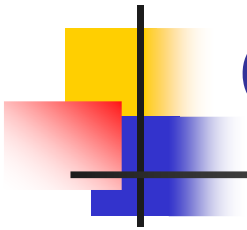
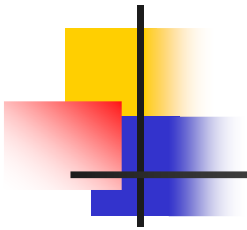
BR-Index: An Indexing Structure for Subgraph Matching in Large Dynamic Graphs

Jiong Yang and Wei Jin
EECS Department
Case Western Reserve University



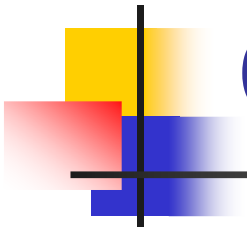
Outline

1. Introduction
2. Problem definitions
3. Properties
4. BR-index data structure
5. BR-index construction
6. Query algorithm
7. Experimental Results



Introduction

- The subgraph indexing problem
- Existing work only deals with static graph
- The large database graph evolve over time
- We propose a partition-based indexing structure.
- The main idea is to break a large graph into small local graphs, and index small features in each small graphs; thus we only need to update locally

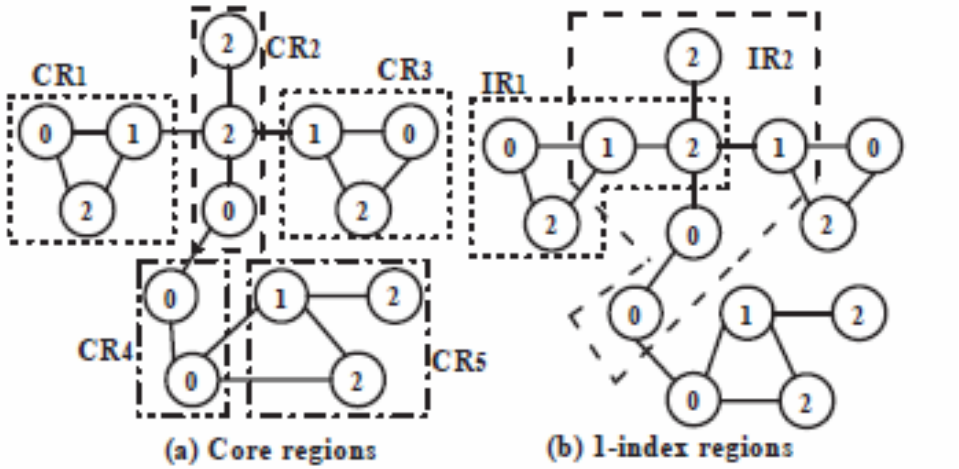


Challenges

- A feature may not be fully contained in one region; the bounding region property
- Organizing features by a feature lattice and using maximal feature to index regions
- Overlapping features and the overlapping region property

Definitions

- K-neighborhood of a vertex v
- A partition of a graph
- The K-index region



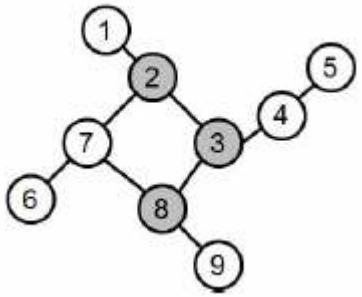
Definitions

- The eccentricity of a vertex *v*, *ecc*(*v*), in a connected graph *G* is the maximum shortest distance from *v* to any other vertex.
- The radius of *G*, *radius*(*G*), is the minimum eccentricity among all vertices in *G*.
- The center of *G*, *center*(*G*), is the set of vertices with the eccentricity equal to the radius.

ecc(1)=4, ecc(2)=ecc(3)=ecc(8)=3, ecc(6)=5

radius(G) = 3

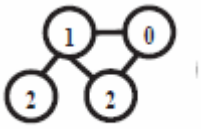
center(G) = {2, 3, 8}

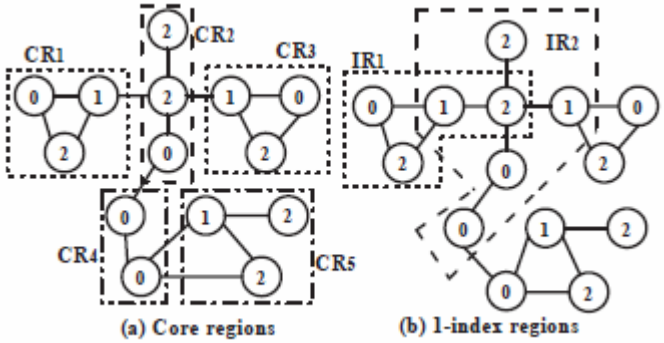


Properties

- The bounding region property:

For any subgraph g of G (radius(g)<=k), there exists at least one k index region IR in any partition of G, such that g is a subgraph of IR

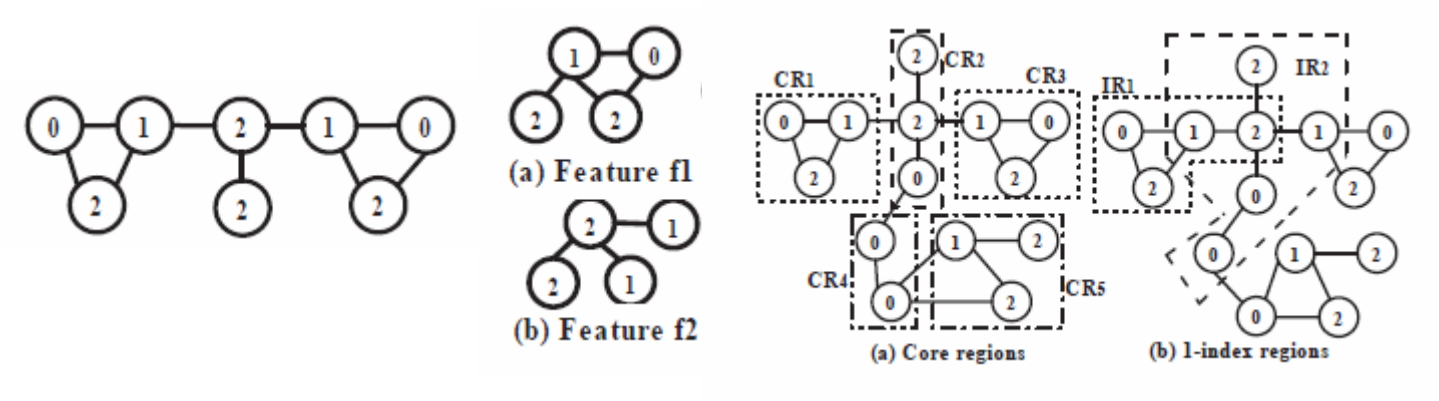


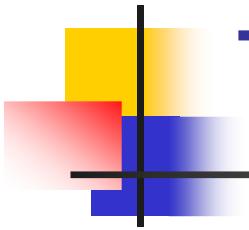


Properties

- The overlapping region property

Let o1 and o2 be the overlapping occurrences of features f1 and f2. For any occurrence o of Q in the database graph G, let IR1 and IR2 be the two index regions that contain o1 and o2 in o. IR1 and IR2 must overlap in G.

