Exploiting New Properties of String Net Frequency for Efficient Computation

Peaker Guo, Patrick Eades, Anthony Wirth, and Justin Zobel
The University of Melbourne
CPM 2024, Fukuoka

Motivation

- Identification of significant strings in a text is a key task in many applications.

Motivation

- Identification of significant strings in a text is a key task in many applications.
- Frequency as a significance measure?

Motivation

- Identification of significant strings in a text is a key task in many applications.
- Frequency as a significance measure?
    - "th" is the most frequent bigram in English ...

Motivation

- Identification of significant strings in a text is a key task in many applications.
- Frequency as a significance measure?
  - "th" is the most frequent bigram in English ...
- Net frequency (NF) mitigates this limitation of frequency:

Motivation

- Identification of significant strings in a text is a key task in many applications.
- Frequency as a significance measure?
  - "th" is the most frequent bigram in English ...
- Net frequency (NF) mitigates this limitation of frequency:
  - the NF of "th" is zero in "<u>th</u>e <u>th</u>eoretical <u>th</u>eme".

Motivation

- Identification of significant strings in a text is a key task in many applications.
- Frequency as a significance measure?
  - "th" is the most frequent bigram in English ...
- Net frequency (NF) mitigates this limitation of frequency:
  - the NF of "th" is zero in "<u>th</u>e <u>th</u>eoretical <u>th</u>eme".
- NF was originally introduced for Chinese NLP tasks [LY01].

Motivation

- Identification of significant strings in a text is a key task in many applications.
- Frequency as a significance measure?
  - "th" is the most frequent bigram in English ...
- Net frequency (NF) mitigates this limitation of frequency:
  - the NF of "th" is zero in "the theoretical theme".
- NF was originally introduced for Chinese NLP tasks [LY01].
- There is a lack of understanding of the properties of NF and the absence of efficient algorithms for computing NF.

Our Contribution

- Reconceptualise NF and simplify its original definition.

Our Contribution

- Reconceptualise NF and simplify its original definition.
- Identify strings with positive NF in a Fibonacci word.

Our Contribution

- Reconceptualise NF and simplify its original definition.
- Identify strings with positive NF in a Fibonacci word.
- Introduce and solve two problems: SINGLE-NF and ALL-NF.

Our Contribution

- Reconceptualise NF and simplify its original definition.
- Identify strings with positive NF in a Fibonacci word.
- Introduce and solve two problems: SINGLE-NF and ALL-NF.
- Prove combinatorial bounds related to ALL-NF.

Our Redefinition of Net Frequency

Definition

The NF of a repeated string S in a text T, denoted as $\phi_T(S)$, is the number of occurrences of S in T that have unique left and right extensions. Such an occurrence is referred to as a *net occurrence*. The NF of a unique string is zero.

Our Redefinition of Net Frequency

Definition

The NF of a repeated string S in a text T, denoted as $\phi_T(S)$, is the number of occurrences of S in T that have unique left and right extensions. Such an occurrence is referred to as a *net occurrence*. The NF of a unique string is zero.

For example, consider $S = \texttt{st}$ and $T = \texttt{\#rstkstcastarstast\$}$.

Our Redefinition of Net Frequency

Definition

The NF of a repeated string S in a text T, denoted as $\phi_T(S)$, is the number of occurrences of S in T that have unique left and right extensions. Such an occurrence is referred to as a *net occurrence*. The NF of a unique string is zero.

$\texttt{\#r}\underline{\texttt{st}}\texttt{kstcastarstast\$}$

Our Redefinition of Net Frequency

Definition

The NF of a repeated string S in a text T, denoted as $\phi_T(S)$, is the number of occurrences of S in T that have unique left and right extensions. Such an occurrence is referred to as a *net occurrence*. The NF of a unique string is zero.

$\texttt{\#r}\underline{\texttt{st}}\texttt{kstcastarstast\$}$

Our Redefinition of Net Frequency

Definition

The NF of a repeated string S in a text T, denoted as $\phi_T(S)$, is the number of occurrences of S in T that have unique left and right extensions. Such an occurrence is referred to as a *net occurrence*. The NF of a unique string is zero.

