Unsupervised Language Learning

Tran Cong Nguyen, Lautaro Quiroz,
Roger Wechsler

Outline

- Overview and goals
- Implementation
- Results

Overview / Goals

Goal: Find the best Tree Substitution Grammar (TSG) for the “numerical” data

Procedure:

- Define a simple **CFG** that generates numbers
- Parse the corpus, randomly extract elementary trees and create an initial **TSG**
- Make small changes to the data and check if data likelihood improves in order to find **best TSG**

Overview / Goals

Extract numeral data from the Penn WSJ Treebank.

Procedure:

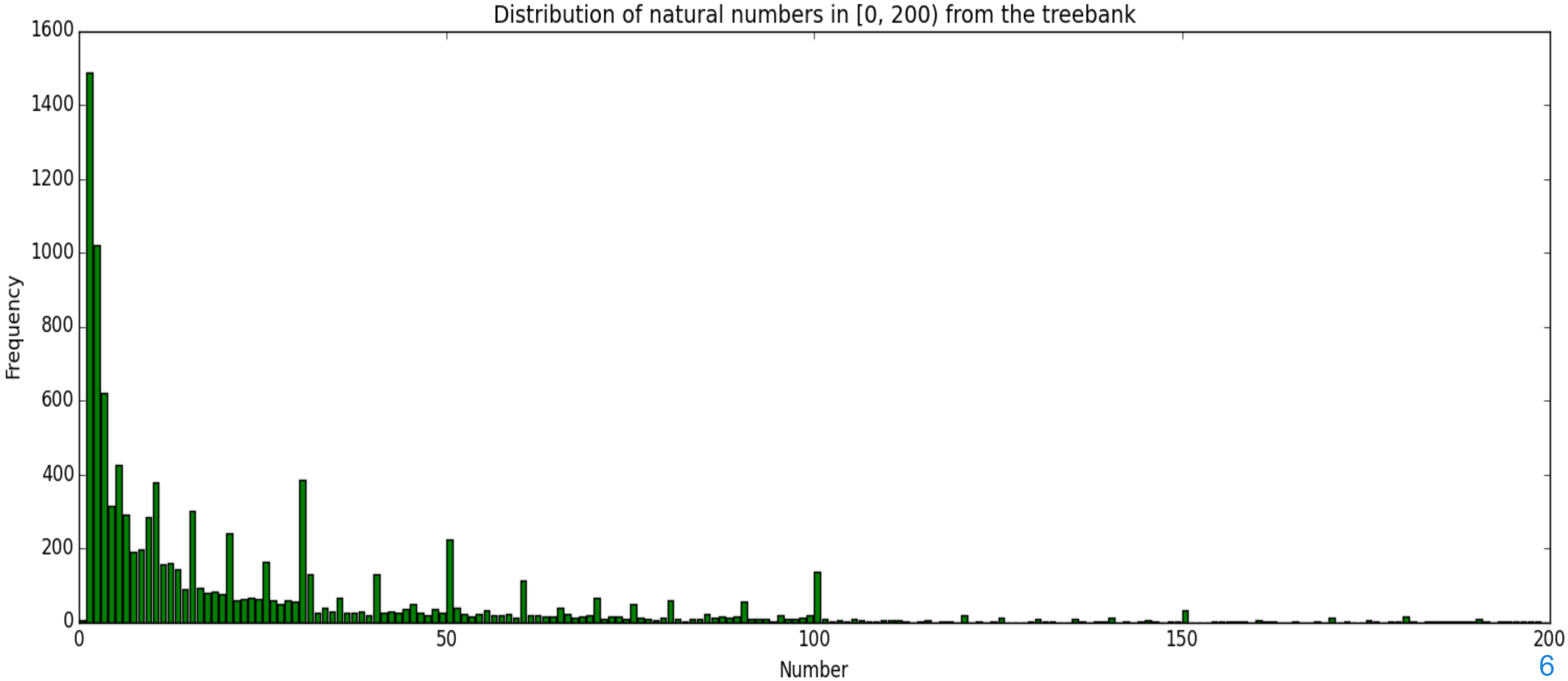
- Look for tags “CD”.
- Extract the values inside this tag.
- If in alphabetic form, translate to number form.

Overview / Goals

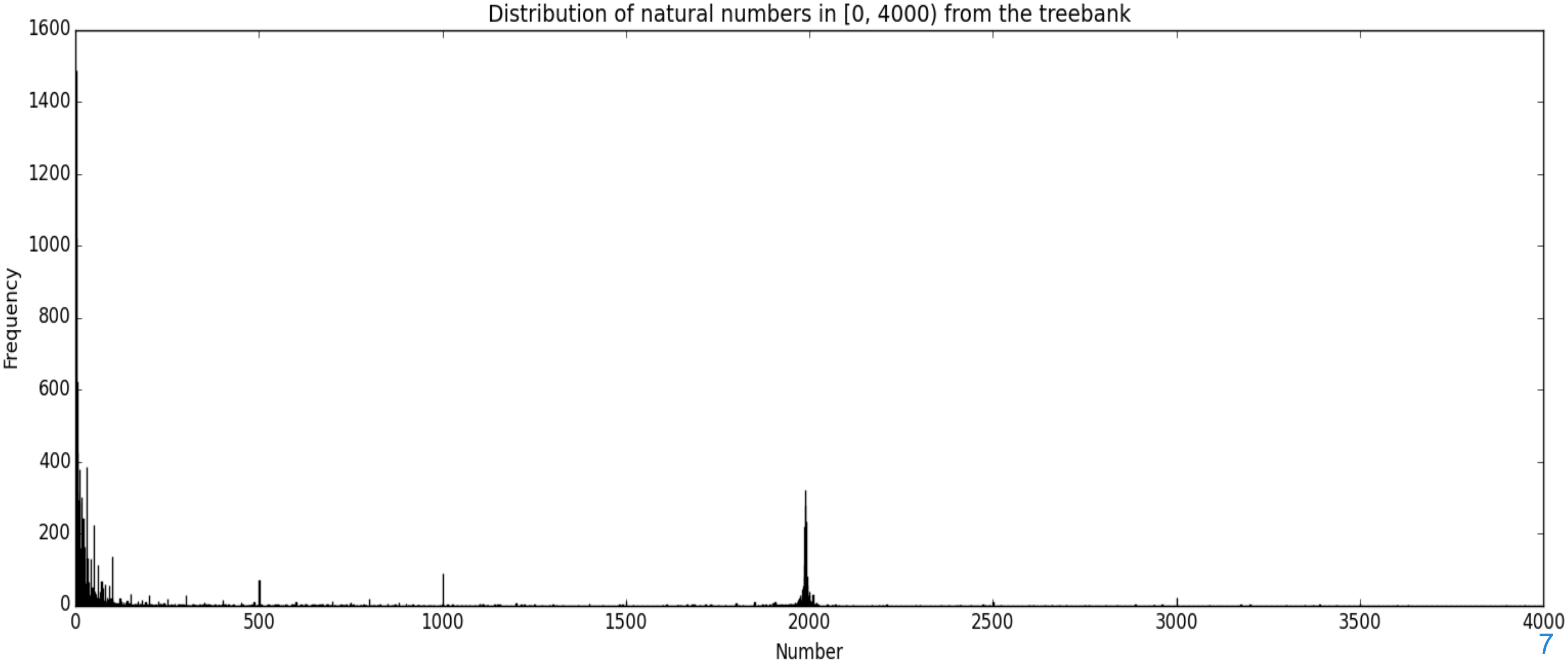
Examples:

- “12” → 12.
- “two hundred” → 200.
- “1.45 million” → 1,450,000.
- “early-1980s” → 1980.
- etc.

