ISCAS 中国科学院软件研究所学术年会'2022
暨计算机科学国家重点实验室开放周

学术论文

Incremental Graph Computations

ACM Trans. Database Syst. 47(2): 6:1-6:44 (2022)

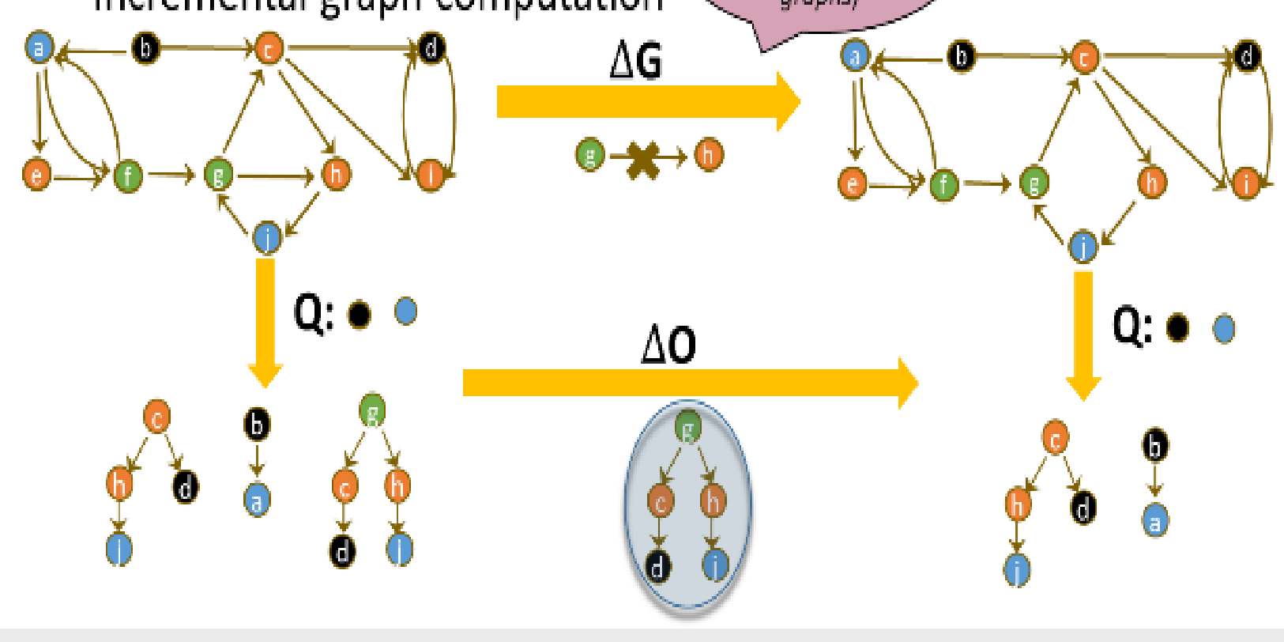
Chao Tian tianchao@iscas.ac.cn

1. Incremental graph computation

◆Batch algorithm vs. Incremental Algorithm

- Graph computations
  - real-life graphs are big -- billions of nodes in Facebook
  - graph queries are expensive -- NP-complete for subgraph isomorphism
- Incremental graph computation

Typically small (5%/week in Web graphs)



Compute new results from old answers

◆Incremental query answering

- Real-life graphs constantly change (ΔG)
- Graph computations are typically iterative

Re-compute Q(G ⊕ ΔG) starting from scratch?

✓ Changes ΔG are typically small

Compute Q(G) once, and then incrementally maintain it

- Incremental query processing:
  - Input: Q, G, Q(G), ΔG — *Changes to the input*; Q(G): *Old output*
  - Output: ΔO such that Q(G ⊕ ΔG) = Q(G) ⊕ ΔO — Q(G ⊕ ΔG): *New output*; ΔO: *Changes to the output*

When changes ΔG to the data G are small, typically so are the changes ΔO to the output Q(G)

Minimizing unnecessary recomputation

◆Complexity of incremental problems

The cost of batch query processing: a function of |G| and |Q|

- incremental algorithms: |CHANGED|, the size of changes in
  - the input: ΔG, and
  - the output: ΔO
- Bounded: the cost is expressible as f(|CHANGED|, |Q|)?

The updating cost that is inherent to the incremental problem itself

G. Ramalingam, Thomas W. Reps: *On the Computational Complexity of Dynamic Graph Problems*. TCS 158(1&2), 1996

Making the cost independent of |G|!

