pSCAN: Fast and Exact Structural Graph Clustering

Never Stand Still | Faculty of Engineering | Computer Science and Engineering

Lijun Chang[1], Wei Li[1], Xuemin Lin[1], Lu Qin[2] , Wenjie Zhang[1]

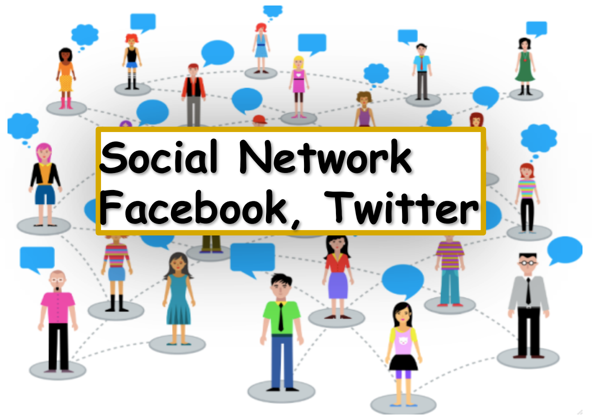
[1]The University of New South Wales, Australia
[2]University of Technology Sydney, Australia

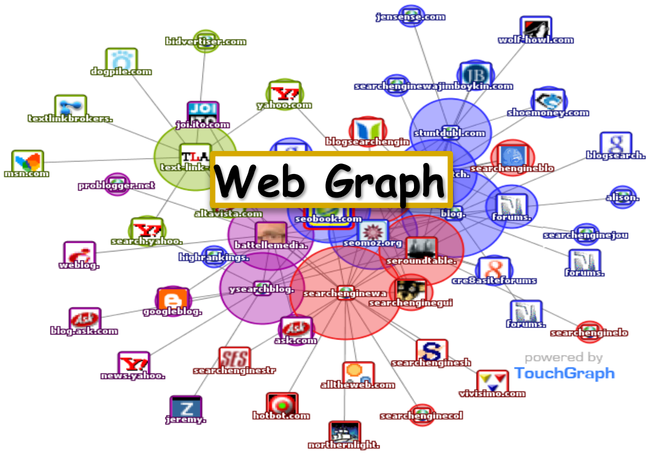
Outline

- **Structural Graph Clustering**
- A Two-Step Framework
- Our pSCAN Approach and Optimizations
- Experimental Studies
- Conclusion

