Sequence Graphs Realizations

Sammy Khalife, Yann Ponty, Laurent Bulteau
LIX, Ecole Polytechnique

Séquences en Bioinformatique, Informatique et Mathématiques
Seqbim

23rd November, 2020

Sequence graphs: introduction

- In natural language, on the contrary of syntax, semantics is not formally defined

Sequence graphs: introduction

- In natural language, on the contrary of syntax, semantics is not formally defined
- Representations of words and textual documents are essential for several tasks (document classification, translation, role labelling, named entity recognition, ...)

Sequence graphs: introduction

- In natural language, on the contrary of syntax, semantics is not formally defined
- Representations of words and textual documents are essential for several tasks (document classification, translation, role labelling, named entity recognition, ...)
- Bag-Of-Words (BOW) representations:

Sequence graphs: introduction

- In natural language, on the contrary of syntax, semantics is not formally defined
- Representations of words and textual documents are essential for several tasks (document classification, translation, role labelling, named entity recognition, ...)
- Bag-Of-Words (BOW) representations:
  - Document similarity can be efficiently computed using sparse linear algebra.

Sequence graphs: introduction

- In natural language, on the contrary of syntax, semantics is not formally defined
- Representations of words and textual documents are essential for several tasks (document classification, translation, role labelling, named entity recognition, ...)
- Bag-Of-Words (BOW) representations:
  - Document similarity can be efficiently computed using sparse linear algebra.
  - The degree of ambiguity grows factorially on the size of the document.

Sequence graphs: introduction

- In natural language, on the contrary of syntax, semantics is not formally defined
- Representations of words and textual documents are essential for several tasks (document classification, translation, role labelling, named entity recognition, ...)
- Bag-Of-Words (BOW) representations:
  - Document similarity can be efficiently computed using sparse linear algebra.
  - The degree of ambiguity grows factorially on the size of the document.
- Co-occurence models (e.g. Graph of words (GOW)) supplement the content of BOW with statistics of co-occurrences, mitigating the degree of ambiguity.

Sequence graphs: introduction

- In natural language, on the contrary of syntax, semantics is not formally defined
- Representations of words and textual documents are essential for several tasks (document classification, translation, role labelling, named entity recognition, ...)
- Bag-Of-Words (BOW) representations:
  - Document similarity can be efficiently computed using sparse linear algebra.
  - The degree of ambiguity grows factorially on the size of the document.
- Co-occurence models (e.g. Graph of words (GOW)) supplement the content of BOW with statistics of co-occurrences, mitigating the degree of ambiguity.

Linux is not UNIX but Linux

Figure: Sequence graph (or Graph of Words) with $w = 3$

Sequence graphs: introduction

- In natural language, on the contrary of syntax, semantics is not formally defined
- Representations of words and textual documents are essential for several tasks (document classification, translation, role labelling, named entity recognition, ...)
- Bag-Of-Words (BOW) representations:
  - Document similarity can be efficiently computed using sparse linear algebra.
  - The degree of ambiguity grows factorially on the size of the document.
- Co-occurence models (e.g. Graph of words (GOW)) supplement the content of BOW with statistics of co-occurrences, mitigating the degree of ambiguity.

Linux is not UNIX but Linux

Figure: Sequence graph (or Graph of Words) with $w = 3$

Sequence graphs: introduction

- In natural language, on the contrary of syntax, semantics is not formally defined
- Representations of words and textual documents are essential for several tasks (document classification, translation, role labelling, named entity recognition, ...)
- Bag-Of-Words (BOW) representations:
  - Document similarity can be efficiently computed using sparse linear algebra.
  - The degree of ambiguity grows factorially on the size of the document.
- Co-occurence models (e.g. Graph of words (GOW)) supplement the content of BOW with statistics of co-occurrences, mitigating the degree of ambiguity.