However,

- Less incremental algorithms are in place than batch algorithms
- Far less incremental algorithms are known bounded or not
- It is hard and ad hoc to prove whether an incremental problem is bounded or not

- *Positive: shortest distance (single source, all pairs)*
- *Negative: reachability (single source), subgraph isomorphism*

✓ *Systematic proof methods?*
✓ *"incrementalizing" popular batch algorithms?*

Is bounded computation within reach in practice?

2. Undoable: unboundness results

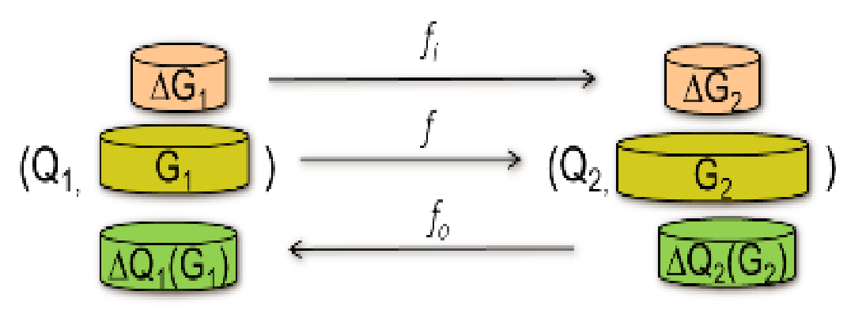
◆Δ-reductions from Q_1 to Q_2

Incremental problems: Q_1 and Q_2, instances (Q_i, G_i)

A triple of functions (f, f_i, f_o) such that for any I_1 = (Q_1, G_1) of Q_1

- f(I_1) is an instance of (Q_2, G_2) of Q_2
- for all updates ΔG_1 to G_1
  - $f_i(\Delta G_1)$ computes updates ΔG_2 to G_2 (updates to input)
  - $f_o(\Delta O_2)$ computes updates ΔO_1 (updates to output)

in PTIME in | ΔG_1 | + | ΔO_1 | and | Q_1 |.



A systematic method to figure out the boundedness

◆Negative results

✓ Theorem: if there exists a Δ-reduction from Q_1 to Q_2 and the incremental problem for Q_2 is bounded, them the incremental problem for Q_1 is bounded

The incremental problem is unbounded for

- regular path queries,
- strongly connected components, and — *Δ-reduction*
- keyword search

even under a unit edge insertion and a unit edge deletion

New unboundedness results

◆Limitations of boundedness

There are efficient incremental algorithms for

✓ regular path queries,
✓ strongly connected components, and
✓ keyword search

although none of the incremental problems is bounded

Boundedness is too strong a criteria

- It does not capture auxiliary structures necessary for algorithms
- It does not reflect the effectiveness of real-life incremental algorithms, which often substantially outperform batch algorithms even when the incremental problem is unbounded

More practical criteria for characterizing the effectiveness?

3. Doable: locality

An incremental algorithm T_Δ for Q is localizable if for each query Q in Q, its cost can be expressed by a function of

✓ |Q|, and
✓ the size of d_Q-neighbors of nodes in ΔG

- d_Q: decided by the size of Q (eg, diameter)
- d_Q-neighbor of node v: within d_Q hops of v

Doable: the incremental problem is localizable for

✓ subgraph isomorphism,
✓ keyword search

although these problems are unbounded!

Effective incremental algorithms are within reach for common queries

4. Doable: relatively bounded incrementalization

Consider a popular batch algorithm *T* for Q

An incremental algorithm T_Δ for Q is bounded relative to *T* if for each query Q in Q, its cost is a polynomial of

✓ |Q|,
✓ |ΔG|, and
✓ |AFF|, the difference between the data inspected by *T* for computing Q(G) and Q(G ⊕ ΔG)

