Constructive Membership Tests In Some Infinite Matrix Groups

Alexander Hulpke
Department of Mathematics
Colorado State University
Fort Collins, CO, 80523, USA
http://www.math.colostate.edu/~hulpke

The Task

Finitely generated group $G = \langle \mathbf{g} \rangle$
with $\mathbf{g}=\{g_1,\ldots,g_k\}$. WLOG $\mathbf{g}^{-1} \subset \mathbf{g}$.

Express $e \in G$ as a *word* in $\mathbf{g}$, i.e. a product of the g_i's that equals e.

Want: Algorithm to find Short(est) word for given e.

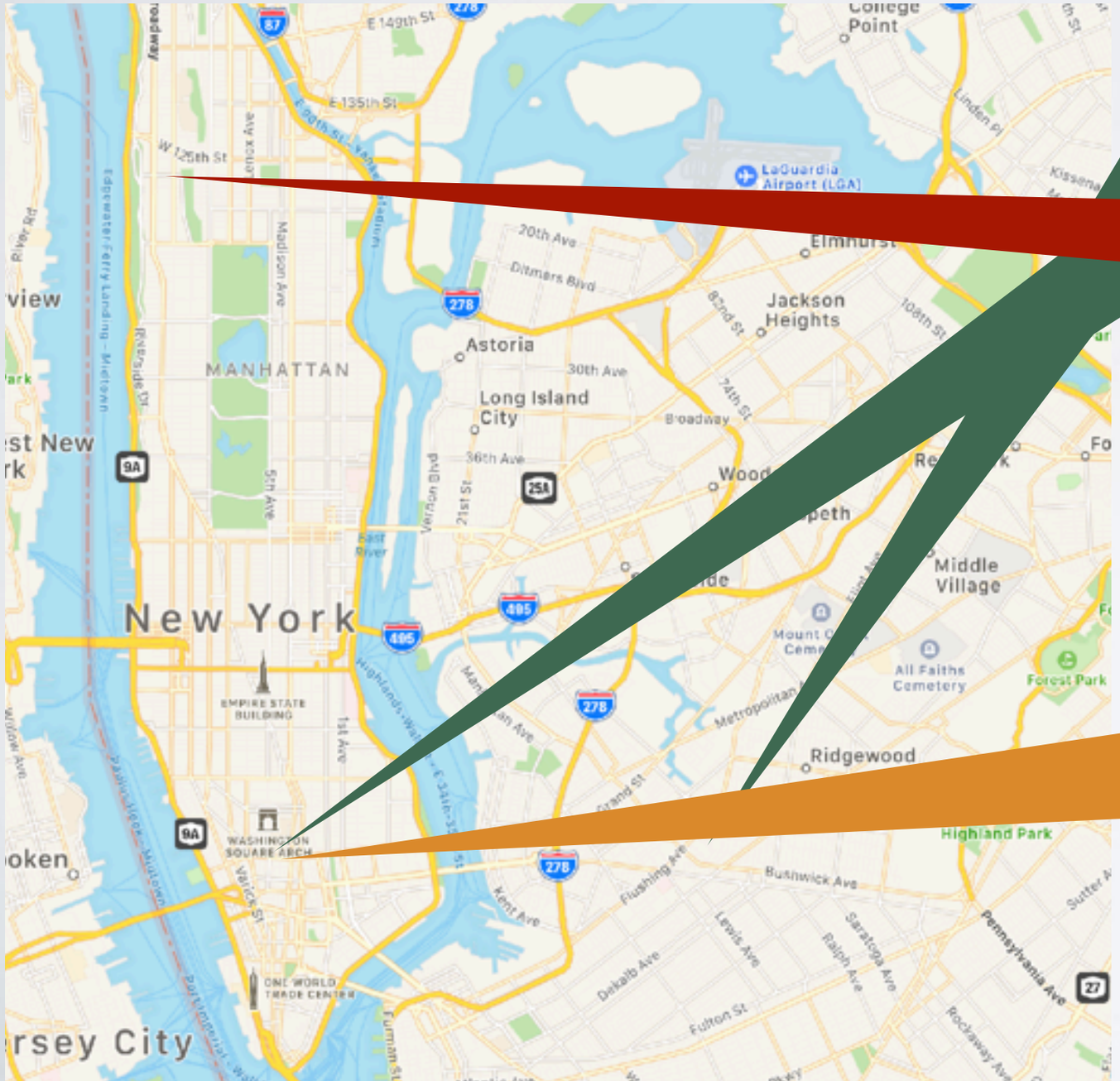
Caveat: Discrete Logarithm for $G = \langle a \mid a^m=1 \rangle$.
Aim for practically usable solution.

Some Applications

- Puzzles (Rubik's Cube and friends)
- Evaluating homomorphisms given on generators.
- (For me:) Finitely generated subgroups of $SL_n(\mathbb{Z})$ or $Sp_{2n}(\mathbb{Z})$: Verify arithmeticity (finite index) through coset enumeration.

Auspicious Location

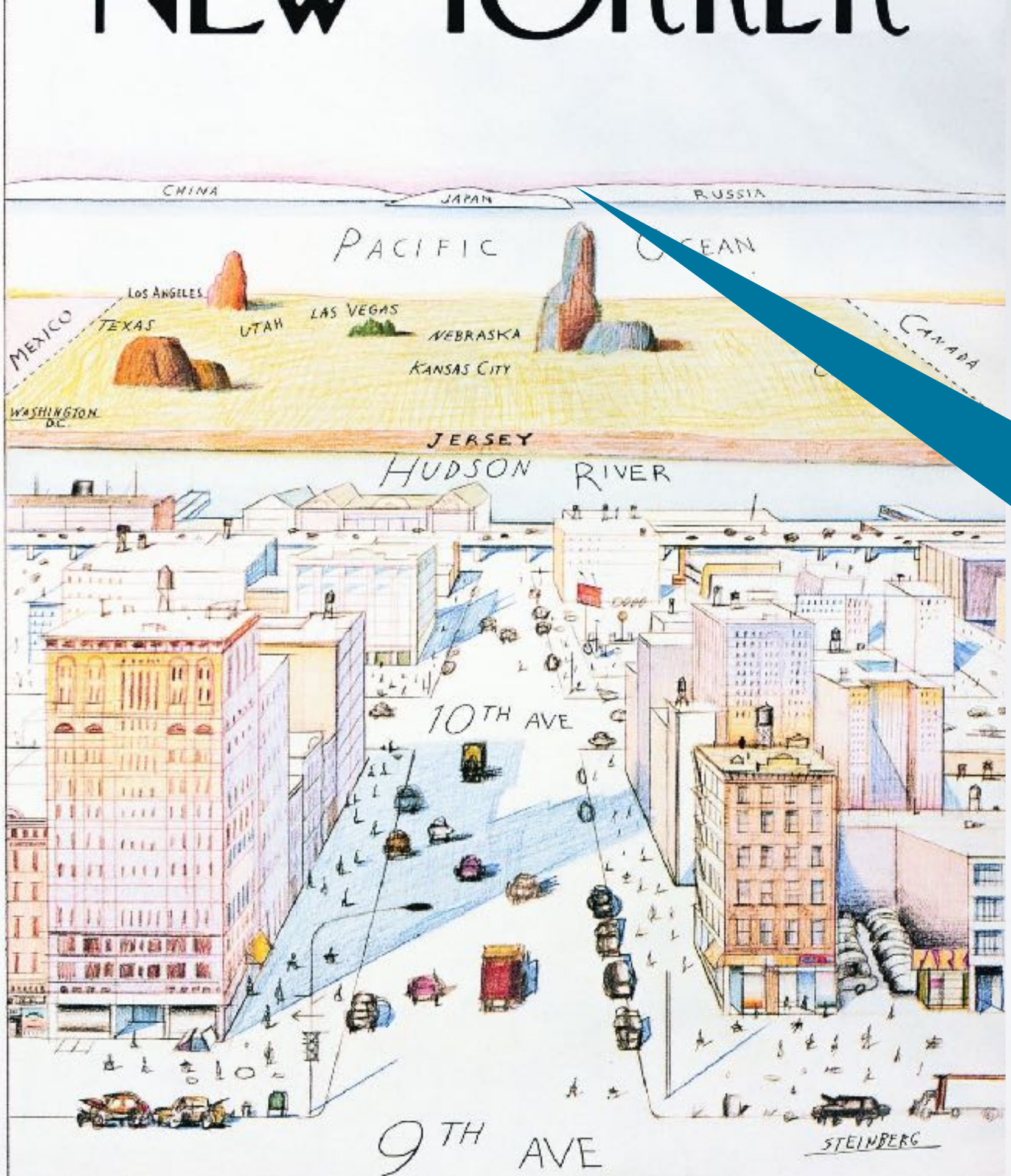
Presentations for $SL_n(\mathbb{Z})$ and $Sp_{2n}(\mathbb{Z})$ originated with residents of NYC:



WILHELM MAGNUS (Courant&Polytech.)

JOAN BIRMAN (Barnard)

PHILLIP GOLD (NYU)



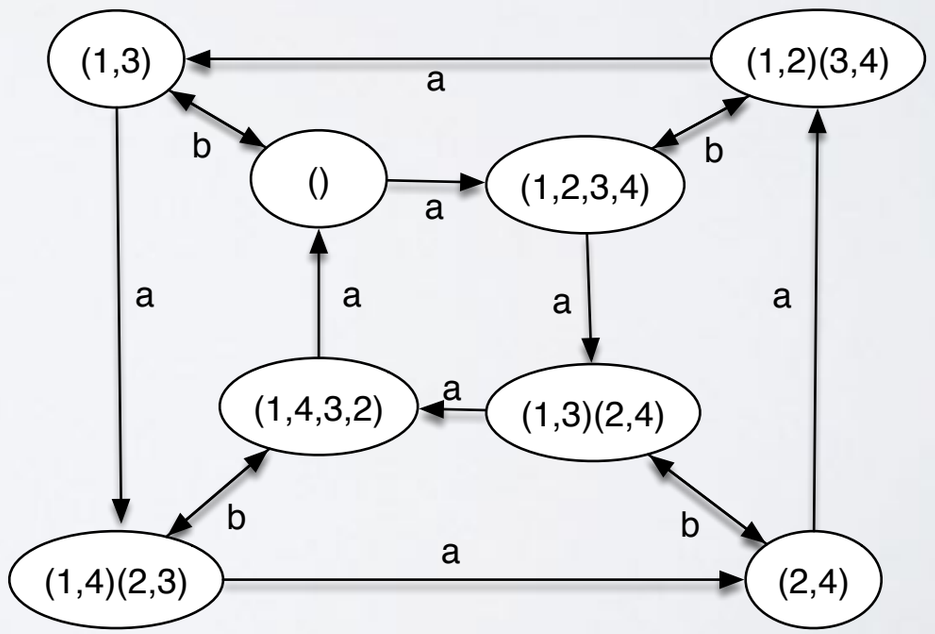
HELMUT KLINGEN
(Freiburg, Germany)

Basic Method: "Floodsearch"