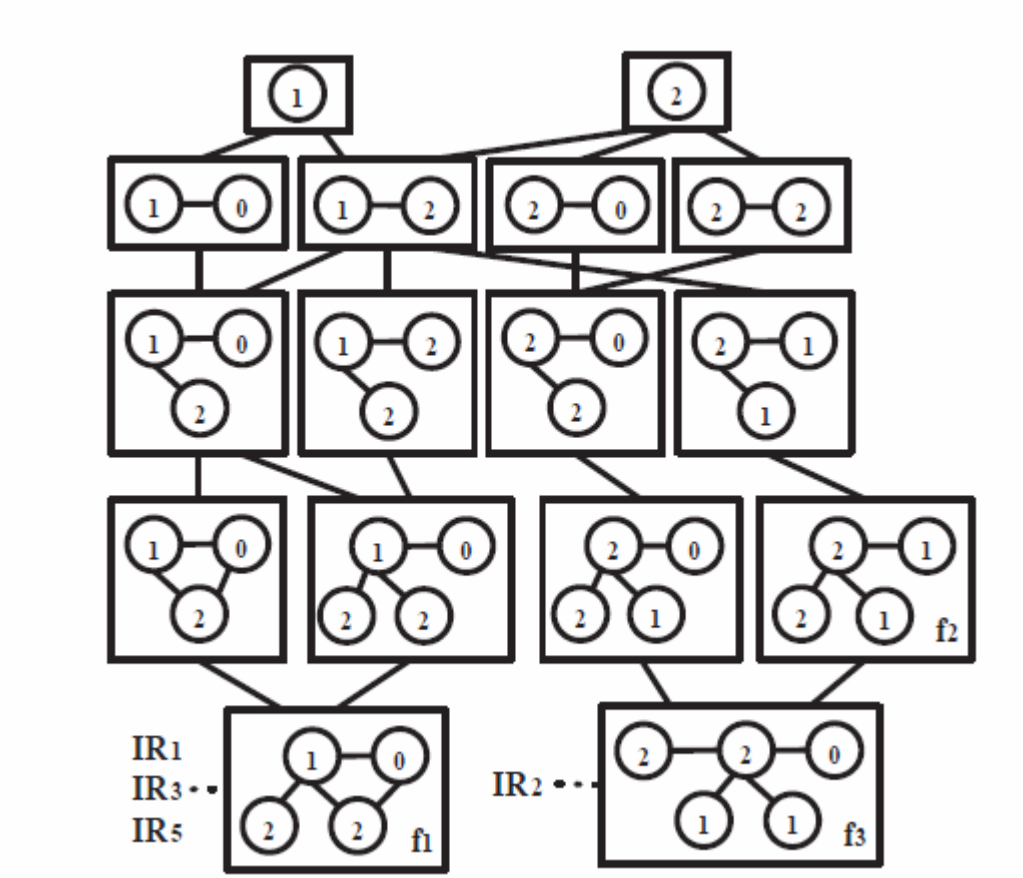




The BR-Index

- A hash table that maps a vertex v to the core region and index regions that contain v
- The feature lattice
- The index regions

The feature lattice



The index region (IR)

- Five fields of the index region

ID

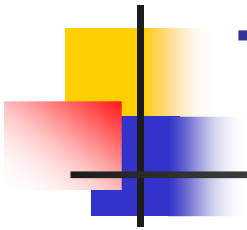
Core vertices

The index region subgraph

Overlapping index regions

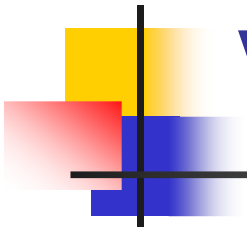
Feature IDs

- Space Analysis: (d^k) times the original database graph



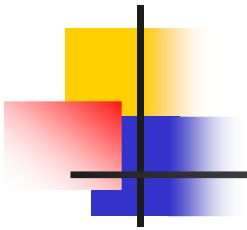
The BR-Index construction

- Vertex insertion
- Edge insertion
- Edge deletion
- Vertex deletion
- Core region merge
- Core region split



Vertex insertion

- Add v to the isolated vertex index region (IV)
- Create a new single label feature if v has a new label
- Update the feature lattice



Edge insertion

- Four cases:

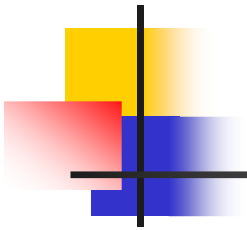
1) u and v are in IV: move them to a core region with fewest vertices;

2) u is in IV and v is in a CR: move u to the CR;

3) u and v are in the same CR: update the edge in all index regions containing u and v;

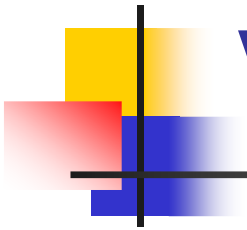
4) u and v are in different CR: v is inserted into IRu, vice versa, update the edge in all index regions containing u and v, update the overlapping list.

- may invoke core region split
- update features



Edge deletion

- Find the index regions containing both u and v
- Remove the edge; if one becomes isolated to any other vertices in the index region, move it to IV
- May invoke core region merge
- Update features

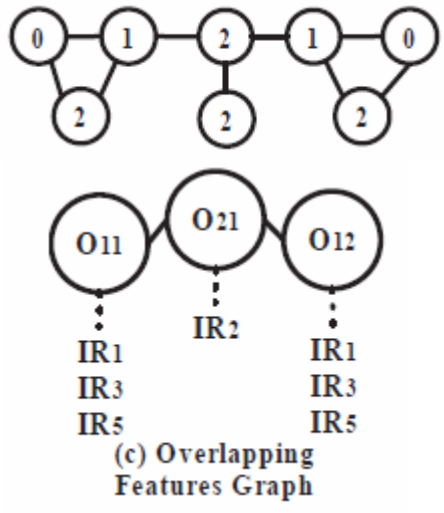


Vertex deletion

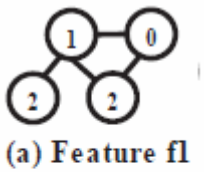
- Delete all the edges attached to v by the edge deletion procedure;
- Delete v;