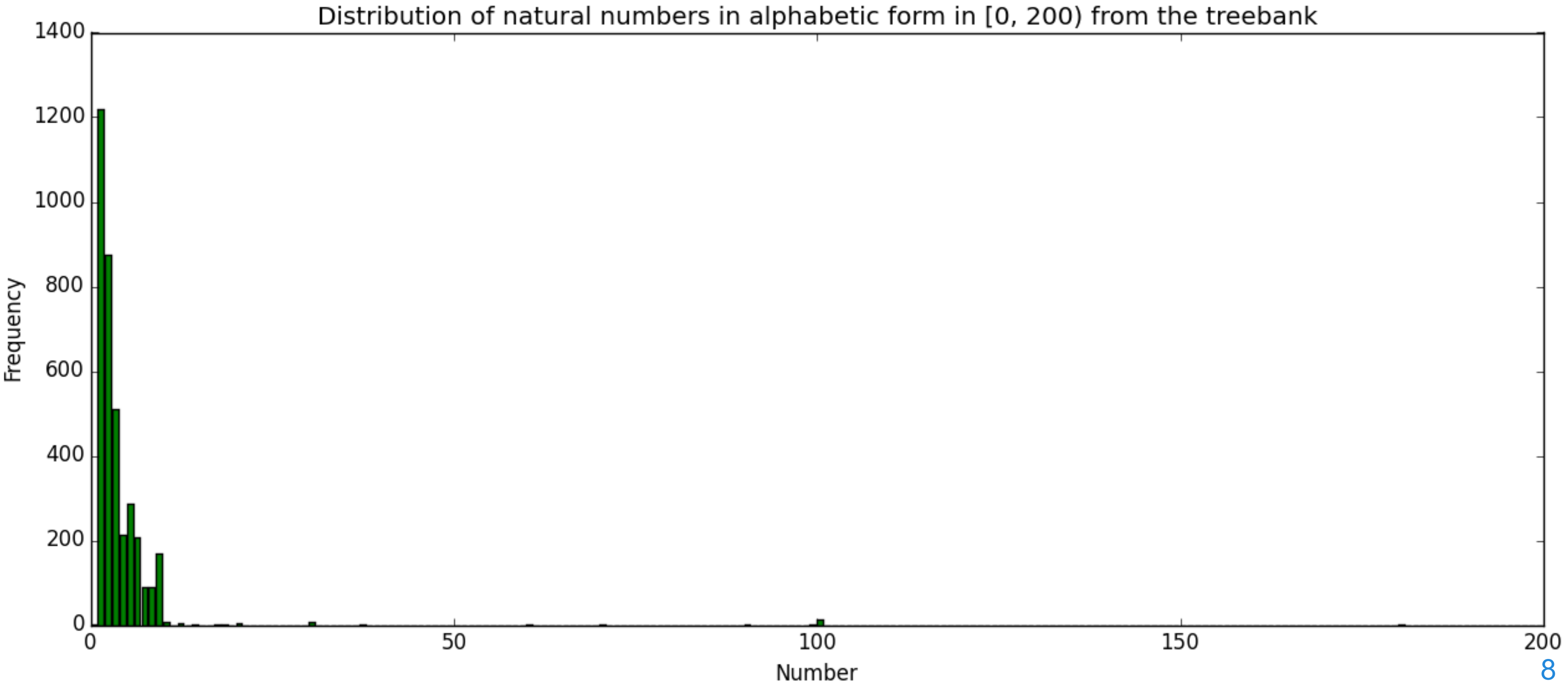
Overview / Goals



Overview / Goals



Overview / Goals



Overview / Goals

Total number of numbers:

- Number form: 9.392
- Alphabetic form: 3.728
- Both forms: 13.120

Overview / Goals

Extension: instead of using the probability of the best derivation of a string, use the **actual likelihood** of a string

p(derivation):

$$\mathrm{P}_G(\psi) = \prod_{r \in R} p(r)^{f_r(\psi)}$$

p(string):

$$\mathrm{P}_G(w) = \sum_{\psi \in \Psi_G(w)} \mathrm{P}_G(\psi)$$

p(best derivation):

$$\max_{\psi \in \Psi_G(w)} \mathrm{P}_G(\psi)$$

Overview / Goals

Expectation: Using the exact data likelihood while sampling leads to a better TSG than using an approximated data likelihood.

Problems:

- Fast implementation is indispensable
- How to measure differences between two approaches

CDEC Decoder

- Cdec is a decoder and aligner used in Machine Translation
- It translates sentences while deriving syntax trees for both the target and source language.
- CFG for source and target required.
- in C++ and Python-wrapper
- provides inside probability of root symbol

Example PCFG

root | Source language | Target language | Probability

```
[S]   ||| [D,1]          ||| [D,1]          ||| LogProb=-0.69314
[S]   ||| [S1,1] [S2,2]  ||| [S1,1] [S2,2]  ||| LogProb=-0.69314
[S1]  ||| [NZ,1] [S2,2]  ||| [NZ,1] [S2,2]  ||| LogProb=-0.69314
[S1]  ||| [NZ,1]         ||| [NZ,1]         ||| LogProb=-0.69314
[S2]  ||| [D,1] [S2,2]   ||| [D,1] [S2,2]   ||| LogProb=-0.69314
[S2]  ||| [D,1]          ||| [D,1]          ||| LogProb=-0.6931
[D]   ||| 1              ||| 1              ||| LogProb=-2.3025850

...
```

Metropolis-Hastings Sampling

- Repeat:
  - Generate a random candidate change
    - Remove or add substitution site marker
  - Compute new likelihood L_{new} of the data with change
  - Accept candidate change in two cases:
    - if $L_{new} > L_{old}$ always accept
    - otherwise accept with probability L_{new} / L_{old}

Grammars for Numerals

Deterministic:

```
[S] → [0]      | [1]      | [2]      | … | [9]
[S] → [1] [P]  | [2] [P]  | [3] [P]  | … | [9] [P]
[P] → [0]      | [1]      | [2]      | … | [9]
[P] → [0] [P]  | [1] [P]  | [2] [P]  | … | [9] [P]
```

Only one derivation is possible

Grammars for Numerals

Half ambiguous:

```
[S]  → [D] [D]       | [S1] [S2]
[S1] → [NZ] [S2]     | [NZ]
[S2] → [D] [S2]      | [D]
[NZ] → [1]           | [2]        | [3] | … | [9]
[D]  → [0]           | [1]        | [2] | … | [9]
```

Not deterministic

Grammars for Numerals

Highly ambiguous:

