



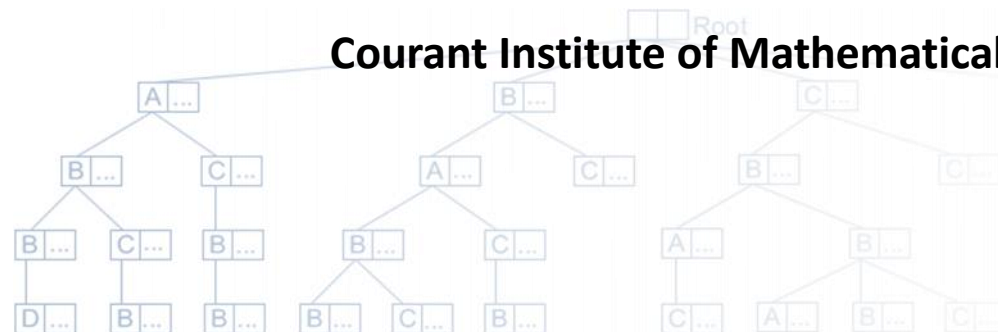
Enhancing Graph Database Indexing By Suffix Tree Structure

V. Bonnici, A. Ferro, R. Giugno, A. Pulvirenti

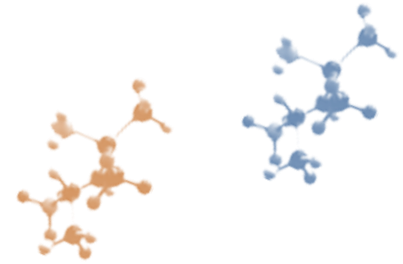
Dipartimento di Matematica e Informatica Università di Catania

D. Shasha

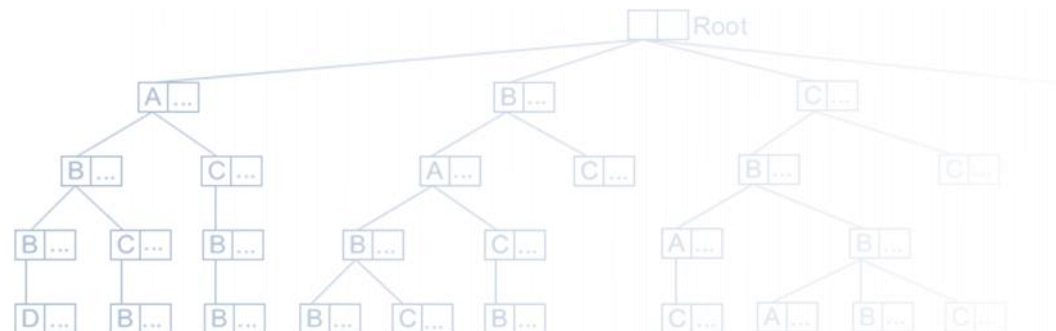
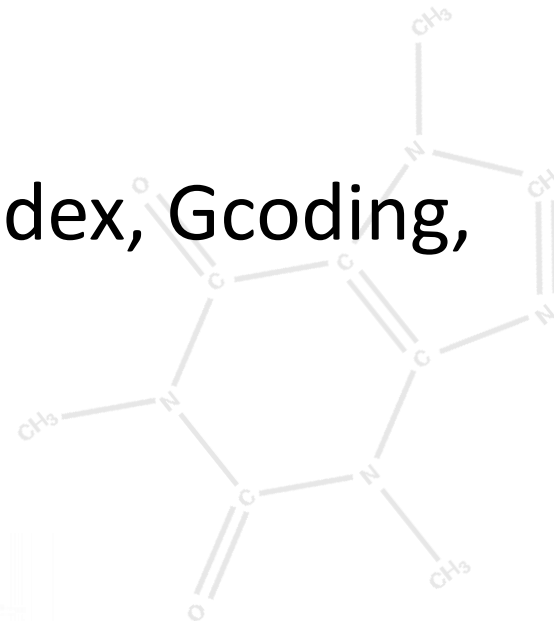
Courant Institute of Mathematical Science, New York University



Outline

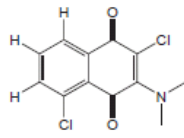
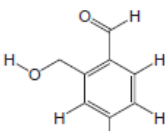
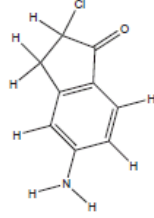
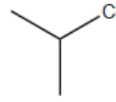
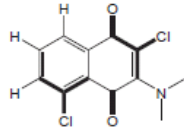
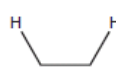
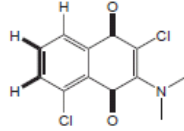
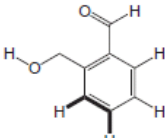
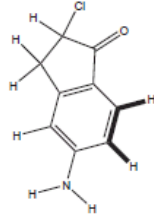


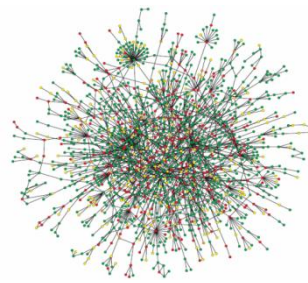
- Biological Motivations
- GraphGrep
- GraphGrepSX
- Experimental results (GIndex, Gcoding, GraphGrep, Ctree, SING)



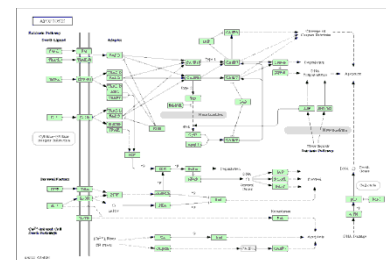
Motivations

- Many applications in industry, science, and engineering share the same problem: given a subgraph, find its occurrences in a database of graphs.
 - Prediction of the functionality of new natural or synthesized compounds
 - Make a compound Q more active
 - Find fragment with the same function among different species
 - Predict protein function, Predict protein interaction

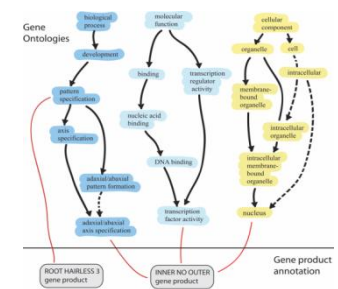
Graph Database	G_1	G_2	G_3
			
Exact Query	Number of Occurrences in Graphs		
 Q_1	 2		
 Q_2	 2	 3	 1



Biological Networks



Pathways



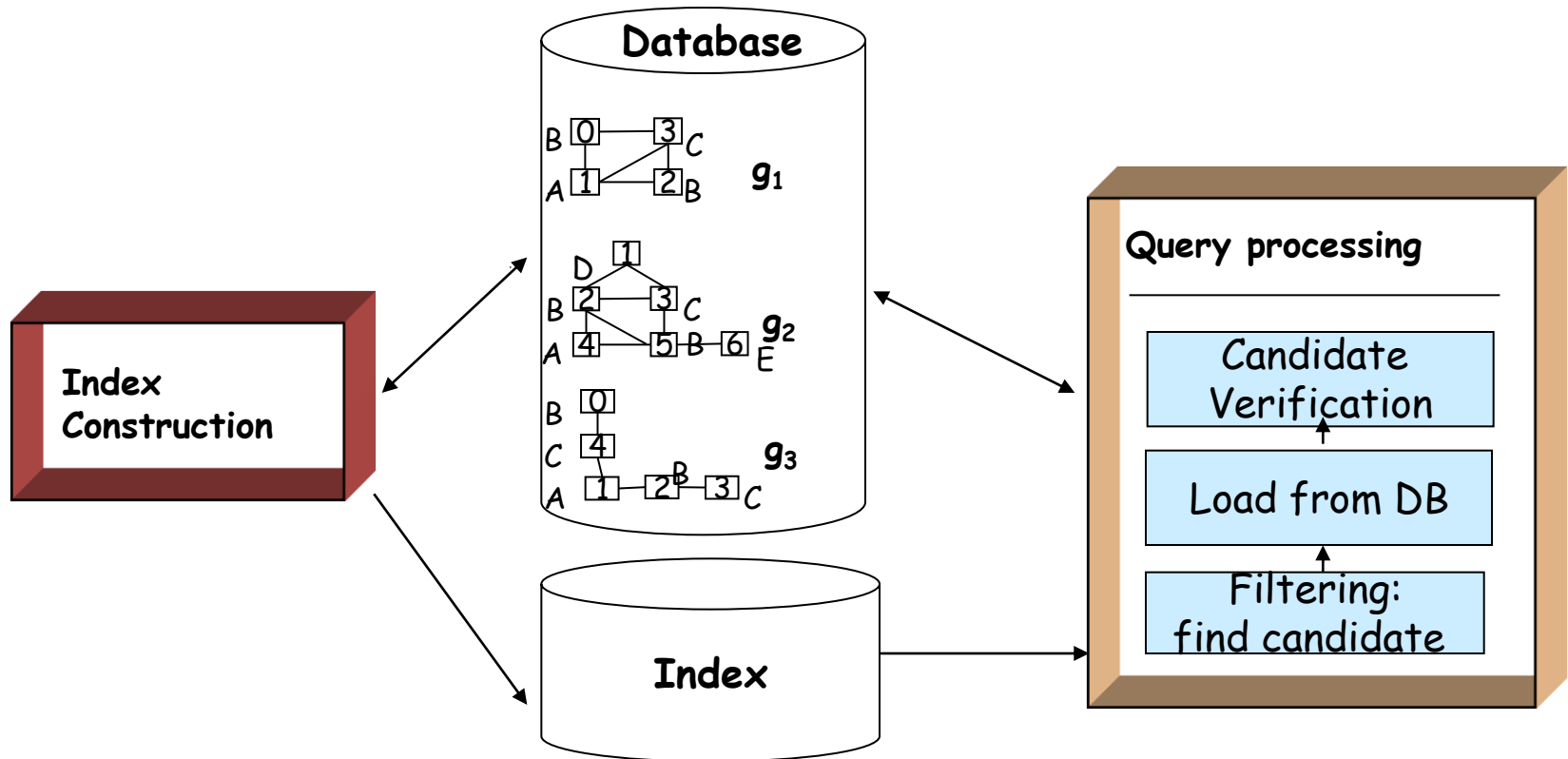
Gene Ontologies

Graph indexing system

- Graph-to-graph matching algorithms can be used, efficiency considerations suggest the use of **specific techniques to reduce the search space and the time complexity.**
- In a **preprocessing phase**, each graph of the database is analyzed in order to extract and store its **discriminatory properties, features.**
- In the **filtering phase**, the graph database index is compared with the query index in order to discard graphs of the database not containing some features present in the query graph.

GraphGrep

Graphs searching is an NP-problem



Indexing is crucial to **reduce the search** space and make the problem affordable!

Shasha D, Wang JL, Giugno R: Algorithmics and Applications of Tree and Graph Searching. Proceeding of the ACM Symposium on Principles of Database Systems (PODS) 2002, :39{52.