$\texttt{\#r}\underline{\texttt{st}}\texttt{kstcastarstast\$}$

Our Redefinition of Net Frequency

Definition

The NF of a repeated string S in a text T, denoted as $\phi_T(S)$, is the number of occurrences of S in T that have unique left and right extensions. Such an occurrence is referred to as a *net occurrence*. The NF of a unique string is zero.

#rstkstcastarstast$

Our Redefinition of Net Frequency

Definition

The NF of a repeated string S in a text T, denoted as $\phi_T(S)$, is the number of occurrences of S in T that have unique left and right extensions. Such an occurrence is referred to as a *net occurrence*. The NF of a unique string is zero.

`#rstkstcastarstast$`

Our Redefinition of Net Frequency

Definition

The NF of a repeated string S in a text T, denoted as $\phi_T(S)$, is the number of occurrences of S in T that have unique left and right extensions. Such an occurrence is referred to as a *net occurrence*. The NF of a unique string is zero.

#r~~st~~kstca~~st~~ar~~st~~a~~st~~$

Our Redefinition of Net Frequency

Definition

The NF of a repeated string S in a text T, denoted as $\phi_T(S)$, is the number of occurrences of S in T that have unique left and right extensions. Such an occurrence is referred to as a *net occurrence*. The NF of a unique string is zero.

#r~~st~~kstca~~st~~ar~~st~~a~~st~~$

Definition

The NF of a repeated string S in a text T, denoted as $\phi_T(S)$, is the number of occurrences of S in T that have unique left and right extensions. Such an occurrence is referred to as a *net occurrence*. The NF of a unique string is zero.

$$\texttt{\#r}\cancel{\texttt{st}}\texttt{k}\underline{\texttt{st}}\texttt{ca}\cancel{\texttt{st}}\texttt{ar}\cancel{\texttt{st}}\texttt{a}\cancel{\texttt{st}}\texttt{\$}$$

Our Redefinition of Net Frequency

Definition

The NF of a repeated string S in a text T, denoted as $\phi_T(S)$, is the number of occurrences of S in T that have unique left and right extensions. Such an occurrence is referred to as a *net occurrence*. The NF of a unique string is zero.

$$\texttt{\#r}\cancel{\texttt{st}}\texttt{k}\underline{\texttt{st}}\texttt{ca}\cancel{\texttt{st}}\texttt{ar}\cancel{\texttt{st}}\texttt{a}\cancel{\texttt{st}}\texttt{\$}$$

Net frequency in Fibonacci words: A Case Study

i	f_i	F_i
1	1	b
2	1	a
3	2	ab
4	3	aba
5	5	abaab
6	8	abaababa

Let f_i be the i^{th} Fibonacci number. Let F_i be the i^{th} Fibonacci word, where $F_1 := \mathtt{b}, F_2 := \mathtt{a}$, and $F_i := F_{i-1}F_{i-2}$ for each $i \geq 3$. Note that $f_i := |F_i|$.

Net frequency in Fibonacci words: A Case Study

i	f_i	F_i
1	1	b
2	1	a
3	2	ab
4	3	aba
5	5	abaab
6	8	abaababa

Let f_i be the i^{th} Fibonacci number. Let F_i be the i^{th} Fibonacci word, where $F_1 := \texttt{b}, F_2 := \texttt{a}$, and $F_i := F_{i-1}F_{i-2}$ for each $i \geq 3$. Note that $f_i := |F_i|$.

Net frequency in Fibonacci words: A Case Study

i	f_i	F_i
1	1	b
2	1	a
3	2	ab
4	3	aba
5	5	abaab
6	8	abaababa

Let f_i be the i^{th} Fibonacci number. Let F_i be the i^{th} Fibonacci word, where $F_1 := \texttt{b}, F_2 := \texttt{a}$, and $F_i := F_{i-1}F_{i-2}$ for each $i \geq 3$. Note that $f_i := |F_i|$.

Net frequency in Fibonacci words: A Case Study

i	f_i	F_i
1	1	b
2	1	a
3	2	ab
4	3	aba
5	5	abaab
6	8	abaababa

Let f_i be the i^{th} Fibonacci number. Let F_i be the i^{th} Fibonacci word, where $F_1 := \texttt{b}, F_2 := \texttt{a}$, and $F_i := F_{i-1}F_{i-2}$ for each $i \geq 3$. Note that $f_i := |F_i|$.