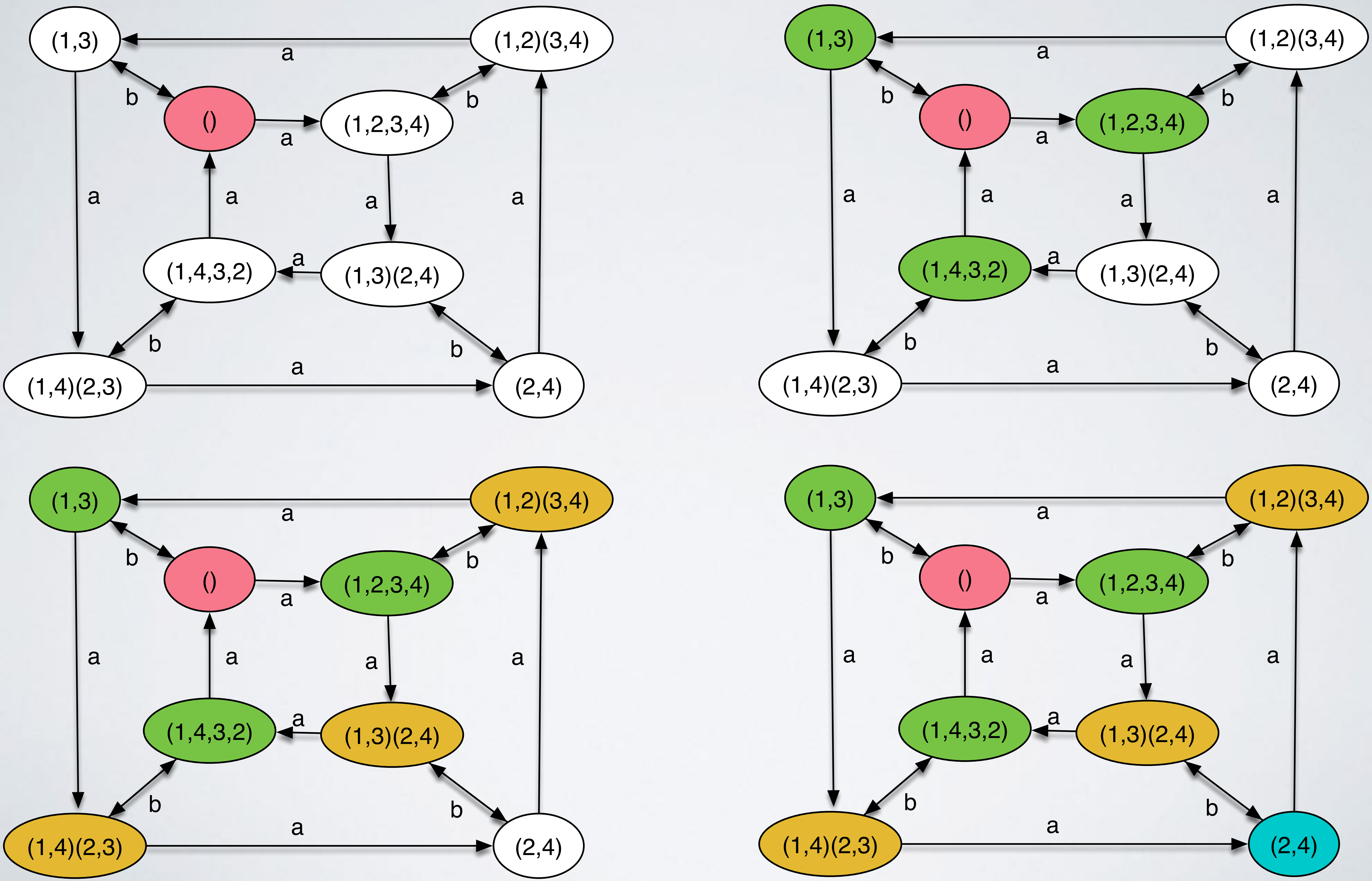
Def: **Cayley Graph** of group G: Vertices are group elements, directed edges $x \to_g y$ iff $x \cdot g = y$ for generator g.

Example $G = D_8 = \langle a=(1,2,3,4), b=(1,3) \rangle$. So $(2,4) = a \cdot b \cdot a^{-1} = a \cdot a \cdot b$

Start at identity, in each step flood neighboring vertices.



Example



Feasibility

+ : Finds word of guaranteed shortest length. Only method to do so in general.

- : Extensive storage requirement (at least 2 bits per element (COOPERMAN, et. al.)), Will only work for finite ball around identity.

Has been principal method to attack Rubik's cube (ROKICKI, building on many others)

Using Normal Forms

The Generators usually chosen for SL are elementary matrices; $t_{i,j}$ is identity with an extra 1 on position i,j

Writing $M \in \mathrm{SL}_n(\mathbb{Z})$ as product of elementary matrices means performing row/column operations to get to identity.

Since we work over $\mathbb{Z}$, identity matrix is Hermite Normal Form, algorithms have been studied extensively (STORJOHANN, SAUNDERS, WAN, **Many more**).

Algorithm HNF

Perform HNF calculation, tracking operations.

Problem: Long words (easily 100k for input being random products of generators of length 500).

Why?

Adding a k-multiple of one row to another records as product of k generators. Intermediate large numbers.

Norm-Based Approach

Do not try to clean out, but to make entries "smaller" towards identity.

This mirrors the experience of using norm-based algorithms for HNF.

Define the *height* of M to be $h(M)=||M-I||^2$ with $||A||^2=\sum_{i,j} a_{i,j}^2$.

Height-Based Algorithm

- For each generator (and inverse) g of $G = SL_n(\mathbb{Z})$ calculate $h(g \cdot M)$ and $h(M \cdot g)$.
- Replace M by product that minimizes height. Repeat.
- If no reduction is possible, process partially reduced M through HNF-based algorithm.

Very elementary heuristics, but produces much shorter words. The HNF fallback is required.

Local Optimization

Reduce reliance on HNF fallback by increasing generating set $\mathbf{g}$:

Use $\mathbf{g} \cup \mathbf{g}^2 \cup \mathbf{g}^3$ (tradeoff between cost and success), get over small bumps in length.

Result works well in experiments in small dimensions.

Symplectic Group

Let $\mathrm{Sp}_{2n}(\mathbb{Z}) = (M \in \mathrm{SL}_{2n}(\mathbb{Z}) \mid MJM^{T} = J)$, where

$J = \begin{pmatrix} 0 & I_n \\ -I_n & 0 \end{pmatrix}$ with I_n denoting an $n \times n$ identity matrix.

Many applications in Geometry, Algebra, Number Theory, ...

Generators: Following KLINGEN/GOLD/BIRMAN:

$$\mathrm{Sp}_{2n}(\mathbb{Z}) = \langle Y_i, U_i, \mathbb{Z}_j \mid 1 \le i \le n, 1 \le j \le n-1 \rangle$$

with $Y_i = t_{i,n+i}^{-1}$, $U_i = t_{n+i,i}$ and

$$Z_i = \left(t_{i+1,n+i} / t_{i+1,n+i+1}\right)^{t_{i,i+1}}$$

the identity with $\begin{pmatrix} -1 & 1 \\ 1 & -1 \end{pmatrix}$ at position $i, n+i$

Symplectic Group

Let $\mathrm{Sp}_{2n}(\mathbb{Z}) = (M \in$ [illegible] $\mid M^T J M = J)$ where

$$J = \begin{pmatrix} 0 & I_n \\ -I_n & 0 \end{pmatrix}$$

[illegible] matrix.

Many applic[illegible]ti[illegible] [illegible]er Theory, ...

Generators: [illegible]

$$\mathrm{Sp}_{2n}(\mathbb{Z}) = \langle Y_i, \quad \leq i \leq n-1 \rangle$$

with $Y_i = t_{i,n+i}^{-1}$, $U_i = t_{i,i}$ [illegible] $Z_i = \left(t_{i+1,n+i} / t_{i+1,n+i+1} \right)^{t_{i,i+1}}$

the identity with $\begin{pmatrix} -1 & 1 \\ 1 & -1 \end{pmatrix}$ at position $i, n+i$

The height-based algorithm fails abysmally.

Further Generators

Also known that

$$Sp_{2n}(Z) = \left\langle \left\{ t_{i,n+j}t_{j,n+i}, t_{n+i,j}t_{n+j,i} \mid 1 \leq i < j \leq s \right\} \cup \left\{ t_{i,n+i}, t_{n+i,i} \mid 1 \leq i \leq n \right\} \right\rangle$$

and add these elements as further generators (as words expressions in U_i, V_i, $\mathbb{Z}_i$, with words from BIRMAN and basic calculations).

Note: These are closer to elementary matrices, so hope for better performance.

Further Generators

Also known that

$$Sp_{2n}(Z) = \left\langle \left\{ t_{i,j} \; t_{i,n+j} \; t_{n+i,j} \; t_{n+i,n+i} \mid 1 \le i \le n \right\} \right\rangle$$

and add the elementary generators (as words expressed as words from Birman and Bas

The height-based algorithm still fails abysmally.

Note: These are closer to elementary matrices, so hope for better performance.

Decomposing The Symplectic Group

Let $R = \left\{ \text{diag}(A, B) \in Sp_{2n}(Z) \mid A, B \in GL_n(Z) \right\} \leq Sp_{2n}(Z)$

and $S = \left\{ \text{diag}(M, M^{-1}) \mid M \in SL_n(Z) \right\} \leq Sp_{2n}(Z) \cong SL_n(Z)$

Then $[R:S]=2$ and $R = \langle S, (Y_1^2 U_1)^2 \rangle$.

Using the algorithm for SL, we can express elements of S as words in generators, using the extra element $(Y_1^2 U_1)^2$ we can do so in R.

Thus it is sufficient to find a word that (by multiplication) brings $e \in \mathrm{Sp}_{2n}(\mathbb{Z})$ into R.

Reduce To Elements Of R

Aim to use partial norms to get the two "off diagonal" blocks to be zero.

First attempt of using

$$h(a)=\sum_{i=1..n}\sum_{j=1..n}(a_{i,n+j}{}^2+a_{n+i,j}{}^2)$$

did not succeed - close but not exactly.

Reason is that making one entry much smaller wins over increasing another entry (in different position) just a bit. This ends in blind alley.

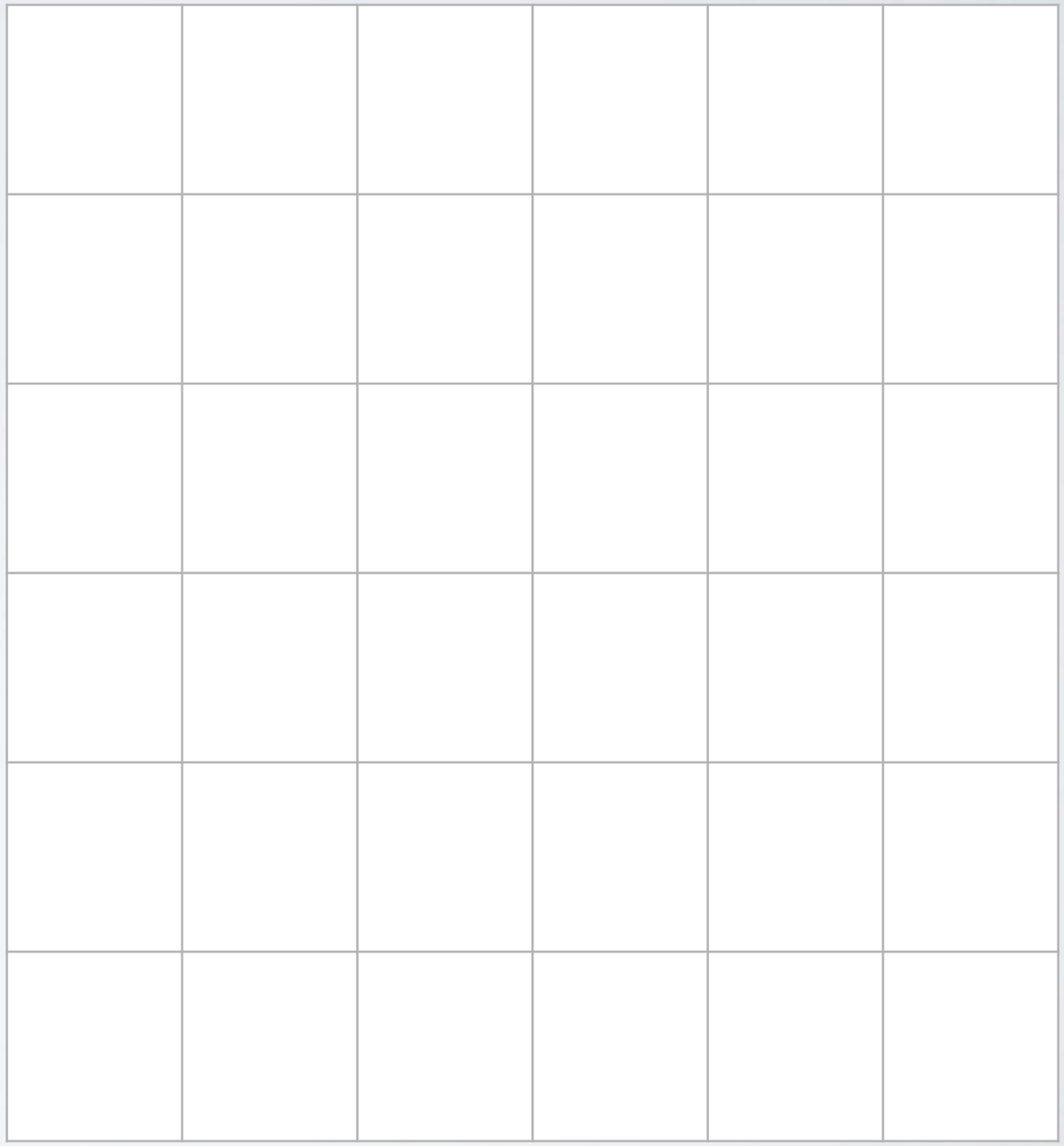
Whac-A-Nonzero

Use multiple heigh
functions to force
entries to be zero
iteratively in more
and more rows.