Doable: the incremental problem is relatively bounded for

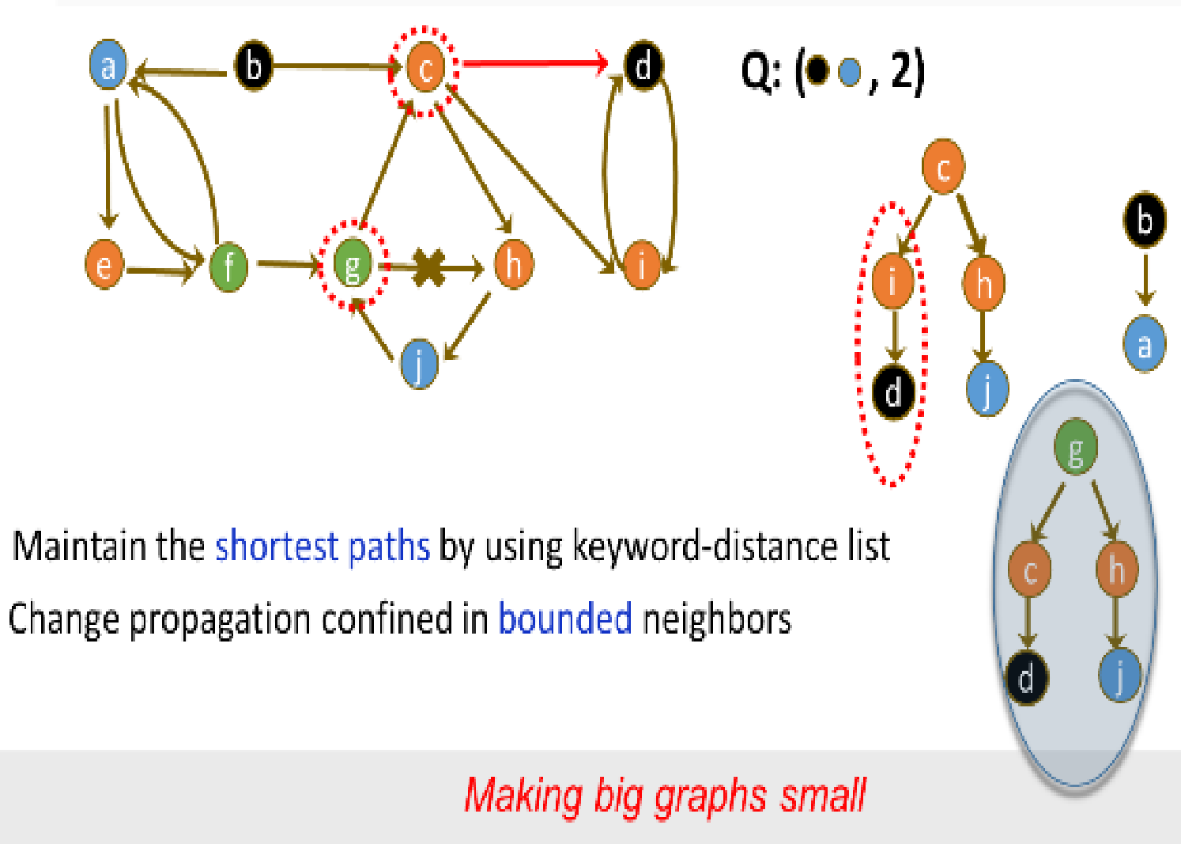
✓ regular path queries, and
✓ strongly connected components

Incrementalizing popular batch algorithm

5. Localizable and relatively bounded algorithms

◆Keyword search

- A keyword query Q is a list of keywords with an integer bound
- Find subtrees of the graph that match keywords on the leaves and have the minimum sum of the distances from the leaves for each distinct root