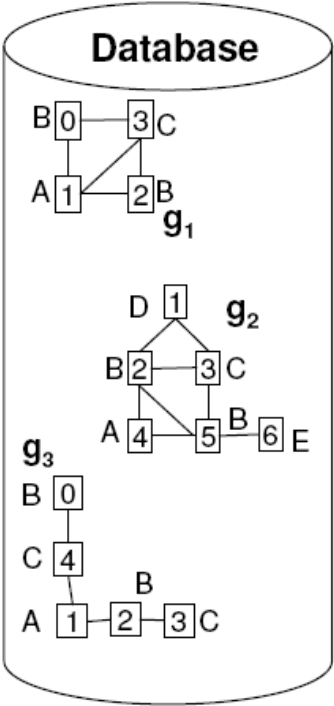
A. Ferro, R. Giugno, M. Mongiovi, A. Pulvirenti, D. Skripin, D. Shasha D. GraphFind: Enhancing Graph Searching by Low Support Data Mining Techniques. *BMC Bioinformatics* 2008, Vol. 9 (Suppl 4) :S10 doi:10.1186/1471-2105-9-S4-S10

GraphGrep: Index building

For each graph in DB:

- **Find** all paths of length from 1 to L (4,10)
- **Save** the paths in a Berkeley DB
- **Count** how many occurrences of each path in each graph
- **Save** the occurrences in an hash table indexed by the strings of the paths

GraphGrep: Index building



Data Storing

(a) Label-Path-Sets (id-paths)

Berkeley DB Tables

A (1)	CB (3,0) (3,2)	A (4)	CB (3,5) (3,2)	A (1)	CB (4,0) (3,2)
BA (0,1) (2,1)	ABC (1,2,3) (1,0,3)	BA (2,4) (5,4)	ABC (4,2,3) (4,5,3)	BA (2,1)	ABC (1,2,3)
ABCA (1,0,3,1) (1,2,3,1)
g₁		g₂		g₃	

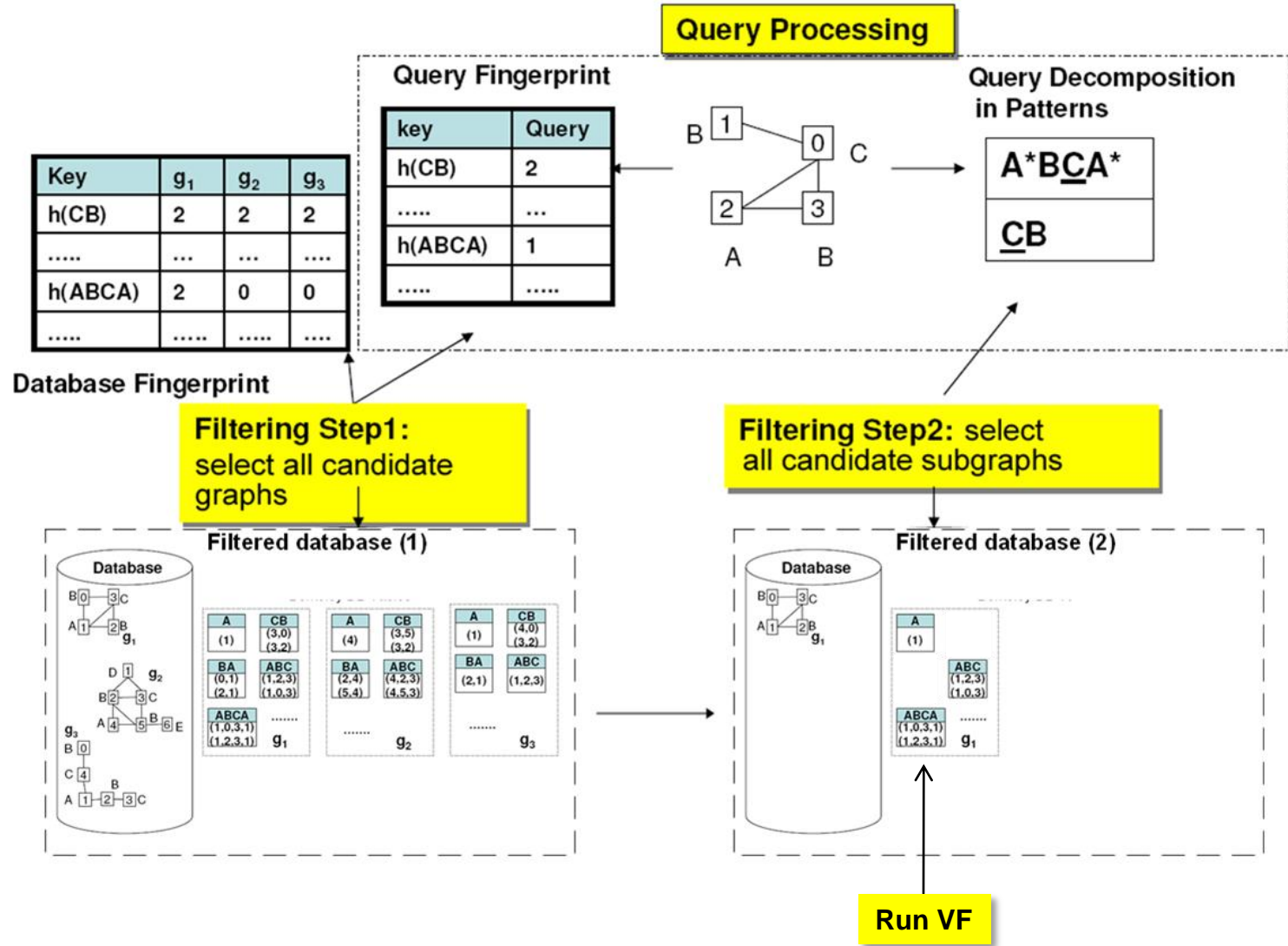
Index

Key	g ₁	g ₂	g ₃
h(CB)	2	2	2
....
h(ABCA)	2	0	0
....

(b) Fingerprint

Berkeley DB HashTable

GraphGrep: Filtering and Matching



GraphGrep: Matching VF_lib

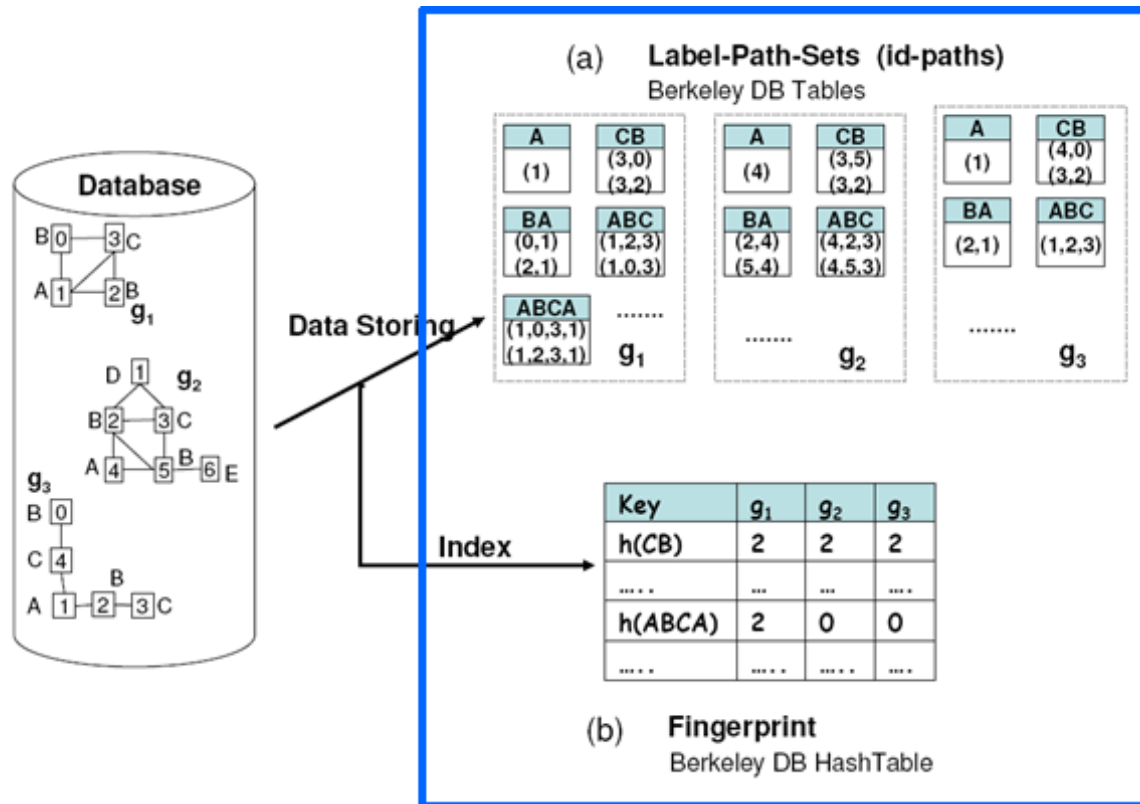
(Cordella et al. IEEE PAMI 2004,

<http://amalfi.dis.unina.it/graph/db/vflib-2.0/doc/vflib.html>)

- Extension of Ullmann matching algorithm (Journal of the ACM, 1976)
- The process of finding the mapping function can be suitably described by means of a TREE called State Space Representation (SSR)
- Each node is a state s of the partial matching process
- Transition from a generic state s to a successor s' represents the addition of a pair matched nodes.
- **k-look-ahead rules** for checking in advance if a consistent state s has no consistent successors after k steps + **Semantic rules**

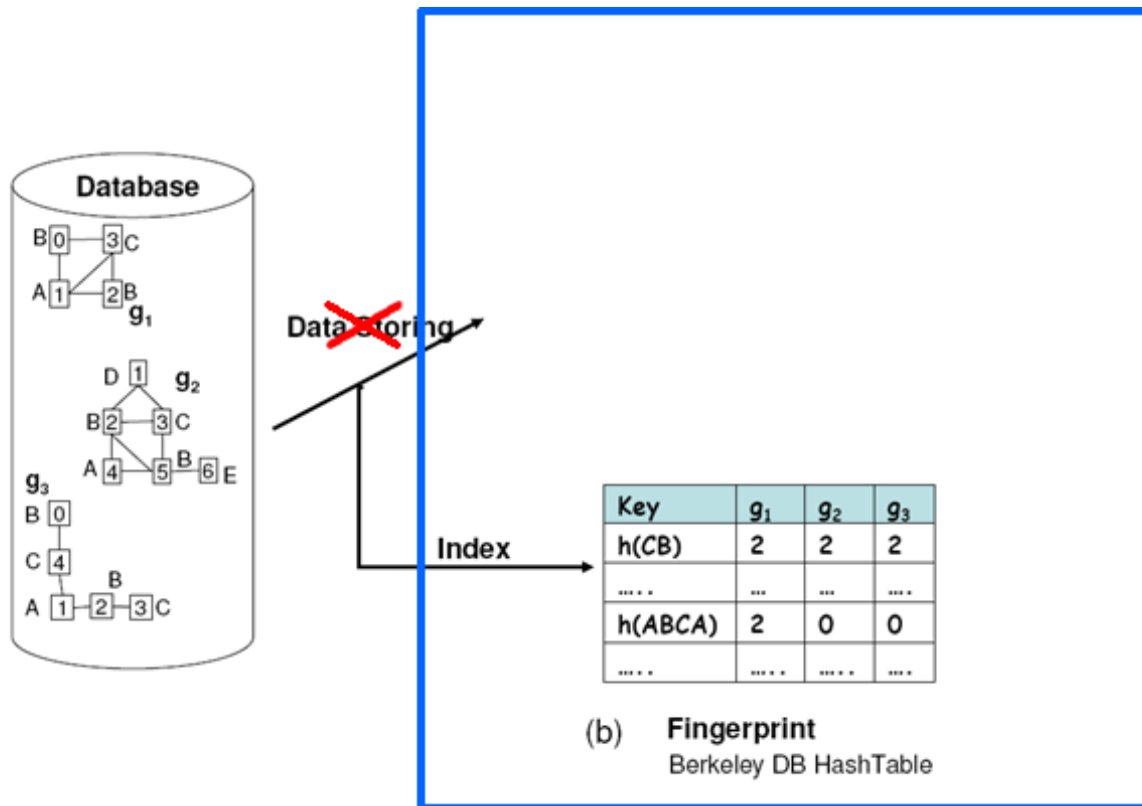
GraphGrepSX: Idea

Realize a compact representation of the index by making use of Suffix trees



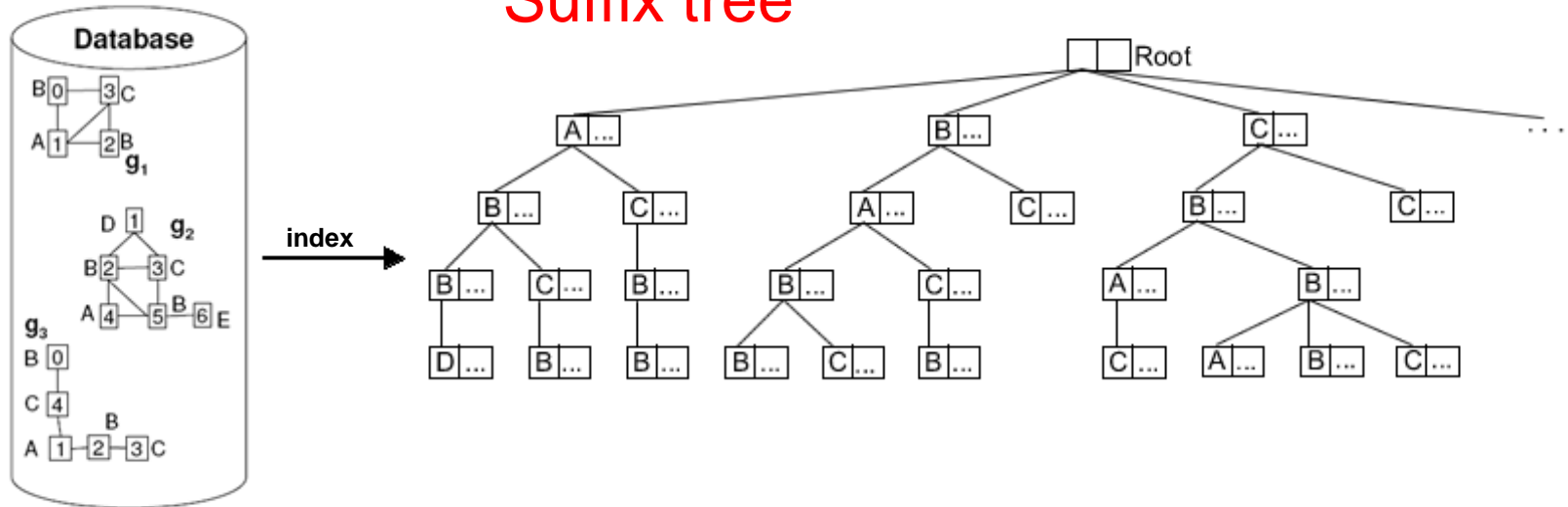
GraphGrepSX: Idea

Realize a compact representation of the index by making use of Suffix trees



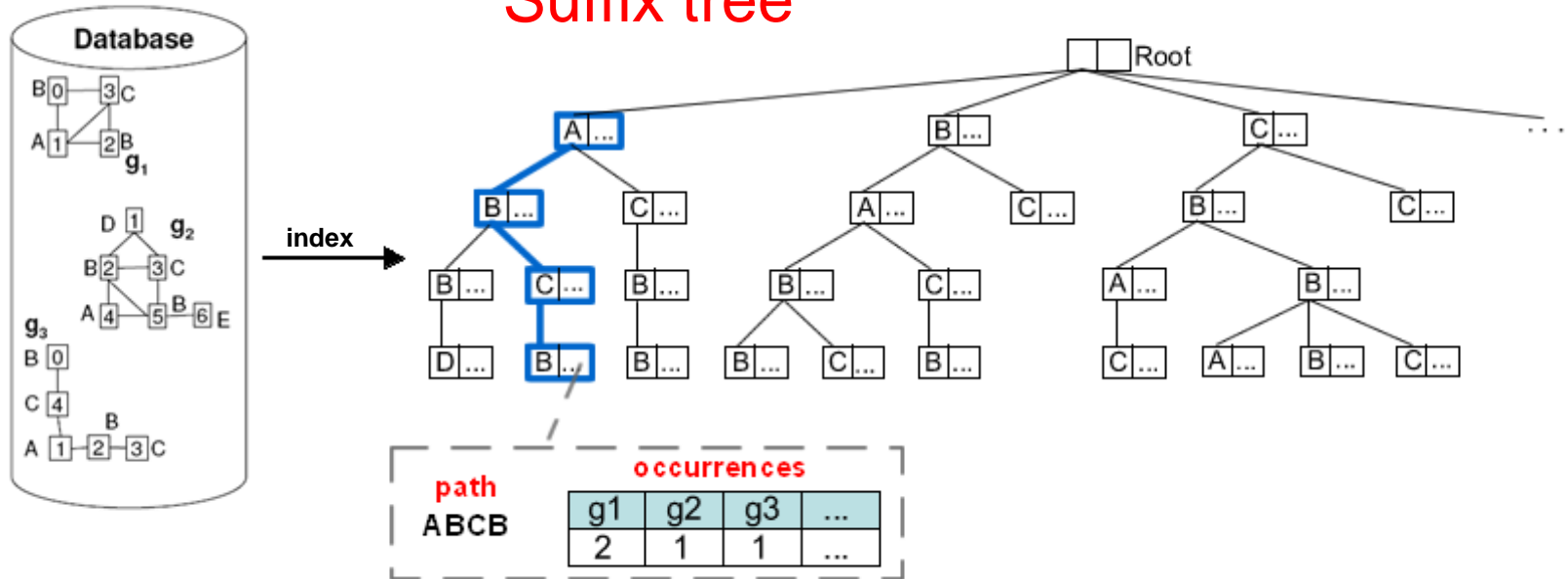
GraphGrepSX: Idea

Realize a compact representation of the index by making use of Suffix trees



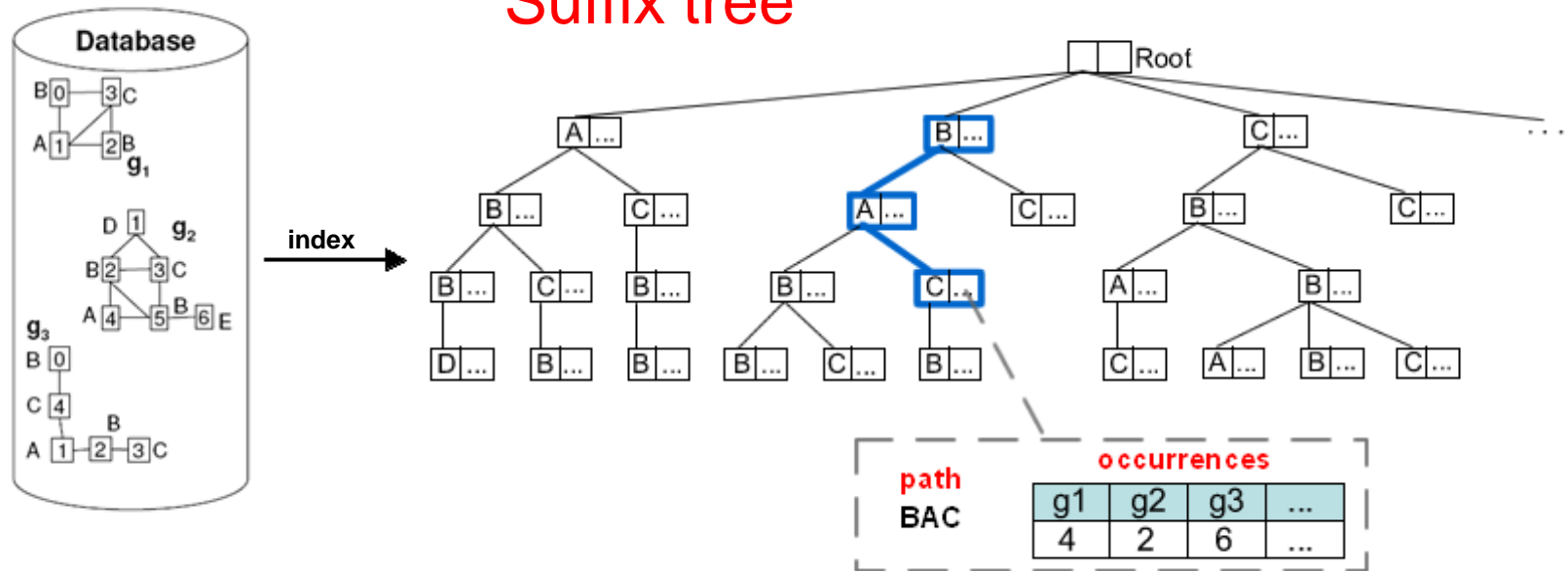
GraphGrepSX: Idea

Realize a compact representation of the index by making use of Suffix trees



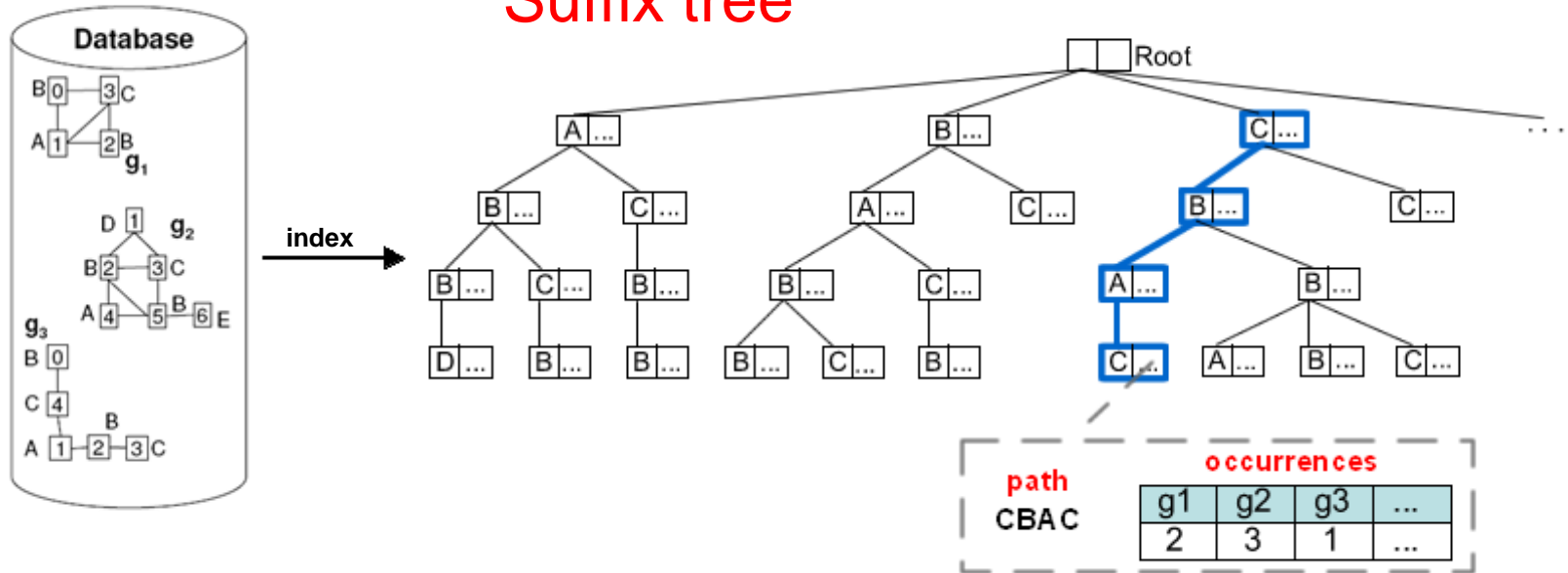
GraphGrepSX: Idea

Realize a compact representation of the index by making use of Suffix trees



GraphGrepSX: Idea

Realize a compact representation of the index by making use of Suffix trees

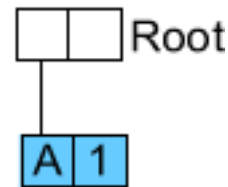
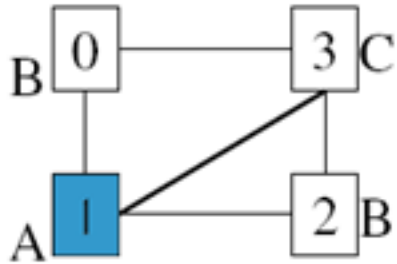


GraphGrepSX

- Preprocessing phase
 - replaces the hash table index by a **suffix tree index**
- Filtering phase
 - Build a **query index tree**
 - The candidate set is constructed by **matching the query index tree and the database index**
- This results in a **more flexible graph indexing system**
 - different ways to build the query index
 - an efficient technique to reduce redundant checks

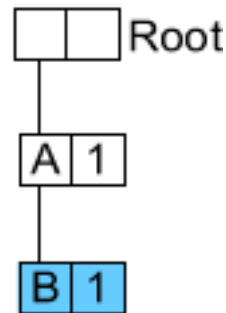
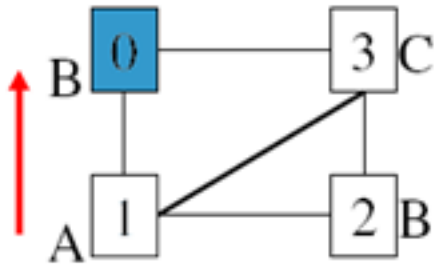
GraphGrepSX: Index structure

GraphGrepSX builds the tree index as follows:



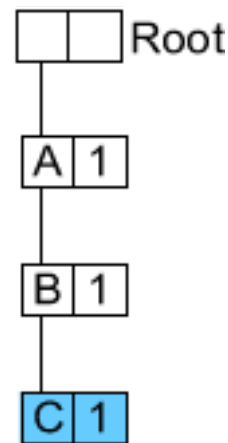
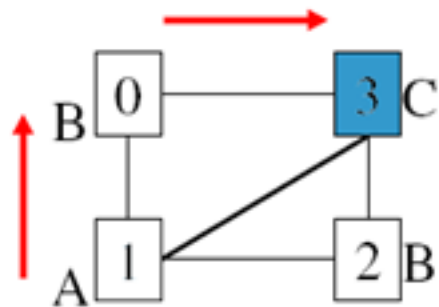
GraphGrepSX : Index structure

GraphGrepSX builds the tree index as follows:



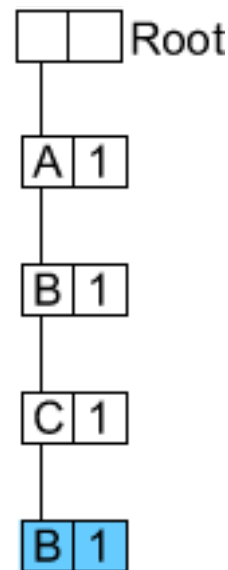
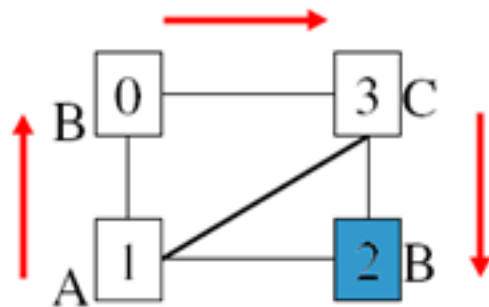
GraphGrepSX : Index structure

GraphGrepSX builds the tree index as follows:



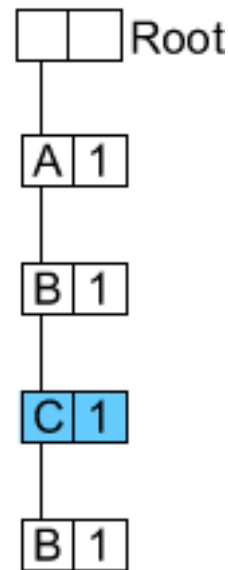
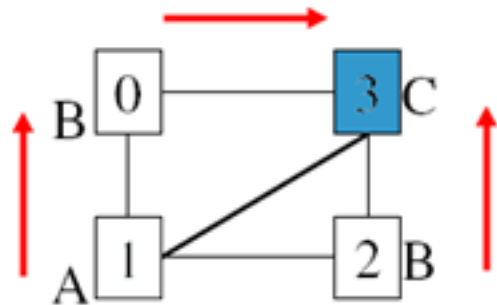
GraphGrepSX : Index structure

GraphGrepSX builds the tree index as follows:



GraphGrepSX : Index structure

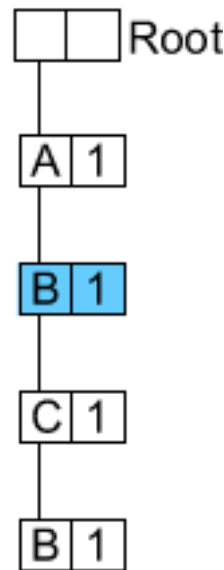
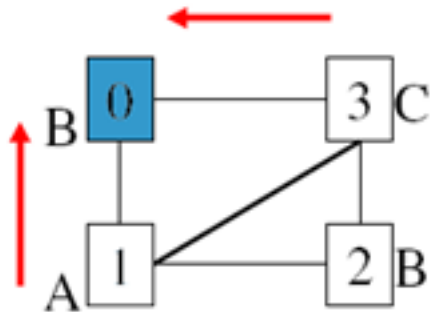
GraphGrepSX builds the tree index as follows:



Computed by DFS visit, the backtracking allows to find paths with the same suffix

GraphGrepSX : Index structure

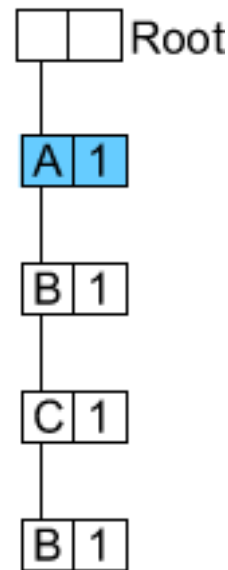
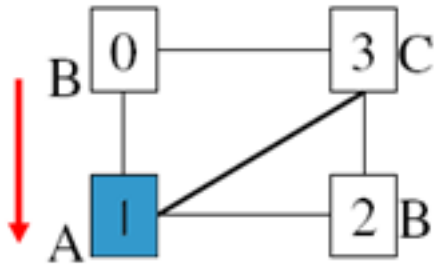
GraphGrepSX builds the tree index as follows:



Computed by DFS visit, the backtracking allows to find paths with the same suffix

GraphGrepSX : Index structure

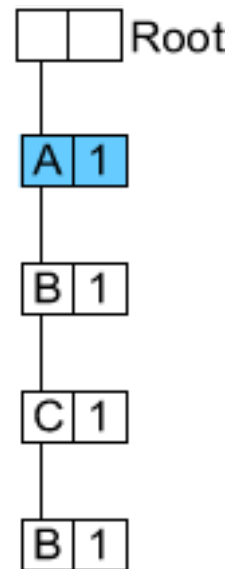
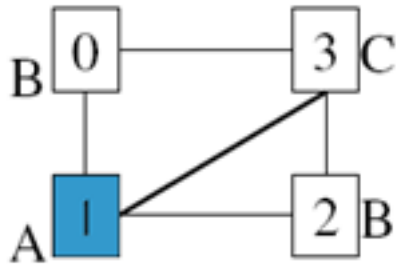
GraphGrepSX builds the tree index as follows:



Computed by DFS visit, the backtracking allows to find paths with the same suffix

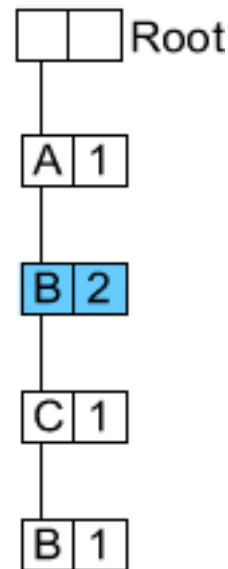
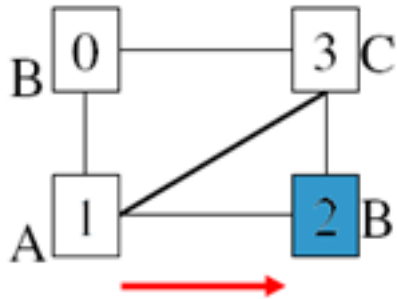
GraphGrepSX : Index structure

GraphGrepSX builds the tree index as follows:



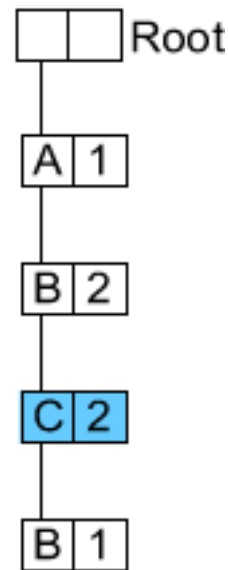
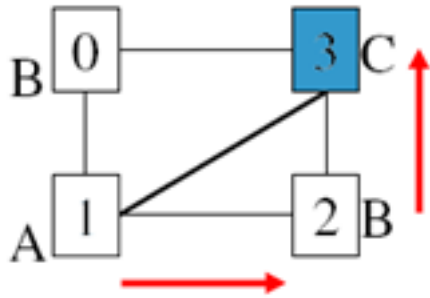
GraphGrepSX : Index structure

GraphGrepSX builds the tree index as follows:



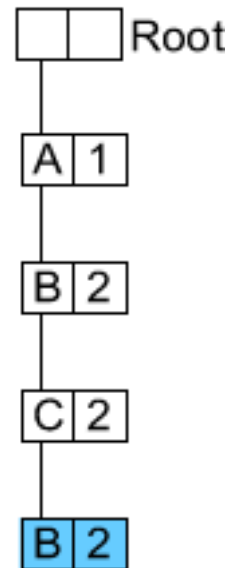
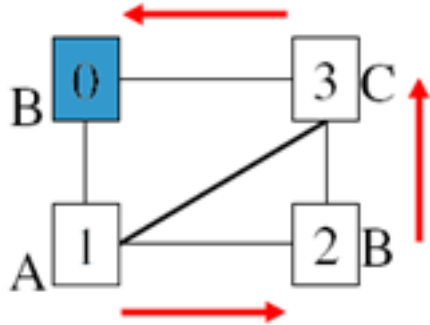
GraphGrepSX : Index structure

GraphGrepSX builds the tree index as follows:



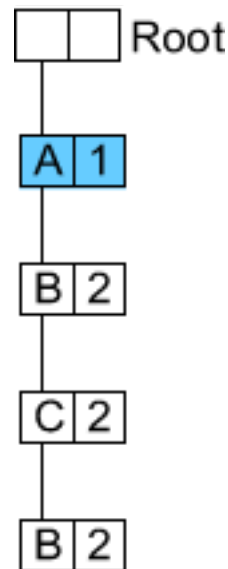
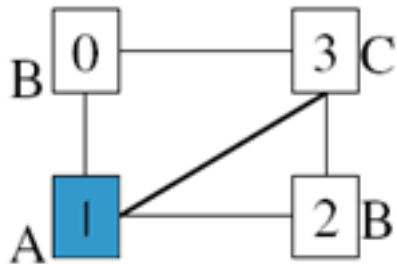
GraphGrepSX : Index structure

GraphGrepSX builds the tree index as follows:



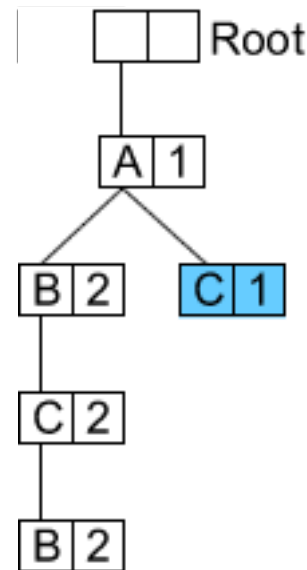
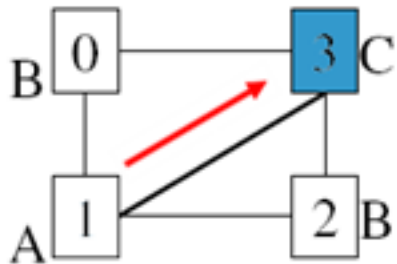
GraphGrepSX : Index structure

GraphGrepSX builds the tree index as follows:



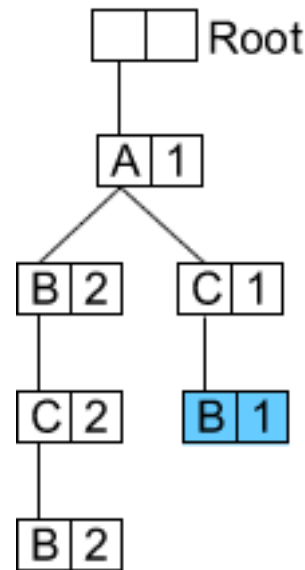
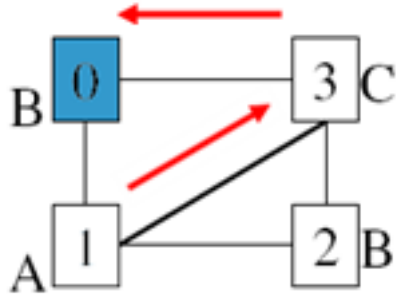
GraphGrepSX : Index structure

GraphGrepSX builds the tree index as follows:



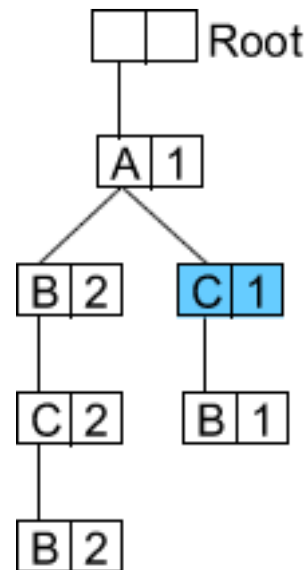
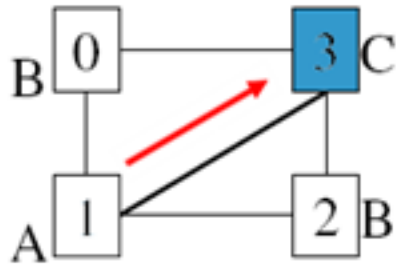
GraphGrepSX : Index structure

GraphGrepSX builds the tree index as follows:



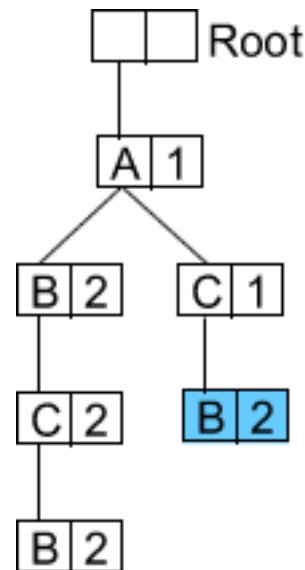
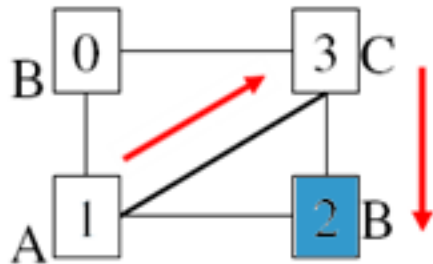
GraphGrepSX : Index structure

GraphGrepSX builds the tree index as follows:

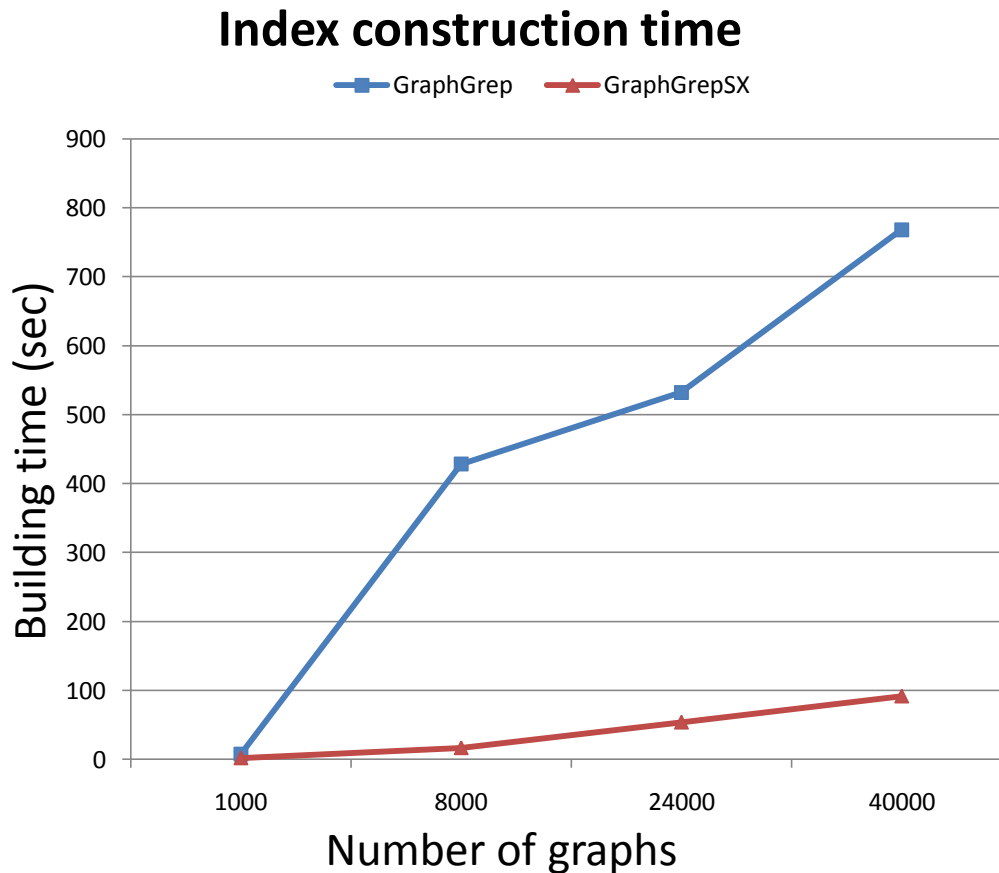


GraphGrepSX : Index structure

GraphGrepSX builds the tree index as follows:

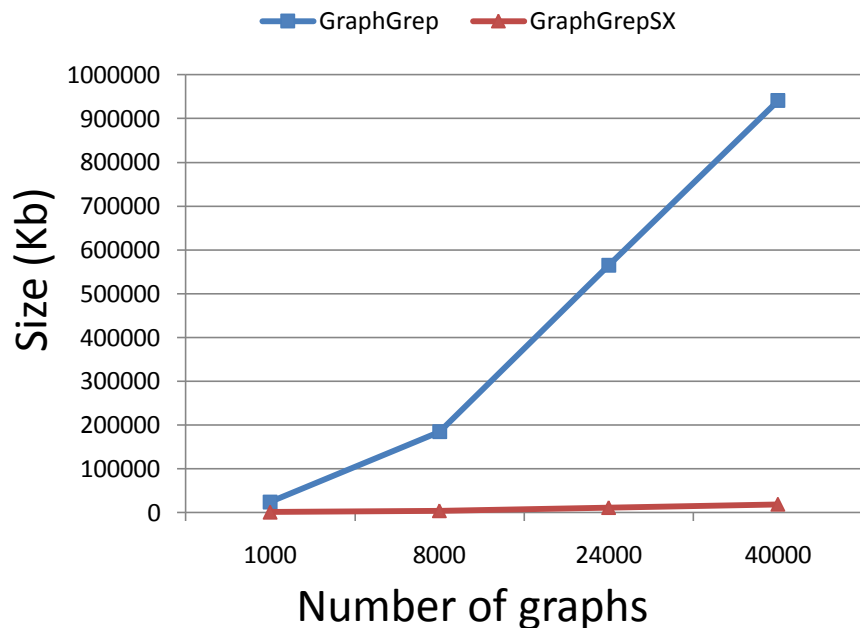


Experimental analysis on molecular dataset

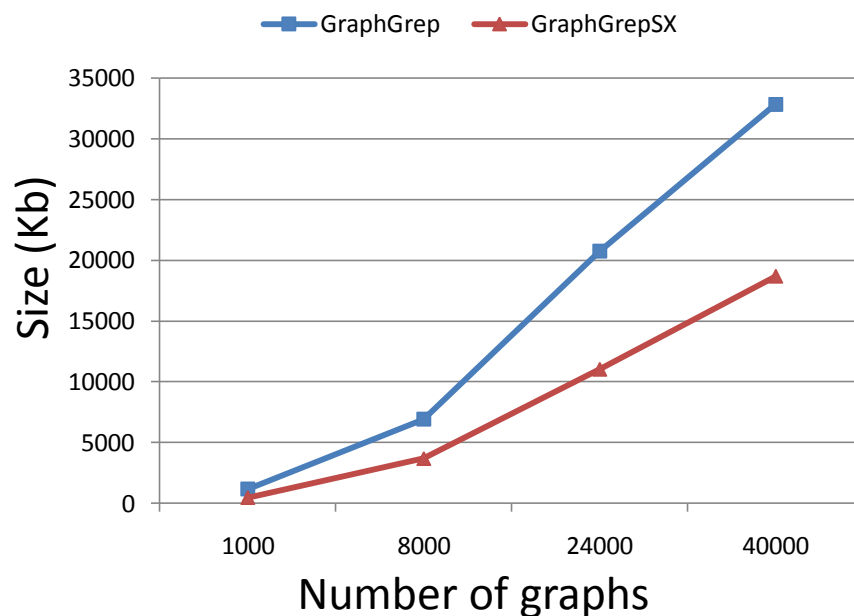


Experimental analysis on molecular dataset

Total index size label-paths table + hashtable



Index fingerprint size hashtable only



GraphGrepSX: Index structure

A path in the index structure is defined as [maximal path](#) if:

GraphGrepSX: Index structure

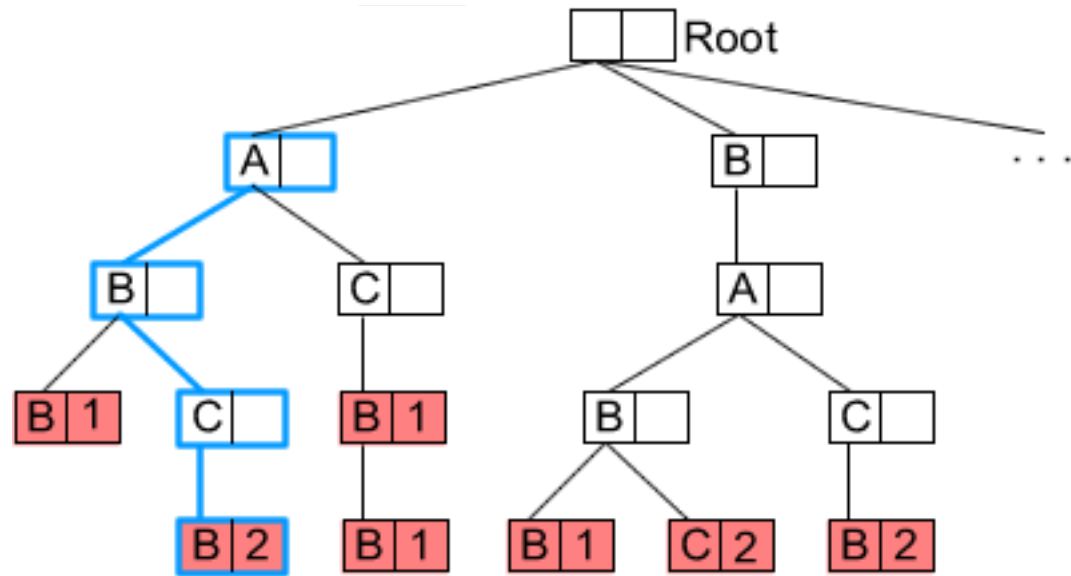
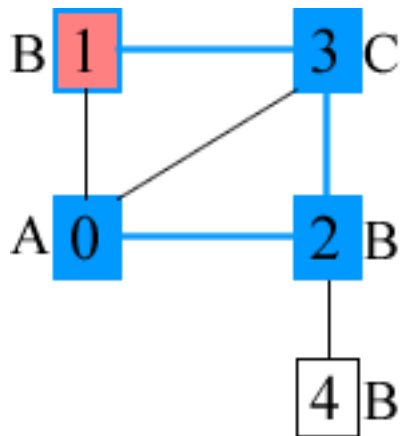
A path in the index structure is defined as [maximal path](#) if:

- its length is L

GraphGrepSX: Index structure

A path in the index structure is defined as **maximal path** if:

- its length is L



GraphGrepSX: Index structure

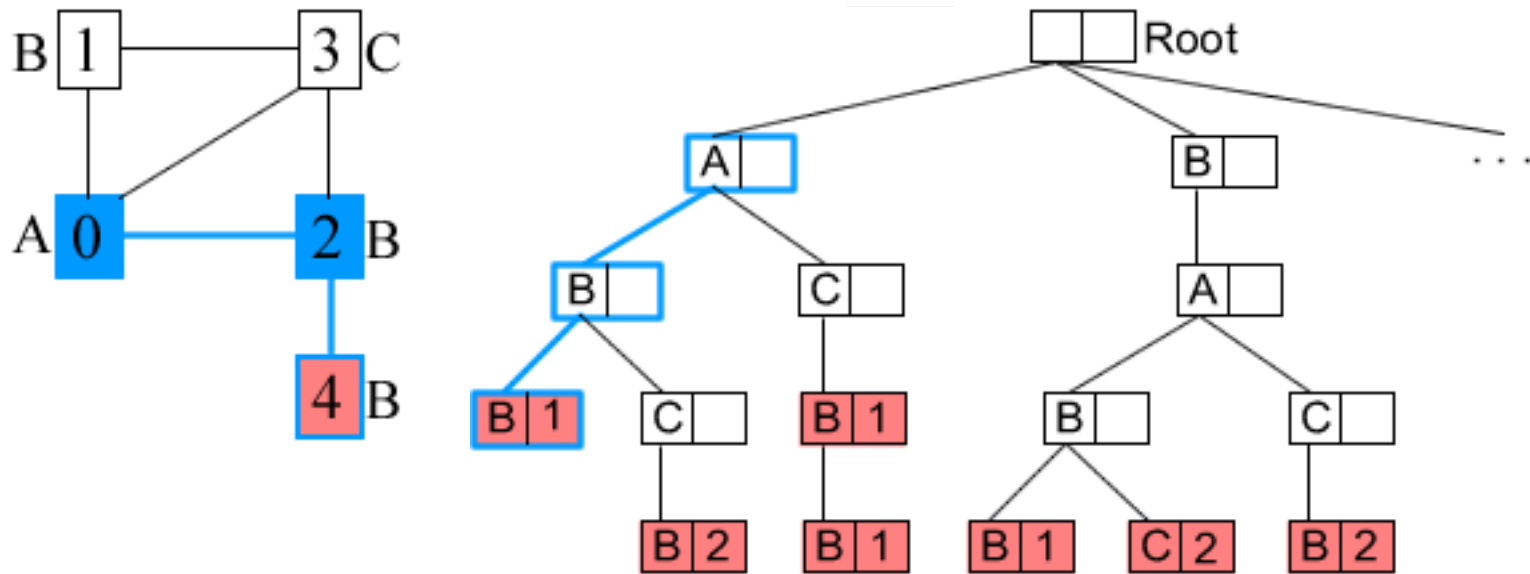
A path in the index structure is defined as **maximal path** if:

- its length is L
- the path has length $< L$ but it cannot be extended

GraphGrepSX: Index structure

A path in the index structure is defined as **maximal path** if:

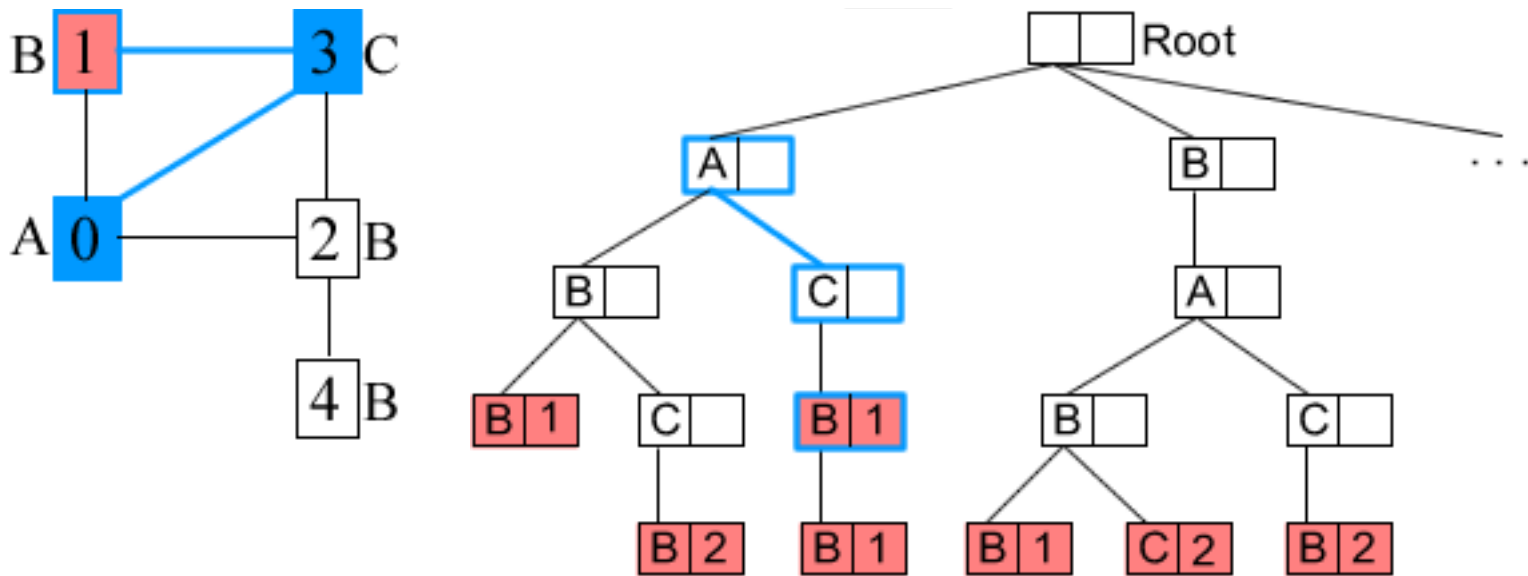
- its length is L
- the path has length $< L$ but it cannot be extended



GraphGrepSX: Index structure

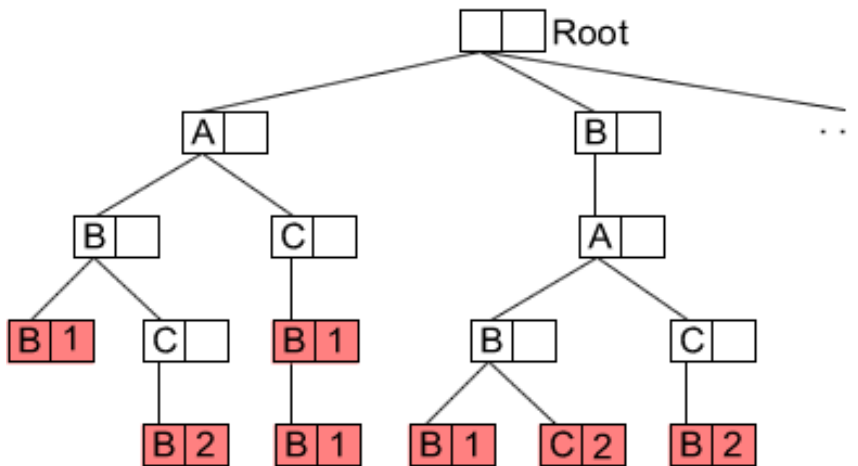
A path in the index structure is defined as **maximal path** if:

- its length is L
- the path has length $< L$ but it cannot be extended



GraphGrepSX: Filtering phase-Query tree index structure construction

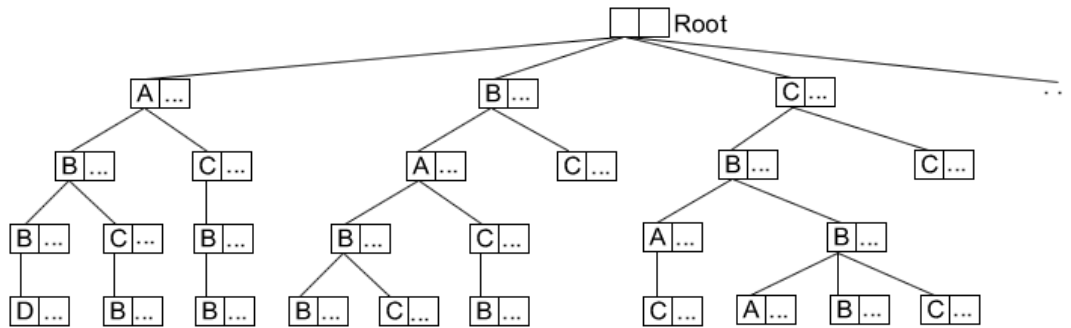
- Discard graphs from the database which do not match the query by analyzing only the **maximal paths**
- the query tree nodes representing these **maximal paths** are **marked**



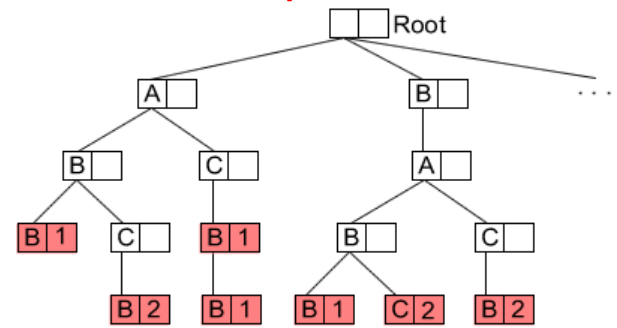
Red nodes represent
End-points of Maximal Paths

GraphGrepSX: Filtering phase-candidates generation

Dataset index

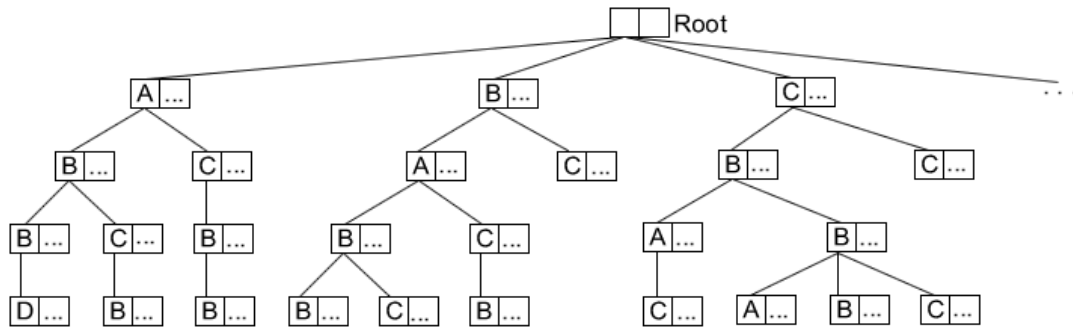


Query index

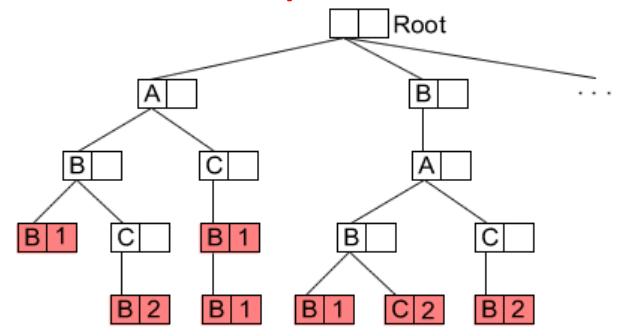


GraphGrepSX: Filtering phase-candidates generation

Dataset index



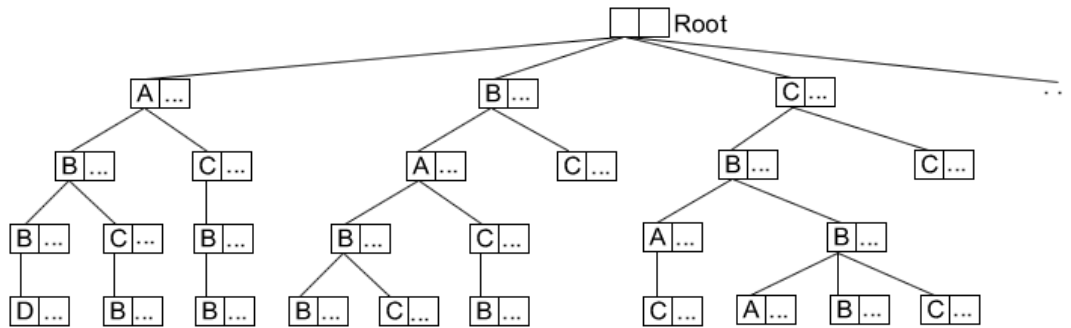
Query index



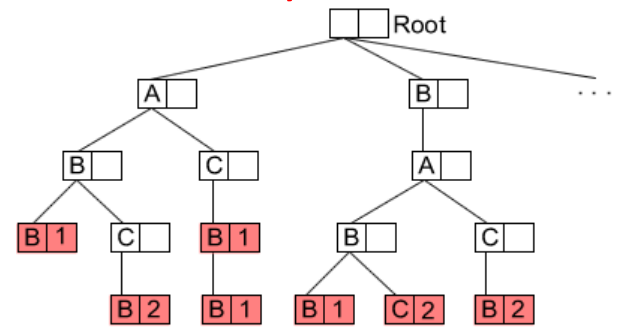
Trees matching

GraphGrepSX: Filtering phase-candidates generation

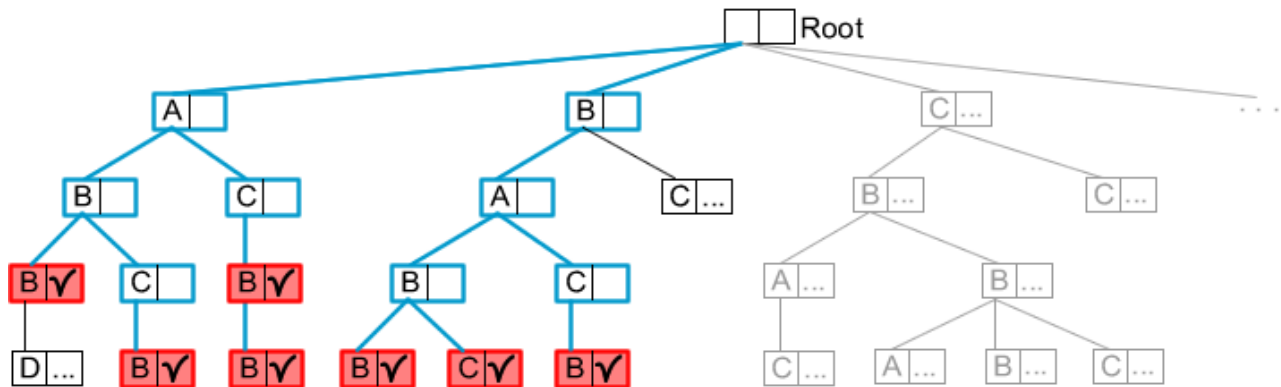
Dataset index



Query index

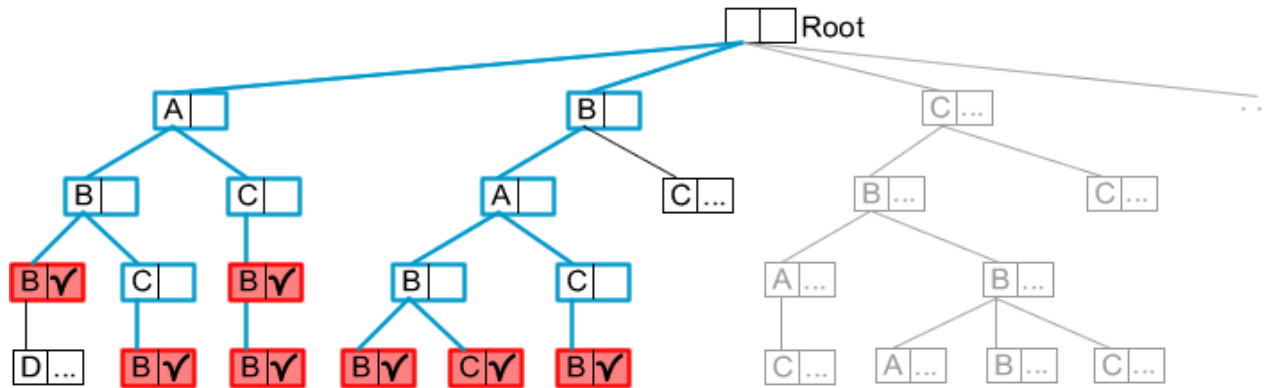


Trees matching

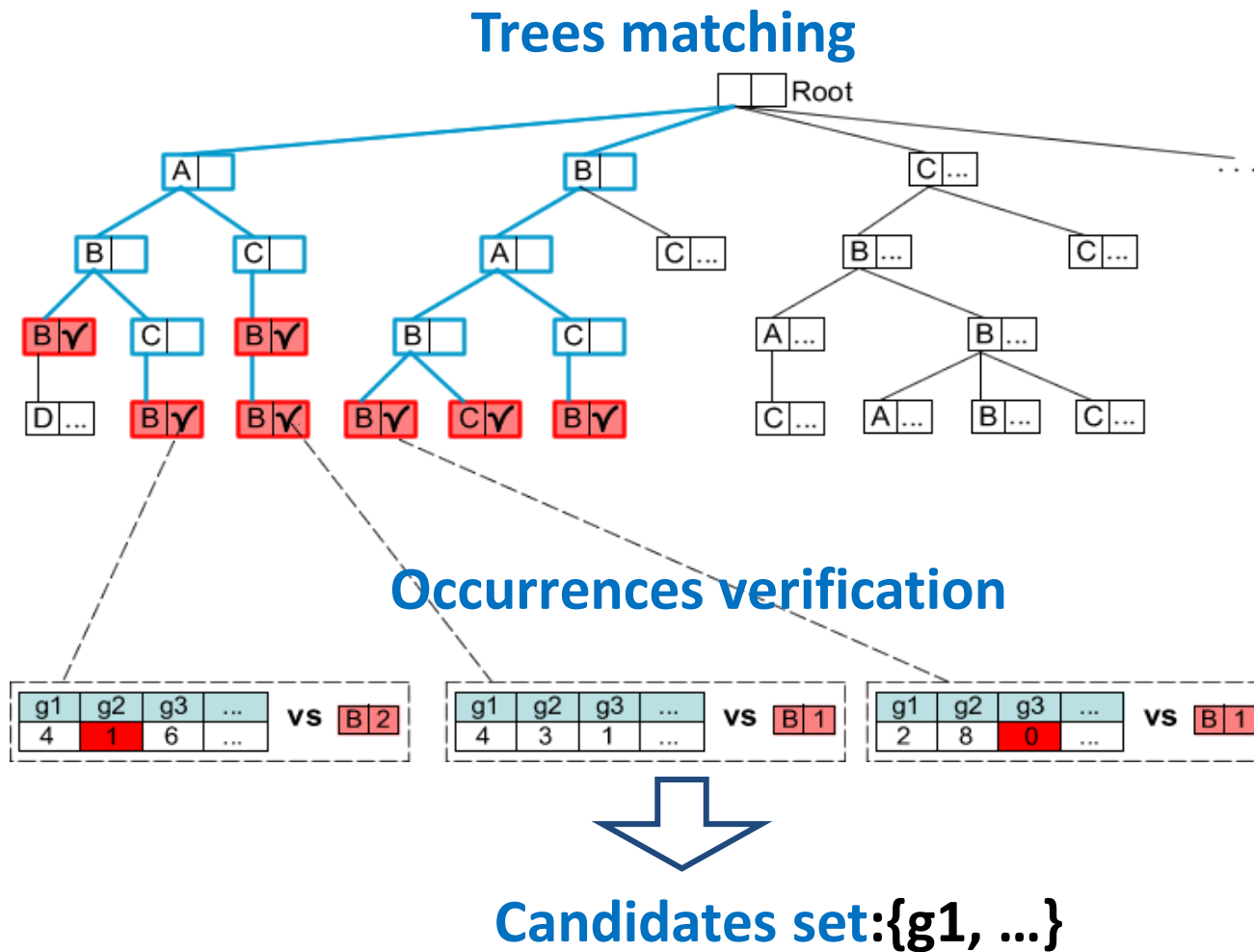


GraphGrepSX: Filtering phase-candidates generation

Trees matching

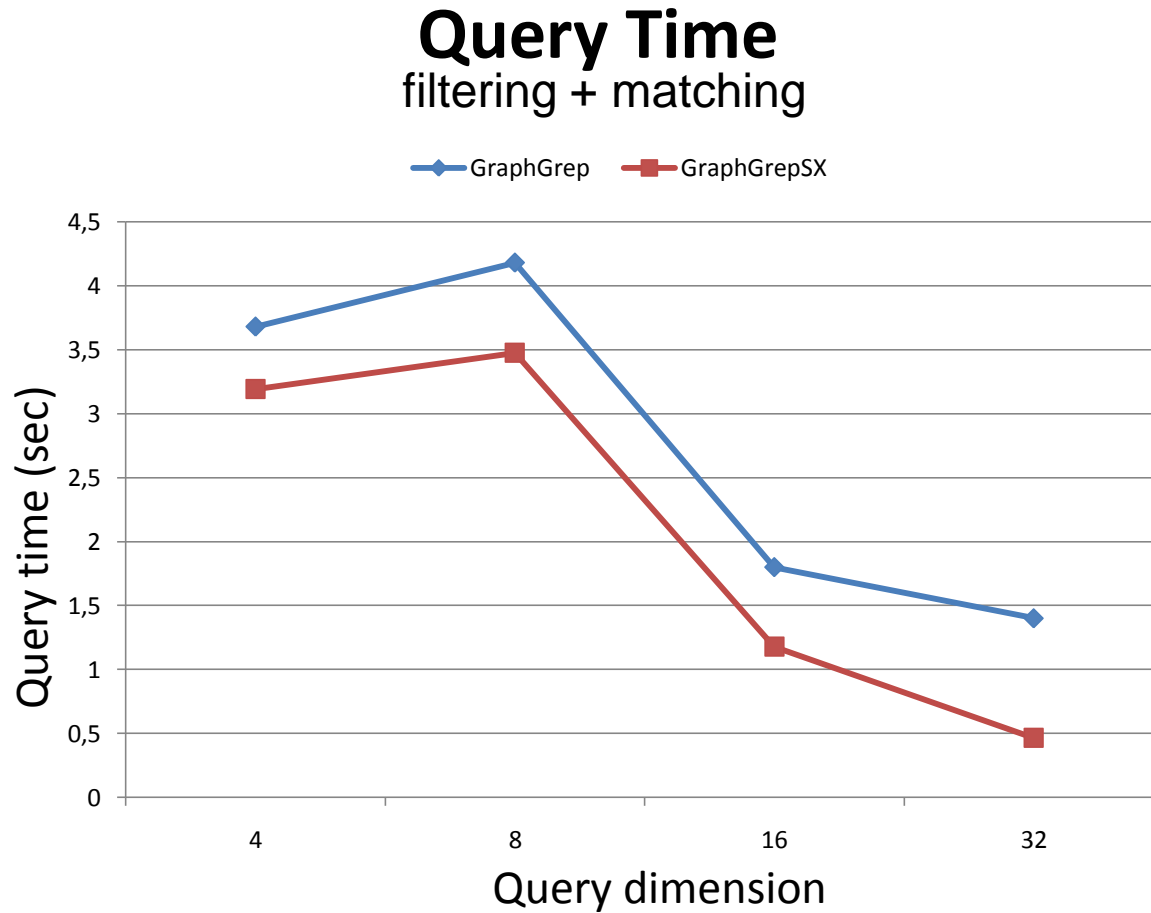


GraphGrepSX: Filtering phase-candidates generation



Experimental analysis

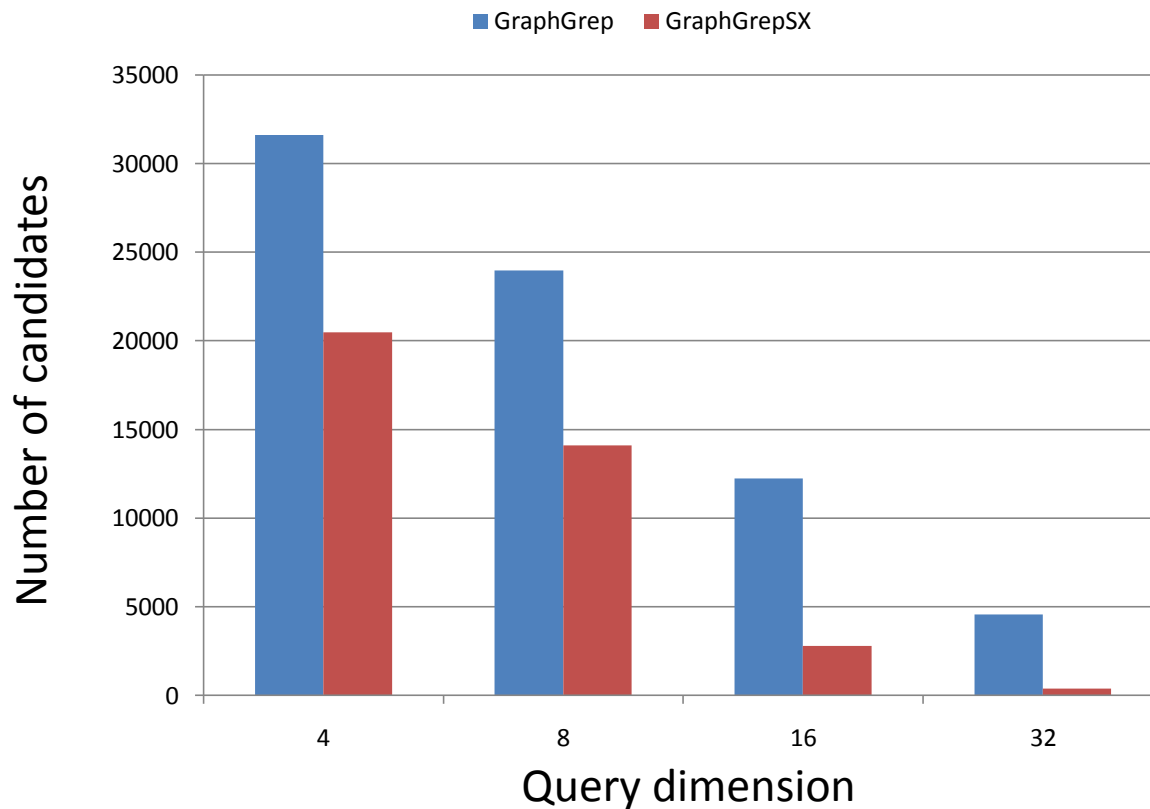
Molecular dataset of 42000 graphs



Experimental analysis

Molecular dataset of 42000 graphs

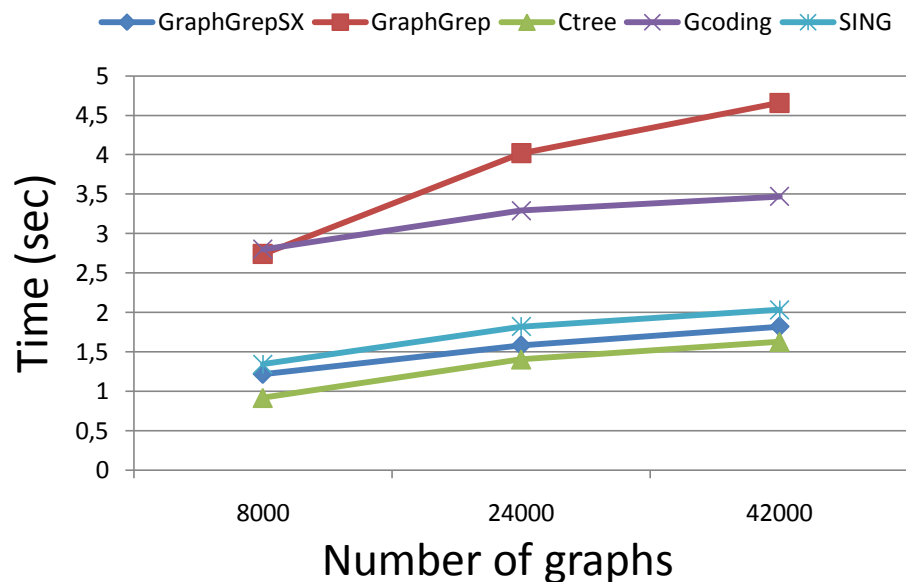
Candidates



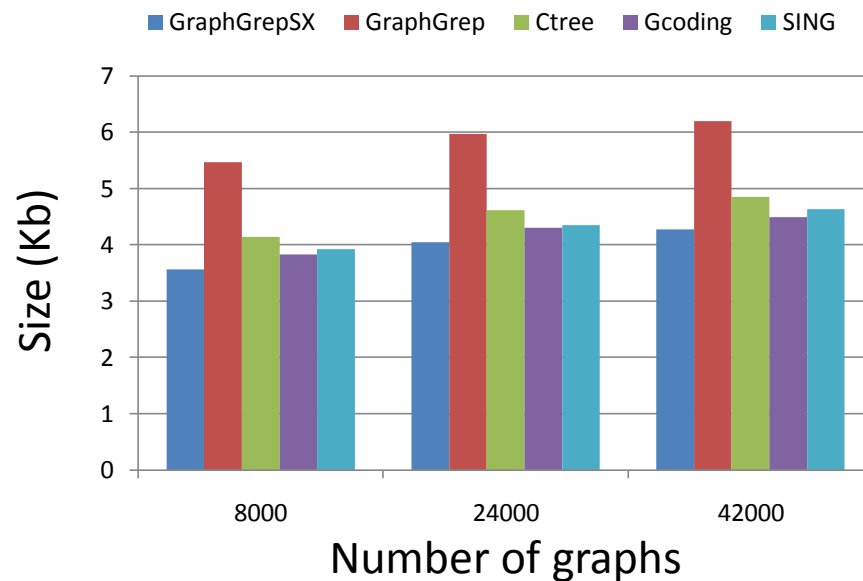
Experimental analysis: CTree, GCoding, GraphGrep, GraphGrepSX

Molecular dataset of 42000 graphs

Index construction time



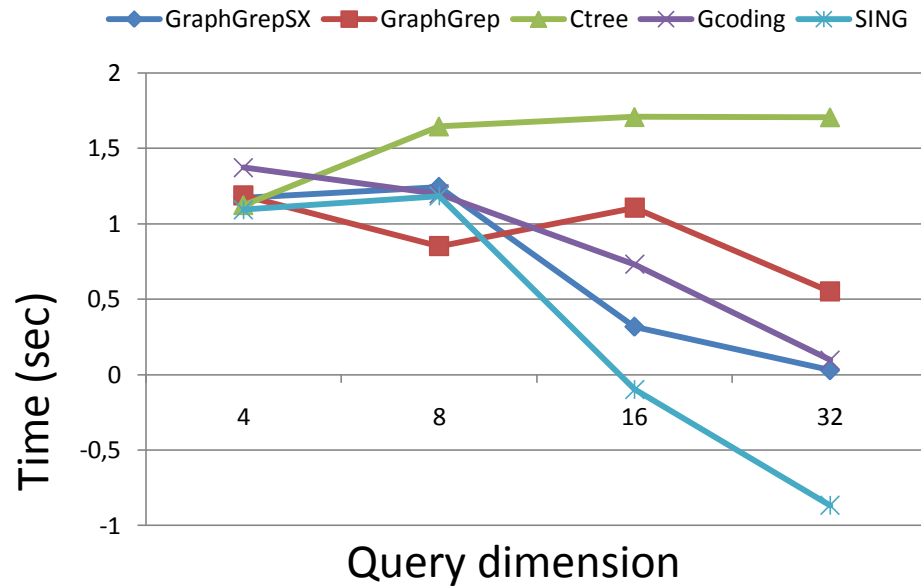
Index size



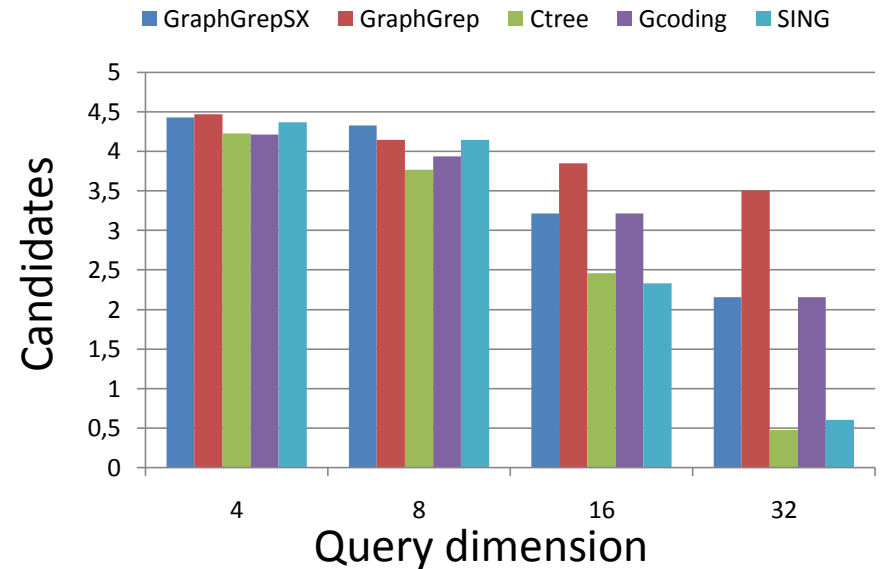
Experimental analysis: CTree, GCoding, GraphGrep, GraphGrepSX

Molecular dataset of 42000 graphs

Query time



Candidates



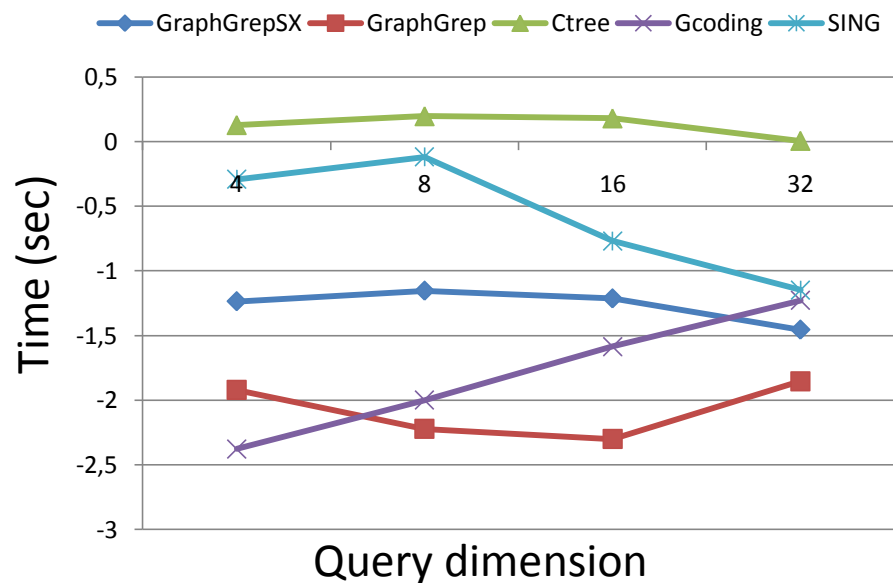
Huahai He Ambuj K. Singh, Closure-Tree: An Index Structure for Graph Queries, ICDE '06

Lei Zou, Lei Chen, Jeffrey Xu Yu, Yansheng Lu, A novel spectral coding in a large graph database, Proceedings of the 11th international conference on Extending database technology, 2008

Experimental analysis: CTree, GCoding, GraphGrep, GraphGrepSX

Molecular dataset of 42000 graphs

Filtering time



Matching time