**This succeeded in
examples tried.**

Whac-A-Nonzero

Use multiple height functions to force entries to be zero iteratively in more and more rows.

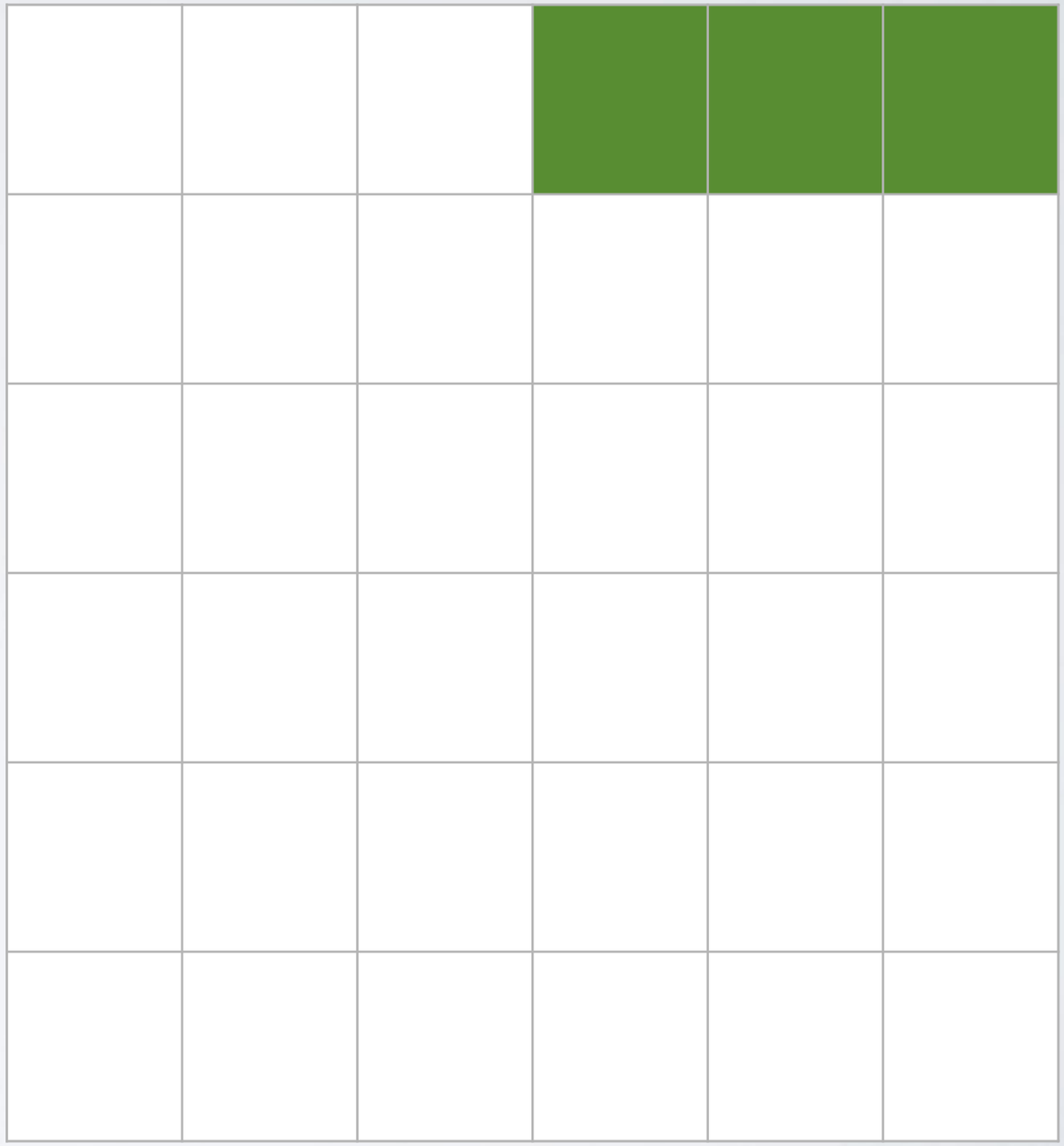


This succeeded in all examples tried.

Whac-A-Nonzero

Use multiple height functions to force entries to be zero iteratively in more and more rows.

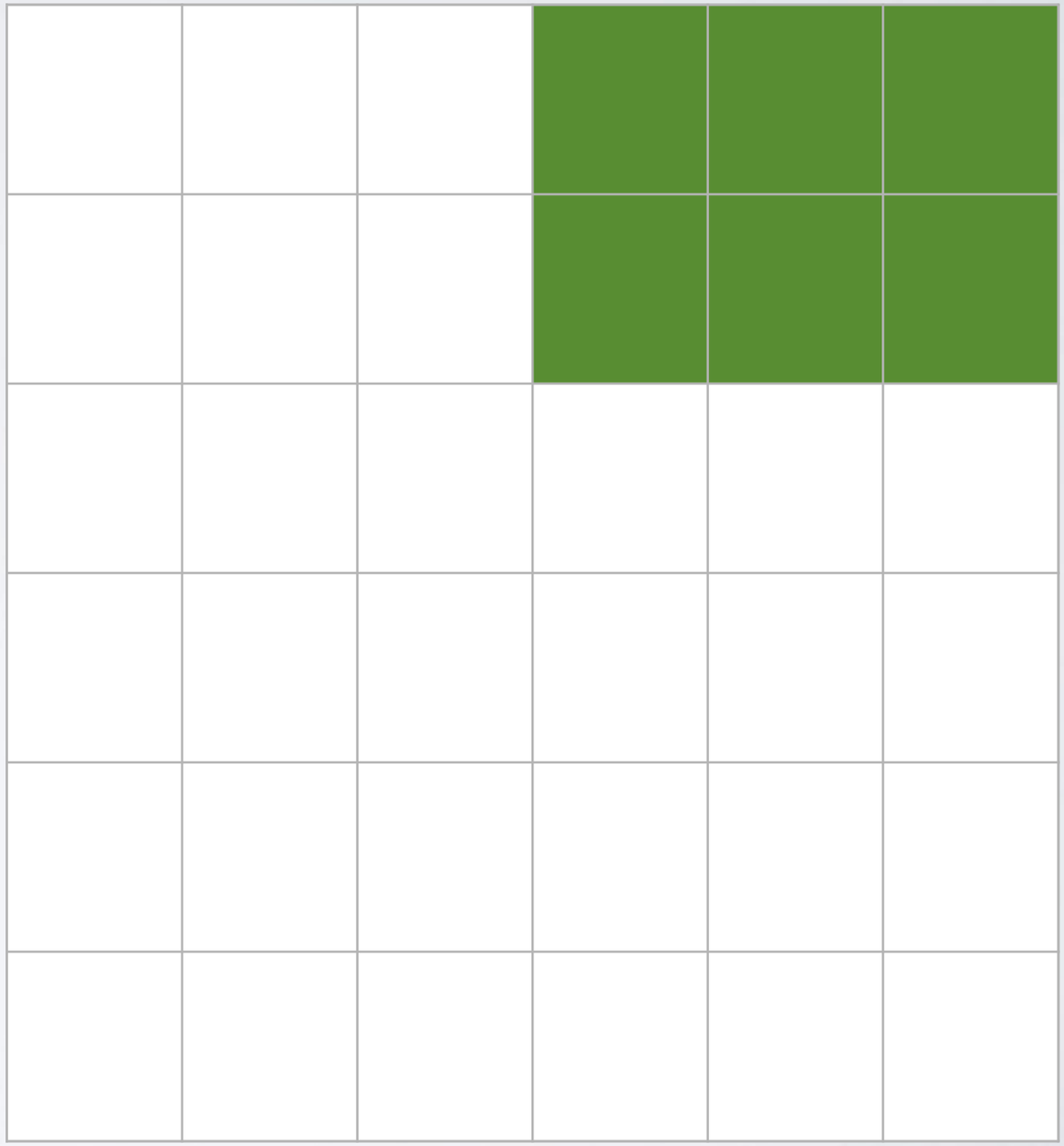
This succeeded in all examples tried.



Whac-A-Nonzero

Use multiple height functions to force entries to be zero iteratively in more and more rows.

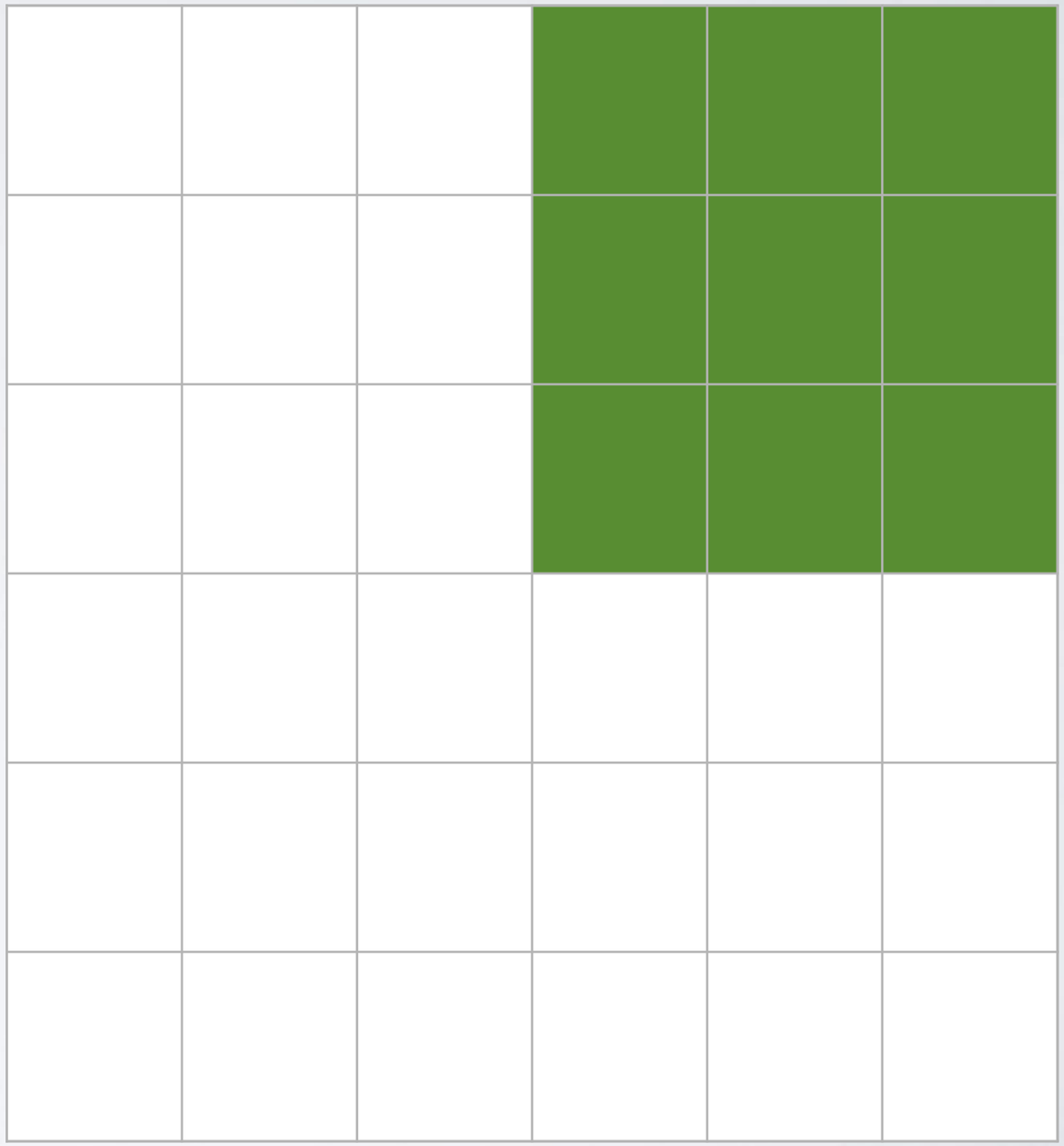
This succeeded in all examples tried.