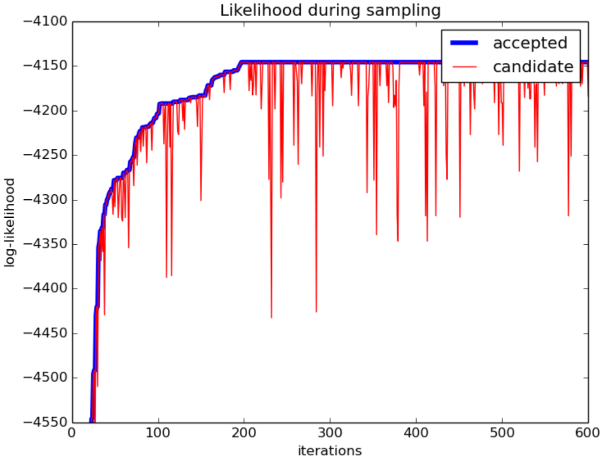
```
[S]  →  [Z]    | [NZ]        | [S1] [S2]
[S1] →  [NZ]   | [NZ] [S2]   | [S1] [S2]
[S2] →  [Z]    | [NZ]        | [Z] [Z]   | [Z] [NZ]  | [NZ] [Z] | [NZ] [NZ]
[NZ] →  [1]    | [2]         | [3]       |   …       | [9]
[Z]  →  [0]
```