Graphs

- Graphs are ubiquitous and can model complex relationships

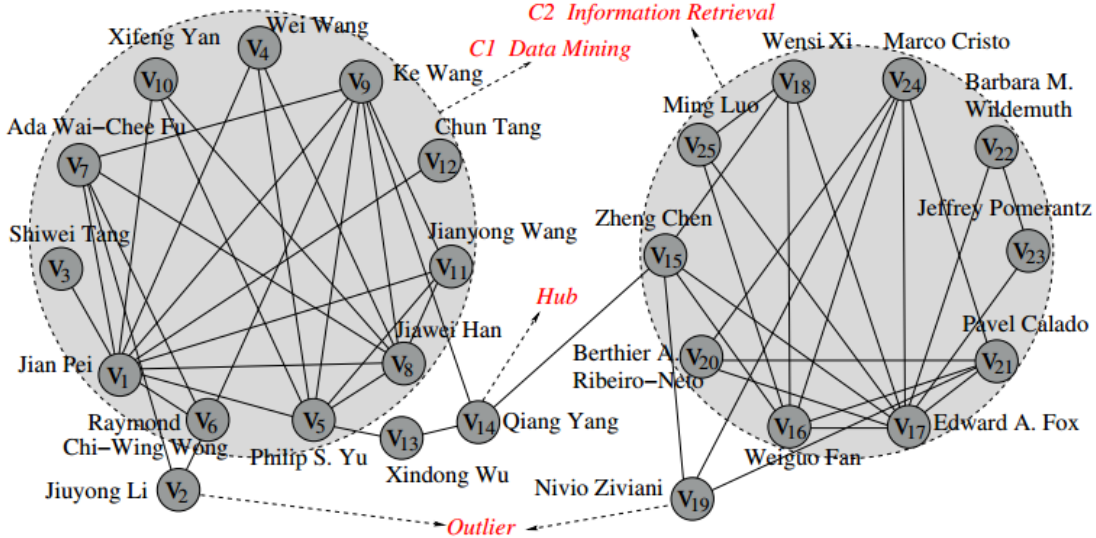




- Graph clustering
  - Group vertices into clusters: dense intra connection and sparse inter connection

Structural Graph Clustering

- SCAN [Xu+, KDD'07]
  - Identifies clusters, hubs, and outliers at the same time
  - Mimics DBSCAN [Ester+, KDD'96] for clustering spatial data



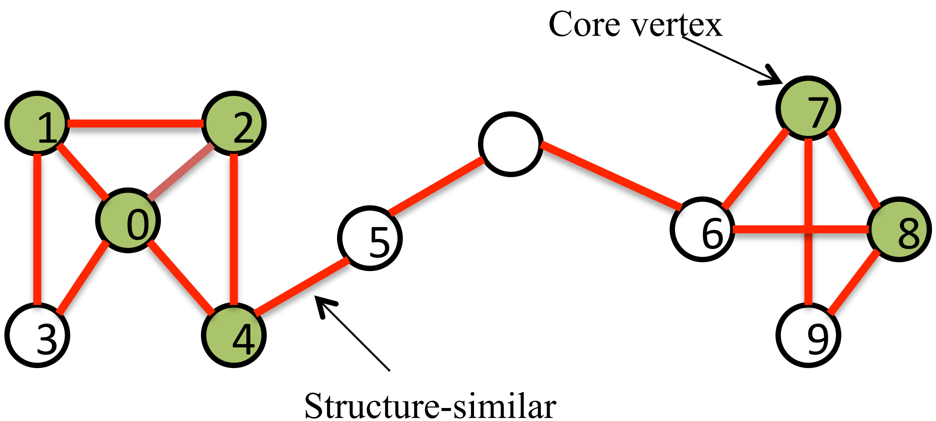
Example structural graph clustering

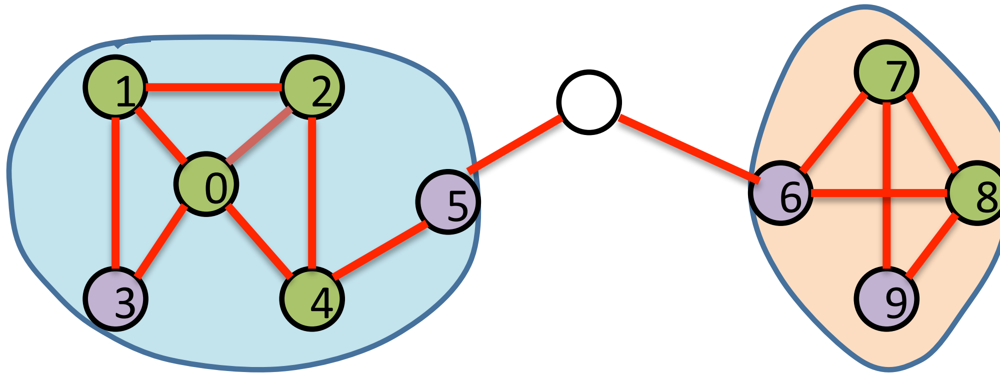
A Cluster = Cores + Borders

Core: vertices that are *structure-similar* to ***many*** other vertices
Border: vertices that are not core but are *structure-similar* to a core

- Structural Similarity: $$\sigma(u,v) = \frac{|N[u] \cap N[v]|}{\sqrt{d[u] \cdot d[v]}}.$$
- Two vertices *u* and *v* are *structure-similar* if
  - Connected
  - Structural similarity $\geq \varepsilon$ (a given similarity threshold)
- Many: $\geq \mu$ (a given size threshold)