Whac-A-Nonzero

Use multiple height functions to force entries to be zero iteratively in more and more rows.

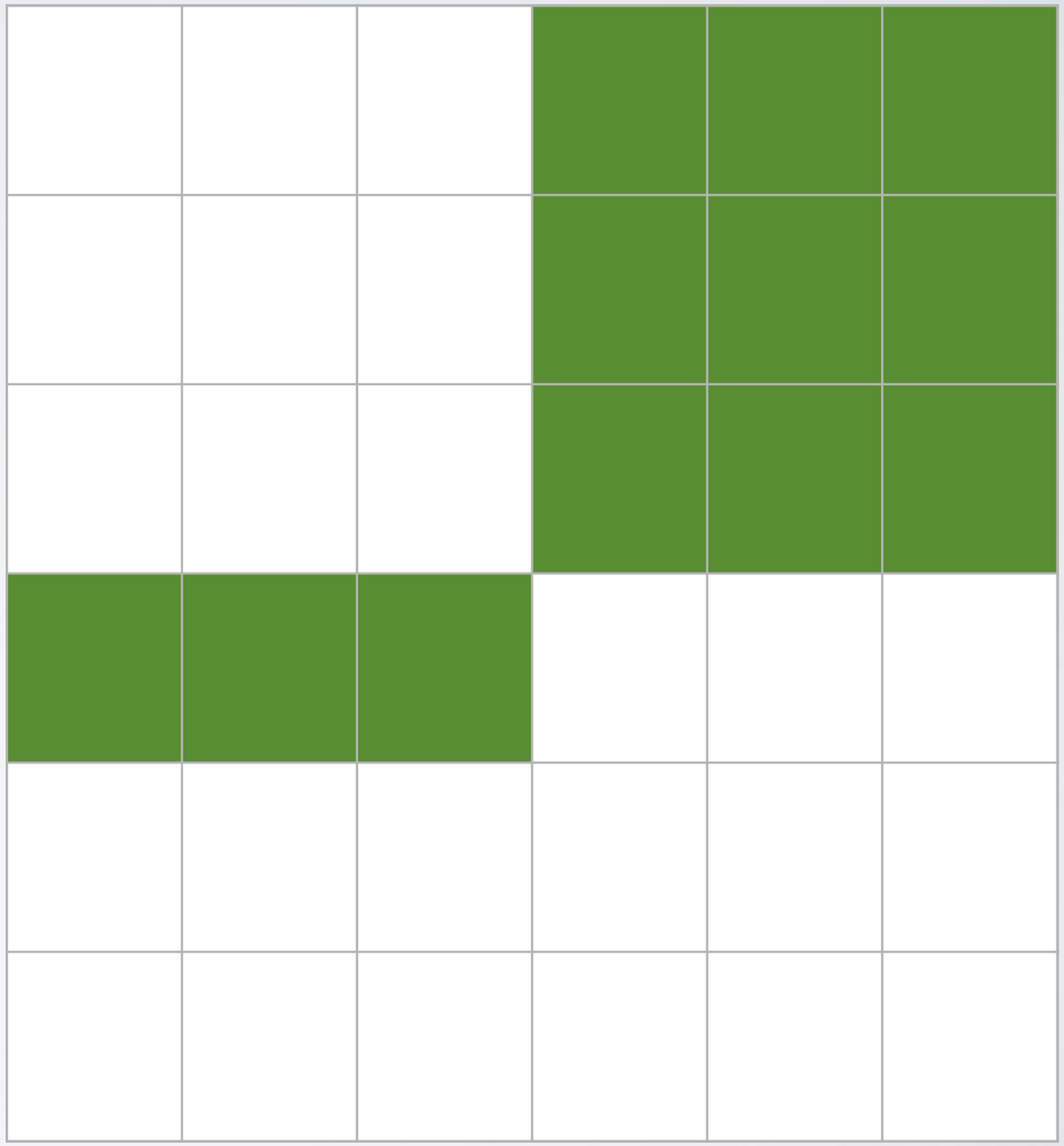
This succeeded in all examples tried.



Whac-A-Nonzero

Use multiple height functions to force entries to be zero iteratively in more and more rows.

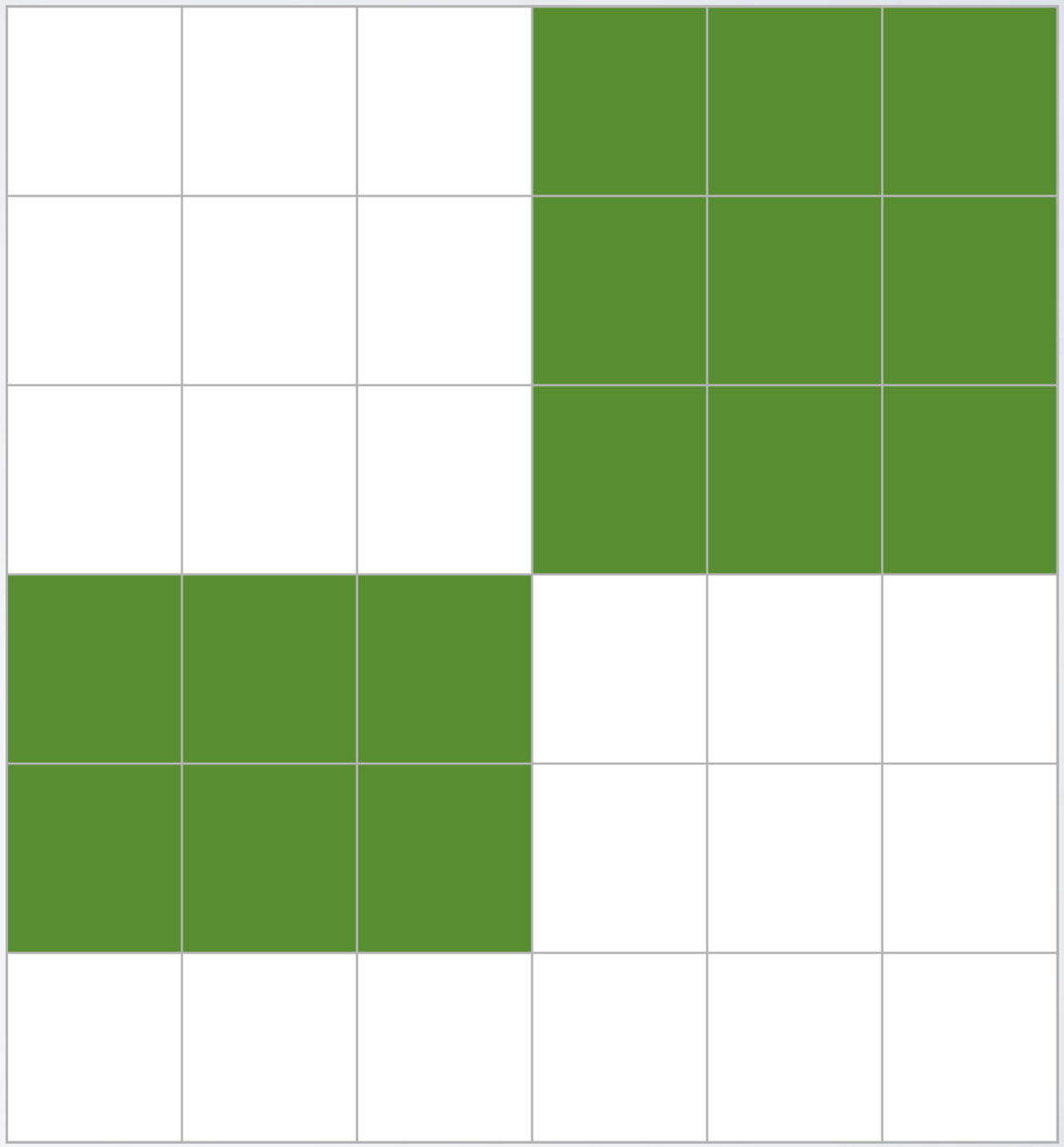
This succeeded in all examples tried.



Whac-A-Nonzero

Use multiple height functions to force entries to be zero iteratively in more and more rows.