Linux is not UNIX but Linux

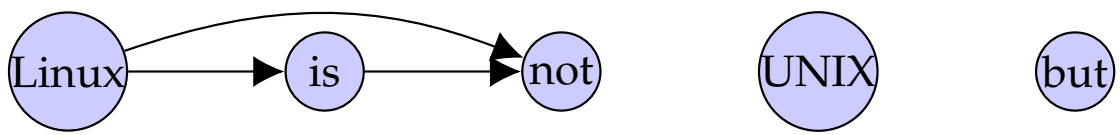


Figure: Sequence graph (or Graph of Words) with $w = 3$

Sequence graphs: introduction

- In natural language, on the contrary of syntax, semantics is not formally defined
- Representations of words and textual documents are essential for several tasks (document classification, translation, role labelling, named entity recognition, ...)
- Bag-Of-Words (BOW) representations:
  - Document similarity can be efficiently computed using sparse linear algebra.
  - The degree of ambiguity grows factorially on the size of the document.
- Co-occurence models (e.g. Graph of words (GOW)) supplement the content of BOW with statistics of co-occurrences, mitigating the degree of ambiguity.

Linux is not UNIX but Linux

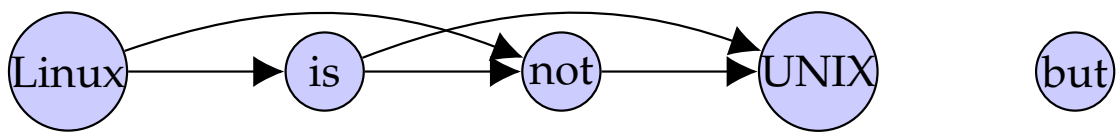


Figure: Sequence graph (or Graph of Words) with $w = 3$

Sequence graphs: introduction

- In natural language, on the contrary of syntax, semantics is not formally defined
- Representations of words and textual documents are essential for several tasks (document classification, translation, role labelling, named entity recognition, ...)
- Bag-Of-Words (BOW) representations:
  - Document similarity can be efficiently computed using sparse linear algebra.
  - The degree of ambiguity grows factorially on the size of the document.
- Co-occurence models (e.g. Graph of words (GOW)) supplement the content of BOW with statistics of co-occurrences, mitigating the degree of ambiguity.

Linux is not UNIX but Linux

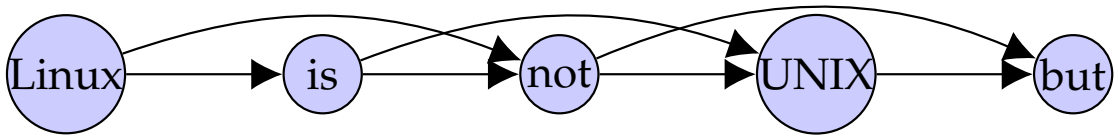


Figure: Sequence graph (or Graph of Words) with $w = 3$

Sequence graphs: introduction

- In natural language, on the contrary of syntax, semantics is not formally defined
- Representations of words and textual documents are essential for several tasks (document classification, translation, role labelling, named entity recognition, ...)
- Bag-Of-Words (BOW) representations:
  - Document similarity can be efficiently computed using sparse linear algebra.
  - The degree of ambiguity grows factorially on the size of the document.
- Co-occurence models (e.g. Graph of words (GOW)) supplement the content of BOW with statistics of co-occurrences, mitigating the degree of ambiguity.