Net frequency in Fibonacci words: A Case Study

i	f_i	F_i
1	1	b
2	1	a
3	2	ab
4	3	aba
5	5	abaab
6	8	abaababa

Let f_i be the i^{th} Fibonacci number. Let F_i be the i^{th} Fibonacci word, where $F_1 := \mathtt{b}, F_2 := \mathtt{a}$, and $F_i := F_{i-1}F_{i-2}$ for each $i \geq 3$. Note that $f_i := |F_i|$.

Strings with Positive NF in F_i

Empirically, only F_{i-2} and $F_{i-1}[1 \ldots f_{i-1} - 2]$ have positive NF in F_i, for each $i \geq 7$ until a reasonably large i.

Strings with Positive NF in F_i

Empirically, only F_{i-2} and $F_{i-1}[1 \ldots f_{i-1} - 2]$ have positive NF in F_i, for each $i \geq 7$ until a reasonably large i.

i	f_i	F_i
5	5	`abaab`
6	8	`abaababa`
7	13	`abaababa`<u>`abaab`</u>

Strings with Positive NF in F_i

Empirically, only F_{i-2} and $F_{i-1}[1 \ldots f_{i-1} - 2]$ have positive NF in F_i, for each $i \geq 7$ until a reasonably large i.

i	f_i	F_i
5	5	abaab
6	8	abaababa
7	13	abaababaabaab

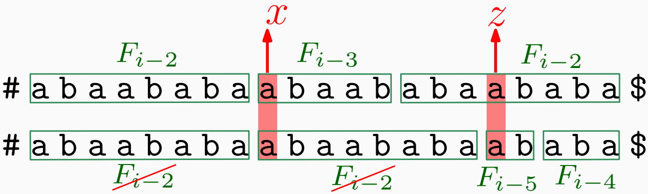
Factorization of F_i

We factorize F_i in two different ways. Take F_8 as an example.

$$\# \underbrace{\boxed{\texttt{a b a a b a b a}}}_{}^{F_{i-2}} \overbrace{\boxed{\texttt{a b a a b}}}^{F_{i-3}} \overbrace{\boxed{\texttt{a b a a b a b a}}}^{F_{i-2}} \$$$

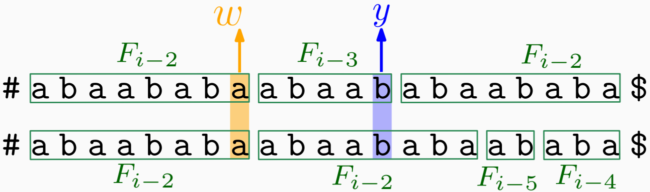
$$\# \underbrace{\boxed{\texttt{a b a a b a b a}}}_{F_{i-2}} \underbrace{\boxed{\texttt{a b a a b a b a}}}_{F_{i-2}} \underbrace{\boxed{\texttt{a b}}}_{F_{i-5}} \underbrace{\boxed{\texttt{a b a}}}_{F_{i-4}} \$$$

NF of F_{i-2} in F_i



$x = z$: the first character of each F_i is always ‘a’ for $i \geq 2$.

NF of F_{i-2} in F_i



$w \neq y$: the last character of consecutive Fibonacci words alternates.

NF of F_{i-2} in F_i

Lemma

Only the last occurrence of F_{i-2} is a net occurrence in F_i.

$$\#\,\texttt{a b a a b a b a a b a a b}\ \overset{F_{i-2}}{\boxed{\texttt{a b a a b a b a}}}\ \$$$

$$\#\,\underset{F_{i-2}}{\boxed{\texttt{a b a a b a b a}}}\ \underset{F_{i-2}}{\boxed{\texttt{a b a a b a b a}}}\ \texttt{a b a b a}\ \$$$

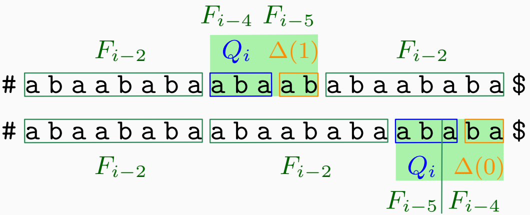
The Near-Commutative Property [Pir97]