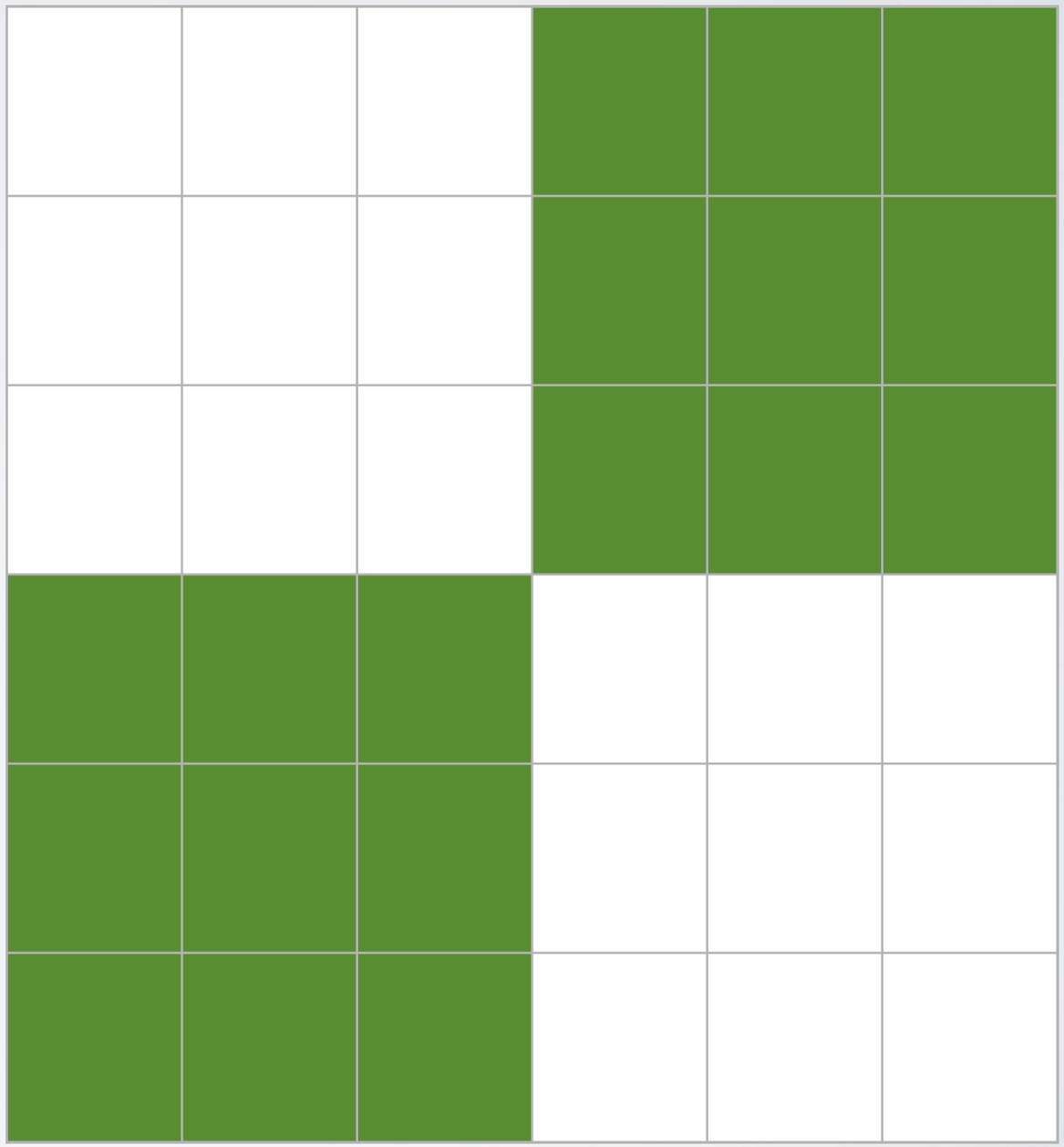
This succeeded in all examples tried.



Whac-A-Nonzero

Use multiple height functions to force entries to be zero iteratively in more and more rows.

This succeeded in all examples tried.



Experimental Observations

Cannot form unbiased random elements of $SL_n(\mathbb{Z})$. Even less calculate what their word length would be.

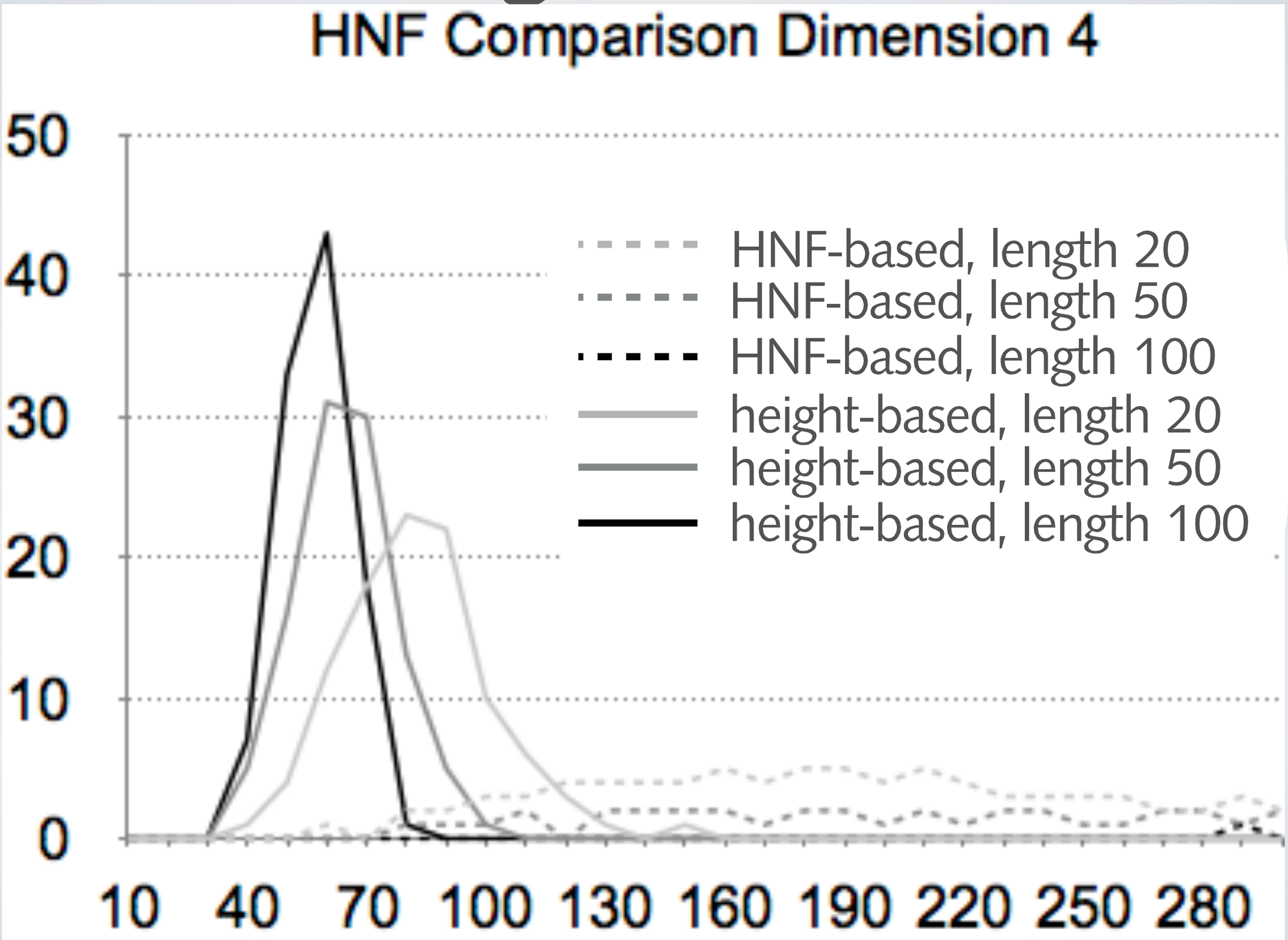
Instead, create random products of prescribed lengths. (This will often be not be minimal!)

Factor using algorithms. Compare length ratios. Plot what percentage of experiments (for given length) gave which (rounded) ratio.

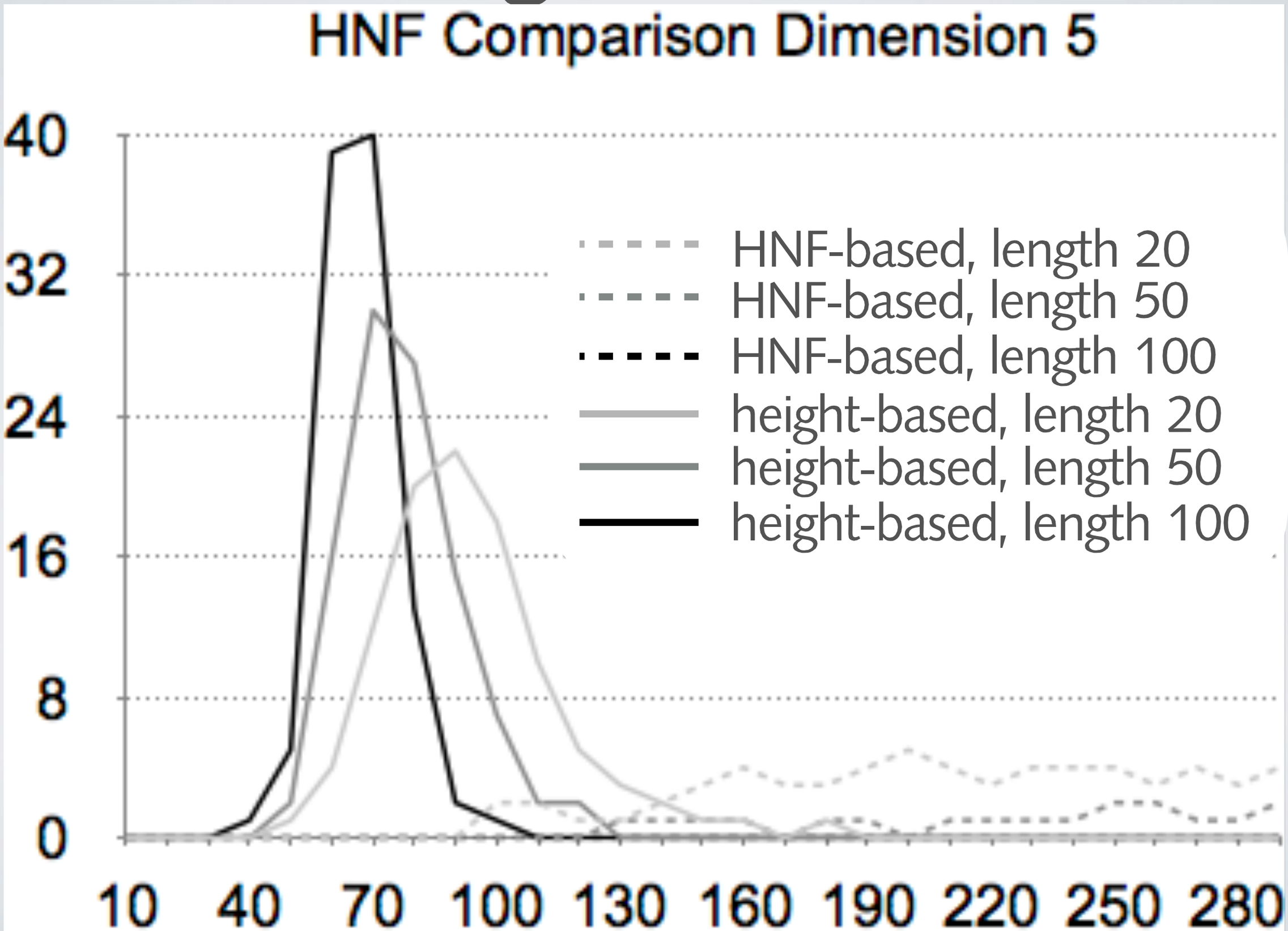
Multiple Lengths, Multiple dimensions.