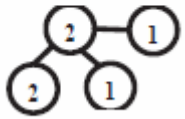
Query Algorithm

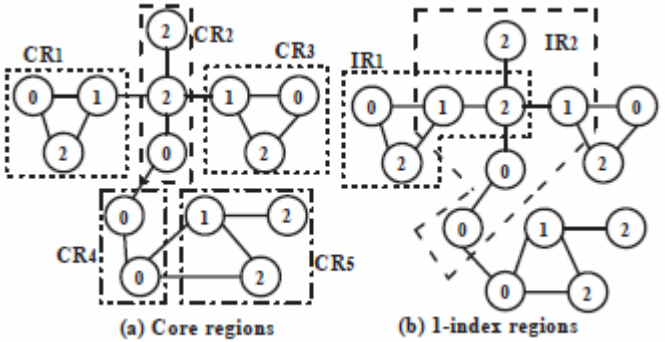
- Three main steps for querying a subgraph
- 1. Maximal features extraction;
- 2. Index region pruning;
- 3. Occurrences discovery by assembling feature matches;



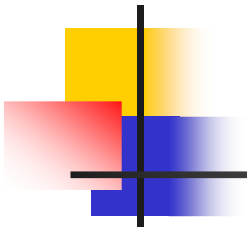
(c) Overlapping Features Graph


(a) Feature f1


(b) Feature f2



(a) Core regions (b) 1-index regions

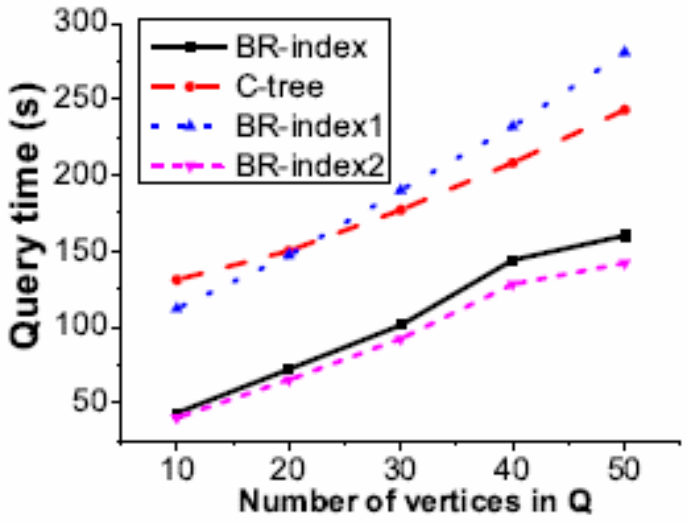


Experimental Results

- BR-index is the full version.
- BR-index1 only uses index regions: *G* is partitioned into a set of index regions and each region is indexed by a set of features.
- BR-index2 uses index regions and the overlapping region information for better pruning power, but without the maximal feature technique.
- Compare with CTree

Query time

- The flickr graph: 1,286,641 vertices, average degree 25.1, label randomly assigned between 1 to 1000

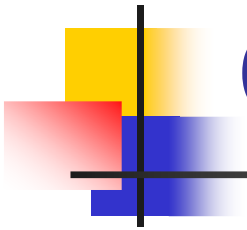


Experiments on a real data set

- Index update time: randomly add 500 edges to the graph and then delete 500 edges

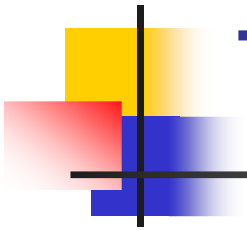
Table 6.1: Index Comparisons on flickr

	Construction Time(s)	Update Time(s)	Size(GB)
BR-index	7310	22	2.5
BR-index1	6782	12	2.2
BR-index2	18210	77	2.9
C-tree	8235	112	2.7



Conclusions

- We address the problem of indexing large dynamic graphs
- The large database graph is partitioned to a set of overlapping index regions.
- Features are extracted and organized as a lattice
- Two properties are identified for accelerating the matching process



Thank You

Q&A