Many parses

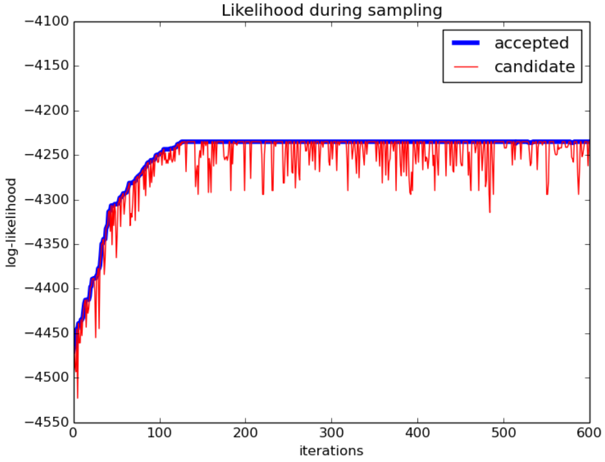
Comparison between grammars

Likelihood

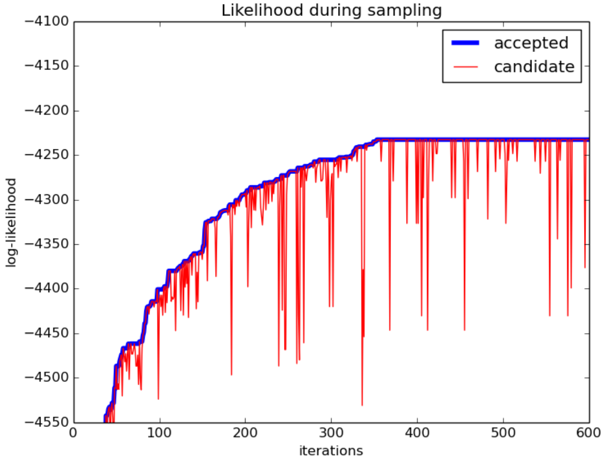
Highly ambiguous



Half ambiguous



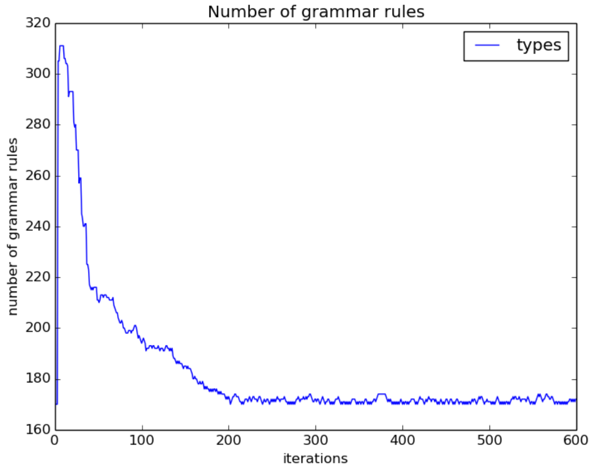
Deterministic



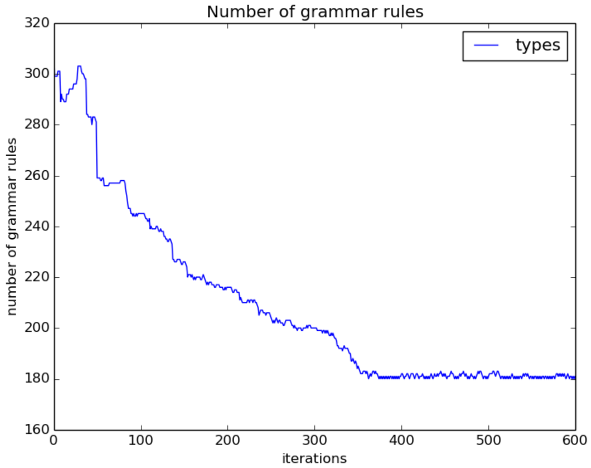
Comparison between grammars

Count of used rules

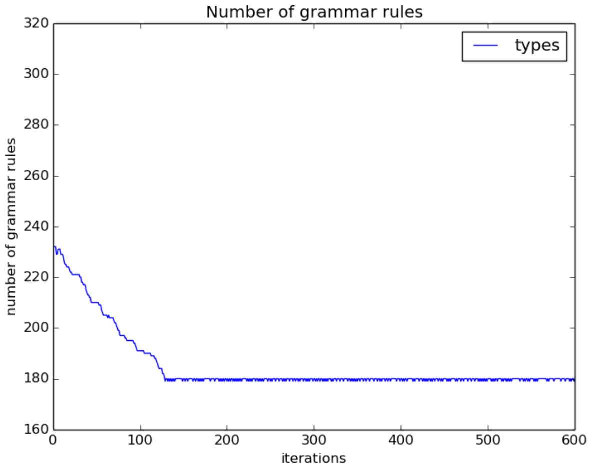
Highly ambiguous



Half ambiguous



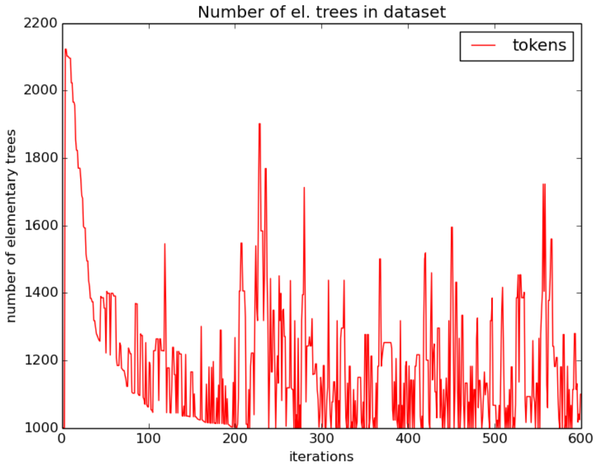
Deterministic



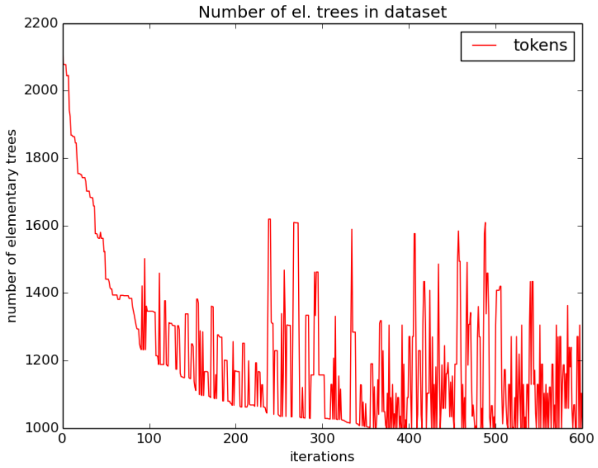
Comparison between grammars

Grammar size

