Minhashing for Graph Similarity Computation

CSCUBS 2016

Can Güney Aksakalli[1] Pascal Welke[2]

RWTH Aachen University, Germany
can.aksakalli@rwth-aachen.de

University of Bonn, Germany
welke@uni-bonn.de

May 25, 2016

Overview

1. Introduction
2. Related Work
3. Graph Minhashing
   - Substructure Extraction
   - Fingerprinting
   - Minhashing
4. Experimental Results
5. Conclusion and Future Work

Introduction

- MinHash [Broder, 2000] for Document Deduplication
  - Invented for AltaVista search engine
  - Filtering duplicated or near-duplicated Web documents
  - Ranking pages correctly
  - Filter out the search results with the same content

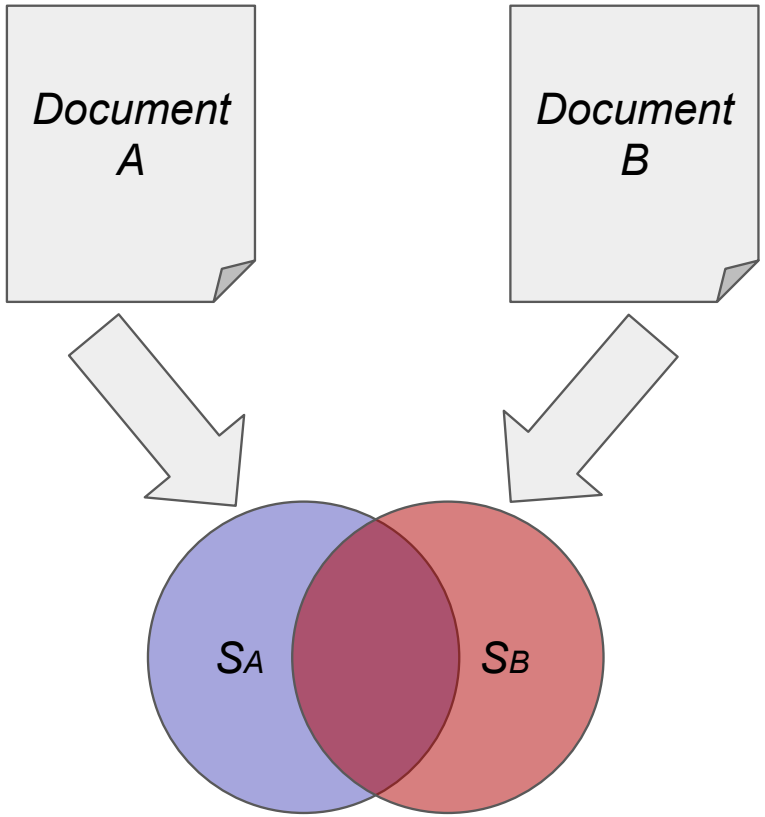
Introduction

Minhashing for documents

1. Extracts chunks of words from text by *w-shingling*
2. Problem is reduced to set intersection for set of fingerprints

$$r(A, B) = \frac{|S_A \cap S_B|}{|S_A \cup S_B|} \quad (1)$$

3. Jaccard similarity of large sets can be approximated by using small fixed sized MinHash *sketches*



Introduction

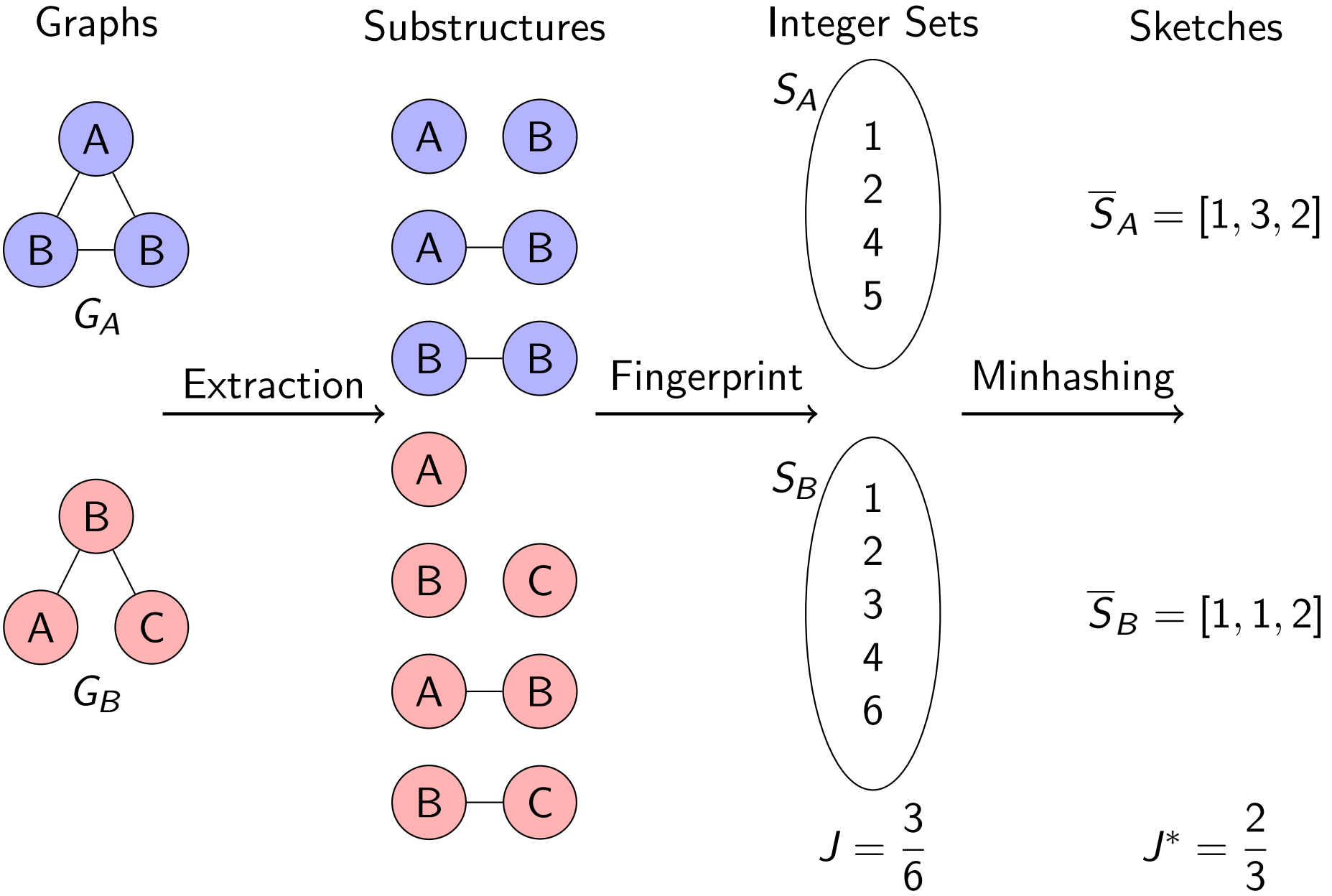
Problem Definition

- Implementing Broder's method for document deduplication for graphs
  - Instead of n-shingles in documents, use (connected) subgraphs with n vertices
  - Construct a hash function h for graphs of size n with the properties
    - If H and H' are isomorphic, then $h(H, k) = h(H', k)$
    - $h(H, k)$ maps H to an integer in the set $1, ..., k$
- Evaluation with real datasets of chemical compounds
  - Molecule databases
    - Atom = Vertex (Node)
    - Bound = Edge

Related Work

- [Broder et al., 1998] Representing all documents as fixed size *sketches*
- [Vishwanathan and Smola, 2003] *tree kernels* for counting shared subtrees
- [Horváth et al., 2004] *cyclic pattern kernels*, counts common occurrences of cycles and trees
  - Misses simple paths
- [Ralaivola et al., 2005] *moleculer fingerprinting*, simple walks on graphs (we used for extraction)
- [Teixeira et al., 2012] MinHash method with graph kernels
  - Unweighted graphs for molecules
  - Type of Molecular Bounds is missed
  - We also investigated weighted graphs

Graph Minhashing



Substructure Extraction

w-Shingling for Text Extraction [Broder, 2000]

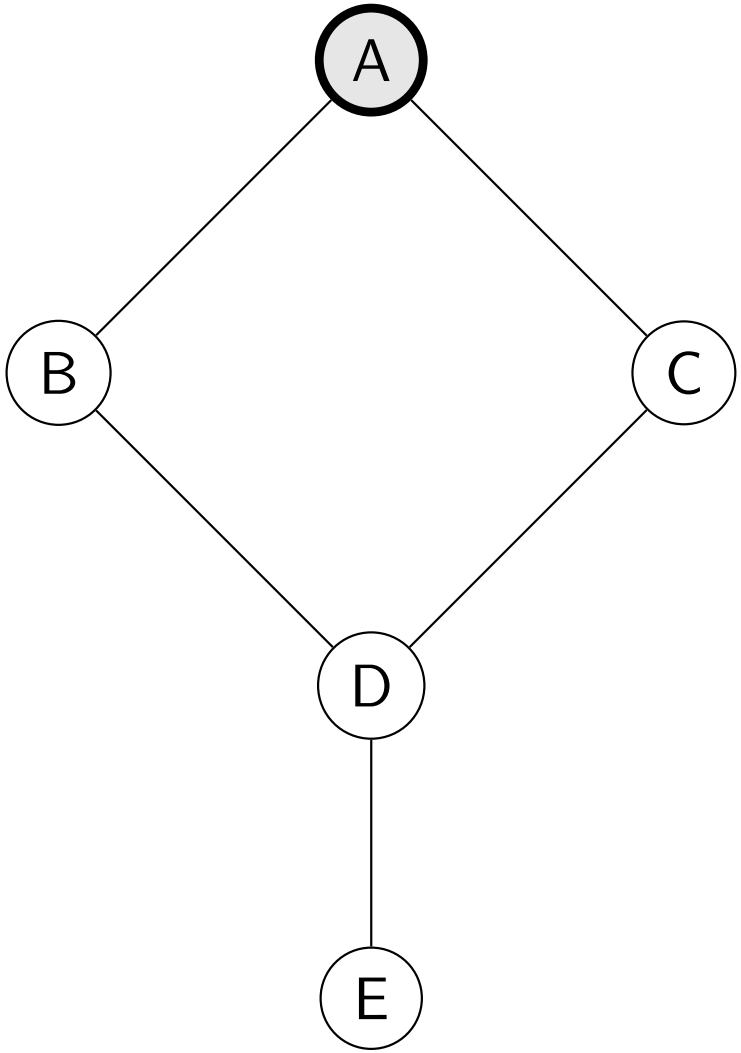
- A contiguous subsequence of words in a text document are defined as *shingle* and size of these chunks as w
- *4-shingle* of a sentence "A rose is a rose is a rose.",

$$\{(a, rose, is, a), (rose, is, a, rose), (is, a, rose, is)\} \tag{2}$$

Simple walks for Graph Extraction [Ralaivola et al., 2005]

- *Depth-first search* with all paths and no cycles
- Slightly modified DFS algorithm which traverses all possible branches up to a depth limit d ($d = 10$ in practice)
- Repeat the search starting from each vertex

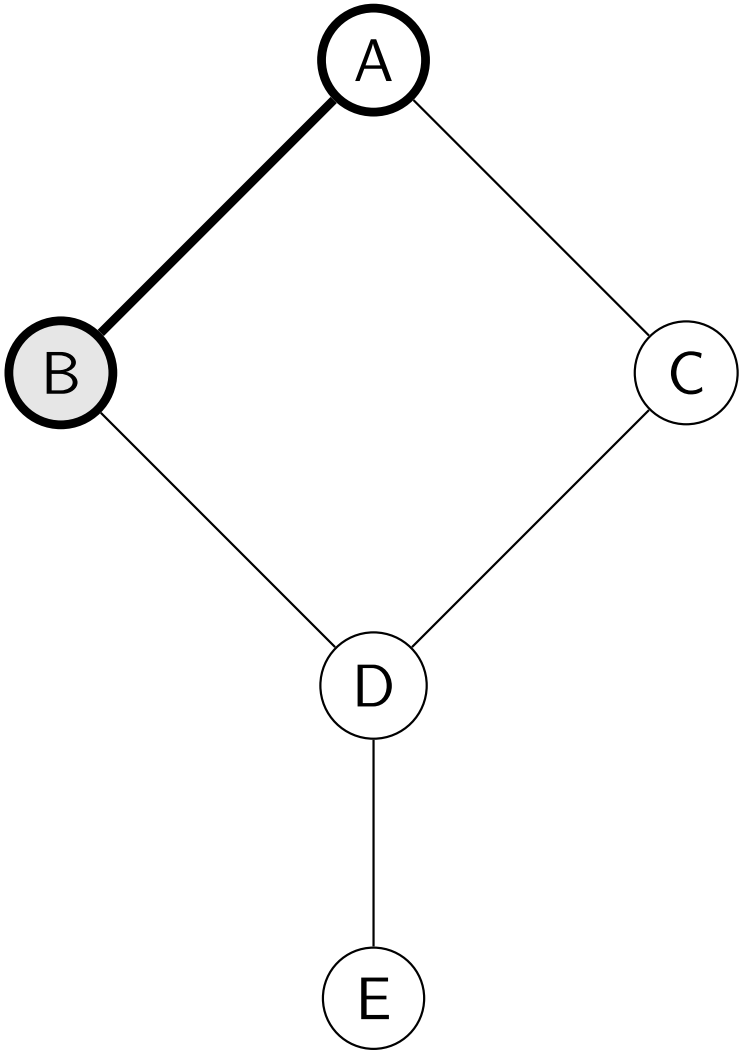
Depth-first Search with all Paths and no Cycles



Extracted paths

- A

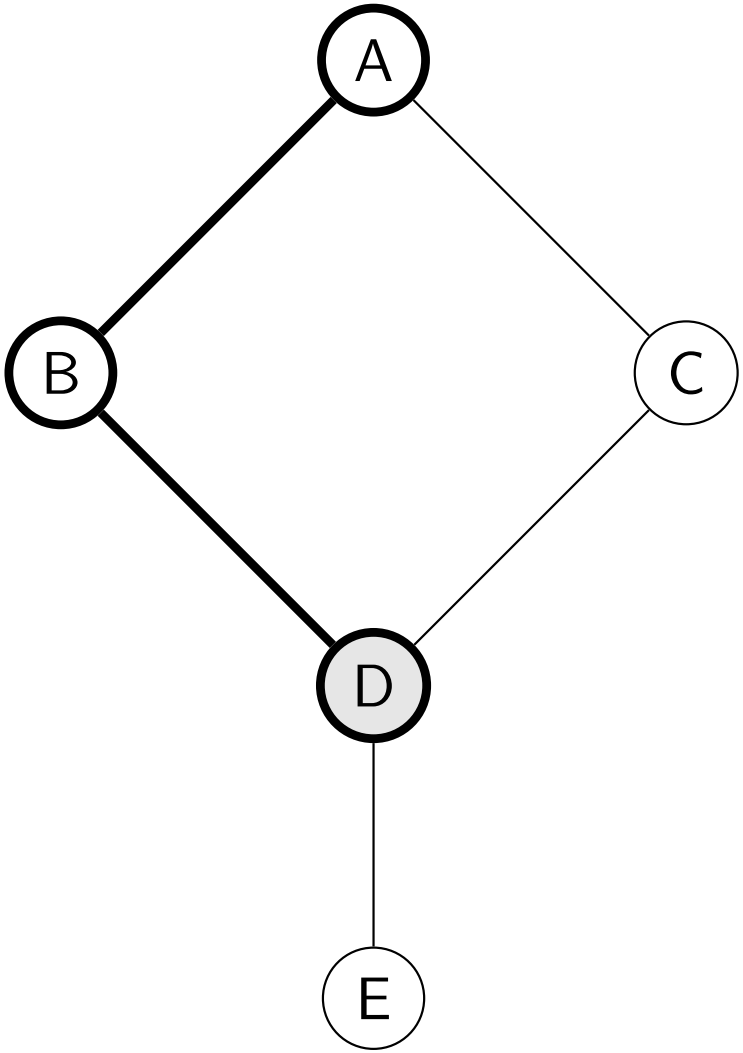
Depth-first Search with all Paths and no Cycles



Extracted paths

- A
- A-B

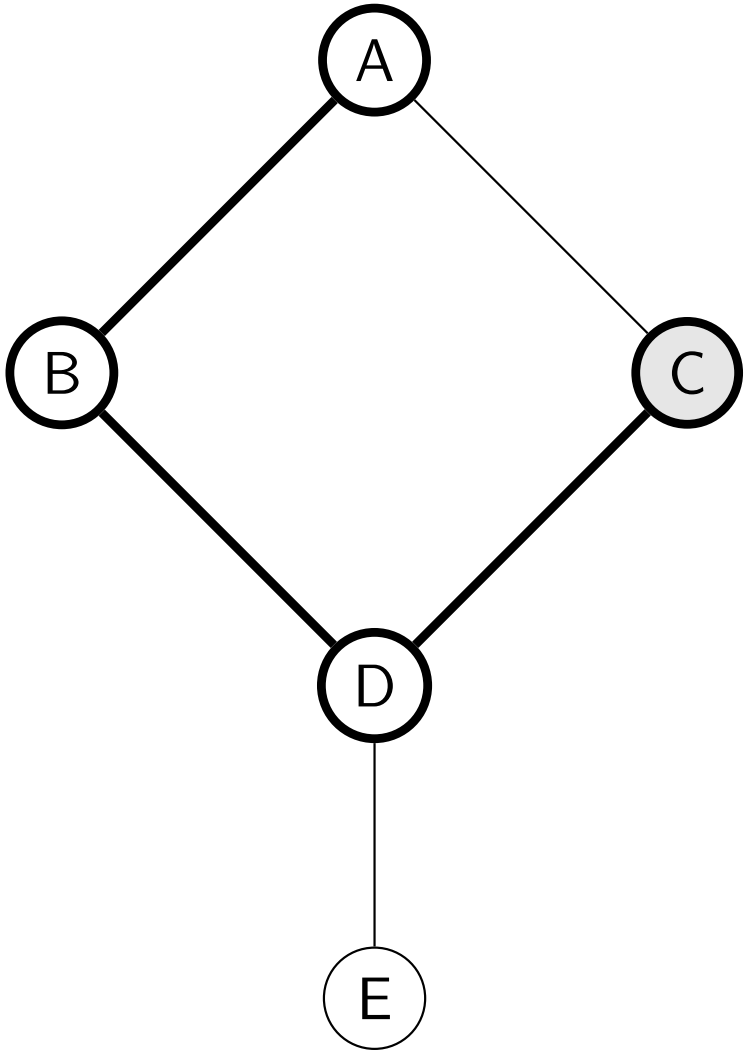
Depth-first Search with all Paths and no Cycles



Extracted paths

- A
- A-B
- A-B-D

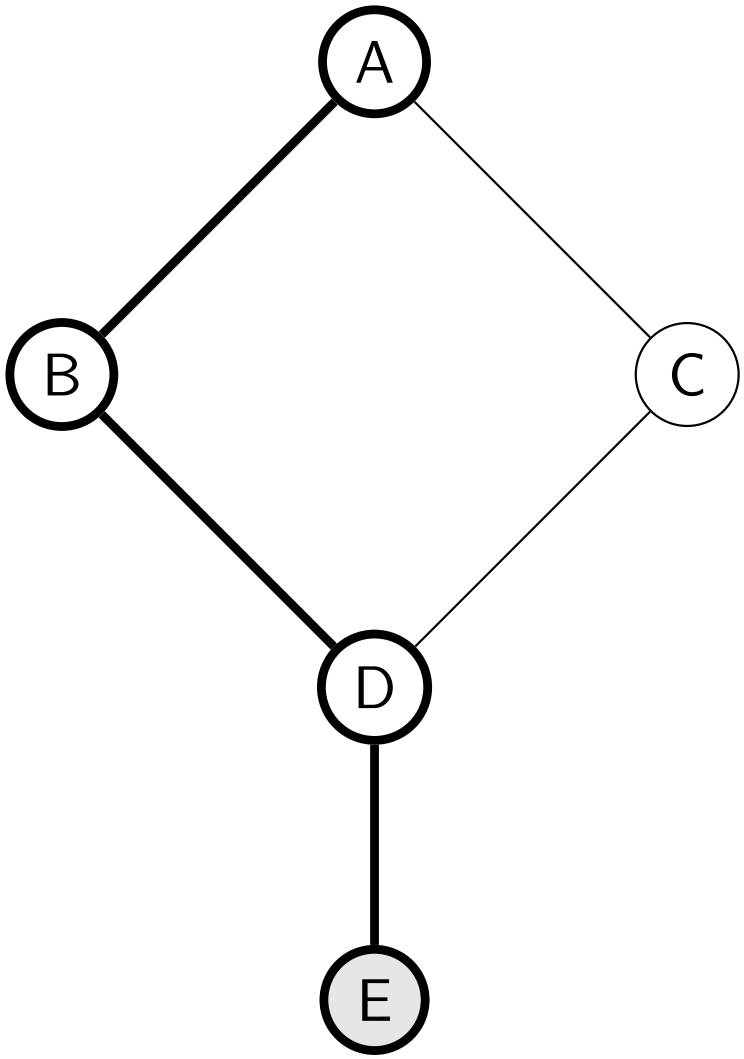
Depth-first Search with all Paths and no Cycles



Extracted paths

- A
- A-B
- A-B-D
- A-B-D-C

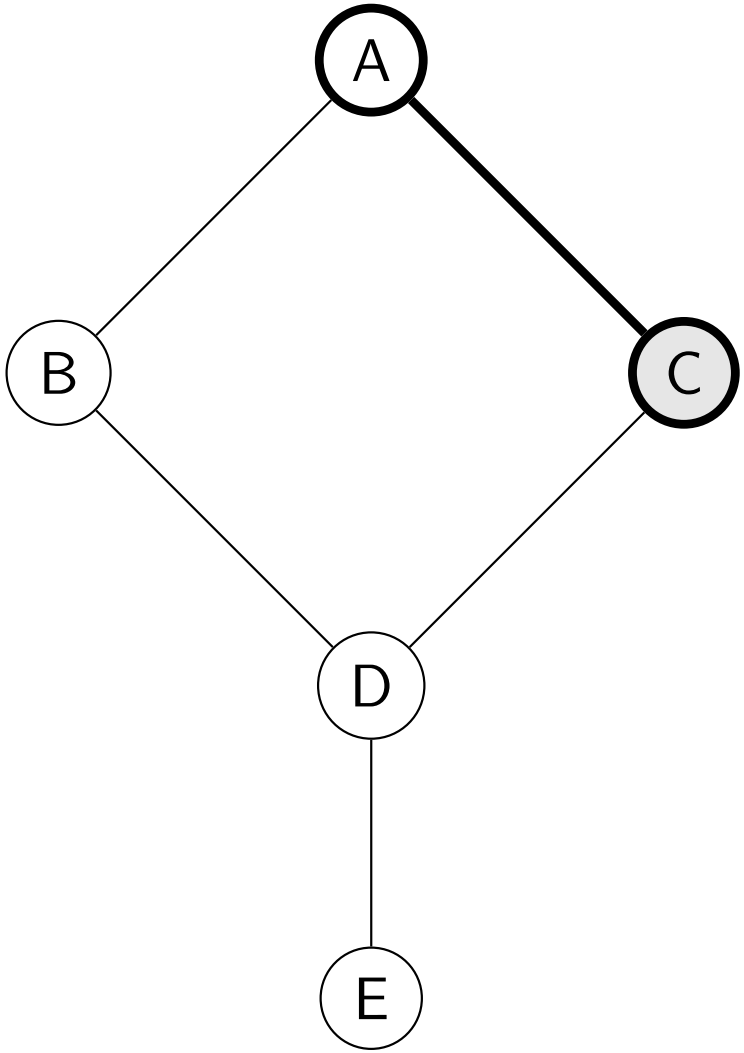
Depth-first Search with all Paths and no Cycles



Extracted paths

- A
- A-B
- A-B-D
- A-B-D-C
- A-B-D-E

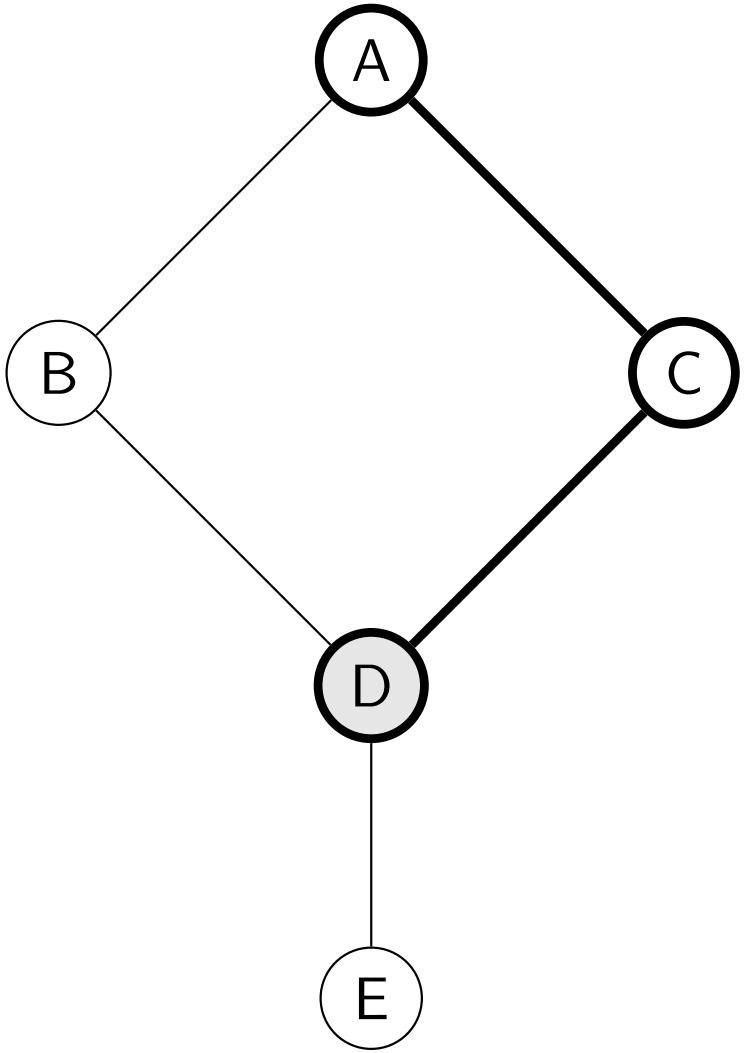
Depth-first Search with all Paths and no Cycles



Extracted paths

- A
- A-B
- A-B-D
- A-B-D-C
- A-B-D-E
- A-C

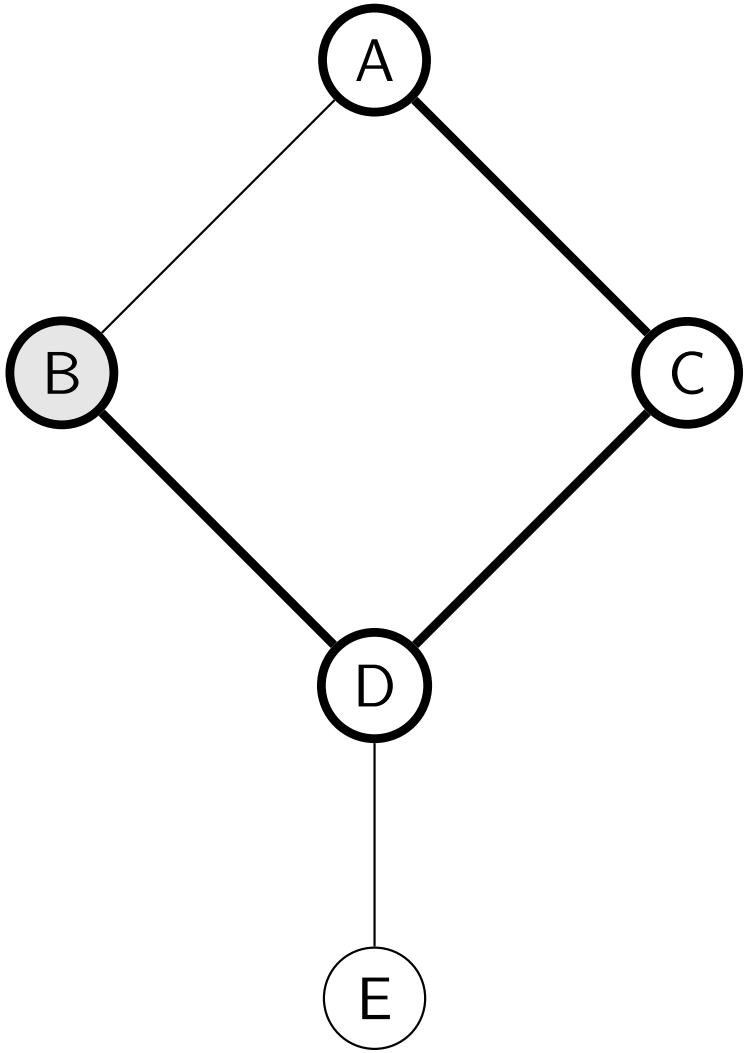
Depth-first Search with all Paths and no Cycles



Extracted paths

- A
- A-B
- A-B-D
- A-B-D-C
- A-B-D-E
- A-C
- A-C-D

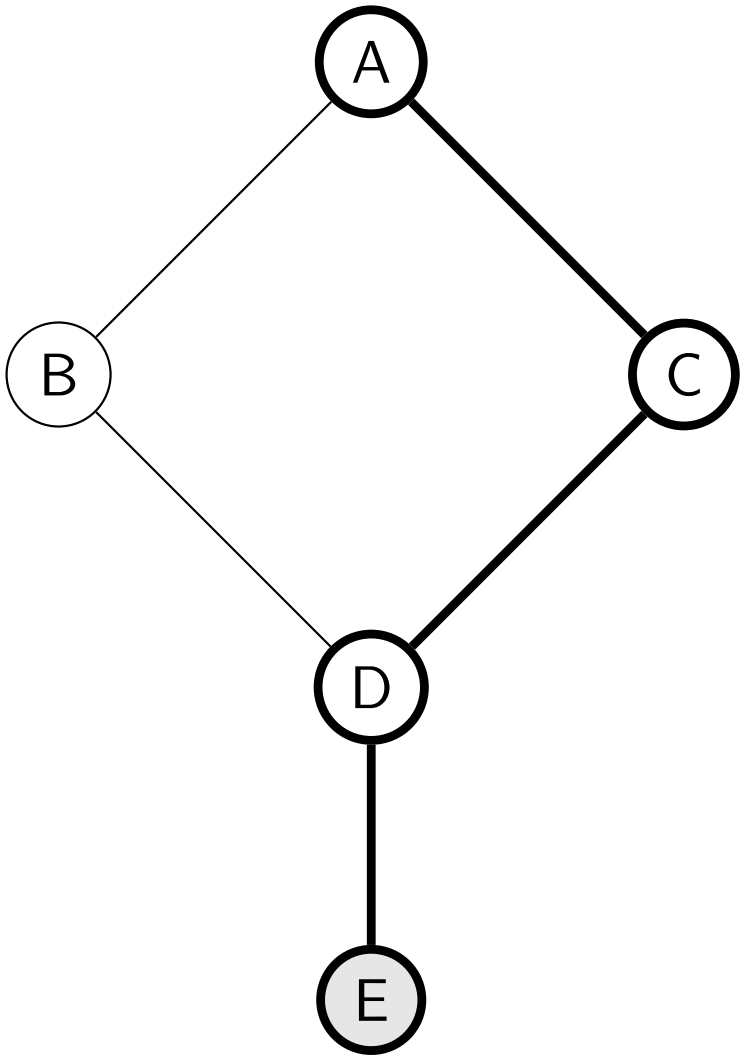
Depth-first Search with all Paths and no Cycles



Extracted paths

- A
- A-B
- A-B-D
- A-B-D-C
- A-B-D-E
- A-C
- A-C-D
- A-C-D-B

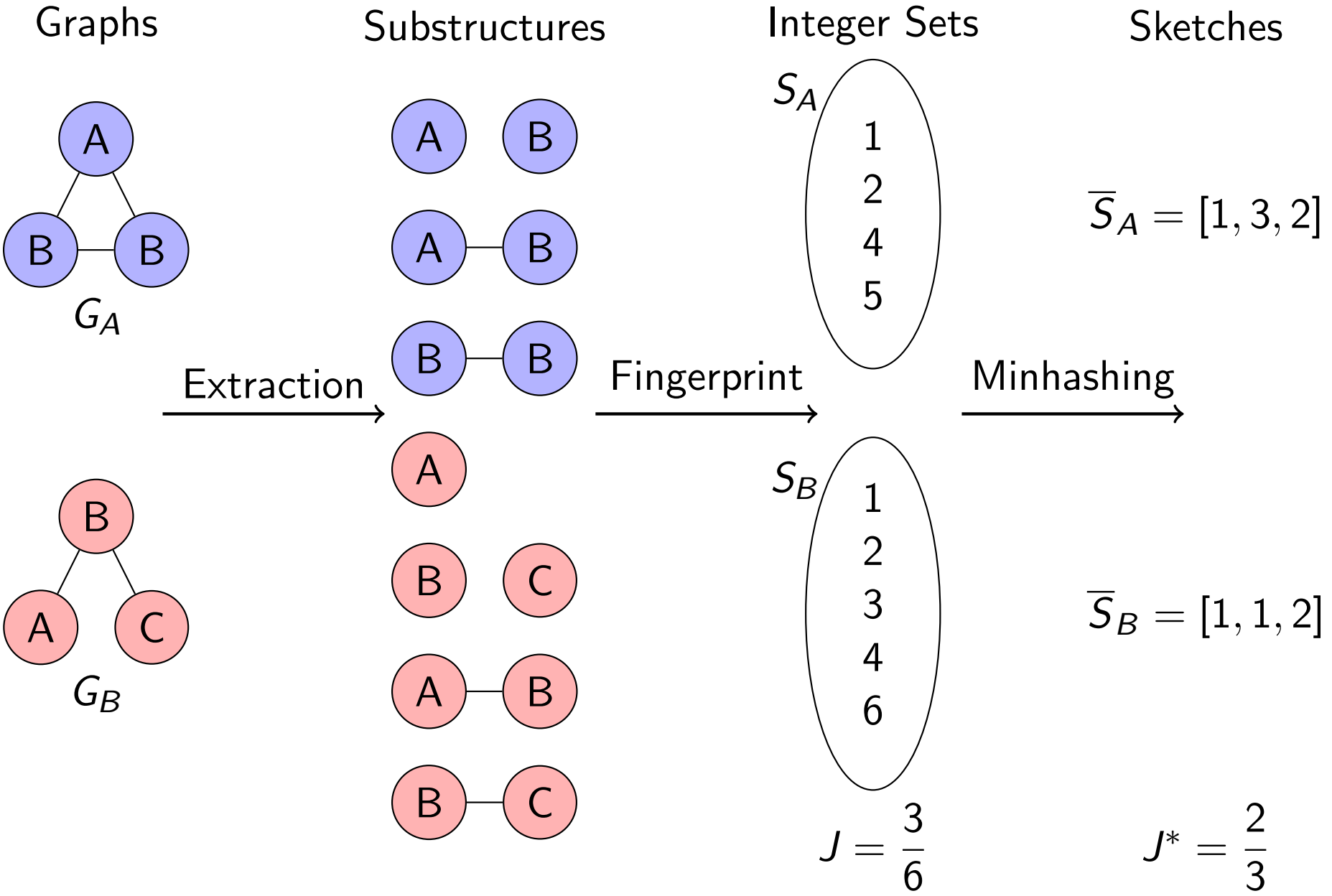
Depth-first Search with all Paths and no Cycles



Extracted paths

- A
- A-B
- A-B-D
- A-B-D-C
- A-B-D-E
- A-C
- A-C-D
- A-C-D-B
- A-C-D-E

Graph Minhashing



Fingerprinting

- After extraction, we have vertex chain $[v_1, v_2 ... v_c]$ which needs to be mapped to an integer value
- *Arrays.deepHashCode* method of Java is used
- $L(v_i)$ gives the code, prime P (in practice $P = 31$)

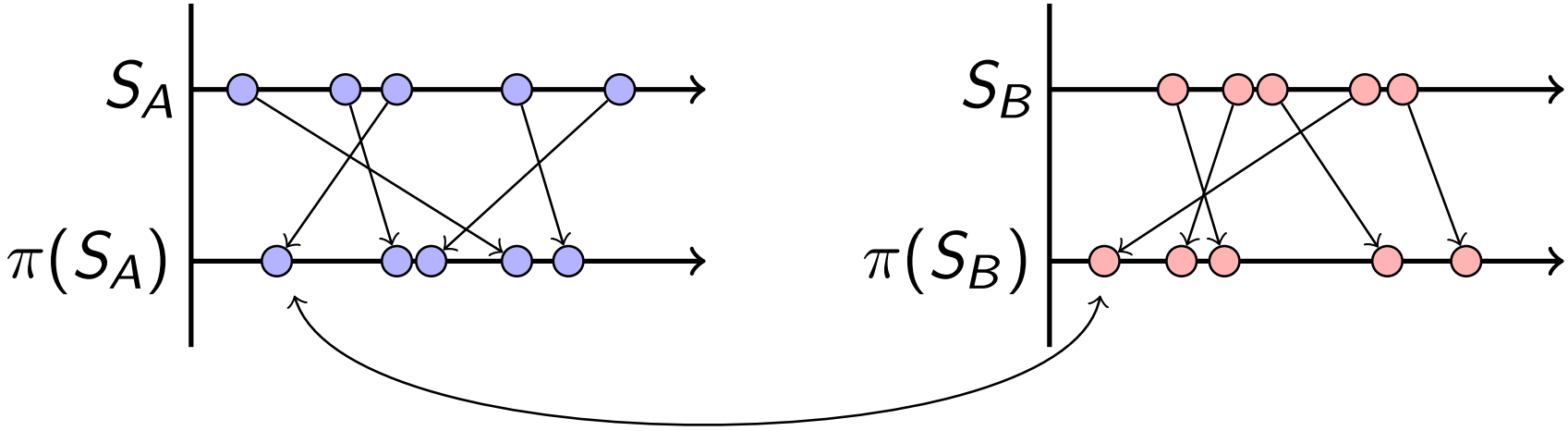
$$integer([v_1, v_2 ... v_c]) = ((P + L(v_1))P + L(v_2))P ... + L(v_c) \quad (3)$$

- For weighted graphs, the edge e_{ij} of v_i and v_j

$$fingerprint' = integer([..., v_i, e_{ij}, v_j, ...]) \quad (4)$$

Minhashing (I)

- After fingerprinting, graphs are represented as sets
  - $G_A \rightarrow S_A$
  - $G_B \rightarrow S_B$
- Thus the problem is reduced to set intersection
- [Broder et al., 1998] let π a uniformly random permutation function



$$min\{\pi(S_A)\} \stackrel{?}{=} min\{\pi(S_B)\}$$

Minhashing (II)

- [Broder et al., 1998] let π a uniformly random permutation function

$$Pr(min\{\pi(S_A)\} = min\{\pi(S_B)\}) = \frac{|S_A \cap S_B|}{|S_A \cup S_B|} = r(A, B) \quad (5)$$

- Any integer value of the range has the same possibility to be the minimum after permutation
- Use a set of random permutations $\pi_1, ..., \pi_t$ and store a *sketch* value for each sets

$$\overline{S}_A = (min\{\pi_1(S_A)\}, min\{\pi_2(S_A)\}, ..., min\{\pi_t(S_A)\}) \quad (6)$$

- The approximate resemblance of A and B is rate of corresponding equal elements in $\overline{S}_A$ and $\overline{S}_B$
- The bigger the sketch size t, smaller the estimated error

Minhashing - Toy Example

		1	2	3	4	5	6	7
h_1	π_1	1	2	3	4	5	6	7
	S_A	1	1	0	1	1	0	0
	S_B	1	1	1	1	0	1	0
h_2	π_2	3	7	1	6	2	5	4
	S_A	0	0	1	0	1	1	1
	S_B	1	0	1	1	1	0	1
h_3	π_3	7	4	3	6	1	2	5
	S_A	0	1	0	0	1	1	1
	S_B	0	1	1	1	1	1	0

Table : Example of minhashing for the toy example.

Implementing the Minhashing method

- In practice, it is impossible to choose a uniform permutation π
- Implementing a smaller set of permutation functions with XOR

```
public List<Integer> minhash(Set<Integer> fingerprintSet) {
  return hashFunctions.stream()
          .map(h -> fingerprintSet.stream()
            .min(Comparator.comparing(i -> i ^ h)).get()
          )
          .collect(Collectors.toList());
}
```

Experimental Results (I)

Evaluation on NCI AIDS Dataset

Total molecules	42 687
Active molecules	422
Avg. vertex (atom)	45.7
Avg. edge (bound)	47.71
Avg. fingerprints unweighted	613.14
Avg. fingerprints weighted	1534.31

Table : AIDS dataset provided by National Cancer Institute

Experimental Results (II)

- Sketch size t settles
- 2^6 gives better result than 2^7
  - Probability of error decreases but not guaranteed

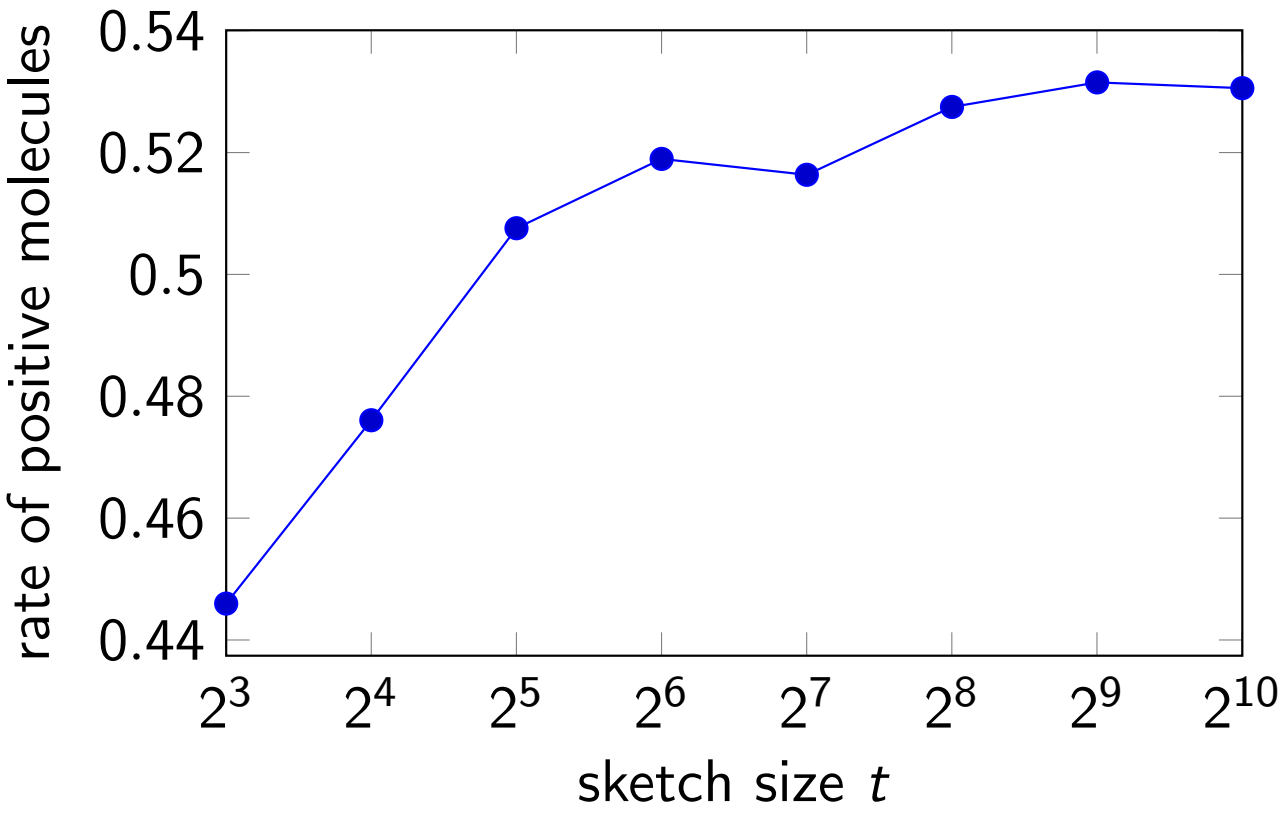


Figure : Precision at k=10 for different sketch sizes t (unweighted graph fingerprinting)

Experimental Results (III)

- Average accuracy is 92% for first item because of collusion

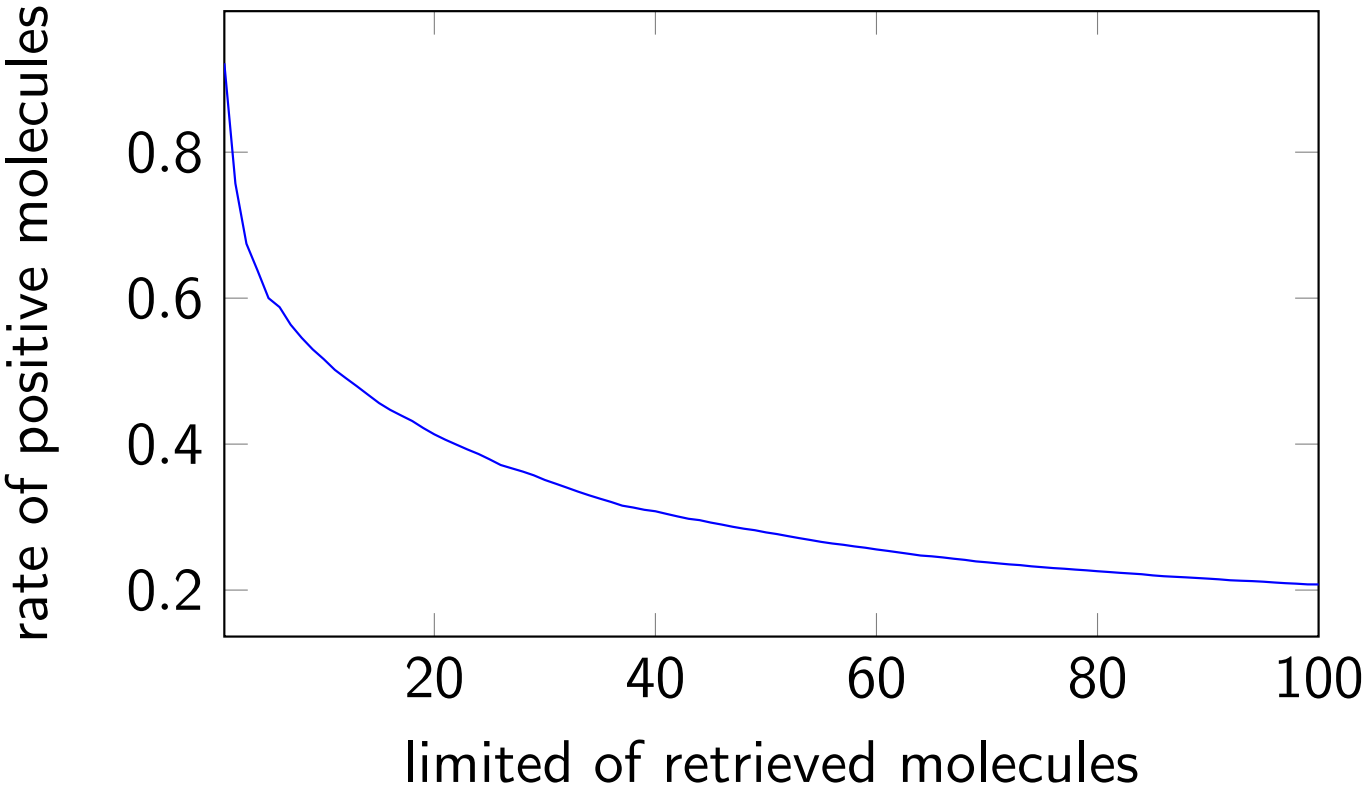


Figure : Precision at k from 1 to 100. (sketch sizes $t = 64$, unweighted graph fingerprinting)

Experimental Results (IV)

Unweighted

		Actual	
		Positive	Negative
Predicted	Positive	216	149
	Negative	206	42116
	ACC= 0.991	TPR= 0.511	TNR= 0.995

Table : The confusion matrix for k-NN classifier, k=3, sketch size t=64, unweighted

- The classes are not balanced, Accuracy (ACC) might be misleading
- True Positive Rate (TPR) is still promising over 1% active molecules

Experimental Results (V)

Weighted

		Actual	
		Positive	Negative
Predicted	Positive	213	160
	Negative	209	42105
	ACC= 0.991	TPR= 0.504	TNR= 0.996

Table : The confusion matrix for k-NN classifier, k=3, sketch size t=64, weighted

- Taking weighted edges into account is not significantly effecting the end result

Conclusion and Future Work

- The idea of minhashing can be applied to graph databases
- A promising graph analysis system was implemented in Java and released under MIT license on GitHub [1]
- An extraction approach with better representation would improve the accuracy in the future

[1] `https://github.com/aksakalli/graph-min-hash`

References I

Broder, A. Z. (2000).
Identifying and filtering near-duplicate documents.
In *Proceedings of the 11th Annual Symposium on Combinatorial Pattern Matching*, COM '00, pages 1–10, London, UK, UK. Springer-Verlag.

Broder, A. Z., Charikar, M., Frieze, A. M., and Mitzenmacher, M. (1998).
Min-wise independent permutations (extended abstract).
In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98, pages 327–336, New York, NY, USA. ACM.

Horváth, T., Gärtner, T., and Wrobel, S. (2004).
Cyclic pattern kernels for predictive graph mining.
In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 158–167, New York, NY, USA. ACM.

References II

Ralaivola, L., Swamidass, S. J., Saigo, H., and Baldi, P. (2005).
Graph kernels for chemical informatics.
Neural Networks, 18(8):1093 – 1110.
Neural Networks and Kernel Methods for Structured Domains.

Teixeira, C. H. C., Silva, A., and Jr., W. M. (2012).
Min-hash fingerprints for graph kernels: A trade-off among accuracy, efficiency, and compression.
Journal of Information and Data Management, 3(3):227–242.

Vishwanathan, S. V. N. and Smola, A. (2003).
Fast Kernels for String and Tree Matching.
Advances in Neural Information Processing Systems, 15.

Questions?

Thank you!