Round 1: HNF versus Height

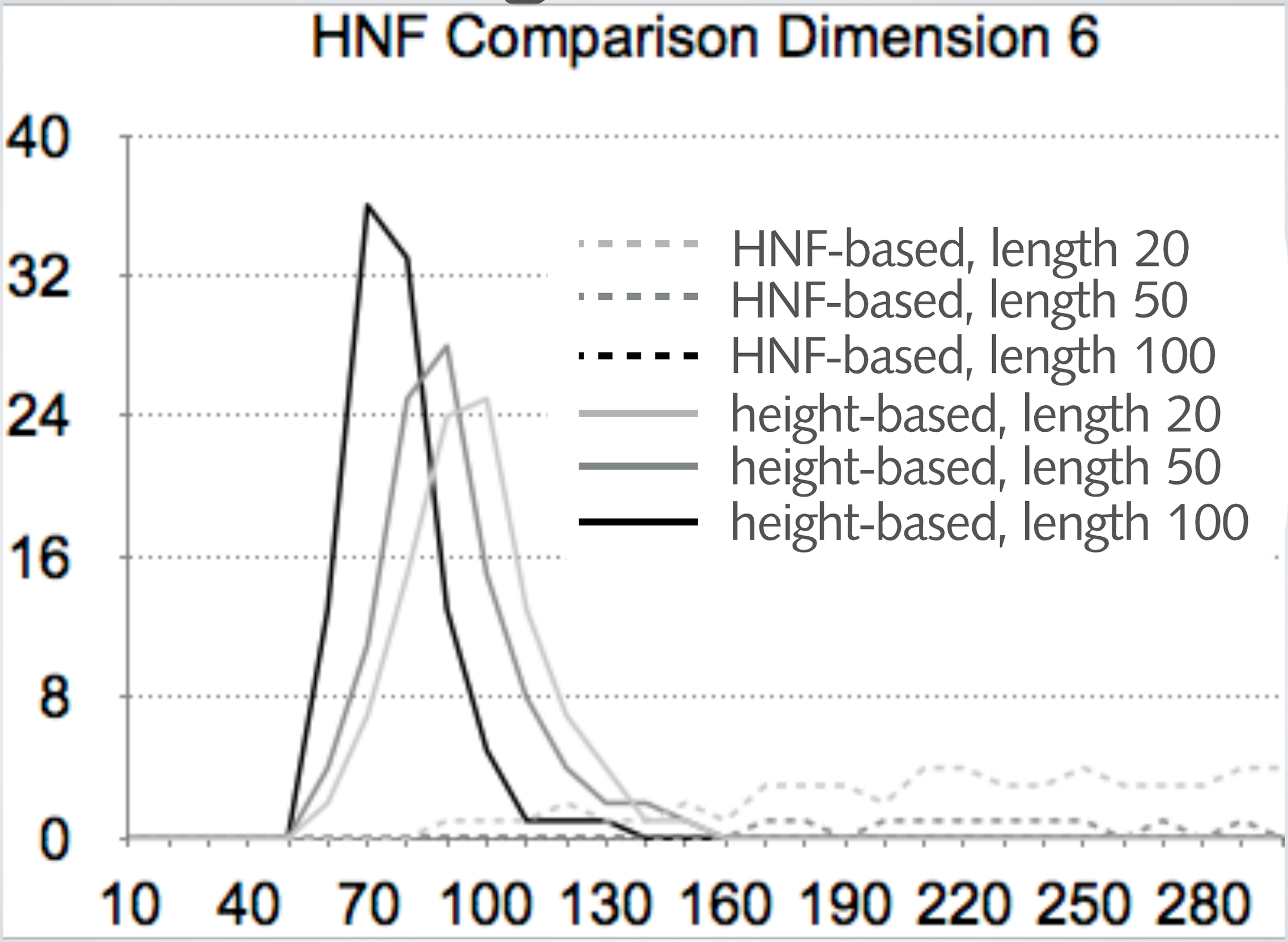
HNF vs. height, Dimension 4



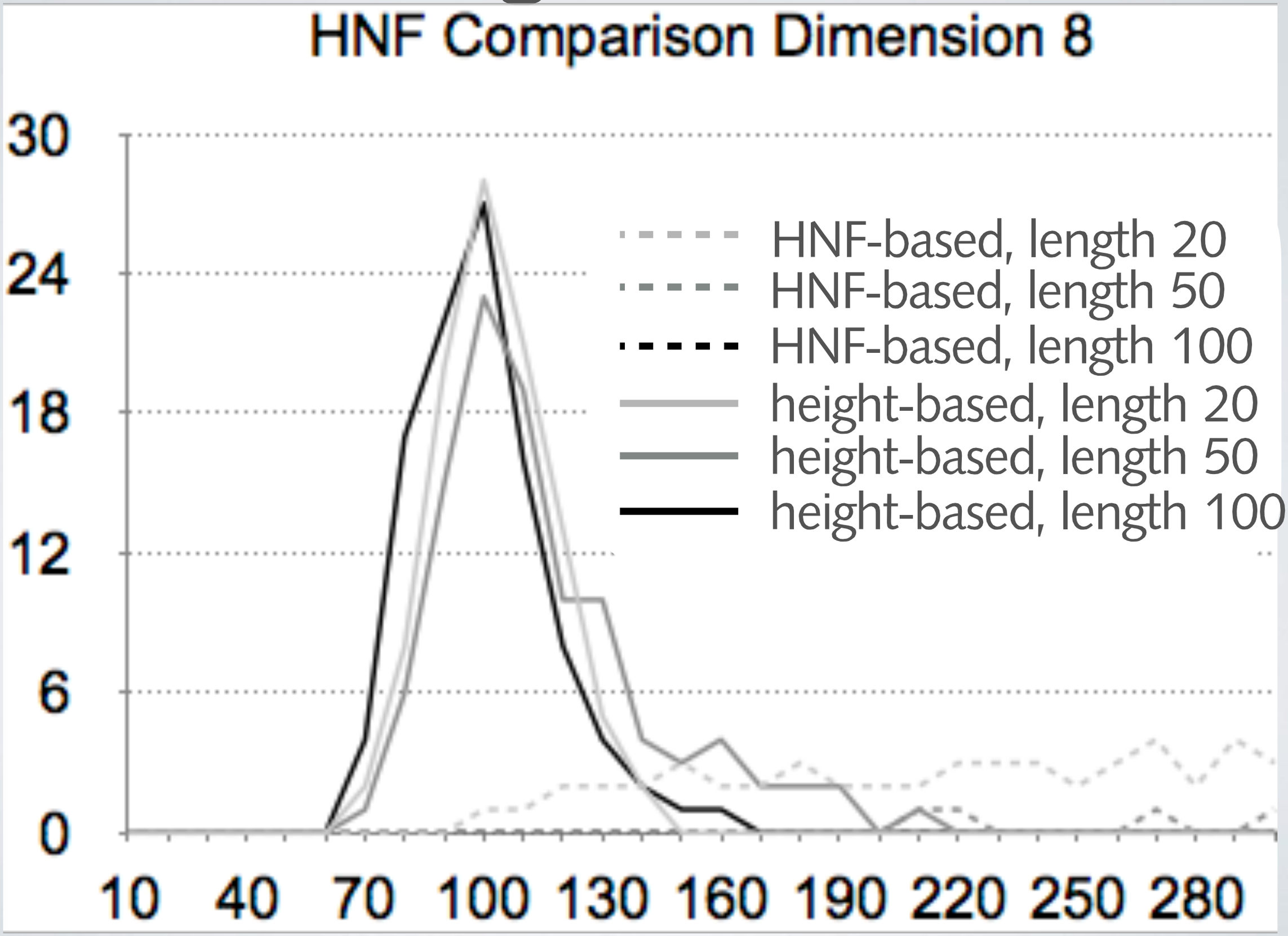
HNF vs. height, Dimension 5



HNF vs. height, Dimension 6

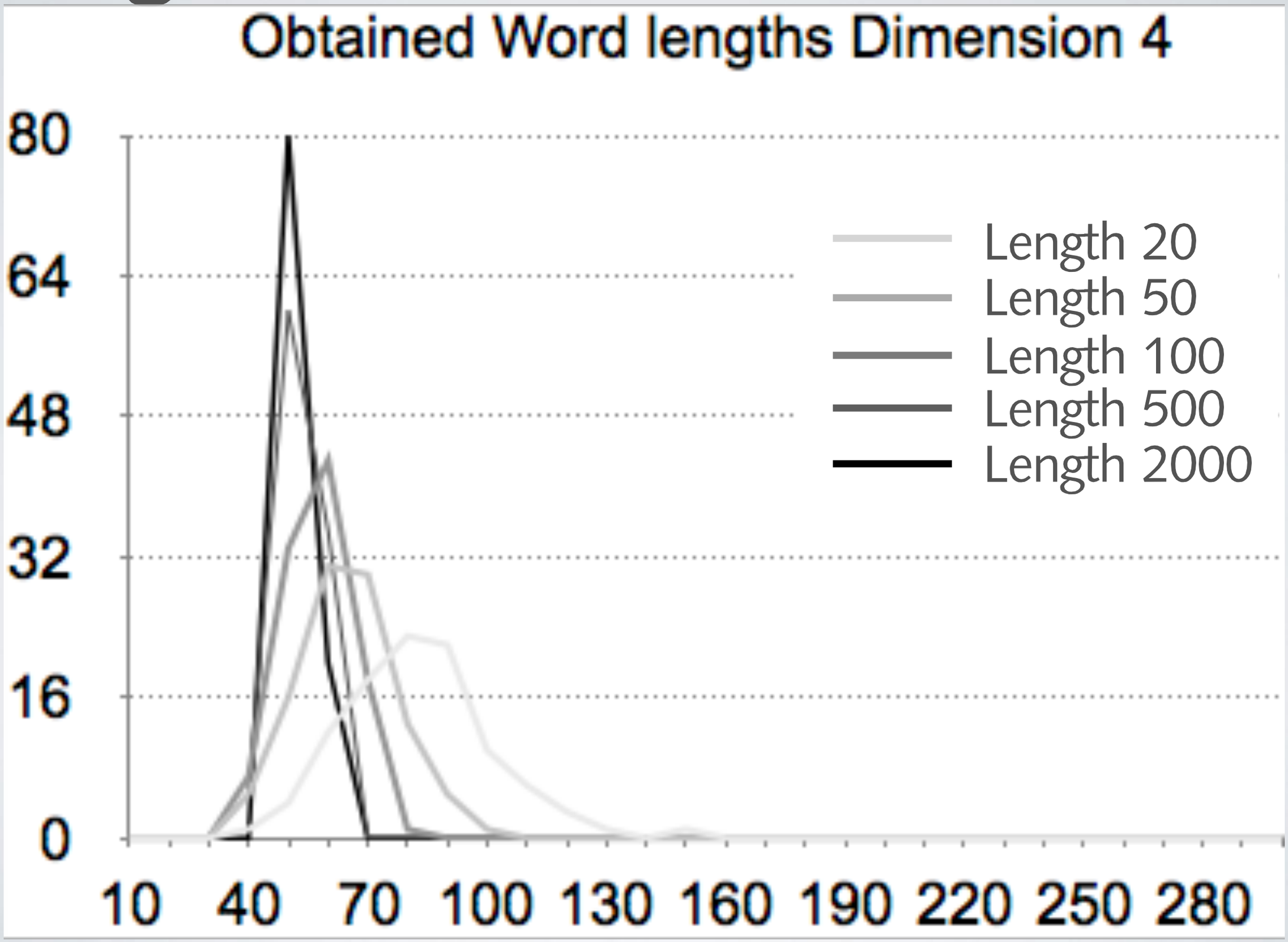


HNF vs. height, Dimension 8

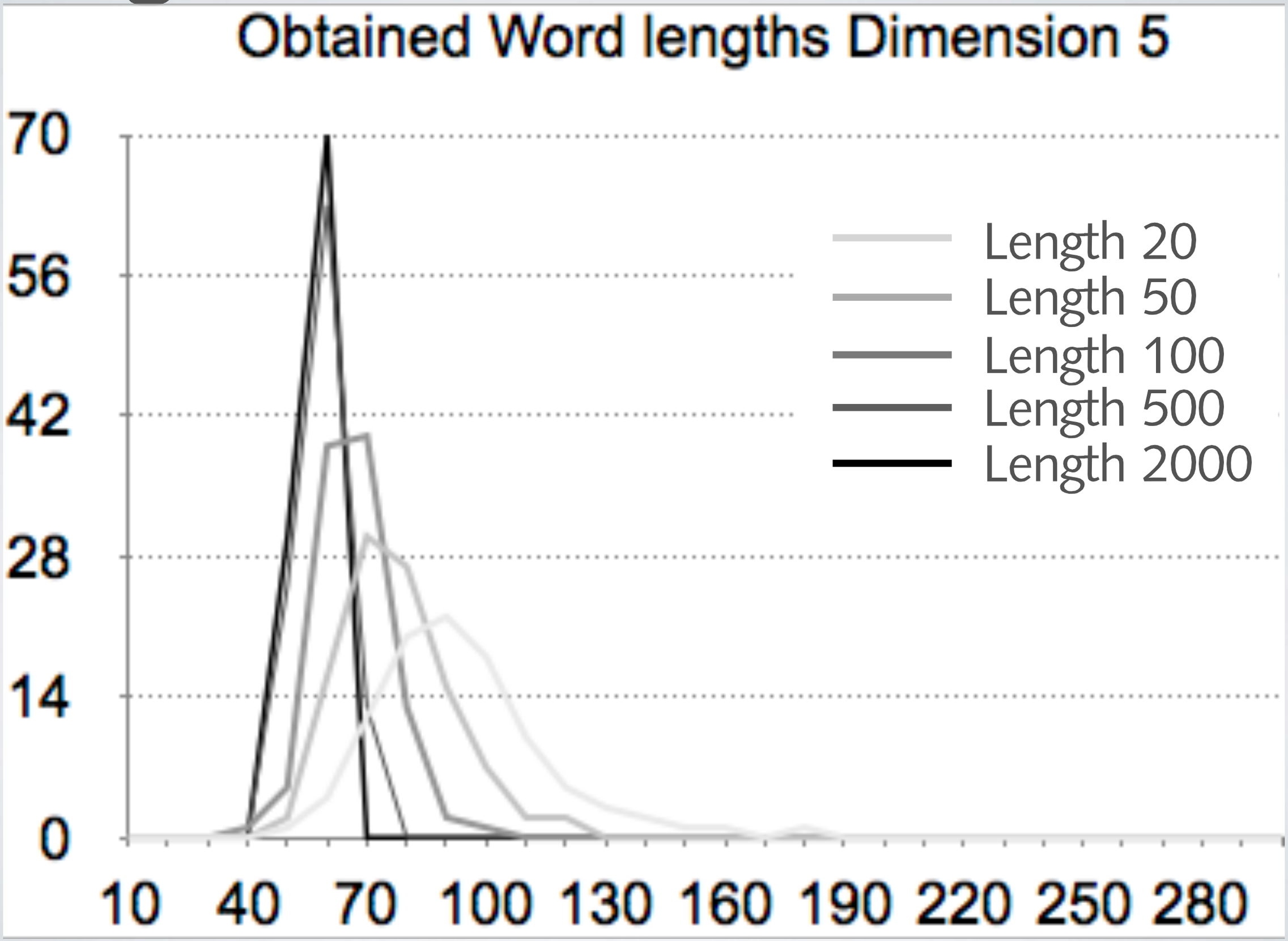


Round 2: Height for Longer Input Lengths

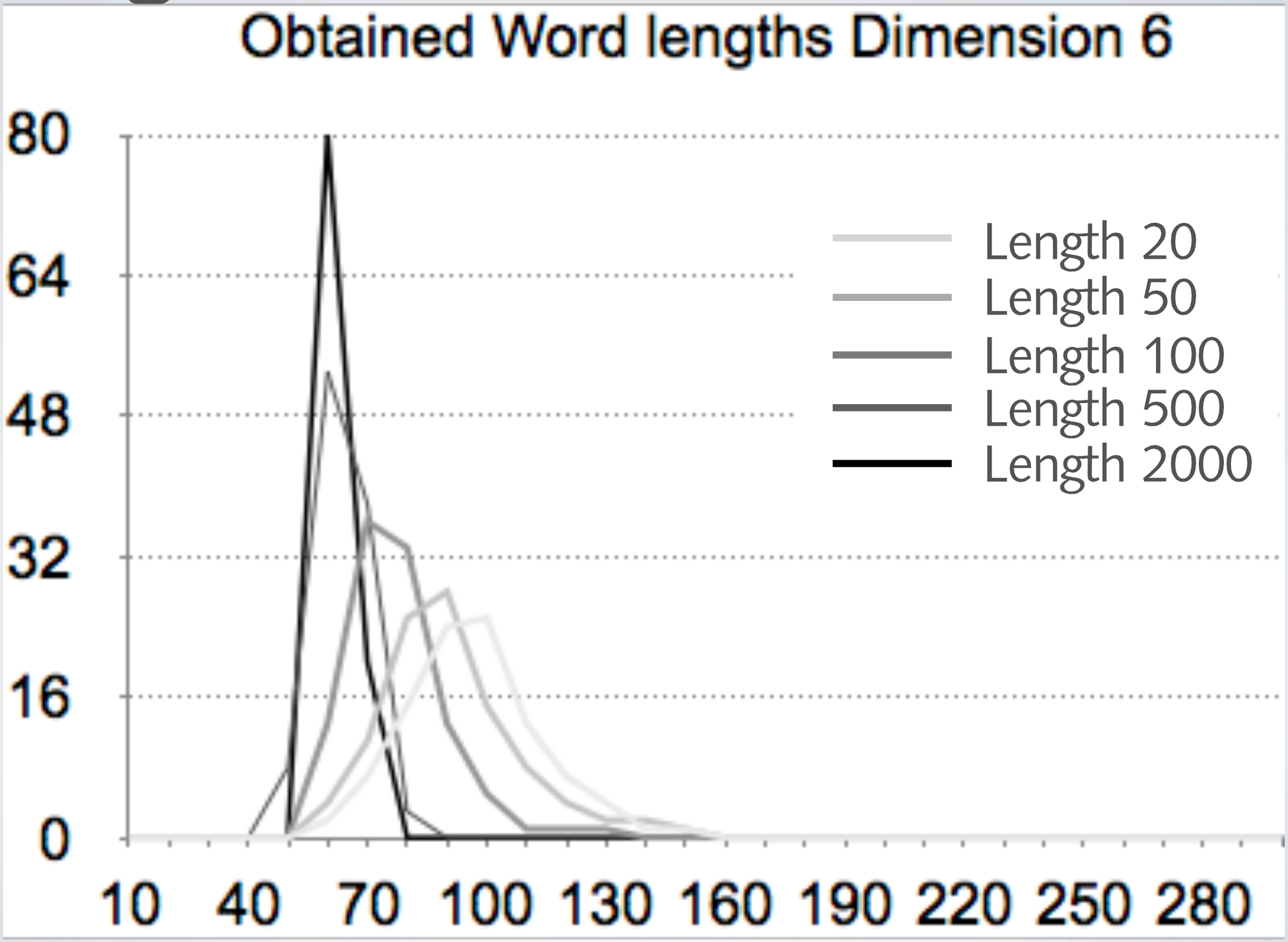