Example (ε=0.0001, μ=3)





Existing Approaches & Challenges

- If the structural similarity between every pair of adjacent vertices has been computed, clusters can be obtained in linear time.
- Existing Approaches:
  - SCAN [Xu+, KDD'07]
  - SCAN++ [Shiokawa+, VLDB'15]

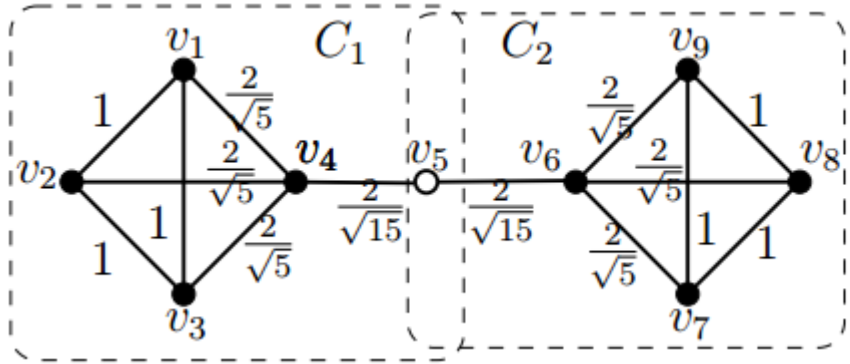
- Challenge-I: a systematic way to reduce the number of structural similarity computations
- Challenge-II: efficiently checking whether two vertices are structure-similar to each other
  - Existing approaches compute the exact structural similarity score

Outline

- Structural Graph Clustering
- **A Two-Step Framework**
- Our pSCAN Approach and Optimizations
- Experimental Studies
- Conclusion

Three Observations Utilized in Our Framework

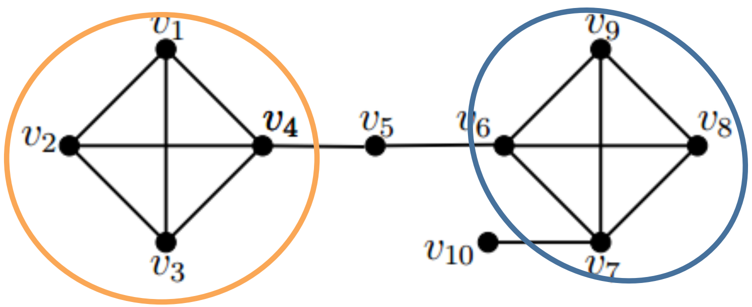
- Observation-I: The Clusters May Overlap



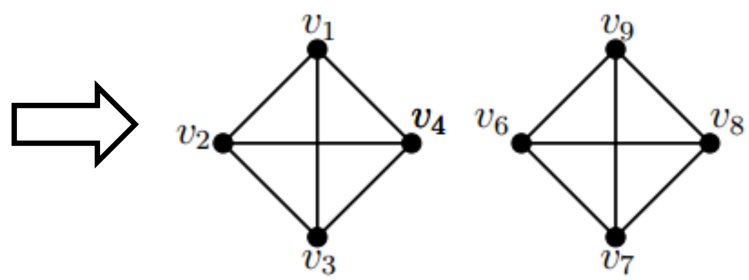
- Observation-II: The Clusters of Core Vertices Are Disjoint
  - Each core vertex belongs to a unique cluster
- Observation-III: The Clusters of Non-core Vertices Are Uniquely Determined By Core Vertices

Two-Step Framework

- **Step-I**: Cluster core vertices
  - Conceptually generate the connectivity graph for core vertices
  - Clusters of core vertices are CCs of the connectivity graph



