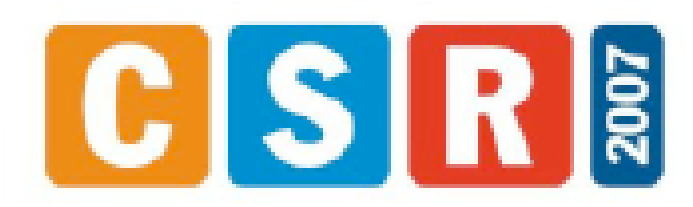
# Maximal Intersection Queries in Randomized Graph Models

**Benjamin Hoffmann**[1] Yury Lifshits[2] Dirk Nowotka[1]

[1]University of Stuttgart

[2]Steklov Institute of Mathematics at St. Petersburg

2nd International Computer Science Symposium in Russia 2007



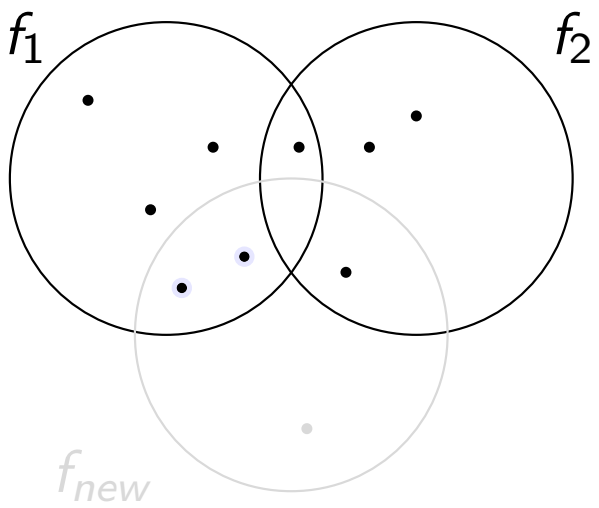
# The Maximal Intersection Problem

**The Maximal Intersection Problem (MaxInt)**

*Database:* A family $\mathcal{F}$ of $n$ sets, $|f| \leq k \ \forall f \in \mathcal{F}$.

*Query:* Given a set $f_{new}$ with $|f_{new}| \leq k$, return $f_i \in \mathcal{F}$ with maximal $|f_{new} \cap f_i|$.

*Constraints:* Preprocessing time $n \cdot polylog(n) \cdot poly(k)$.
Query time $polylog(n) \cdot poly(k)$.



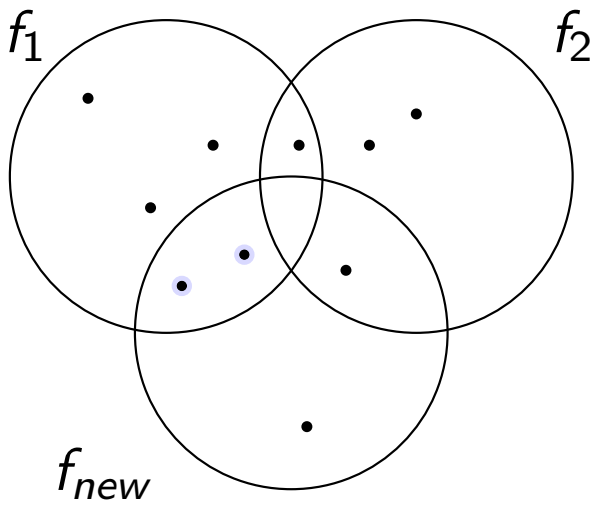
# The Maximal Intersection Problem

**The Maximal Intersection Problem (MaxInt)**

*Database:* A family $\mathcal{F}$ of $n$ sets, $|f| \leq k \ \forall f \in \mathcal{F}$.

*Query:* Given a set $f_{new}$ with $|f_{new}| \leq k$, return $f_i \in \mathcal{F}$ with maximal $|f_{new} \cap f_i|$.

*Constraints:* Preprocessing time $n \cdot polylog(n) \cdot poly(k)$.
Query time $polylog(n) \cdot poly(k)$.