Maintain the shortest paths by using keyword-distance list

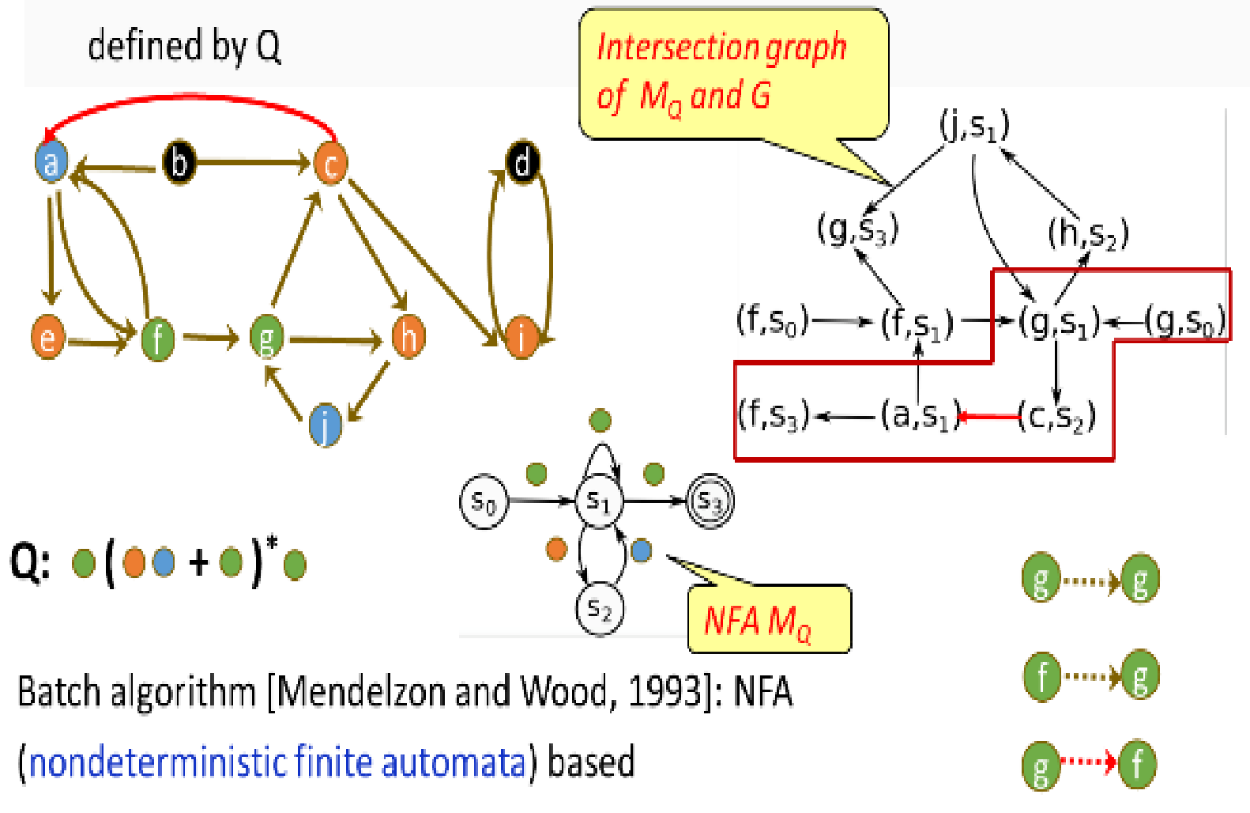
Change propagation confined in bounded neighbors

Making big graphs small

◆Regular path queries

- A regular path query Q is a regular expression defined on an alphabet of labels
- Find node pairs that are connected by paths with labels in the regular language defined by Q

Intersection graph of M_Q and G



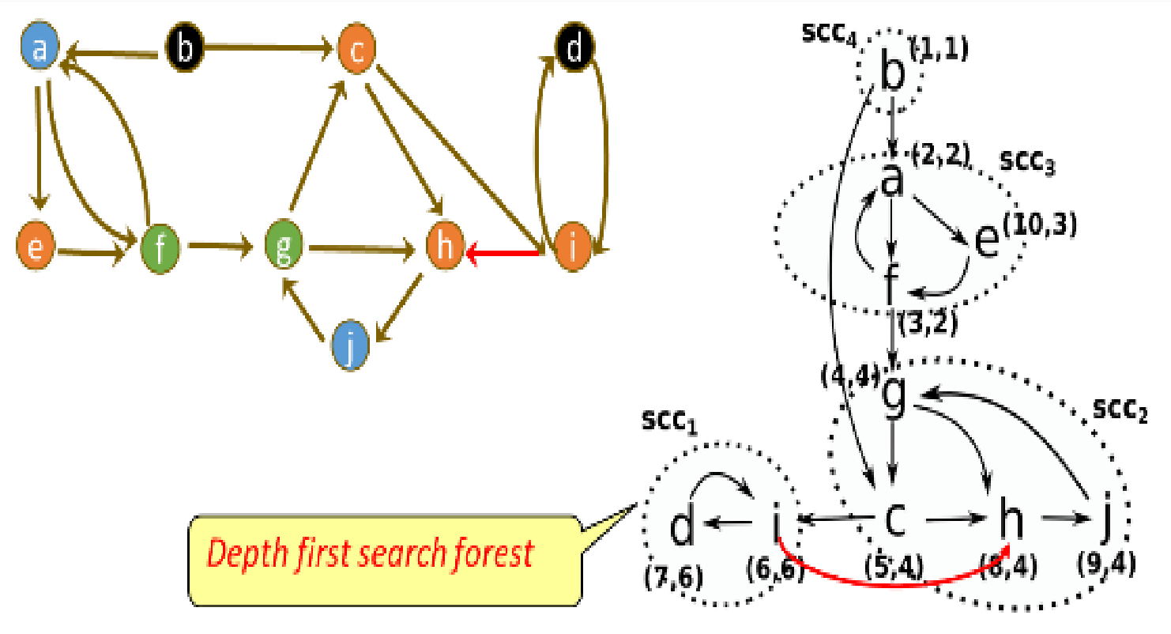
NFA M_Q

Batch algorithm [Mendelzon and Wood, 1993]: NFA (nondeterministic finite automata) based

