Efficient Regular Path Query Evaluation in PGX

Author:
Xuming Meng

Supervisor:
dr. G.H.L. Fletcher

15-08-2016

Technische Universiteit
Eindhoven
University of Technology

Technische Universiteit
Eindhoven
University of Technology

Where innovation st

Introduction & Problem Statement

Regular Path Query (**RPQ**) in **PGX**.

- an in-memory parallel graph analytics framework, developed by Oracle Lab.

- Space requirement
- Performance requirement
- Commitment to deliver result

Introduction & Problem Statement

RPQ: (X, knows∘like$^+$∘(like*∘dislike)$^+$, Y)

Three types of clauses:

- Non-Kleene star clause, i.e. *knows*
- Non-nested Kleene star clause, i.e. *like*$^+$
- Nested Kleene star clause, i.e. *(like***∘ dislike)*$^+$

Algorithm & possible optimizations:

- Naive: search in the graph by standard algorithms, such as BFS or DFS
- Cache: speed-up with materialization (space/speed trade-off)
- Context-specific: specialized in-memory search

Existing Approaches

Index-based

- *k*-path index (*Fletcher et al. 2016*)
- Reachability index (*Gubichev et al. 2013*)

Automata-based

- Automata-based approach (*Koschmieder et al. 2012*)

Datalog-based

- Datalog-based relational database (*Saumen C. Dey et al. 2013*)

Transitive Closure-based

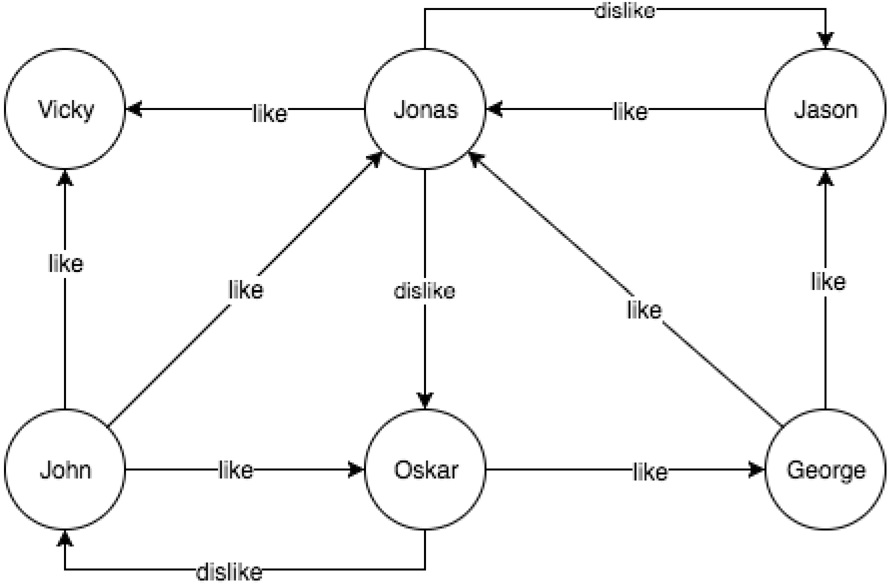
- Full *Transitive Closure* (*Rakesh Agrawal 1988*)