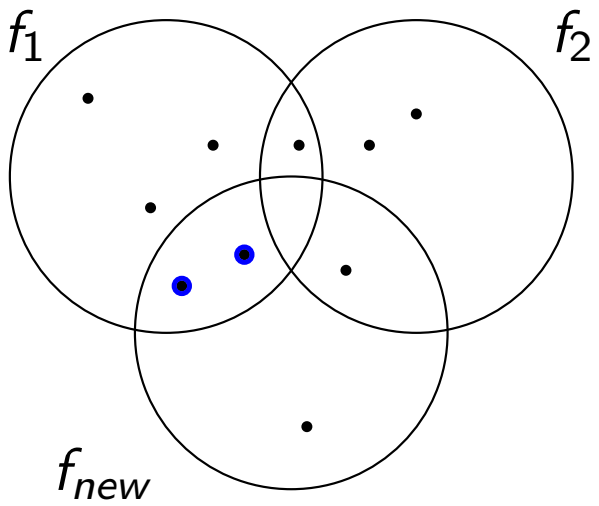
# The Maximal Intersection Problem

**The Maximal Intersection Problem (MaxInt)**

*Database:* A family $\mathcal{F}$ of $n$ sets, $|f| \leq k \ \forall f \in \mathcal{F}$.

*Query:* Given a set $f_{new}$ with $|f_{new}| \leq k$, return $f_i \in \mathcal{F}$ with maximal $|f_{new} \cap f_i|$.

*Constraints:* Preprocessing time $n \cdot polylog(n) \cdot poly(k)$.
Query time $polylog(n) \cdot poly(k)$.



# Motivation

**Possible applications of MaxInt**:

- *Advertisement matching*: Which website should an advertisement be placed on (e.g. Google AdSense)?
- *Text clustering/classification*: Find a document in a database that has a maximal number of common terms with a newcomer document (e.g. Reuters).
- *Recommendation systems, near-duplicate detection, code plagiarism detection, search engines, ...*

# Relation to Nearest Neighbor

**Nearest Neighbor Problem**: Determine in a general metric space a point that is closest to a given query point (Zezula et al., *Similarity Search - The Metric Space Approach, Springer, 2006*).

**MaxInt**: special case of Nearest Neighbor.

Similarity: size of intersection.

# Relation to Nearest Neighbor

**Nearest Neighbor Problem**: Determine in a general metric space a point that is closest to a given query point (Zezula et al., *Similarity Search - The Metric Space Approach, Springer, 2006*).

**MaxInt**: special case of Nearest Neighbor.

Similarity: size of intersection.

# Main result

**Assumption**: Input is taken from some predefined distribution.

There exists an algorithm that finds with very high probability an almost optimal solution in time logarithmic in the size of the family.

# Zipf's law

Zipf[1] (1932): In natural language texts the (absolute) frequency $f$ of a term is approximately inversely proportional to its rank $r$ in the frequency table.

$$\exists \text{ constant } c \text{ such that } f \cdot r \approx c.$$

[1] George Kingsley Zipf, 1902 – 1950

# Zipf's law

| Word | Freq. | Rank | $f \cdot r$ | Word | Freq. | Rank | $f \cdot r$ |
|---|---|---|---|---|---|---|---|
| the | 3332 | 1 | 3332 | turned | 51 | 200 | 10200 |
| and | 2972 | 2 | 5944 | you'll | 30 | 300 | 9000 |
| a | 1775 | 3 | 5235 | name | 21 | 400 | 8400 |
| he | 877 | 10 | 8770 | comes | 16 | 500 | 8000 |
| but | 410 | 20 | 8400 | group | 13 | 600 | 7800 |
| be | 294 | 30 | 8820 | lead | 11 | 700 | 7700 |
| there | 222 | 40 | 8880 | friends | 10 | 800 | 8000 |
| one | 172 | 50 | 8600 | begin | 9 | 900 | 8100 |
| about | 158 | 60 | 9480 | family | 8 | 1000 | 8000 |
| more | 138 | 70 | 9660 | brushed | 4 | 2000 | 8000 |
| never | 124 | 80 | 9920 | sins | 2 | 3000 | 6000 |
| Oh | 116 | 90 | 10440 | Could | 2 | 4000 | 8000 |
| two | 104 | 100 | 10400 | Applausive | 1 | 8000 | 8000 |