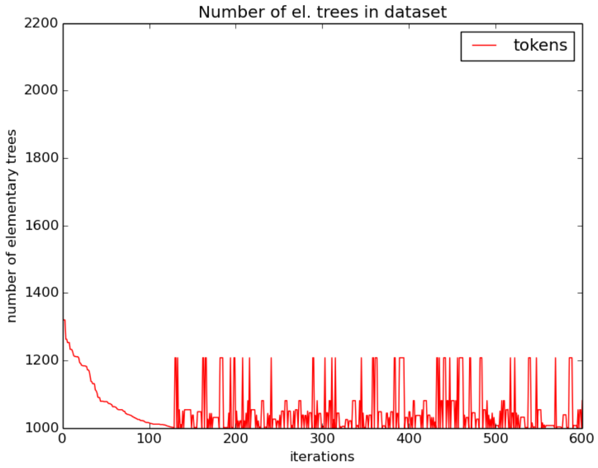
Highly ambiguous



Half ambiguous



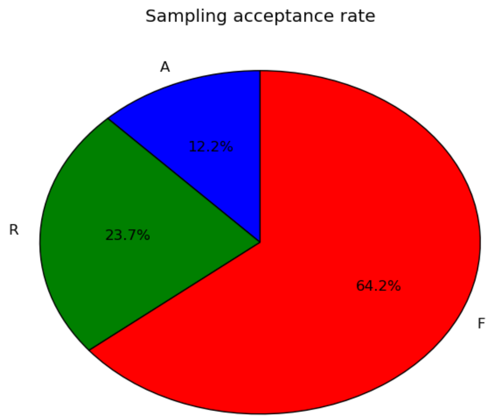
Deterministic



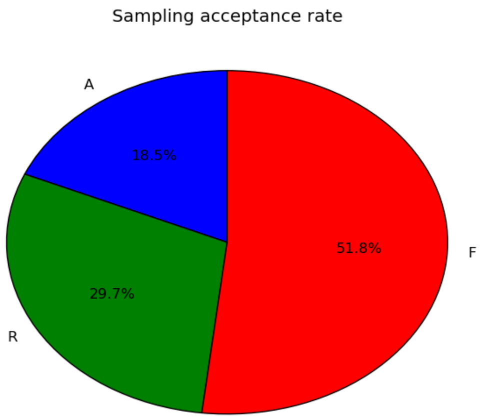
Comparison between grammars

Acceptance rate

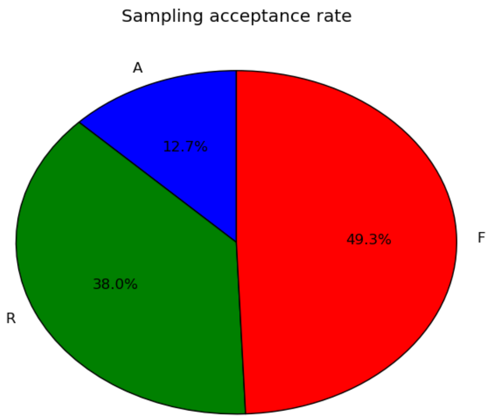
Highly ambiguous



Half ambiguous



Deterministic



Comparison: viterbi vs inside

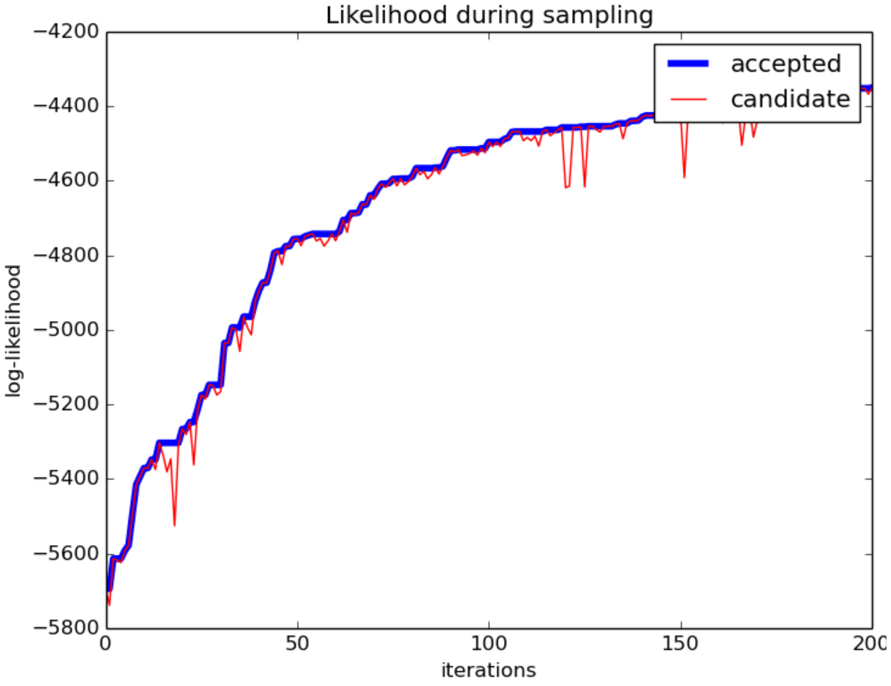
Experiment: use same data set and sample

- using inside probability
- using viterbi probability

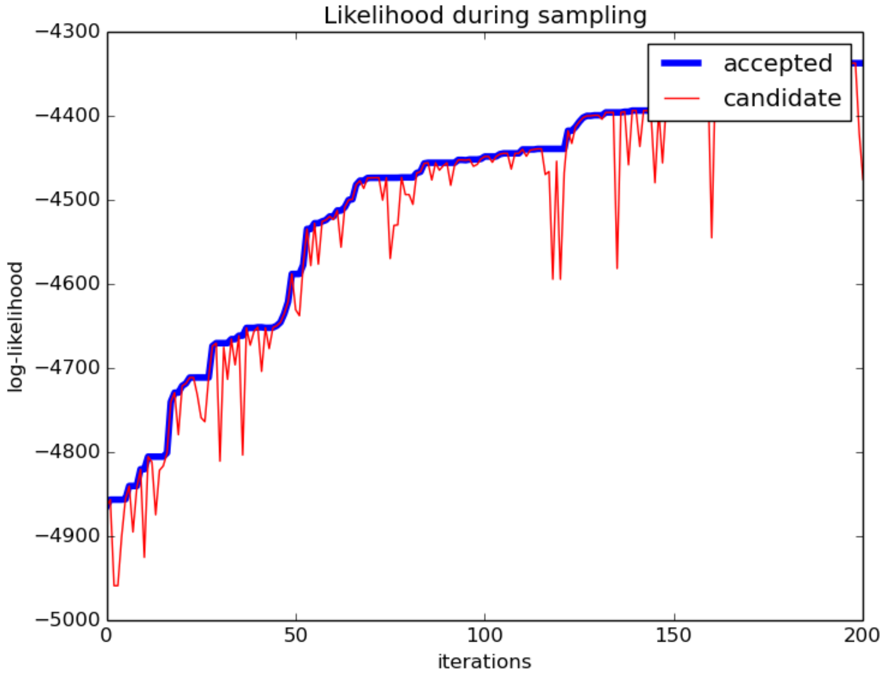
What are the differences?