Linux is not UNIX but Linux

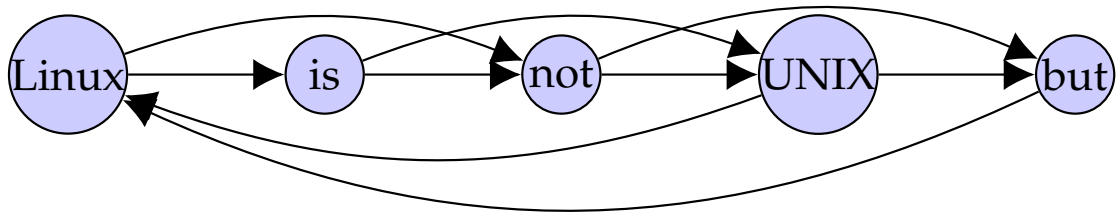


Figure: Sequence graph (or Graph of Words) with $w = 3$

Sequence graphs: introduction

- In natural language, on the contrary of syntax, semantics is not formally defined
- Representations of words and textual documents are essential for several tasks (document classification, translation, role labelling, named entity recognition, ...)
- Bag-Of-Words (BOW) representations:
  - Document similarity can be efficiently computed using sparse linear algebra.
  - The degree of ambiguity grows factorially on the size of the document.
- Co-occurence models (e.g. Graph of words (GOW)) supplement the content of BOW with statistics of co-occurrences, mitigating the degree of ambiguity.

Linux is not UNIX but Linux

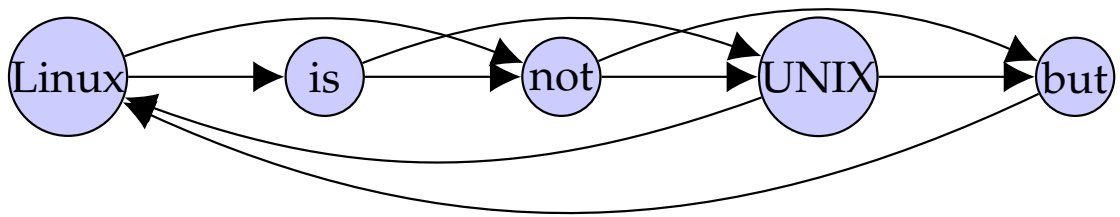


Figure: Sequence graph (or Graph of Words) with $w = 3$

Sequence graphs: introduction

- Several models (word2vec, Glove, fastText, Latent random walk) also use the same type of information, and allowed to increase performance for several tasks in natural language processing.
- Each word is attributed a representation using co-occurence statistics.
- Such models still induce ambiguity due to the fact that several sequences can *a priori* create the same graph.

Definition of a sequence graph

Let $x = x_1, x_2, ..., x_p$ be a sequence over some vocabulary $X = \{1, ..., n\}$.

Definition 1

$G = (V, E)$ is the graph of the sequence x with window size $w \in \mathbb{N}^*$ if and only if $V = \{x_i \mid i \in \{1, ..., p\}\}$, and

$$(i, j) \in E \iff \exists (k, k') \in I_p^2,\ |k - k'| \leq w - 1\ x_k = i \text{ and } x_{k'} = j$$

$$(i, j) \in E \iff i \text{ and } j \text{ “appears closer” than } w \text{ in } x$$

$$x \text{ is a } w\text{-realization of } G$$

Sequence graphs: definitions

- Definition can be generalized to digraphs and weighted graphs by counting the number of co-occurences.
- Can be computed in linear time

The defines a correspondence between the sequence set S_X into the graph set $\mathcal{G}$:

$$\phi_w : S_X \to \mathcal{G}, x \mapsto G_w(x)$$

To study the degree of ambiguity, we are interested in

$$\mathsf{Card}\,\mathsf{Im}\,\phi_w^{-1}(G)$$

When G is a given graph.

Sequence graphs: definitions

Problem 1 (Weighted-REALIZABLE (W-REALIZABLE))

Input: *Possibly directed graph G, matrix weights Π, window size w*
Output: *True if (G, Π) is the w-sequence graph of some sequence x, False otherwise.*

Problem 2 (Unweighted-REALIZABLE (U-REALIZABLE))

Input: *Possibly directed graph G, window size w*
Output: *True if G is the w-sequence graph of some sequence x, False otherwise.*

Sequence graphs: definitions

Problem 3 (Unweighted-NumRealizations (U-NumRealizations))

Input: *Possibly directed graph G, window size w*
Output: *The number of realizations of G, i.e. preimages of G through ϕ_w such that $|\{x \in X^\star \mid \phi_w(x) = G\}|$ if finite, or $+\infty$ otherwise.*

Problem 4 (Weighted-NumRealizations (W-NumRealizations))

Input: *Possibly directed graph G, matrix weights Π, window size w*
Output: *The number of realizations of G in the weighted sense.*