General Drawbacks

- Large potential intermediate results
- Impractical precomputation cost

RPQ Operator Design

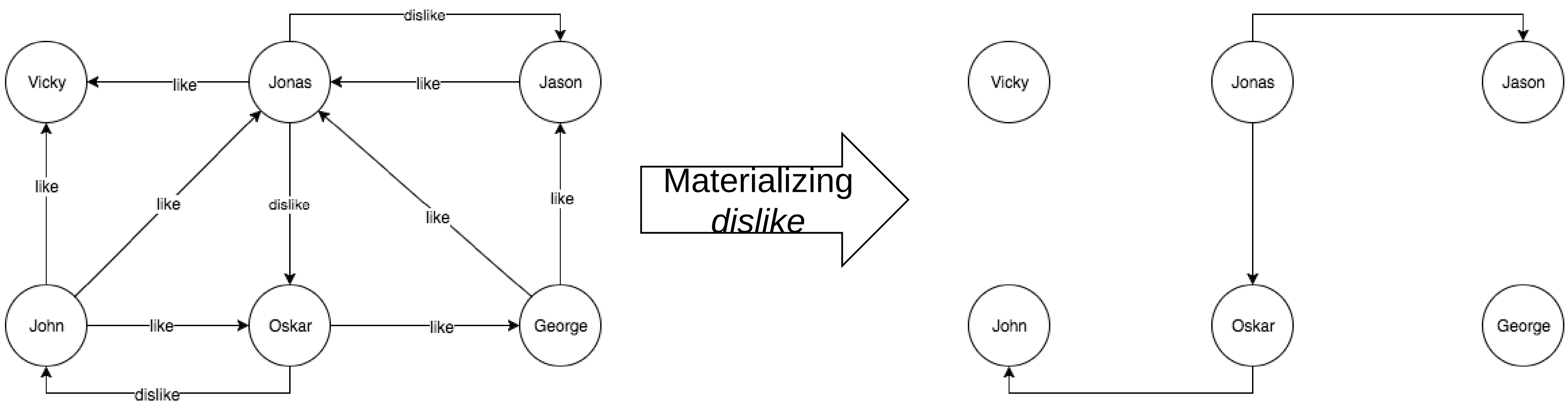
How to **adapt** transitive closure algorithms to solve *non-nested Kleene star clause* on labeled digraphs?



RPQ Operator Design

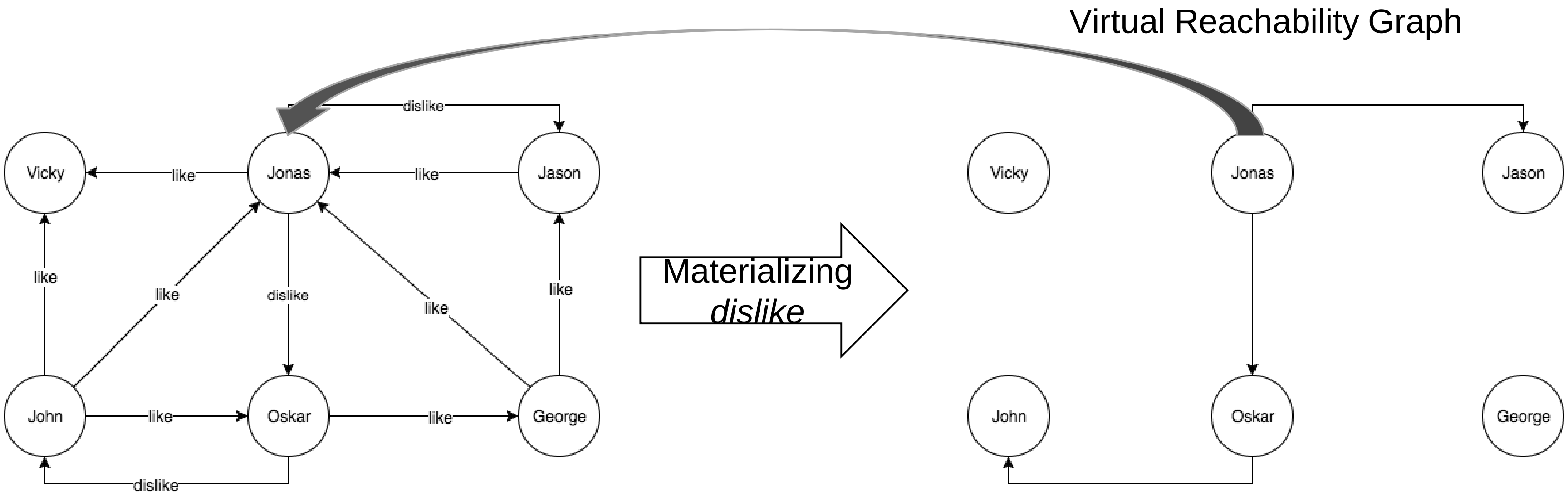
RPQ: (X, $dislike^{+}$, *Y*)

Reachability Graph (*R.G.*)



RPQ Operator Design

Question: what if there is not enough memory for R.G.?