Comparison: viterbi vs inside

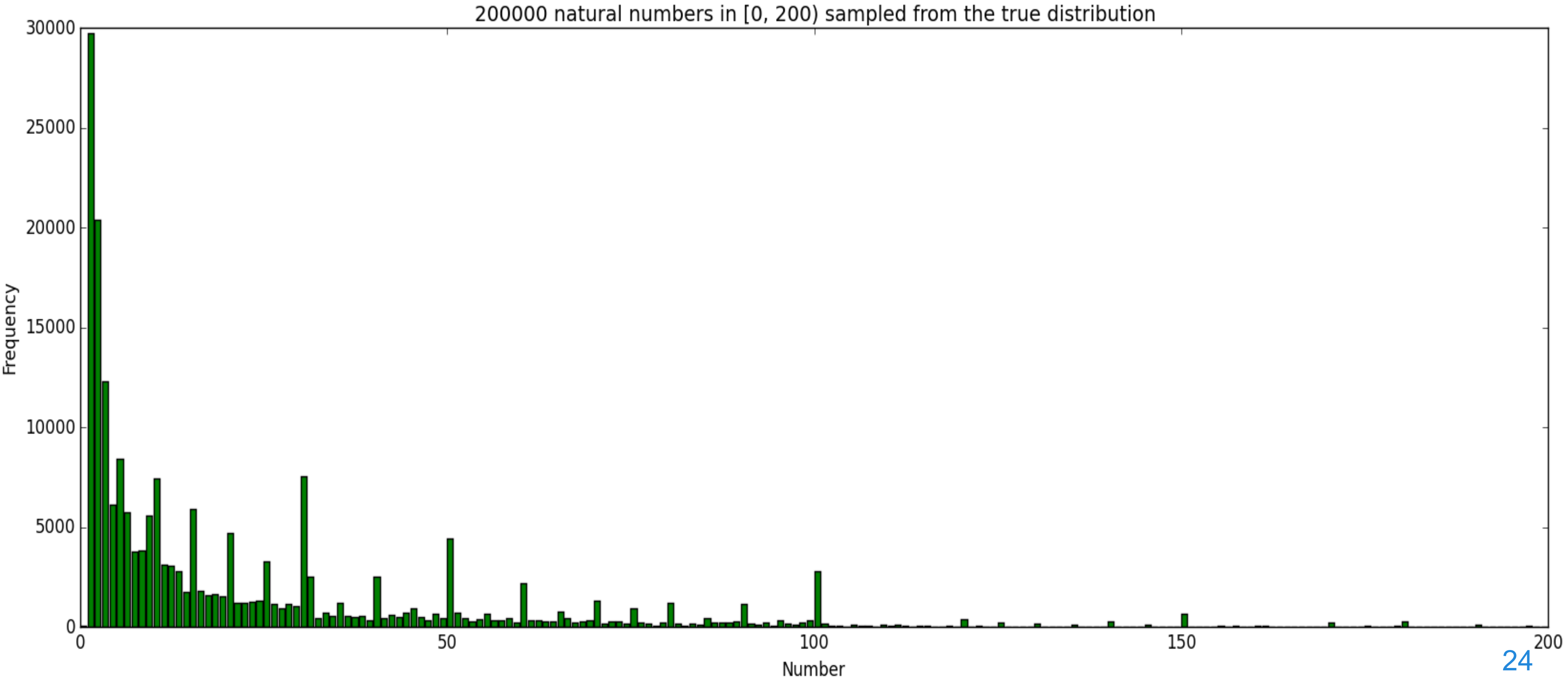
viterbi



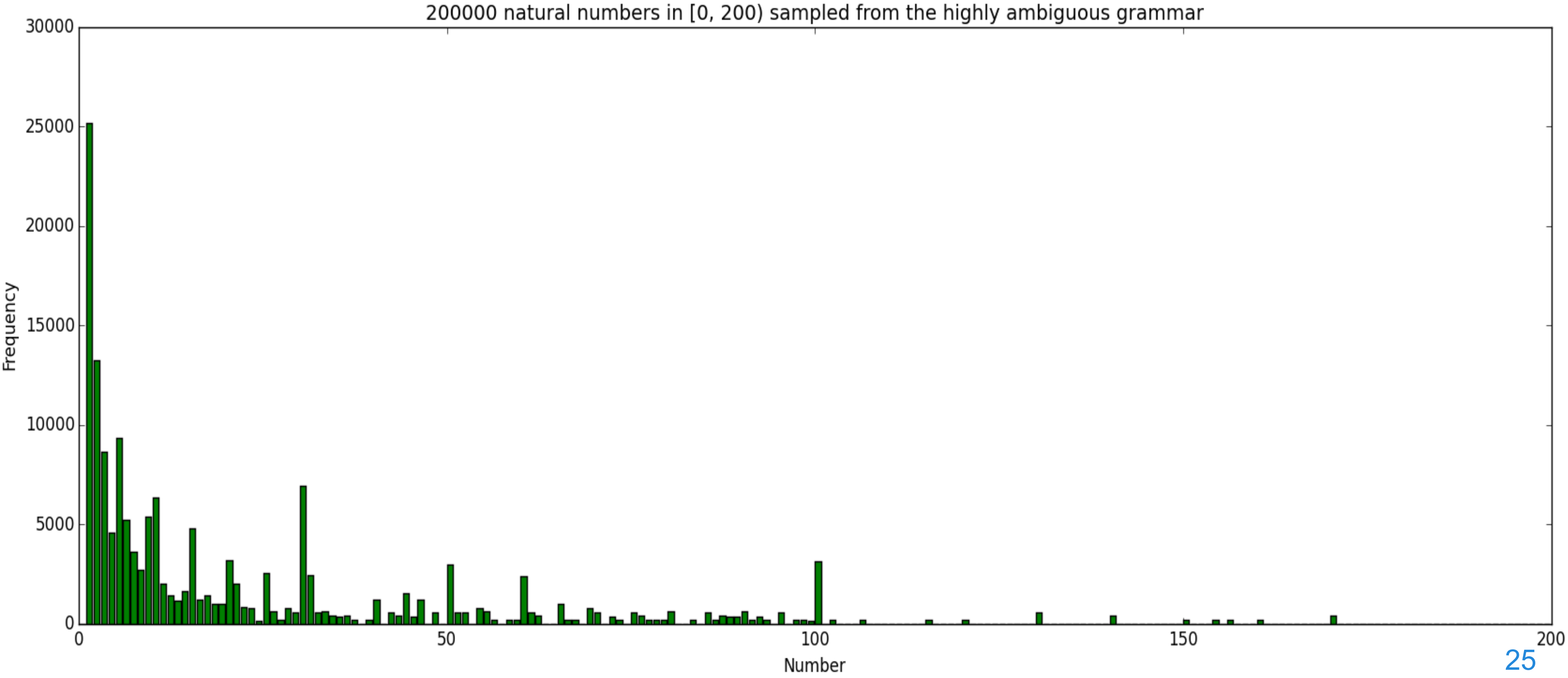
inside



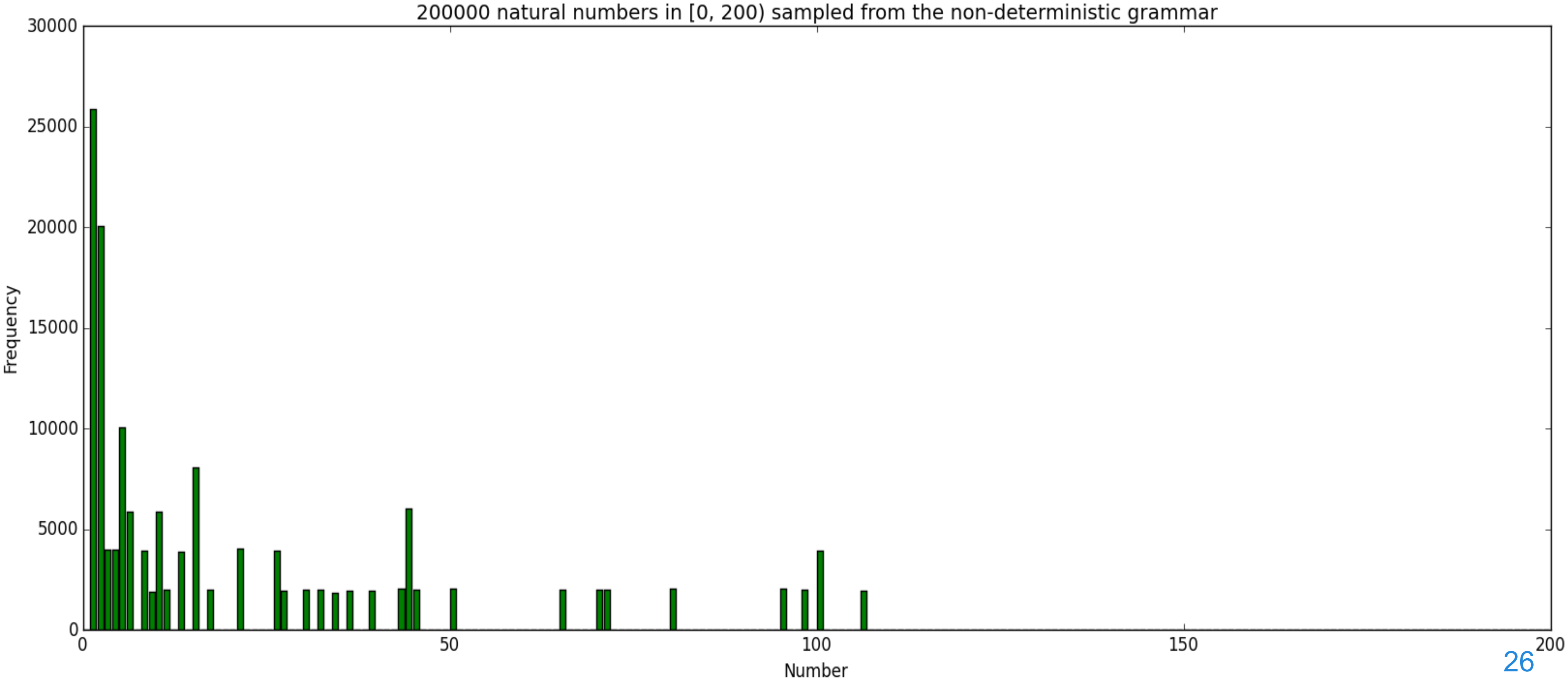
Comparison: Sampled distributions



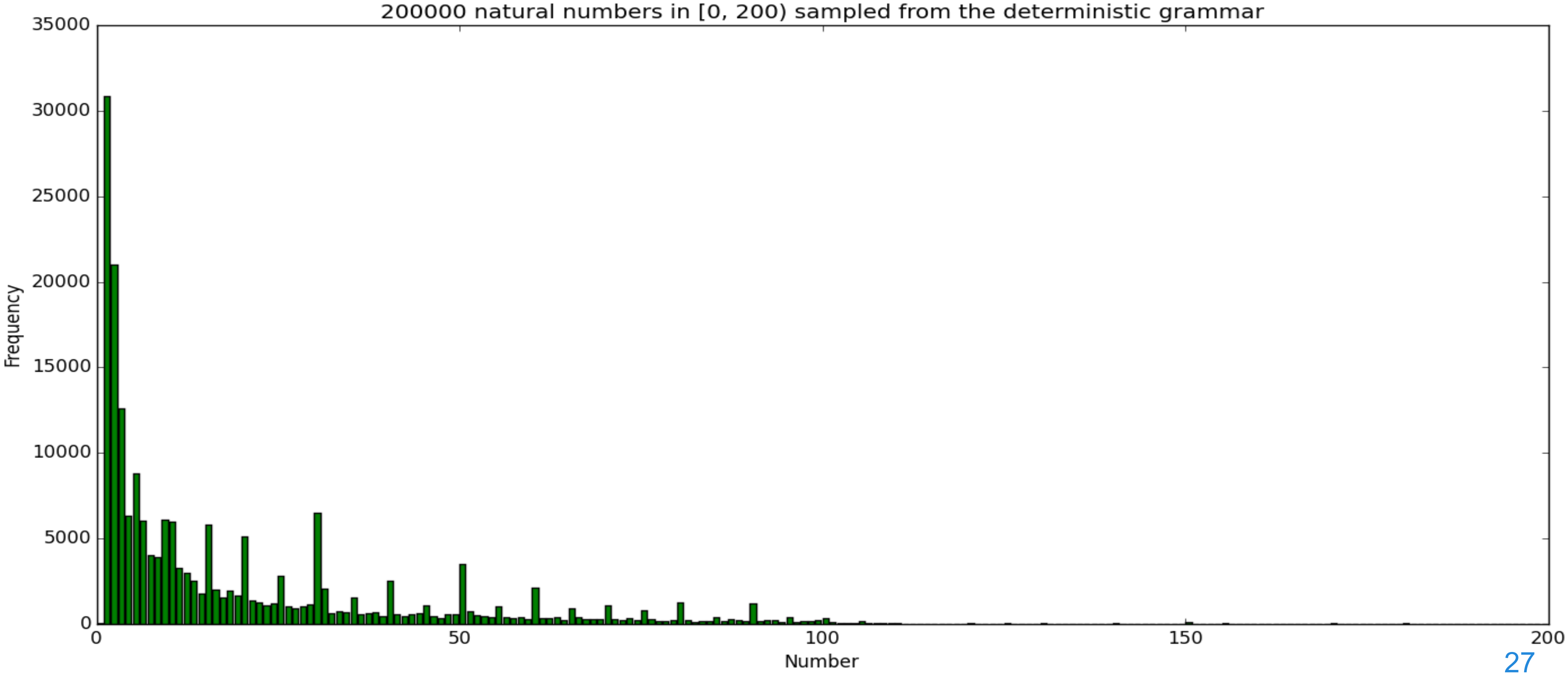