- F_i is defined as $F_{i-1}\ F_{i-2}$, but what if we reverse the order of the concatenation?
- $F_{i-1}\ F_{i-2}$ and $F_{i-2}\ F_{i-1}$ only differ in the last two characters:

$$F_6\ F_5 = \texttt{abaababa|aba}\underline{\texttt{ab}}$$

$$F_5\ F_6 = \texttt{abaab|abaaba}\underline{\texttt{ba}}$$

NF of $F_{i-1}[1 \dots f_{i-1} - 2] = F_{i-2} Q_i$ in F_i



Lemma

$F_{i-4}\ F_{i-5} = Q_i\ \Delta\,(1 - (i \bmod 2))$ *and*

$F_{i-5}\ F_{i-4} = Q_i\ \Delta(i \bmod 2)$

… where $Q_i := F_{i-5}\ F_{i-6} \cdots F_3\ F_2$, $\Delta(0) :=$ ba, and $\Delta(1) :=$ ab.

Strings with Positive NF in F_i

Theorem

For each $i \geq 7$, $\phi_{F_i}(F_{i-2}) = 1$ and $\phi_{F_i}(F_{i-2}\ Q_i) = 2$.

Conjecture

There are only three net occurrences in F_i.

Idea: a net occurrence may overlap with another net occurrence, but cannot contain it entirely.

Net Frequency Computation

We consider the following computational problems:

- SINGLE-NF: process an input text; report the NF of a query string in the input text.

Net Frequency Computation

We consider the following computational problems:

- SINGLE-NF: process an input text; report the NF of a query string in the input text.
- ALL-NF: report an occurrence and the NF of each string of positive NF in an input text.

Net Frequency Computation

We consider the following computational problems:

- SINGLE-NF: process an input text; report the NF of a query string in the input text.
- ALL-NF: report an occurrence and the NF of each string of positive NF in an input text. For example, the output of ALL-NF on abaababaabaab is: $((1,6),2),((9,14),1)$.

SINGLE-NF Algorithm

Augment the suffix array (SA) with LCP array and LF mapping.

SINGLE-NF Algorithm

Augment the suffix array (SA) with LCP array and LF mapping.

Theorem

Let $\langle l, r \rangle$ be the SA interval of a string S. For each $i \in \langle l, r \rangle$, let $\ell(i) := \max(LCP[i], LCP[i+1])$, then, $(SA[i], SA[i] + |S| - 1)$ is a net occurrence if $|S| = \ell(i) \geq \ell(LF[i])$.

SINGLE-NF Algorithm

Augment the suffix array (SA) with LCP array and LF mapping.

Theorem

Let $\langle l, r\rangle$ be the SA interval of a string S. For each $i \in \langle l, r\rangle$, let $\ell(i) := \max(LCP[i], LCP[i+1])$, then, $(SA[i], SA[i] + |S| - 1)$ is a net occurrence if $|S| = \ell(i) \geq \ell(LF[i])$.

Proof idea: S is repeated if $|S| \leq \ell(i)$ and is unique if $|S| > \ell(i)$.

SINGLE-NF Algorithm (Example)

i	$T[SA[i]\dots n]\,T[1\dots SA[i]-1]$	$LCP[i]$	$\ell(i)$	$LF[i]$
...				
7	`kstcastarstast$#rst`	0	0	19
8	`rstast$#rstkstcasta`	0	3	3
9	`rstkstcastarstast$#`	3	3	1
10	`st$#rstkstcastarsta`	0	2	4
11	`starstast$#rstkstca`	2	3	5
12	`stast$#rstkstcastar`	3	3	8
13	`stcastarstast$#rstk`	2	2	7
14	`stkstcastarstast$#r`	2	2	9
...				

Remainder: $\ell(i) := \max(LCP[i], LCP[i+1])$

SINGLE-NF Algorithm (Example)

The SA interval of "st"

i	$T[SA[i]\ldots n]T[1\ldots SA[i]-1]$	$LCP[i]$	$\ell(i)$	$LF[i]$
...				
7	kstcastarstast$#rst	0	0	19
8	rstast$#rstkstcasta	0	3	3
9	rstkstcastarstast$#	3	3	1
10	st$#rstkstcastarsta	0	2	4
11	starstast$#rstkstca	2	3	5
12	stast$#rstkstcastar	3	3	8
13	stcastarstast$#rstk	2	2	7
14	stkstcastarstast$#r	2	2	9
...				