Empirical evaluation of Zipf's law on Tom Sawyer
(Manning/Schütze, *Foundations of statistical natural language processing, MIT Press, 1999*)

# Zipf model

Documents $\mathcal{D} = \{d_1, \ldots, d_n\}$
Terms $\mathcal{T} = \{t_1, \ldots, t_m\}$, $m \leq poly(n)$

**Generating a document collection**:

- Every document is generated independently.
- Term occurences are also independent.
- A document contains term $t_i$ with probability $\frac{1}{i}$.
- Expected number of terms in a document: $\ln m$.

# Zipf model

Documents $\mathcal{D} = \{d_1, \ldots, d_n\}$
Terms $\mathcal{T} = \{t_1, \ldots, t_m\}$, $m \leq poly(n)$

**Generating a document collection**:

- Every document is generated independently.
- Term occurences are also independent.
- A document contains term $t_i$ with probability $\frac{1}{i}$.
- Expected number of terms in a document: $\ln m$.

# Zipf model (cont.)

*Definition* (Relative frequency of a term $t$ in a document collection $\mathcal{D}$):

$$\frac{|\{d \in \mathcal{D} \mid t \in d\}|}{|\mathcal{D}|}$$

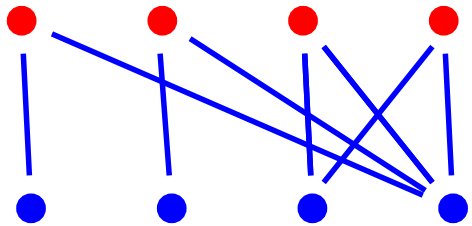
In the **Zipf model**:

- Expected relative frequency for $t_i$: $\frac{1}{i}$
- Expected frequency rank for $t_i$: $i$-th value among those of all terms.

$\Rightarrow$ Zipf model reflects in a natural way Zipf's law.

# Zipf model: relation to random graphs

Representing the data as a graph: *bipartite graph* (documents/terms)



$\Rightarrow$ Zipf's law for distribution of term degrees, $P(k) = \frac{1}{k}$.

# Threshold phenomenon

Assume that the terms of a query document are ordered by their frequency in the document collection.

Two events:

- **Any $q$-match**: $\exists d \in D$ that has at least $q$ common terms with the query document.
- **Prefix $q$-match**: $\exists d \in D$ that has at least $q$ "top" terms with the query document.

For a document collection following the Zipf model the following holds:

- The probability for both events is close to one for small $q$.
- At some magic level the probability for both events falls to nearly zero.

# Threshold phenomenon

Assume that the terms of a query document are ordered by their frequency in the document collection.

Two events:

- **Any $q$-match**: $\exists d \in D$ that has at least $q$ common terms with the query document.
- **Prefix $q$-match**: $\exists d \in D$ that has at least $q$ "top" terms with the query document.

For a document collection following the Zipf model the following holds:

- The probability for both events is close to one for small $q$.
- At some magic level the probability for both events falls to nearly zero.

# Threshold phenomenon

Assume that the terms of a query document are ordered by their frequency in the document collection.

Two events:

- **Any $q$-match**: $\exists d \in D$ that has at least $q$ common terms with the query document.
- **Prefix $q$-match**: $\exists d \in D$ that has at least $q$ "top" terms with the query document.

For a document collection following the Zipf model the following holds:

- The probability for both events is close to one for small $q$.
- At some magic level the probability for both events falls to nearly zero.

# Threshold phenomenon

Assume that the terms of a query document are ordered by their frequency in the document collection.

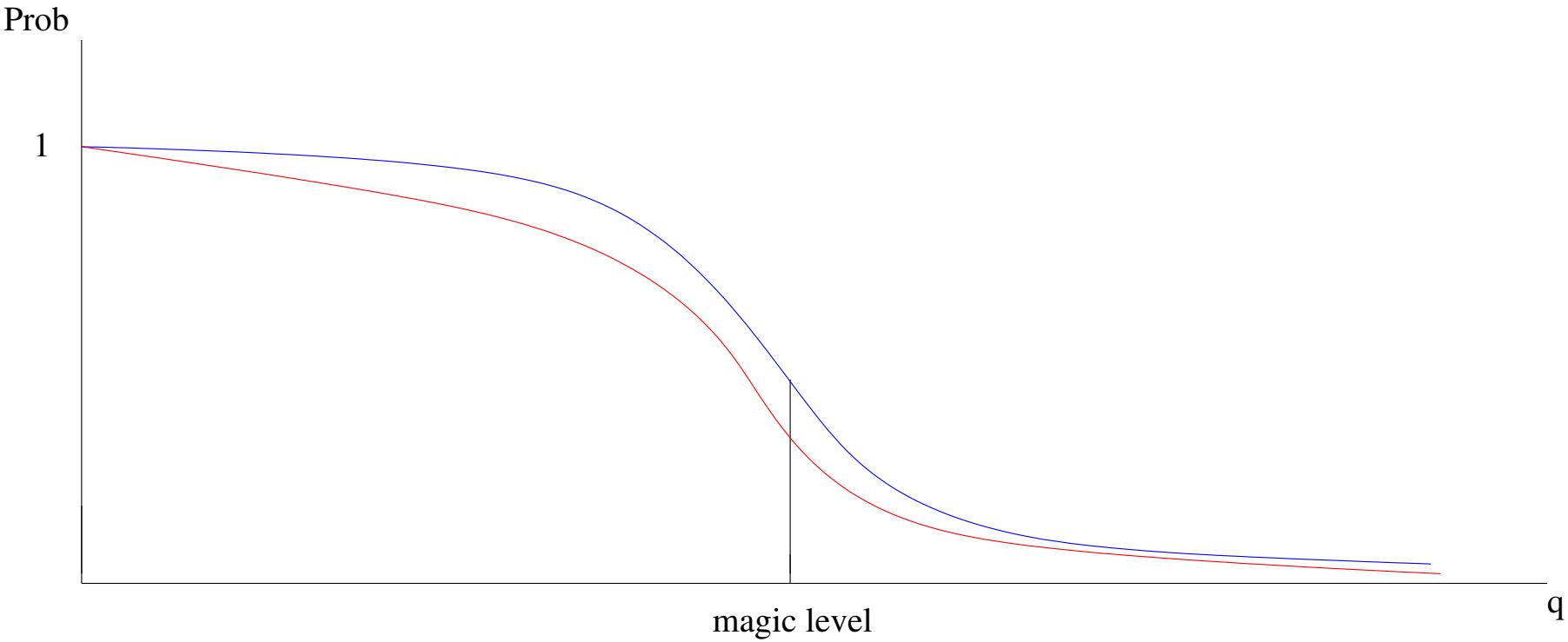
Two events:

- **Any $q$-match**: $\exists d \in D$ that has at least $q$ common terms with the query document.
- **Prefix $q$-match**: $\exists d \in D$ that has at least $q$ "top" terms with the query document.

For a document collection following the Zipf model the following holds:

- The probability for both events is close to one for small $q$.
- At some magic level the probability for both events falls to nearly zero.

# Threshold phenomenon




Exemplary probability curves for any $q$-match and prefix $q$-match.
($q$ = no. of matched terms)

# Zipf model (cont.)

Partitioning of $\mathcal{T}$:

$$\underbrace{t_1\ t_2}_{P_1} \quad \underbrace{t_3 \cdots t_7}_{P_2} \quad \ldots$$

Group $P_i$ includes terms from $t_{\lceil e^{i-1} \rceil}$ to $t_{\lfloor e^i \rfloor}$.

A document that contains $\ln m$ terms $p_1 \ldots p_{\ln m}$, $p_i \in P_i$, will be called regular.

# Zipf model (cont.)

Partitioning of $\mathcal{T}$:

$$\underbrace{t_1\ t_2}_{P_1} \quad \underbrace{t_3 \cdots t_7}_{P_2} \quad \ldots$$

Group $P_i$ includes terms from $t_{\lceil e^{i-1} \rceil}$ to $t_{\lfloor e^i \rfloor}$.

A document that contains $\ln m$ terms $p_1 \ldots p_{\ln m}$, $p_i \in P_i$, will be called regular.

# Magic level for the Zipf model

**Magic level:**

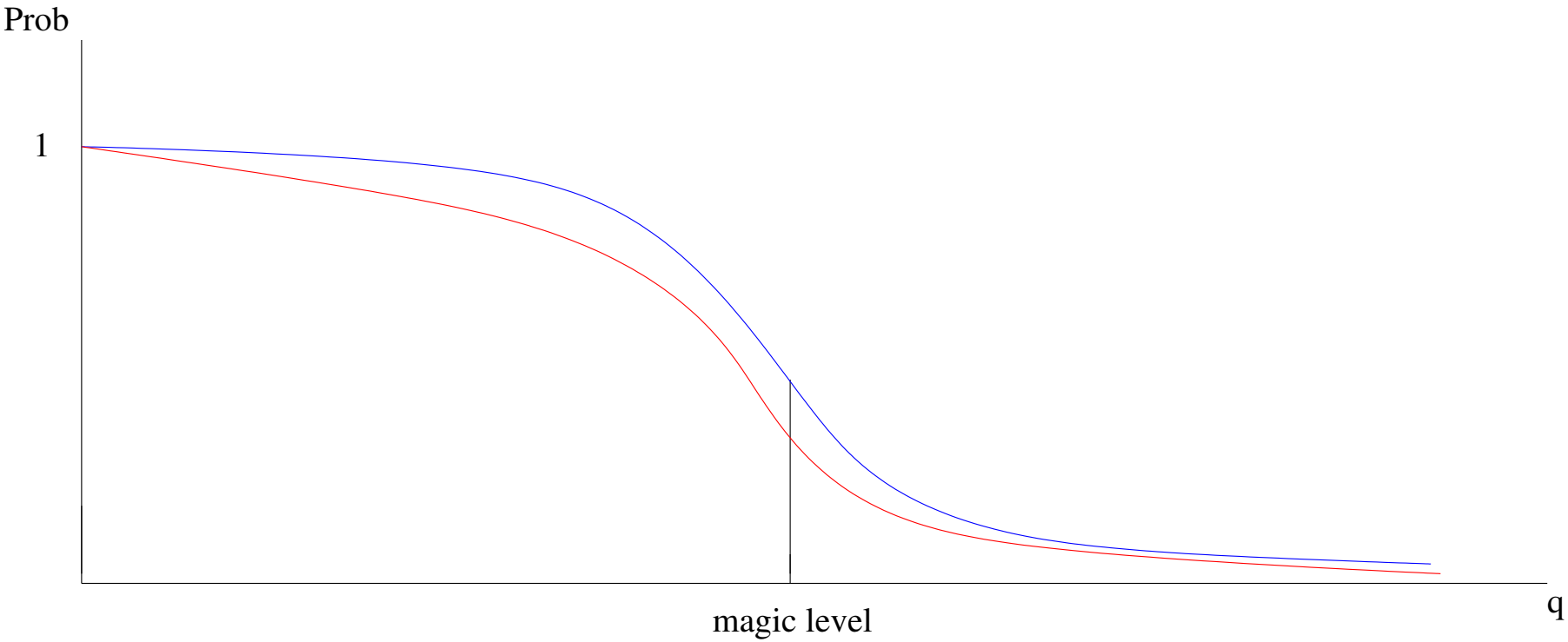
$$q = \sqrt{2 \ln n} \qquad (n = \text{no. of documents}).$$

**Theorem**

*Let $3 \leq \gamma < q - 1, \gamma \in \mathbb{N}$. Fix $n, m$ and a regular query document $d_{new}$. Then for a document collection following the Zipf model the following holds:*

1. *The probability that there exists a document $d \in \mathcal{D}$ that contains the first $q - \gamma$ terms of $d_{new}$ is greater than $1 - 2^{-e^{\frac{q(\gamma+1)}{2}}}$.*
2. *The probability that there exists a document $d \in \mathcal{D}$ that contains at least $q + \gamma$ terms of $d_{new}$ is smaller than $\frac{1}{e^{(\gamma-2)q-1}}$.*

# Magic level for the Zipf model




Exemplary probability curves for any $q$-match and prefix $q$-match.
($q$ = no. of matched terms)

# Magic level for the Zipf model

**Magic level:**

$$q = \sqrt{2 \ln n} \qquad (n = \text{no. of documents}).$$

**Theorem**

*Let $3 \leq \gamma < q - 1, \gamma \in \mathbb{N}$. Fix $n, m$ and a regular query document $d_{new}$. Then for a document collection following the Zipf model the following holds:*

1. *The probability that there exists a document $d \in \mathcal{D}$ that contains the first $q - \gamma$ terms of $d_{new}$ is greater than $1 - 2^{-e^{\frac{q(\gamma+1)}{2}}}$.*
2. *The probability that there exists a document $d \in \mathcal{D}$ that contains at least $q + \gamma$ terms of $d_{new}$ is smaller than $\frac{1}{e^{(\gamma-2)q-1}}$.*

# MaxInt algorithm in the Zipf model

**Preprocessing**

1. For every document: Sort the term list according to the position of the term in the frequency table.
2. For every document: Generate the set of all possible regular $(q - \gamma)$-lists.
3. Sort these regular lists and store for every list a pointer to the corresponding document.

Complexity: $\mathcal{O}(\log m \cdot n \cdot \log n)$

**Query**

Find a regular $(q - \gamma)$-list having the maximal common prefix with the query document by binary search. Return the document corresponding to this list.

Complexity: $\mathcal{O}(\log^2 n)$

# MaxInt algorithm in the Zipf model

**Preprocessing**

1. For every document: Sort the term list according to the position of the term in the frequency table.
2. For every document: Generate the set of all possible regular $(q - \gamma)$-lists.
3. Sort these regular lists and store for every list a pointer to the corresponding document.

Complexity: $\mathcal{O}(\log m \cdot n \cdot \log n)$

**Query**

Find a regular $(q - \gamma)$-list having the maximal common prefix with the query document by binary search. Return the document corresponding to this list.

Complexity: $\mathcal{O}(\log^2 n)$

# MaxInt algorithm in the Zipf model

**Preprocessing**

1. For every document: Sort the term list according to the position of the term in the frequency table.
2. For every document: Generate the set of all possible regular $(q - \gamma)$-lists.
3. Sort these regular lists and store for every list a pointer to the corresponding document.

Complexity: $\mathcal{O}(\log m \cdot n \cdot \log n)$

**Query**

Find a regular $(q - \gamma)$-list having the maximal common prefix with the query document by binary search. Return the document corresponding to this list.

Complexity: $\mathcal{O}(\log^2 n)$

# MaxInt algorithm in the Zipf model

**Preprocessing**

1. For every document: Sort the term list according to the position of the term in the frequency table.
2. For every document: Generate the set of all possible regular $(q-\gamma)$-lists.
3. Sort these regular lists and store for every list a pointer to the corresponding document.

Complexity: $\mathcal{O}(\log m \cdot n \cdot \log n)$

**Query**

Find a regular $(q-\gamma)$-list having the maximal common prefix with the query document by binary search. Return the document corresponding to this list.

Complexity: $\mathcal{O}(\log^2 n)$

# Future Work

- Does it hold in real life (empirical studies)?
- Does a threshold phenomenon also hold for other randomized models (e.g. preferential attachment model)?
- Does an exact algorithm for MaxInt exist (preserving our time constraints)?

Thanks for your attention!

# Future Work

- Does it hold in real life (empirical studies)?
- Does a threshold phenomenon also hold for other randomized models (e.g. preferential attachment model)?
- Does an exact algorithm for MaxInt exist (preserving our time constraints)?

Thanks for your attention!

# Future Work

- Does it hold in real life (empirical studies)?
- Does a threshold phenomenon also hold for other randomized models (e.g. preferential attachment model)?
- Does an exact algorithm for MaxInt exist (preserving our time constraints)?

Thanks for your attention!

# Future Work

- Does it hold in real life (empirical studies)?
- Does a threshold phenomenon also hold for other randomized models (e.g. preferential attachment model)?
- Does an exact algorithm for MaxInt exist (preserving our time constraints)?

Thanks for your attention!