Comparison: Sampled distributions



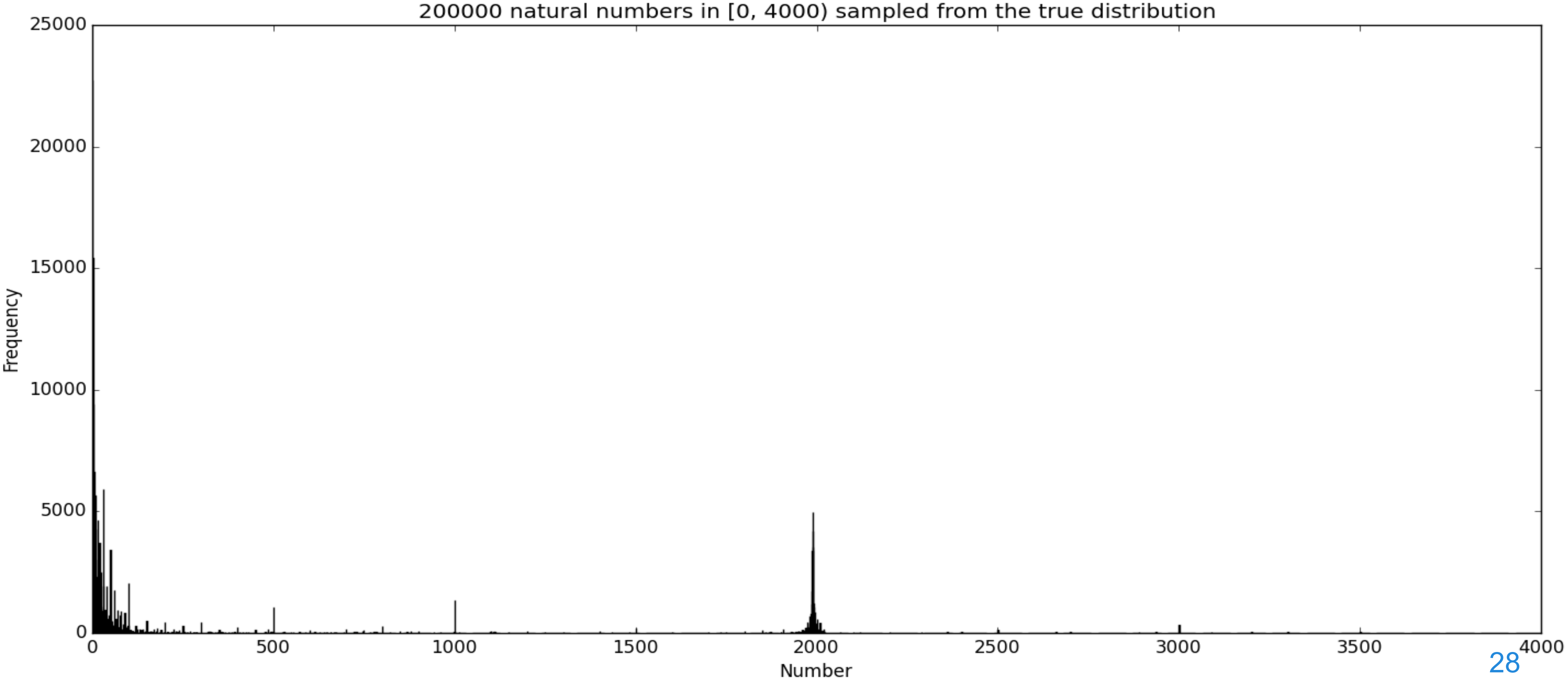
Comparison: Sampled distributions



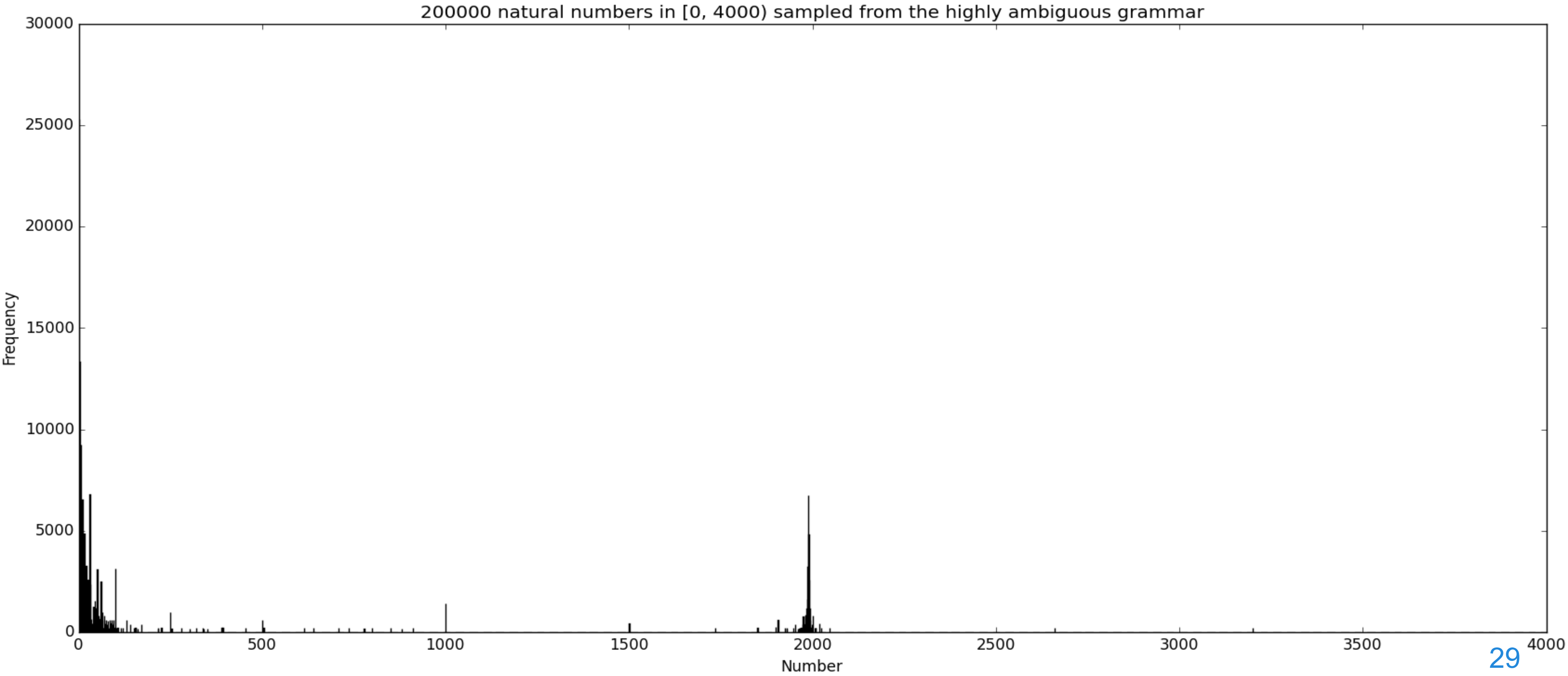
Comparison: Sampled distributions



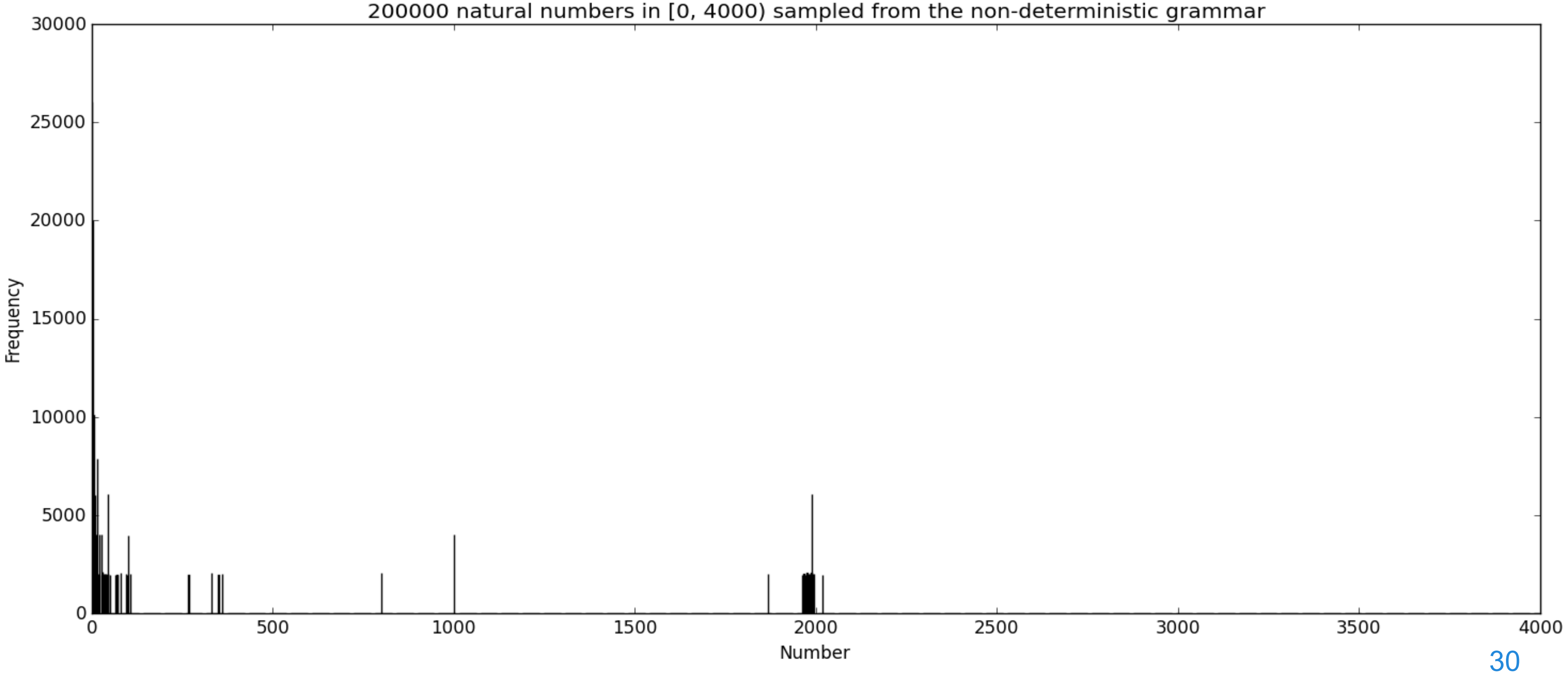
