all” and “some” (acyclic 2-state FA);
  - “odd” and “even” (2-state FA);
  - “less than 8” and “more than 7” (FA);
Procedure

- 8 different quantifiers divided into four groups.
  - “all” and “some” (acyclic 2-state FA);
  - “odd” and “even” (2-state FA);
  - “less than 8” and “more than 7” (FA);
  - “less than half” and “more than half” (PDA).

Each quantifier was presented in 10 trials.
The sentence true in the picture in half of the trials.
Quantity of target items near the criterion of validation.
Practice session followed by the experimental session.
Each quantifier problem was given one 15.5 s event.
Subjects were asked to decide the truth-value.
Procedure

- 8 different quantifiers divided into four groups.
  - “all” and “some” (acyclic 2-state FA);
  - “odd” and “even” (2-state FA);
  - “less than 8” and “more than 7” (FA);
  - “less than half” and “more than half” (PDA).
- Each quantifier was presented in 10 trials.
Procedure

- 8 different quantifiers divided into four groups.
  - “all” and “some” (acyclic 2-state FA);
  - “odd” and “even” (2-state FA);
  - “less than 8” and “more than 7” (FA);
  - “less than half” and “more than half” (PDA).

- Each quantifier was presented in 10 trials.
- The sentence true in the picture in half of the trials.
Procedure

- 8 different quantifiers divided into four groups.
  - “all” and “some” (acyclic 2-state FA);
  - “odd” and “even” (2-state FA);
  - “less than 8” and “more than 7” (FA);
  - “less than half” and “more than half” (PDA).
- Each quantifier was presented in 10 trials.
- The sentence true in the picture in half of the trials.
- Quantity of target items near the criterion of validation.
Procedure

- 8 different quantifiers divided into four groups.
  - “all” and “some” (acyclic 2-state FA);
  - “odd” and “even” (2-state FA);
  - “less than 8” and “more than 7” (FA);
  - “less than half” and “more than half” (PDA).

- Each quantifier was presented in 10 trials.
- The sentence true in the picture in half of the trials.
- Quantity of target items near the criterion of validation.
- Practice session followed by the experimental session.
Procedure

- 8 different quantifiers divided into four groups.
  - “all” and “some” (acyclic 2-state FA);
  - “odd” and “even” (2-state FA);
  - “less than 8” and “more than 7” (FA);
  - “less than half” and “more than half” (PDA).

- Each quantifier was presented in 10 trials.
- The sentence true in the picture in half of the trials.
- Quantity of target items near the criterion of validation.
- Practice session followed by the experimental session.
- Each quantifier problem was given one 15.5 s event.
Procedure

- 8 different quantifiers divided into four groups.
  - “all” and “some” (acyclic 2-state FA);
  - “odd” and “even” (2-state FA);
  - “less than 8” and “more than 7” (FA);
  - “less than half” and “more than half” (PDA).
- Each quantifier was presented in 10 trials.
- The sentence true in the picture in half of the trials.
- Quantity of target items near the criterion of validation.
- Practice session followed by the experimental session.
- Each quantifier problem was given one 15.5 s event.
- Subjects were asked to decide the truth-value.
## Analysis of accuracy

<table>
<thead>
<tr>
<th>Quantifier group</th>
<th>Examples</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aristotelian FO</td>
<td>all, some</td>
<td>99</td>
</tr>
<tr>
<td>Parity</td>
<td>odd, even</td>
<td>91</td>
</tr>
<tr>
<td>Cardinal FO</td>
<td>less than 8, more than 7</td>
<td>92</td>
</tr>
<tr>
<td>Proportional</td>
<td>less than half, more than half</td>
<td>85</td>
</tr>
</tbody>
</table>

The percentage of correct answers
Analysis of RT

- All differences significant;
- Aristotelian,
- parity,
- cardinal,
- proportional.