Size Estimation Overview

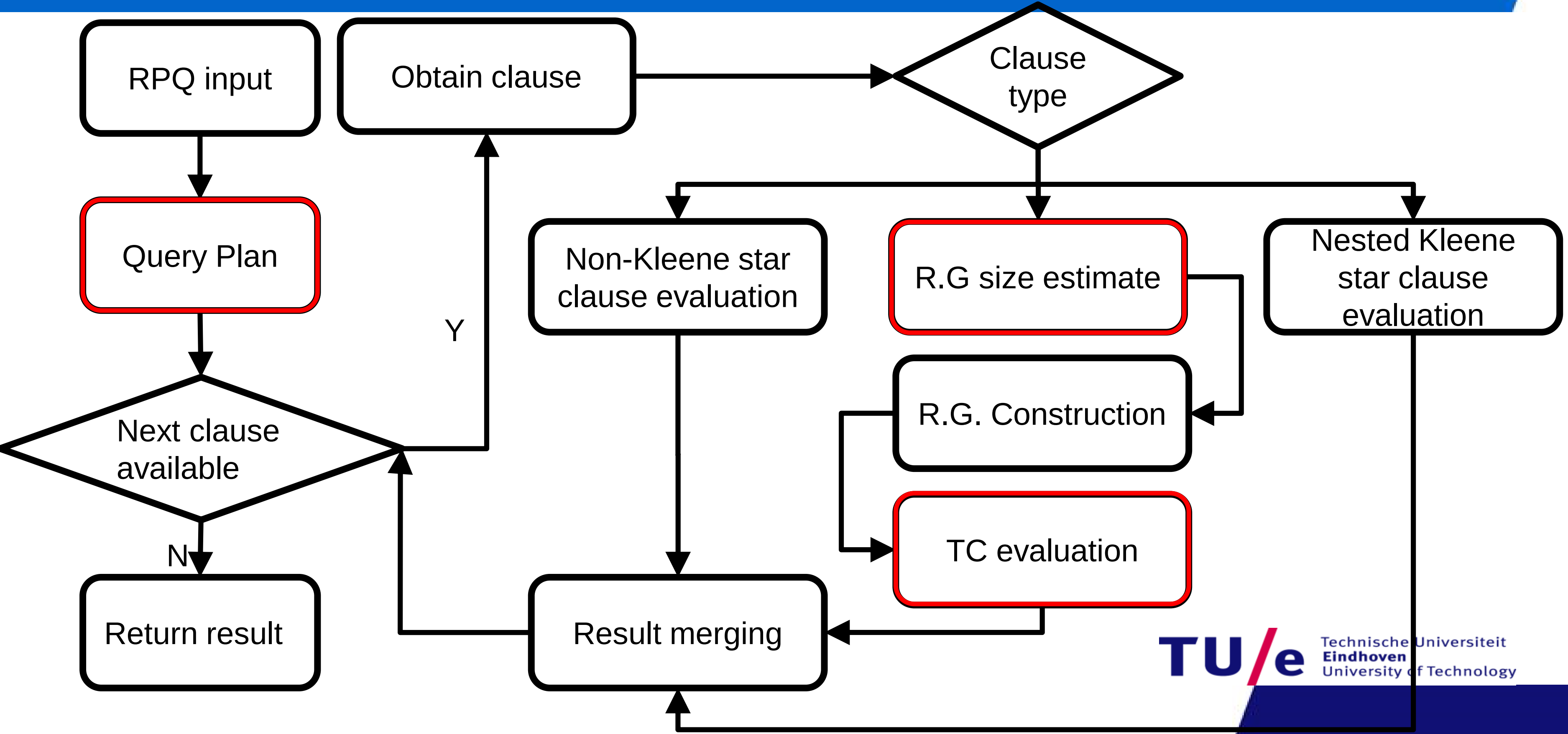
Non-Kleene

- Capturing correlations between labels in paths is critical to a precise estimate
- We adopt the method in (*Ashraf Aboulnaga et al. 2001*) that captures certain degree of *co-relationship* between edge labels in paths

Kleene star

- Need estimates for transitive closures, E.g. $like^{+}$
- Traditional methods produce poor estimates due to lack of *deduction*
- We use min-hash sketch (*Edith Cohen, 1997*) for estimation

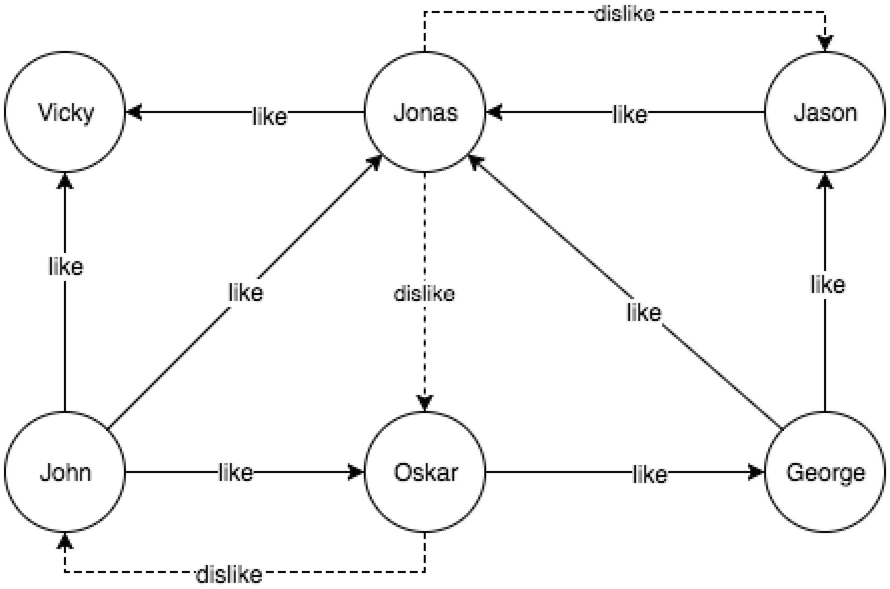
RPQ Life Cycle



RPQ Operator Implementation

Depending on whether the R.G. has small-world property

- Bitmap-based BFS (*M. Yang and C. Zaniolo, 2014*)
- Multi-source BFS (*M. Then et al., 2014*)



Experiments & Result analysis

Objectives

- Effectiveness of materializing reachability graph.
- Performance impact of reachability graph construction.
- Performance impact of reachability graph type and algorithm choice

	LDBC	LUBM
Num. of Vertex	7,352,398	21,715,109
Num. of Edge	7,689,947	66,199,001
Num. of Person	11,000	-
Num. of Uni.	-	1,000

TABLE 6.4: Characteristics of Test Data Set

NOTICE:
All queries are designed with Kleene star clause
Below, only results from LDBC dataset are presented.

Technische Universiteit **Eindhoven** University of Technology

Experiments & Result analysis

Query ID	BFSb (ms)	BFSbRG (ms)	RGCons (ms)	$Speedup_1$	$Speedup_2$
1	238,380	2,428	2,773	89x	45x
2	275,000	497	830	553x	207x
3	843,000	1,727	382	488x	399x
4	717,700	1,054	331	680x	518x
5	232,766	198	281	1175x	492x
6	> 11 hours	408,166	2,619	-	-
7	232,449	206	265	1128x	493x
8	241,440	227	456	1063x	353x

TABLE 6.5: LDBC-SNB: Ratios between the performance of bitmap-based BFS with and without reachability graph (i.e. *BFSbRG*, *BFSb* respectively). Reachability graph construction time (*RGCons*) is not included in $Speedup_1$ but included in $Speedup_2$.

Experiments & Result analysis

Query ID	BFSbRG (ms)	MSBFSRG (ms)	RGCons (ms)	$Share_1$	$Share_2$
1	2,428	4,138,652	2,773	53.31%	$\approx 0\%$
2	497	241	830	62.54%	77.49%
3	1,727	289	382	18.11%	56.92%
4	1,054	277	331	23.90%	54.44%
5	198	203	281	58.66%	58.07%
6	408,166	21,336	2,619	0.63%	10.93%
7	206	208	265	56.26%	56.02%
8	227	274	456	66.76%	62.46%

TABLE 6.6: LDBC-SNB: Share of reachability graph construction in total query processing. $Share_1$ indicates the graph construction share in bitmap-based BFS solution. $Share_2$ indicates the graph construction share in MS-BFS solution.

Conclusion & Future work

Achievement

- Boosting *RPQ* evaluation using partial materialization
- Switching physical *TC* operator depending on graph type
- Trading performance for space if necessary

Possible Improvement

- A better query estimation method
- An efficient in-memory *RPQ* evaluation solution without *R.G.*
- Facilitating graph traversal with effective cache usage

Thank You