SINGLE-NF Algorithm (Example)

When $i = 13$, the right extension "stc" is unique:

i	$T[SA[i]...n]T[1...SA[i]-1]$	$LCP[i]$	$\ell(i)$	$LF[i]$
...				
7	kstcastarstast$#rst	0	0	19
8	rstast$#rstkstcasta	0	3	3
9	rstkstcastarstast$#	3	3	1
10	st$#rstkstcastarsta	0	2	4
11	starstast$#rstkstca	2	3	5
12	stast$#rstkstcastar	3	3	8
13	stcastarstast$#rstk	2	2	7
14	stkstcastarstast$#r	2	2	9
...				

Remainder: check for $|S| + 1 > \ell(i)$ where $\ell(i) := \max(LCP[i], LCP[i+1])$

SINGLE-NF Algorithm (Example)

The left extension "kst" is located using LF:

i	$T[SA[i] \ldots n] T[1 \ldots SA[i]-1]$	$LCP[i]$	$\ell(i)$	$LF[i]$
...				
7	kstcastarstast$#rst	0	0	19
8	rstast$#rstkstcasta	0	3	3
9	rstkstcastarstast$#	3	3	1
10	st$#rstkstcastarsta	0	2	4
11	starstast$#rstkstca	2	3	5
12	stast$#rstkstcastar	3	3	8
13	stcastarstast$#rstk	2	2	7
14	stkstcastarstast$#r	2	2	9
...				

SINGLE-NF Algorithm (Example)

The left extension "kst" is unique:

i	$T[SA[i]...n]T[1...SA[i]-1]$	$LCP[i]$	$\ell(i)$	$LF[i]$
...				
7	kstcastarstast$#rst	0	0	19
8	rstast$#rstkstcasta	0	3	3
9	rstkstcastarstast$#	3	3	1
10	st$#rstkstcastarsta	0	2	4
11	starstast$#rstkstca	2	3	5
12	stast$#rstkstcastar	3	3	8
13	stcastarstast$#rstk	2	2	7
14	stkstcastarstast$#r	2	2	9
...				

Remainder: check for $|S| + 1 > \ell(LF[i])$ where $\ell(i) := \max(LCP[i], LCP[i+1])$

Improved SINGLE-NF Algorithm

Lemma (Coloured range listing (CRL) [Mut02])

After processing a text T of length n in $O(n)$ time, given a range $i, \ldots, j$, the position of each distinct character ("colour") in $T[i \ldots j]$ can be listed in $O(1)$ time.

Improved SINGLE-NF Algorithm

Lemma (Coloured range listing (CRL) [Mut02])

After processing a text T of length n in $O(n)$ time, given a range $i, \ldots, j$, the position of each distinct character ("colour") in $T[i \ldots j]$ can be listed in $O(1)$ time.

For example, when $T = \texttt{b}\underline{\texttt{ana}}\texttt{na}$, $CRL_T(2, 4) = \{2, 3\}$.

Improved SINGLE-NF Algorithm

Lemma (Coloured range listing (CRL) [Mut02])

After processing a text T of length n in $O(n)$ time, given a range $i, \ldots, j$, the position of each distinct character ("colour") in $T[i \ldots j]$ can be listed in $O(1)$ time.

For example, when $T = \mathtt{ba}\underline{\mathtt{nana}}$, $CRL_T(3, 6) = \{3, 4\}$.

Improved SINGLE-NF Algorithm

Lemma (Coloured range listing (CRL) [Mut02])

After processing a text T of length n in $O(n)$ time, given a range $i, \ldots, j$, the position of each distinct character ("colour") in $T[i \ldots j]$ can be listed in $O(1)$ time.

i	$T[SA[i]...n]$ $T[1...SA[i]-1]$
...	
10	st$#rstkstcastarsta
11	starstast$#rstkstca
12	stast$#rstkstcastar
13	stcastarstast$#rstk
14	stkstcastarstast$#r
...	

$$CRL_{BWT}(10, 14) = \{10, 12, 13\}$$

SINGLE-NF Result

Theorem

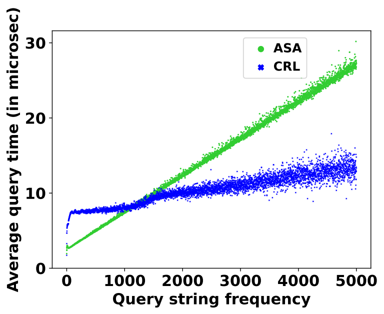
After processing a text T of length n in $O(n)$ time, the NF of a query string of length m can be computed in $O(m + d)$ time where d is the number of distinct left extension characters of S.