Height-based SL, Dimension 4



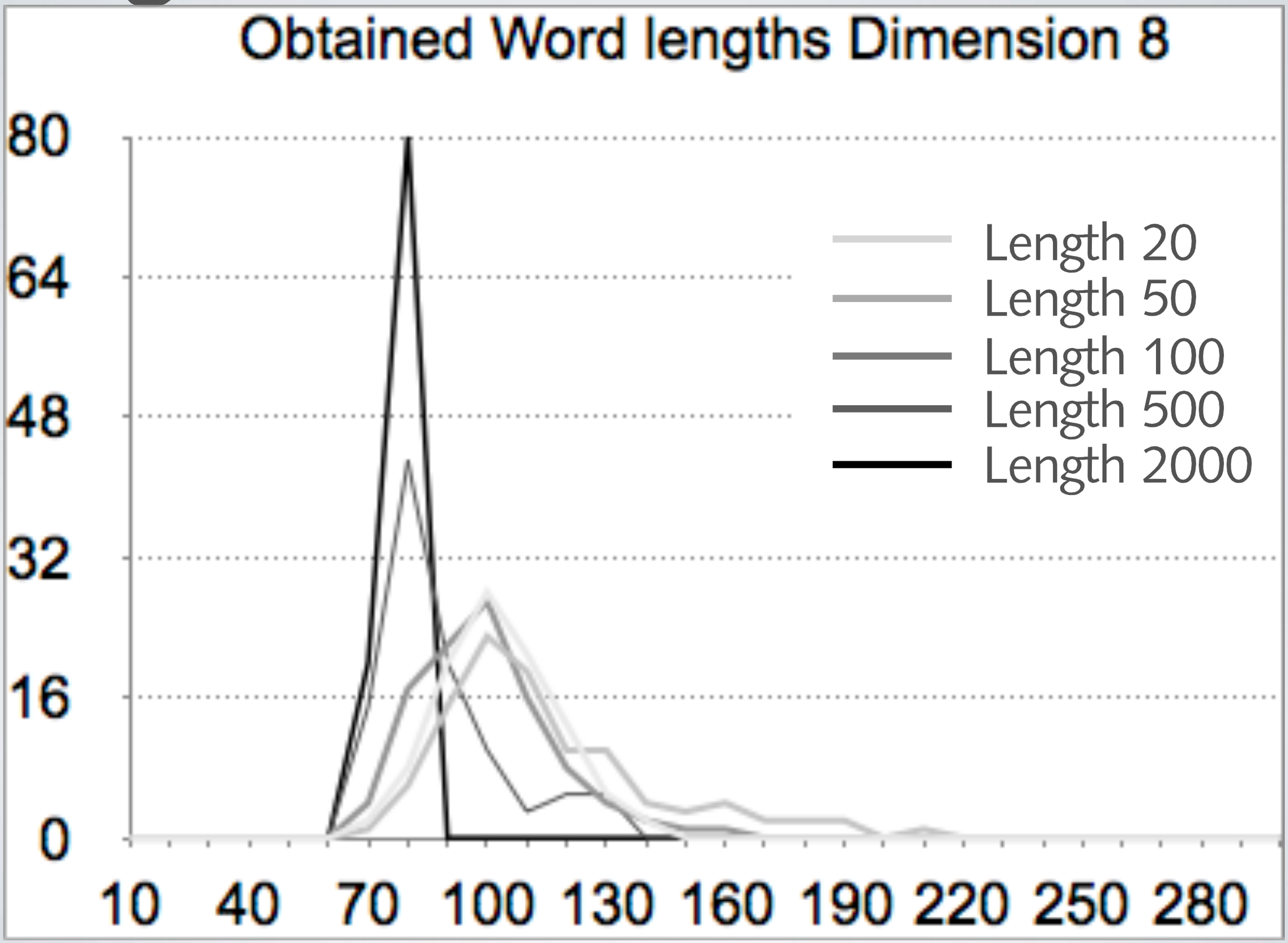
Height-based SL, Dimension 5



Height-based SL, Dimension 6

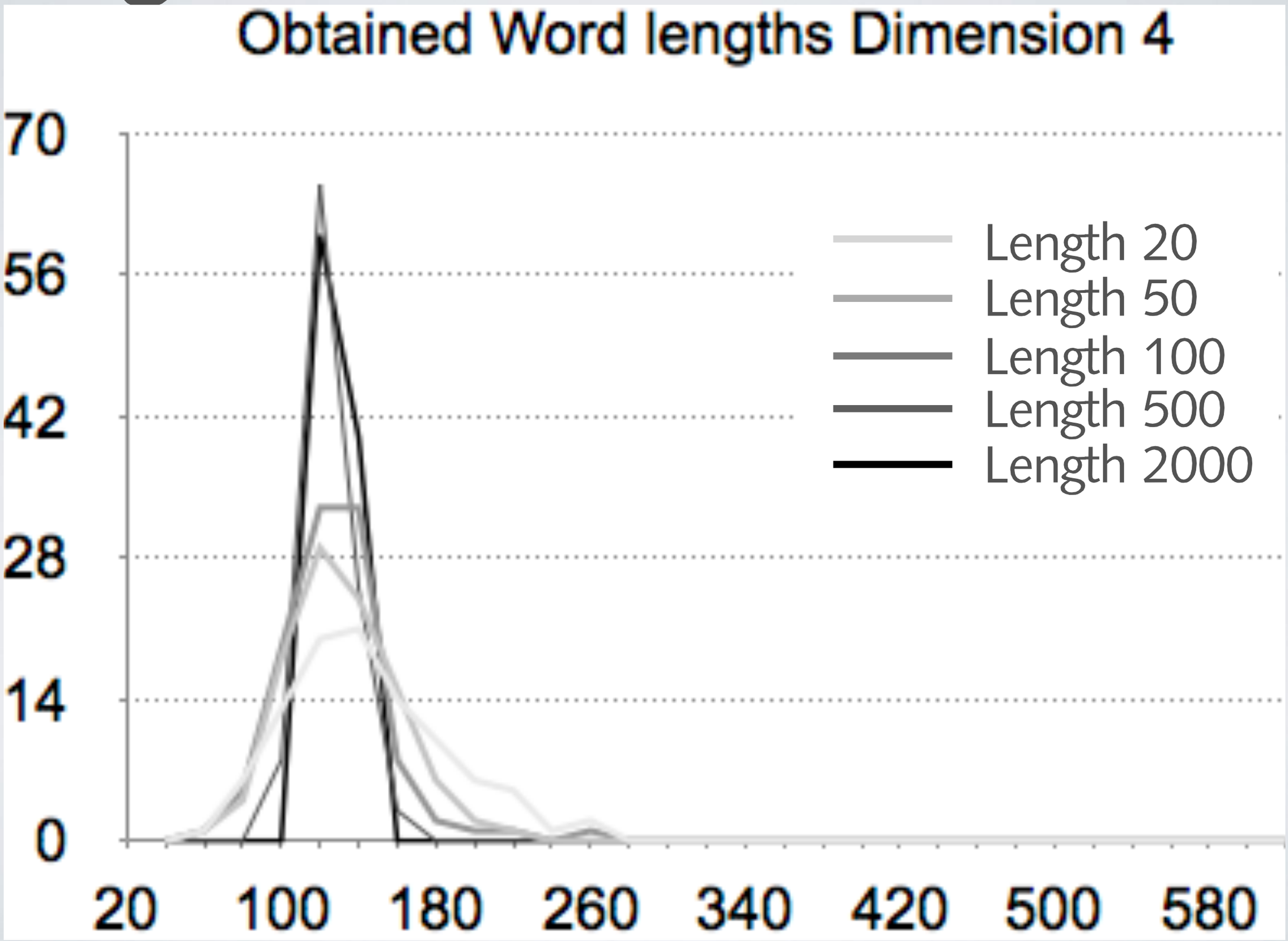


Height-based SL, Dimension 8

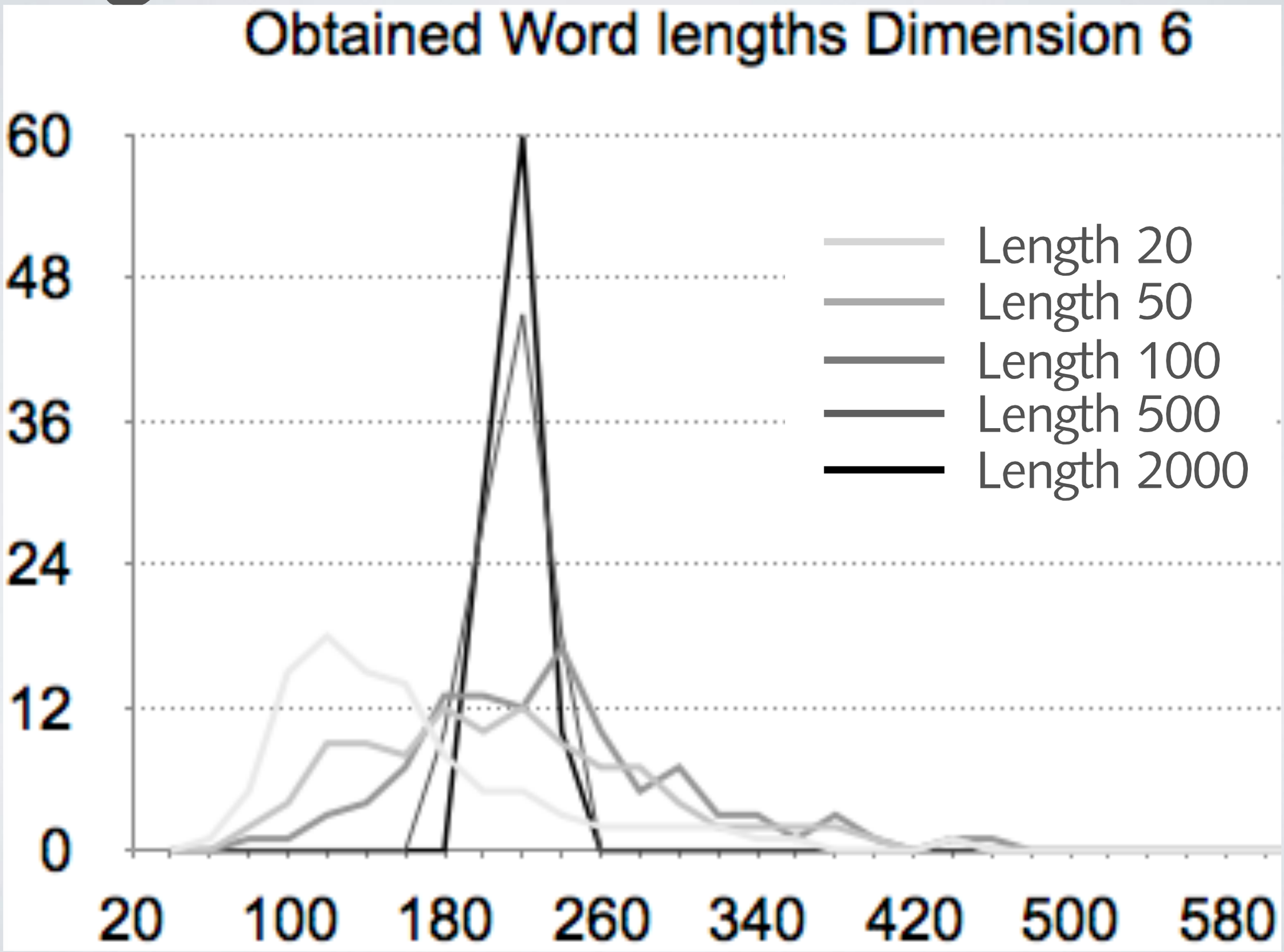


Round 3: Symplectic

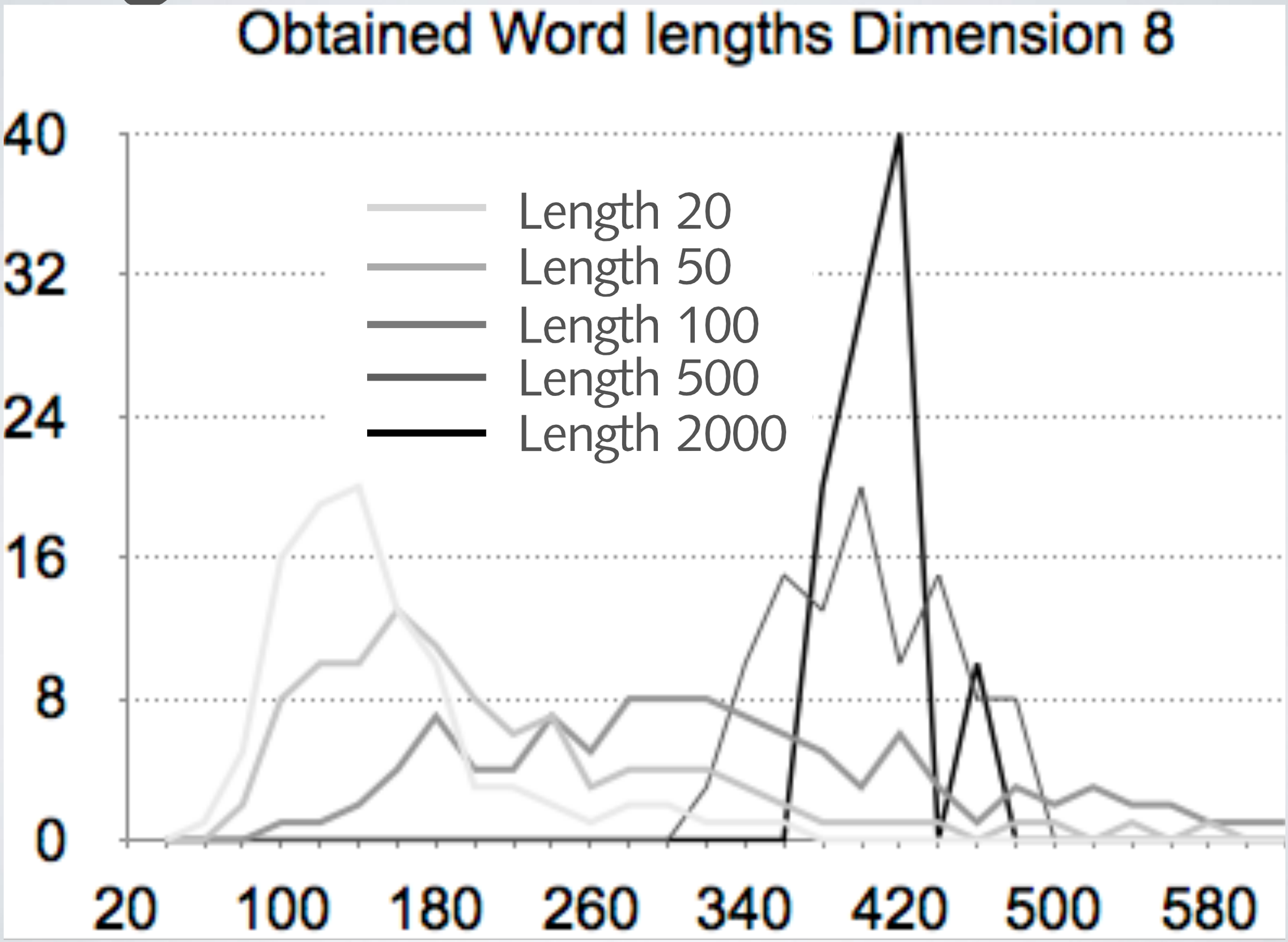
Height-Based SP, Dimension 4



Height-Based SP, Dimension 6



Height-Based SP, Dimension 8



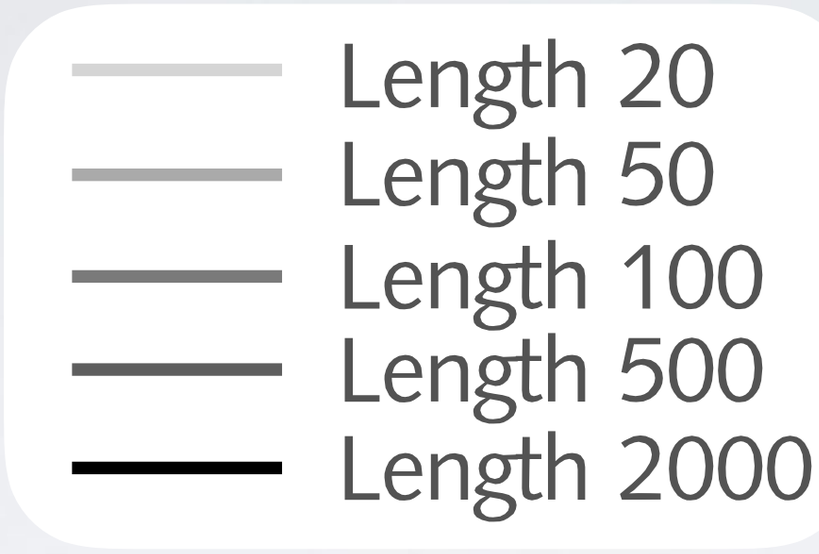
Closing Remarks

Can one *prove* that (or in which cases?) the algorithm succeeds for SP?

The norm-based approach very much requires working in characteristic 0. No help for finite characteristic.

HNF: Do short words correlate with small entries in transition matrices?

Better performance for SP (in larger dimensions)?


Length 20
Length 50
Length 100
Length 500
Length 2000