Sequence graphs: definitions

Prefix for the directed or undirected versions:

DW Directed weighted

GW Undirected weighted

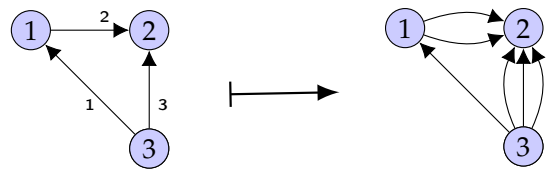
DU Directed unweighted

GU Undirected unweighted

2-sequence graphs

First results: The case $w = 2$ leads to simple characterization and algorithmic treatment:

- Weighted contraction $G \mapsto R^+(G)$ in a DAG of its strongly connected components.
- Eulerian reductions.



G $\qquad$ $\psi(G)$

Table: Complexity for various instances ($w = 2$)

	NumRealizations$_2$		Realizable$_2$	
Data Instance	Complexity	#Sequences	Complexity	Characterization
Unweighted graph	P	$\{0, +\infty\}$	P	G connected
Weighted graph	#P-hard	$\{0, 1\} \cup 2\mathbb{N}^*$	P	$\psi(G)$ (semi)Eulerian
Unweighted digraph	P	$\{0, 1, +\infty\}$	P	$R^+(G)$ straight line with unit weights
Weighted digraph	P	$\mathbb{N}$ (BEST Theorem)	P	$\psi(G)$ (semi)Eulerian

Results for $w \geq 3$

Contributions:

- Complexity results for all the variations of REALIZABLE and NUMREALIZATIONS$_w$
- Practical algorithms when w is fixed.

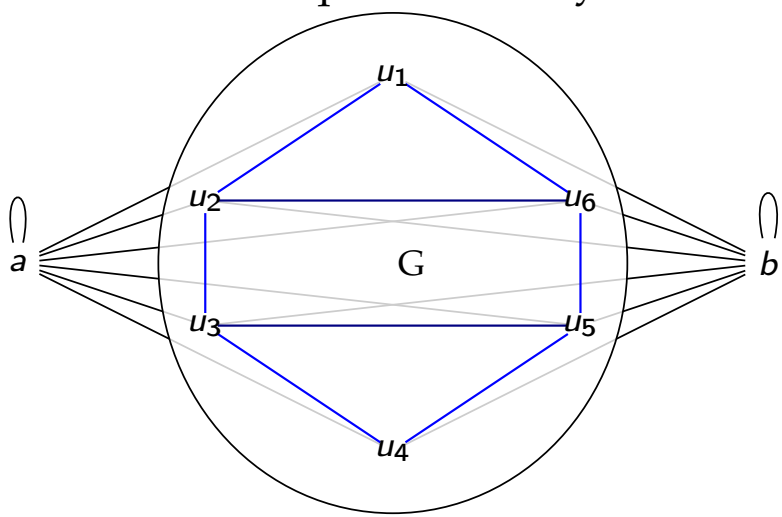
Complexity results for $w \geq 3$

Proposition 1

CLIQUE *admits a polynomial time parameterized reduction into* GU-REALIZABLE.

Proof.

Let $G = (V, E)$ be a simple graph. Let G' be a graph constructed from G adding two nodes a and b with loops, such that a and b are connected to each vertex of G. Let k be a strictly positive integer and $w = k + 1$. Then G has a k-clique if and only if G' is w-realizable.



Results for $w \geq 3$

Similarly, using hamiltonian (non-trivial) reductions:

- DU-REALIZABLE$_w$ is NP-hard for any $w \geq 3$
- GW-REALIZABLE$_w$, DW-REALIZABLE$_w$ are NP-hard for any $w \geq 3$

which allow to characterize the complexity of our problems:

Table: Complexity for various instances of our problems ($w \geq 3$)

Variation	NUMREALIZATIONS$_w$ Complexity	REALIZABLE$_w$ Complexity	NUMREALIZATIONS Complexity	REALIZABLE Complexity
GU	?	?	W[1]-hard	W[1]-hard
GW	NP-hard $\forall w \geq 3$	NP-hard $\forall w \geq 3$	NP-hard	NP-hard
DU	NP-hard $\forall w \geq 3$	NP-hard $\forall w \geq 3$	W[1]-hard	W[1]-hard
DW	NP-hard $\forall w \geq 3$	NP-hard $\forall w \geq 3$	NP-hard	NP-hard

Practical algorithms for $w \geq 3$

We propose two practical algorithms:

- a) A direct recursive formulation for REALIZABLE_w in $O(n^w)$
- b) A dynamic programming formulation for NUMREALIZATIONS_w (exponential in worst-case scenario)

General case: direct formulation

- If $G = (V, E)$ then

$$H(G) = (E, H_E) \quad \text{such that}$$

$$\forall e = (v_1, v_2) \in E \quad \text{and} \quad \forall f = (v_3, v_4) \in E$$

$$(e, f) \in H_E \iff v_2 = v_3 \ \text{ and } \ (v_1, v_4) \in E$$

$H(G)$ is the ***adjoint graph*** of G ("line graph") for which some edges have been deleted.

$\rightarrow$ If $w = 3$, a realization of G is a walk in $H(G)$

Practical algorithm: direct formulation

Idea: For $w > 3$, recursively iterate the process for $k \leq w - 2$:

$$E^{(k)} = \{u_{1:(k+1)} \in V^{k+1} \mid u_{1:k} \in E^{(k-1)},\ u_{2:(k+1)} \in E^{(k-1)} \\ \text{s.t}(u_1, u_{k+1}) \in E\}$$

$$H^{(k)} \triangleq (E^{(k)}, E^{(k+1)})$$

GU-REALIZABLE $\in$ XP

Proposition 2

Let $x = x_1, ..., x_p$ be a w-realization of a graph $G = (V, E)$. If $w \leq p$, x, then x is an authentic sequence of a walk of length $p - w + 1$ on $H^{(w-2)}$.

$\rightarrow$ Proof by induction on $k \in \{1, ...p\}$

Theorem 2

GU-REALIZABLE $\in$ *XP (or* GU-REALIZABLE$_w \in P$*)*

$\rightarrow$ Proof: Extract connected components of $H^{(w-2)}$. For each component, there exists a walk covering all vertices at least once. A potential realization cannot generate more edges than these walks. Use the definition to compute the realizations and compare.

Complexity results: summary

Table: Complexity for various instances of our problems ($w \geq 3$)

Variation	NumRealizations$_w$ Complexity	Realizable$_w$ Complexity	NumRealizations Complexity	Realizable Complexity
GU	P $\forall w \geq 3$	P $\forall w \geq 3$	W[1]-hard	W[1]-hard
GW	NP-hard $\forall w \geq 3$	NP-hard $\forall w \geq 3$	NP-hard	NP-hard
DU	NP-hard $\forall w \geq 3$	NP-hard $\forall w \geq 3$	W[1]-hard	W[1]-hard
DW	NP-hard $\forall w \geq 3$	NP-hard $\forall w \geq 3$	NP-hard	NP-hard

General case: dynamic programming formulation

Recursion

$$N_w\left[\Pi, p, \mathbf{u}\right] = \sum_{v \in V} \Phi\left(\Pi'_{(\mathbf{u},v)}, p-1, \mathbf{u}\right)$$

General case: dynamic programming formulation

Recursion

$$N_w\left[\Pi, p, \mathbf{u}\right] = \sum_{v \in V} \Phi\left(\Pi'_{(\mathbf{u},v)}, p-1, \mathbf{u}\right)$$

$$\Phi(\Pi'_{(\mathbf{u},v)}, p-1, \mathbf{u}) = \begin{cases} N_w\left[\Pi'_{(\mathbf{u},v)}, p-1, (u_1, ..., u_{|u|}, v)\right] & \text{if } |\mathbf{u}| < w-1 \\ N_w\left[\Pi'_{(\mathbf{u},v)}, p-1, (u_2, ..., u_{w-1}, v)\right] & \text{if } |\mathbf{u}| = w-1 \end{cases}$$

General case: dynamic programming formulation

Recursion

$$N_w\left[\Pi, p, \mathbf{u}\right] = \sum_{v \in V} \Phi\left(\Pi'_{(\mathbf{u},v)}, p-1, \mathbf{u}\right)$$

$$\Phi(\Pi'_{(\mathbf{u},v)}, p-1, \mathbf{u}) = \begin{cases} N_w\left[\Pi'_{(\mathbf{u},v)}, p-1, (u_1, ..., u_{|u|}, v)\right] & \text{if } |\mathbf{u}| < w-1 \\ N_w\left[\Pi'_{(\mathbf{u},v)}, p-1, (u_2, ..., u_{w-1}, v)\right] & \text{if } |\mathbf{u}| = w-1 \end{cases}$$

Π' is an update of Π following:

$$\Pi'_{(\mathbf{u},v)} := \left(\pi_{ij} - \#\left\{k \in \{1, ..., |\mathbf{u}|\} \mid (u_k, v) = (i,j)\right\}\right)_{(i,j) \in V^2}$$

General case: dynamic programming formulation

Initialisation

$$\forall\ \Pi,\ \ N_w[\Pi, 0, \mathbf{u}] = \begin{cases} 1 & \text{if } \Pi = (0)_{(i,j)\in V^2} \\ 0 & \text{otherwise.} \end{cases}$$

The total number of admissible sequences is $N_w[\Pi, p, \varepsilon]$, ε being the empty prefix.

Relation with Language Models

Experiment: Ambiguity w.r.t window size

Relation with Language Models

Experiment: Ambiguity w.r.t window size

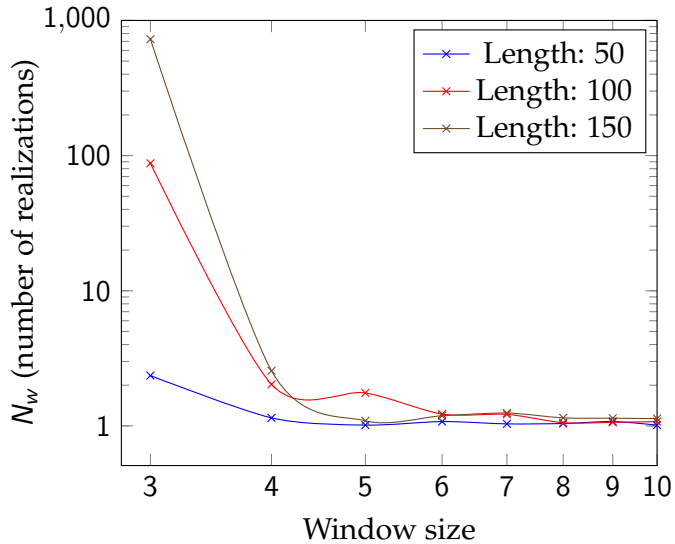


Figure: Estimation of the average number of realizations N_w.
500 documents of English Wikipedia, 2016.

Relation with Language Models

Experiment: Ambiguity w.r.t window size

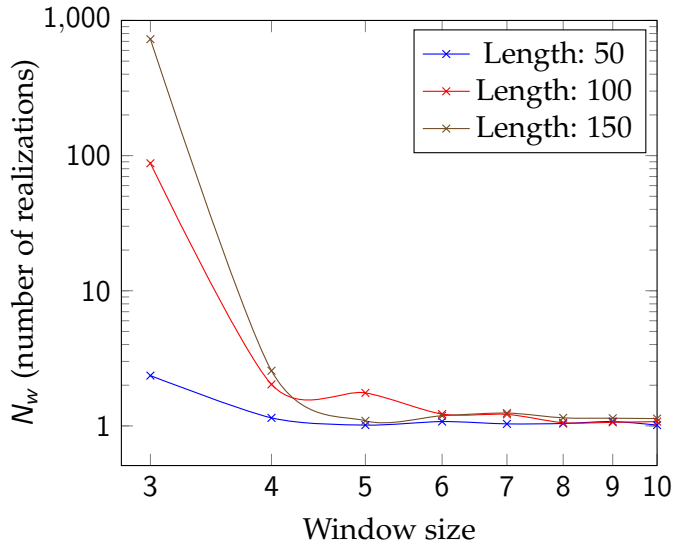


Figure: Estimation of the average number of realizations N_w.
500 documents of English Wikipedia, 2016.