Experimental Results

ASA (augmented SA) vs CRL (ASA & coloured range listing):

$O(m + occ)$ vs $O(m + d)$

where m, occ, and d are the length, the number of occurrences and the number of distinct left extension characters of the query string, respectively.



ALL-NF-Related Bounds

Let Λ be the set of strings with positive NF in a text T; $n = |T|$.

ALL-NF-Related Bounds

Let Λ be the set of strings with positive NF in a text T; $n = |T|$.

Lemma

$\sum_{S\in\Lambda} \phi(S) \leq n$ *and* $|\Lambda| \leq n$.

ALL-NF-Related Bounds

Let Λ be the set of strings with positive NF in a text T; $n = |T|$.

Lemma

$\sum_{S \in \Lambda} \phi(S) \leq n$ *and* $|\Lambda| \leq n$.

Let $M := \sum_{S \in \Lambda} |S|$ and $L := \sum_{S \in \Lambda} \phi(S) \cdot |S|$.

ALL-NF-Related Bounds

Let Λ be the set of strings with positive NF in a text T; $n = |T|$.

Lemma

$\sum_{S \in \Lambda} \phi(S) \leq n$ *and* $|\Lambda| \leq n$.

Let $M := \sum_{S \in \Lambda} |S|$ and $L := \sum_{S \in \Lambda} \phi(S) \cdot |S|$.

When $T = \mathtt{abaababaabaab}$,

$\Lambda = \{\mathtt{abaaba}, \mathtt{abaab}\}$, $M = 6 + 5$, and $L = 6 \times 2 + 5 \times 1$.

ALL-NF-Related Bounds

Let Λ be the set of strings with positive NF in a text T; $n = |T|$.

Lemma

$\sum_{S \in \Lambda} \phi(S) \leq n$ *and* $|\Lambda| \leq n$.

Let $M := \sum_{S \in \Lambda} |S|$ and $L := \sum_{S \in \Lambda} \phi(S) \cdot |S|$.

Theorem

$M \in \Omega(n)$ *and* $L \in O(n \log \delta)$.

... where $\delta := \max \{S(k)/k : k \in [n]\}$, $S(k)$ is the # of distinct strings of length k.

Proof idea: sum of irreducible LCP values.

Conclusion and Future Work

Conclusion:

- Properties of NF:
  - redefinition, Fibonacci words, bounds (more in the paper)
- $O(m + d)$-time SINGLE-NF algorithm:
  - extensive experiments (more results in the paper)
- $O(n)$-time ALL-NF algorithm
  - two variants of ALL-NF (more details in the paper)

Conclusion and Future Work

Conclusion:

- Properties of NF:
  - redefinition, Fibonacci words, bounds (more in the paper)
- $O(m + d)$-time SINGLE-NF algorithm:
  - extensive experiments (more results in the paper)
- $O(n)$-time ALL-NF algorithm
  - two variants of ALL-NF (more details in the paper)

Future work:

- Only F_{i-2} and $F_{i-2}Q_i$ have positive NF in F_i
- Closing the gap of $\Omega(n) \leq M \leq L \leq O(n \log \delta)$
- A lower bound for SINGLE-NF
- Online computation of NF

Acknowledgement

The authors thank:

- William Umboh, now one of my supervisors, for his insightful discussions
- Anonymous reviewers for their valuable suggestions
- Hideo Bannai for bringing to our attention a paper on Fibonacci words during the summer school last week

References

[LY01] Yih-Jeng Lin and Ming-Shing Yu. Extracting Chinese frequent strings without dictionary from a Chinese corpus and its applications. *Journal of Information Science and Engineering*, 17(5):805–824, 2001.

[Mut02] S. Muthukrishnan. Efficient algorithms for document retrieval problems. In *Proceedings of the Thirteenth Annual ACM-SIAM Symposium on Discrete Algorithms, January 6-8, 2002, San Francisco, CA, USA*, pages 657–666. ACM/SIAM, 2002.

[Pir97] Giuseppe Pirillo. Fibonacci numbers and words. *Discrete Mathematics*, 173(1-3):197–207, 1997.

Thank you for your time! Questions?

Full paper including code: