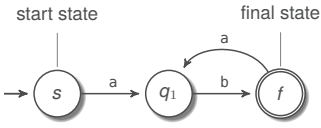
An Analysis of the Feasibility of Graph Compression Techniques for Indexing Regular Path Queries

Frank Tetzel, Hannes Voigt, Marcus Paradies, Wolfgang Lehner

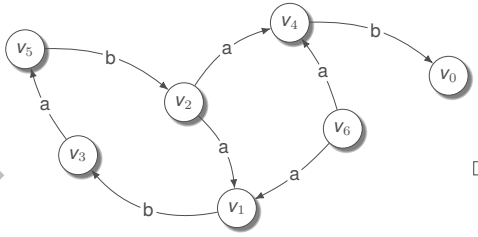
May 19, 2017

Regular Path Queries (RPQs)

- Matching paths conforming to regular expression
- Only distinct (start, end) vertex pairs in result set



Automaton representing (ab)+



Search in data graph

start	end
v_2	v_0, v_2, v_3
v_3	v_0, v_2, v_3
v_6	v_0, v_2, v_3

Final result set

Processing strategies

Baseline

- Guided search with automaton on data graph
- Adjacency list on column store

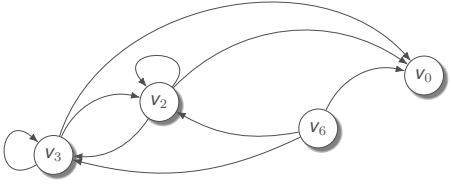
Processing strategies

Baseline

- Guided search with automaton on data graph
- Adjacency list on column store

MR-Index

- Store results of RPQs for future use
- Treat vertex pairs as edges of a reachability graph
- Use graph compression for reachability graph



Reachability graph

K^2-tree graph compression

Dresden Database Systems Group

- Compact representation of a binary relation, e.g., an adjacency matrix
- Hierarchical graph compression

	0	1	2	3	4	5	6	
0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0
2	1	0	1	1	0	0	0	0
3	1	0	1	1	0	0	0	0
4	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0
6	1	0	1	1	0	0	0	0
	0	0	0	0	0	0	0	0

Adjacency matrix

K^2-tree graph compression

- Compact representation of a binary relation, e.g., an adjacency matrix
- Hierarchical graph compression

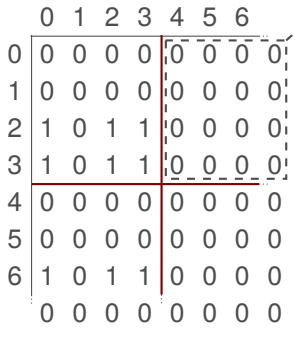
	0	1	2	3	4	5	6	
0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0
2	1	0	1	1	0	0	0	0
3	1	0	1	1	0	0	0	0
4	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0
6	1	0	1	1	0	0	0	0
	0	0	0	0	0	0	0	0

Adjacency matrix

Conceptual K^2-tree for $k = 2$

K^2-tree graph compression

- Compact representation of a binary relation, e.g., an adjacency matrix
- Hierarchical graph compression

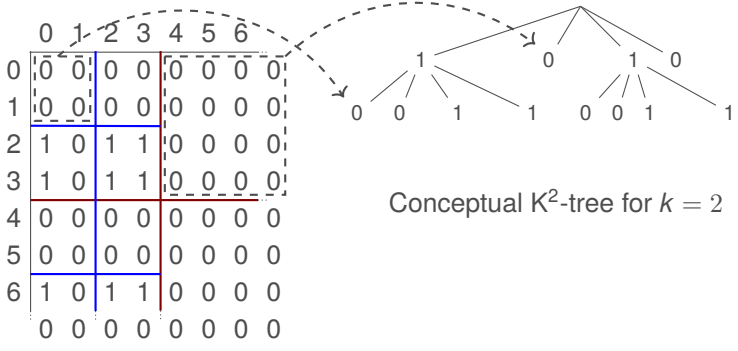


Adjacency matrix

Conceptual K^2-tree for $k = 2$

K^2-tree graph compression

- Compact representation of a binary relation, e.g., an adjacency matrix
- Hierarchical graph compression

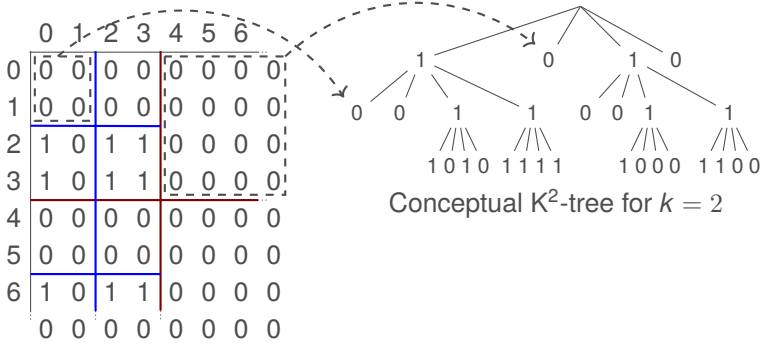


Adjacency matrix

Conceptual K^2-tree for $k = 2$

K^2-tree graph compression

- Compact representation of a binary relation, e.g., an adjacency matrix
- Hierarchical graph compression

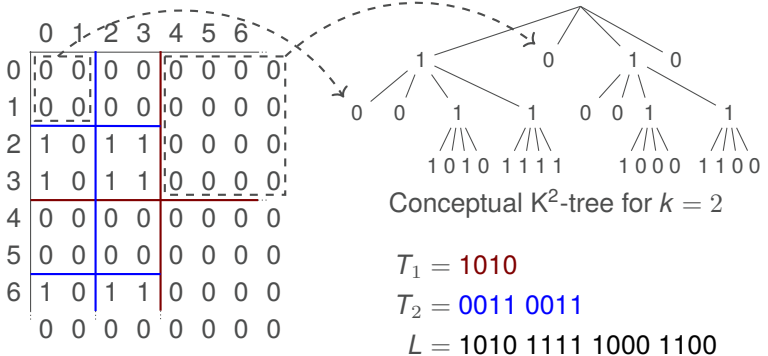


Adjacency matrix

Conceptual K^2-tree for $k = 2$

K^2-tree graph compression

- Compact representation of a binary relation, e.g., an adjacency matrix
- Hierarchical graph compression



Adjacency matrix

Conceptual K^2-tree for $k = 2$

$T_1 = 1010$

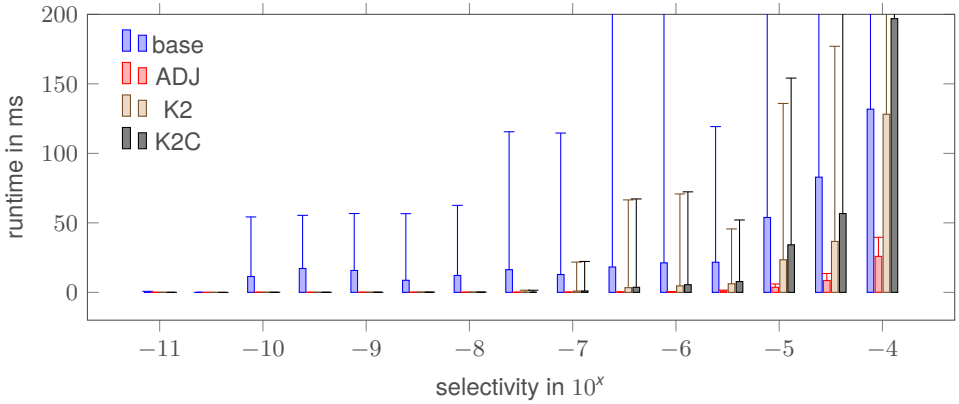
$T_2 = 0011\ 0011$

$L = 1010\ 1111\ 1000\ 1100$

Bitvector representation

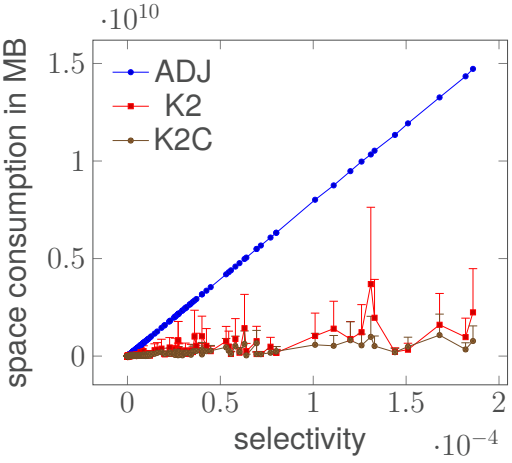
Measurements

- LDBC social network dataset with scale factor 1 (3 million vertices and 17 million edges)
- generate all possible RPQs with path length up to 3 hops, and recursions of them



(ADJ - adjacency list, K2 - K^2-tree, K2C - K^2-tree with leaf compression)

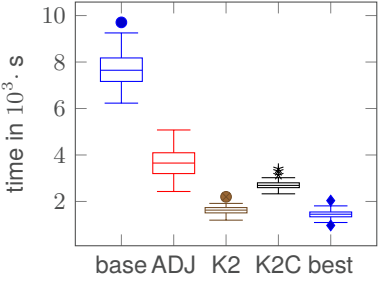
Space consumption

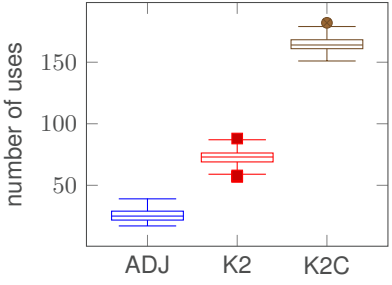


Space consumption of queries

Measurements

- Batch of 300 queries, memory budget of 10 GB





Batch processing with sampled query sets

Conclusion and Future Work

- Graph compression promising for storing reachability information
- K^2-trees not beneficial for all results sets
  - Too much overhead for tiny result sets
  - No good compression for huge result sets

Future Work

- Experiment with other K^2-trees to compress uniform 1-regions as well
- Improve access time by providing specialized range queries, extracting submatrices
- Compare memory consumption and access time with other compact reachability indices like FERRARI

Conclusion and Future Work

- Graph compression promising for storing reachability information
- K^2-trees not beneficial for all results sets
  - Too much overhead for tiny result sets
  - No good compression for huge result sets

Future Work

- Experiment with other K^2-trees to compress uniform 1-regions as well
- Improve access time by providing specialized range queries, extracting submatrices
- Compare memory consumption and access time with other compact reachability indices like FERRARI

Thank You