Affected area: changes to the intersection graph, including the associated shortest distance information

◆Strongly connected components

- Find components that contain a directed path between every pair of nodes in it
- Incrementalize Tarjan's algorithm [Tarjan, 1977]



Depth first search forest

Affected area: changes to the information maintained in the DFS forest

6. Experimental Study

✓ *Real-life datasets*

- *DBpedia: 4.3 million nodes, 40.3 million edges, 495 labels*
- *LiveJournal: 4.9 million nodes, 68.5 million edges, 100 labels*

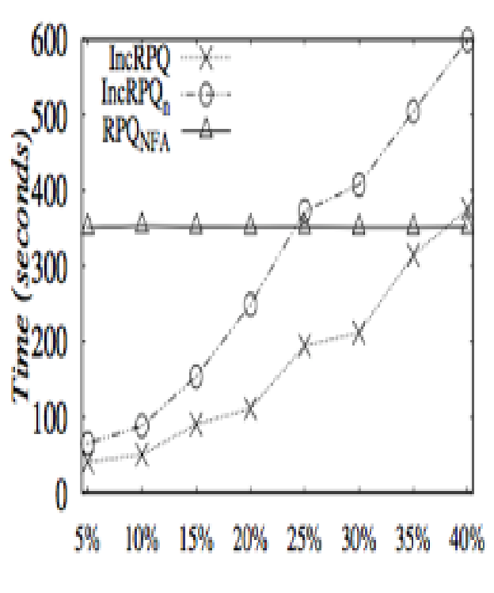
✓ *Synthetic graph*

- *up to 100 million nodes, 500 million edges*
- *labels drawn from an alphabet of 6000 symbols*

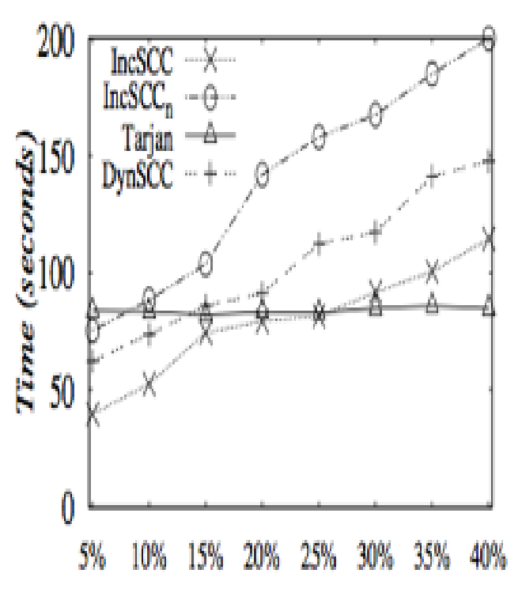
✓ *Updates*

- *randomly generated*
- *controlled by size |ΔG| and ratio of edge insertions to deletions*

◆ Scalability with |ΔG|

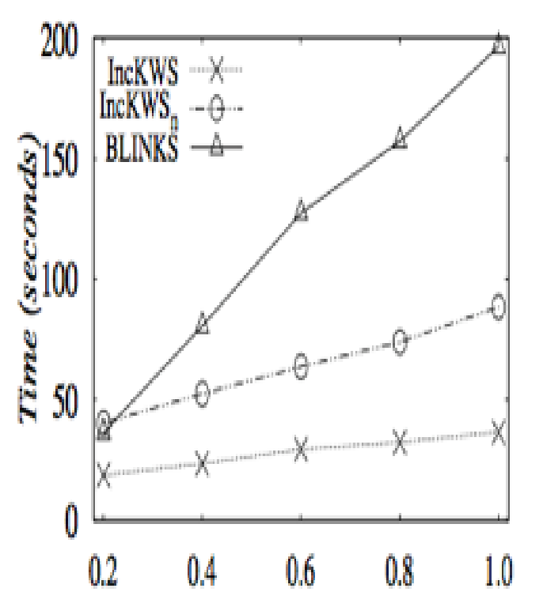


Varying ΔG, RPQ (DBpedia)



Varying ΔG, SCC (liveJ)

◆ Scalability with |G|



Varying G, KWS (Synthetic)

7. Summary

We established undoable and doable results for incremental graph computations:

✓ the incremental problems for regular path queries, strongly connected components and keyword search are unbounded under unit updates
✓ two new characterizations for the effectiveness of incremental graph computations
✓ localizable and relatively bounded incremental algorithms