- Average number of distinct realizations tends to 1 w.r.t window size.

Relation with Language Models

Experiment: Ambiguity w.r.t window size

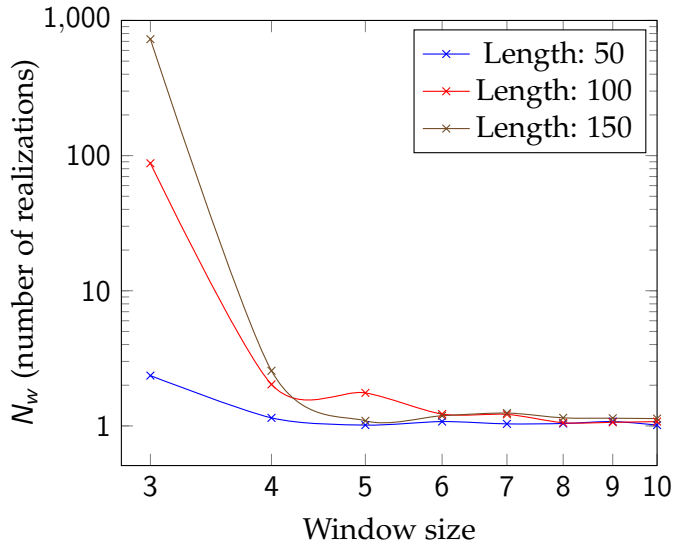


Figure: Estimation of the average number of realizations N_w.
500 documents of English Wikipedia, 2016.

- Average number of distinct realizations tends to 1 w.r.t window size.
- Different realizations exist, even for $w = 10$

Relation with Language Models

Experiment: Ambiguity w.r.t window size

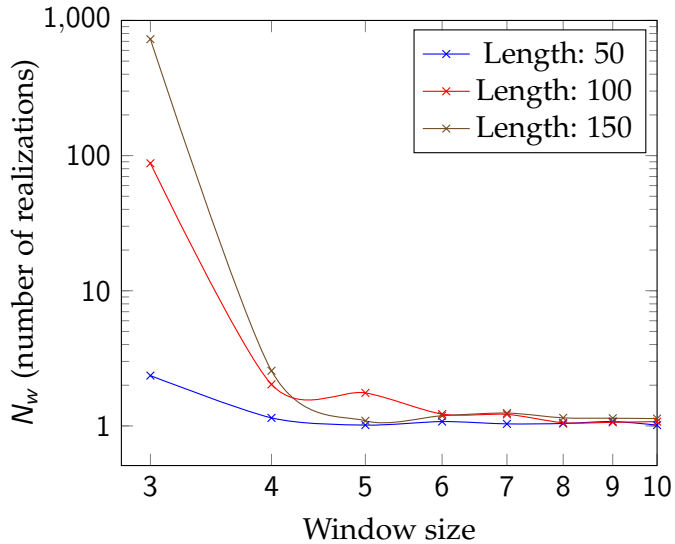


Figure: Estimation of the average number of realizations N_w.
500 documents of English Wikipedia, 2016.

- Average number of distinct realizations tends to 1 w.r.t window size.
- Different realizations exist, even for $w = 10$
- Recommendation of window size w.r.t document length.

Conclusion

- The ambiguity of graph-of-word representations can be formulated as an inverse problem
- We characterized the complexity of the corresponding inverse problems
- Dynamic programming formulation still suffers from exponential complexity but good in practice for relatively large instances ($p \leq 500, w \leq 10$)
- Near future: compare ambiguity of these graph-of-word representations with other representations such as those in neural networks.