Analysis of RT

RT determined by quantifier type:
Analysis of RT

RT determined by quantifier type:
- All differences significant;

![Graph showing RT vs. quantifier type]
Analysis of RT

RT determined by quantifier type:
- All differences significant;
  - Aristotelian,
  - parity,
  - cardinal,
  - proportional.

Outline

Languages and automata

Quantifiers are classes of models

Quantifier automata

Complexity and reaction time

Complexity and working memory

Outlook
McMillan et al. fMRI studies

Differences in brain activity.
McMillan et al. fMRI 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. fMRI 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.

But definability seems not to be fine grained enough!

McMillan et al., Neural basis for generalized quantifiers comprehension, 2005
Szymanik, A Note on some neuroimaging study of natural language quantifiers comprehension, Neuropsychologia, 2007
Baddeley’s model

WM unified system responsible for the performance in complex tasks.
Baddeley’s model

WM unified system responsible for the performance in complex tasks.

- The model consists of:
  - temporary storage units:
    - phonological loop;
    - visual loop;
  - a controlling system (central executive).

Baddeley, Working memory and language: an overview, 2003
Span test

To assess the working memory construct.

Subjects read sentences.

They are asked to:

- Remember the final words.
- Comprehend the story.

What is:

- The number of correctly memorized words?
- The degree of understanding?
- Engagement of processing and storage functions.

Daneman and Carpenter, Individual differences in working memory, 1980
Span test

- To assess the working memory construct.
Span test

- To assess the working memory construct.
- Subjects read sentences.

Daneman and Carpenter, Individual differences in working memory, 1980
Span test

- To assess the working memory construct.
- Subjects read sentences.
- They are asked to:
  - remember the final words.
  - comprehend the story.
Span test

- To assess the working memory construct.
- Subjects read sentences.
- They are asked to:
  - remember the final words.
  - comprehend the story.
- What is:
  - the number of correctly memorized words?
  - the degree of understanding?

Daneman and Carpenter, Individual differences in working memory, 1980
Span test

- To assess the working memory construct.
- Subjects read sentences.
- They are asked to:
  - remember the final words.
  - comprehend the story.
- What is:
  - the number of correctly memorized words?
  - the degree of understanding?
- Engagement of processing and storage functions.

Daneman and Carpenter, Individual differences in working memory, 1980
‘Computational’ theory of WM

Observation

*A trade-off between processing and storage functions.*
‘Computational’ theory of WM

Observation
A trade-off between processing and storage functions.

Hypothesis
One cognitive resource – competition for a limited capacity.

Daneman and Merikle, Working memory and language comprehension, 1996
Experimental setup

Question

How additional memory load influences quantifier verification?
Experimental setup

Question

*How additional memory load influences quantifier verification?*

Combined task:

- memorize sequences of digits;
- verify quantifier sentences;
- recall digits.
Predictions

Difficulty (RT and accuracy) should decrease as follows:
- proportional quantifiers,
- numerical quantifiers of high rank,
- parity quantifiers,
- numerical quantifiers of low rank.

Additionally:
- processing of the PQs should influence storage functions;
- the effect should be stronger in more demanding situation.
Predictions

Difficulty (RT and accuracy) should decrease as follows:

- proportional quantifiers,
- numerical quantifiers of high rank,
- parity quantifiers,
- numerical quantifiers of low rank.

Additionally:

- processing of the PQs should influence storage functions;
- the effect should be stronger in more demanding situation.
Sentence verification

64 grammatically simple propositions in Polish, like:

1. More than 7 cars are blue.
2. An even number of cars is yellow.
3. Less than half of the cars are black.
Sentence verification

64 grammatically simple propositions in Polish, like:

1. More than 7 cars are blue.
2. An even number of cars is yellow.
3. Less than half of the cars are black.

► 8 different quantifiers divided into four groups.
Sentence verification

64 grammatically simple propositions in Polish, like:

1. More than 7 cars are blue.
2. An even number of cars is yellow.
3. Less than half of the cars are black.

- 8 different quantifiers divided into four groups.
  1. numerical quantifiers of relatively low rank, NQ4/5;
Sentence verification

64 grammatically simple propositions in Polish, like:

1. More than 7 cars are blue.
2. An even number of cars is yellow.
3. Less than half of the cars are black.

▶ 8 different quantifiers divided into four groups.
   1. numerical quantifiers of relatively low rank, NQ4/5;
   2. numerical quantifiers of relatively high rank, NQ7/8;
Sentence verification

64 grammatically simple propositions in Polish, like:

1. More than 7 cars are blue.
2. An even number of cars is yellow.
3. Less than half of the cars are black.

► 8 different quantifiers divided into four groups.
   1. numerical quantifiers of relatively low rank, NQ4/5;
   2. numerical quantifiers of relatively high rank, NQ7/8;
   3. parity quantifiers, DQ;
64 grammatically simple propositions in Polish, like:

1. More than 7 cars are blue.
2. An even number of cars is yellow.
3. Less than half of the cars are black.

- 8 different quantifiers divided into four groups.
  1. numerical quantifiers of relatively low rank, NQ4/5;
  2. numerical quantifiers of relatively high rank, NQ7/8;
  3. parity quantifiers, DQ;
  4. proportional quantifiers, PQ.
Memory task

- At the beginning of each trial a sequence of digits.

- 4 digits
- 6 digits

After verification task: recall the string.
Memory task

- At the beginning of each trial a sequence of digits.
- 2 experimental conditions:
  - 4 digits
  - 6 digits
Memory task

- At the beginning of each trial a sequence of digits.
- 2 experimental conditions:
  - 4 digits
  - 6 digits
- After verification task: recall the string.
RT in verification task

- PQ solved longer than others;
- NQ 4/5 processed shorter than the rest;
- No difference between DQ and NQ 7/8.

6-digit condition:
- NQ 4/5 had the shortest average RT.

Only PQ differed between memory load conditions.
RT in verification task

RT determined by quantifier type in 4-digit:

- PQ solved longer than others;
- NQ 4/5 processed shorter than the rest;
- No difference between DQ and NQ 7/8.

6-digit condition:
- NQ 4/5 had the shortest average RT.
- Only PQ differed between memory load conditions.
RT in verification task

RT determined by quantifier type in 4-digit:

- PQ solved longer than others;

- NQ 4/5 processed shorter than the rest;

- No difference between DQ and NQ 7/8.

6-digit condition:

- NQ 4/5 had the shortest average RT.

- Only PQ differed between memory load conditions.
RT determined by quantifier type in 4-digit:

- PQ solved longer than others;
- NQ 4/5 processed shorter than the rest;
RT in verification task

RT determined by quantifier type in 4-digit:
▶ PQ solved longer than others;
▶ NQ 4/5 processed shorter than the rest;
▶ No difference between DQ and NQ 7/8.

Only PQ differed between memory load conditions.
RT in verification task

RT determined by quantifier type in 4-digit:
- PQ solved longer than others;
- NQ 4/5 processed shorter than the rest;
- No difference between DQ and NQ 7/8.

6-digit condition:

<table>
<thead>
<tr>
<th>RT in msec</th>
<th>Digits</th>
</tr>
</thead>
<tbody>
<tr>
<td>PQ</td>
<td>4</td>
</tr>
<tr>
<td>NQ 7/8</td>
<td>4</td>
</tr>
<tr>
<td>DQ</td>
<td>6</td>
</tr>
<tr>
<td>NQ 4/5</td>
<td>6</td>
</tr>
</tbody>
</table>

RT determined by quantifier type in 4-digit:
- PQ solved longer than others;
- NQ 4/5 processed shorter than the rest;
- No difference between DQ and NQ 7/8.
RT in verification task

RT determined by quantifier type in 4-digit:
▷ PQ solved longer than others;
▷ NQ 4/5 processed shorter than the rest;
▷ No difference between DQ and NQ 7/8.

6-digit condition:
▷ NQ 4/5 had the shortest average RT.
RT determined by quantifier type in 4-digit:
- PQ solved longer than others;
- NQ 4/5 processed shorter than the rest;
- No difference between DQ and NQ 7/8.

6-digit condition:
- NQ 4/5 had the shortest average RT.

Only PQ differed between memory load conditions.
Accuracy in verification task

All quantifiers differed significantly, besides DQ and NQ 7/8.

Large effect for PQ!

Subjects performed worse in 4-digit condition.
Accuracy in verification task

- All quantifiers differed significantly,
- besides DQ and NQ 7/8.
All quantifiers differed significantly, besides DQ and NQ 7/8. Large effect for PQ!
Accuracy in verification task

- All quantifiers differed significantly,
- besides DQ and NQ 7/8.
- Large effect for PQ!

Subjects performed worse in 4-digit condition.
Memory task: recall accuracy

![Graph showing accuracy of digits recalled versus quantifier for 4 and 6 digit recall tasks.]

- In 4-digit recall, the worst performance is observed with PQ.
- In 6-digit recall, there are no significant differences.
Memory task: recall accuracy

- In 4-digit with PQ: the worst;

![Graph showing accuracy of digits recalled vs quantifier.
Accuracy is highest for NQ 4/5 and lowest for PQ.
Numbers of digits recalled range from 3.5 to 6.5.]
Memory task: recall accuracy

- In 4-digit with PQ: the worst;
- In 6-digit: no differences.
Summary

- In 4-digit automata were good predictors of difficulty.
Summary

- In 4-digit automata were good predictors of difficulty.
- Discrepancy under two memory load conditions:
  - The real differences occurred only in 4-digit condition.
  - Holding six elements in memory was probably too difficult.
  - Trade-off between processing and storage.
In 4-digit automata were good predictors of difficulty.

Discrepancy under two memory load conditions:
- The real differences occurred only in 4-digit condition.
- Holding six elements in memory was probably too difficult.
- Trade-off between processing and storage.

Number of states is a good predictor of the cognitive load.

Szymanik & Zajenkowski, *Quantifiers and working memory*, LNCS, 2010
Proportional quantifiers

- 4-digit strings accompanying this class were recalled worst.
- But no differences in 6-digit condition:
  - RT decreased: subjects ignored recalling.
- WM engagement PQ processing is qualitatively different.
Further evidence

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

![Bar chart showing reaction time in milliseconds for different quantifiers: Aristotelian, Numerical, Parity, and Proportional. The chart compares patients and control groups.](image-url)
RT data
Conclusion

*Automata model is psychologically plausible.*
Summary

Conclusion
Automata model is psychologically plausible.

Conclusion
Computational complexity \(\approx\) cognitive difficulty.
Conclusion

*Automata model is psychologically plausible.*

Conclusion

*Computational complexity $\approx$ cognitive difficulty.*

- As far as we know this is the first empirical proof.
- Between Marr’s level 1 and 2.
Outline

Languages and automata

Quantifiers are classes of models

Quantifier automata

Complexity and reaction time

Complexity and working memory

Outlook
Bigger picture

► Enrich the model:
  1. Approximate Number System;
  2. Visual clues;
Neurocognitive computational modeling

- Mechanism selection;
- Translate to neurocognitive setting;
- fMRI experiments.
Modeling example

(a) Visual display

Verification of ‘most dots are black’

Approximate number system (ANS)
Approx. cardinalities ($b$ and $w$)
Cardinality comparison (Is $b > w$?)

true

(b) Visual display

Verification of ‘most dots are black’

Sample small subsets of ($\leq 4$) dots
Subitize
Count ‘wins’
Cardinality comparison

true

(c) Visual display

Verification of ‘most dots are black’

Pair each white dot with a black dot
Non-paired blacks dots
Test of 1:1 correspondence (black dots left?)

true