Comparison: Sampled distributions



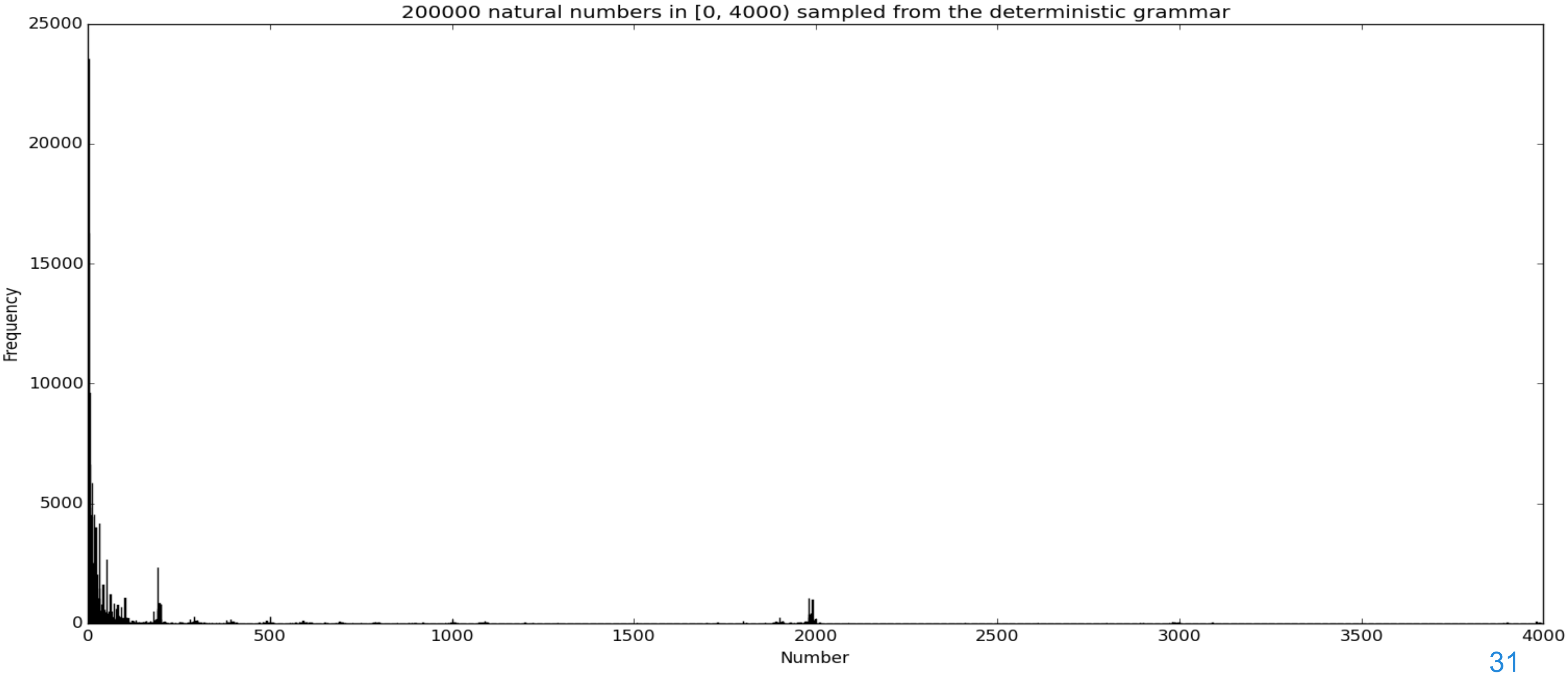
Comparison: Sampled distributions



Comparison: Sampled distributions



Comparison: Sampled distributions



Questions & Answers

Thank you for your attention