Original graph



Connectivity graph

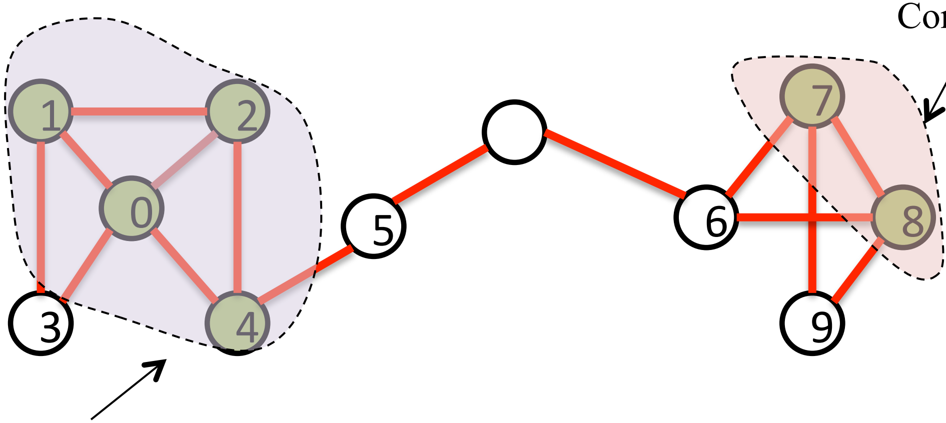
  - This presents optimization opportunity, since not all edges are needed for computing CCs

Two-Step Framework

- **Step-II**: Cluster non-core vertices
  - A non-core vertex belongs to the same cluster of a set of core vertices if it is structure-similar to one of the core vertices

Example (ε=0.0001, μ=3)

Core cluster 2



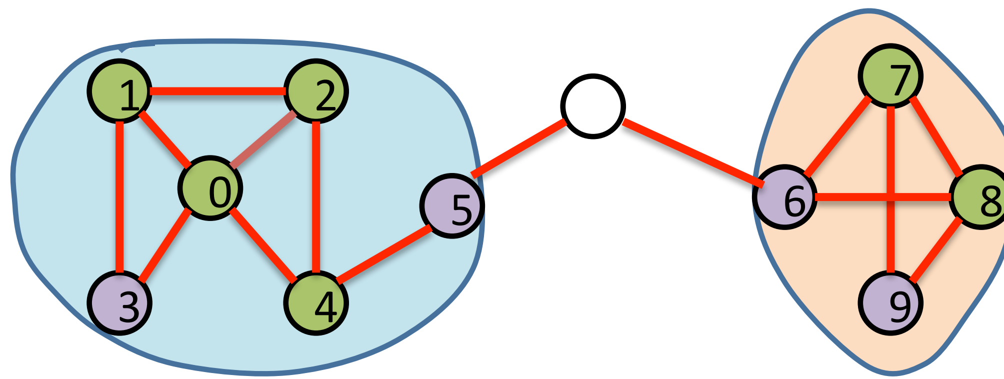
Core cluster 1

Step-I: cluster core vertices

Computes connectivity graph, and finds core clusters

Step-II: cluster non-core vertices

Assign non-core vertices to core clusters



Outline

- Structural Graph Clustering
- A Two-Step Framework
- **Our pSCAN Approach and Optimizations**
- Experimental Studies
- Conclusion

Our pSCAN Approach

- Determine core vertices
  - Maintain *sd(u)*, *ed(u)* for each vertex *u*
  - *sd(u)*: similarity degree of *u*, the number of neighbors that have been confirmed to be structure-similar to *u*
    - *u* is a core vertex if $sd(u) \geq \mu$
  - *ed(u)*: effective degree of *u*, *sd(u)* + the number of neighbors whose structural similarities to *u* have not been computed
    - *u* is non-core vertex if $ed(u) < \mu$
- We check core vertices in *non-increasing effective degree* order
  - After computing the structural similarity between *u* and *v*, we also update *sd(v)* or *ed(v)*

Our pSCAN Approach

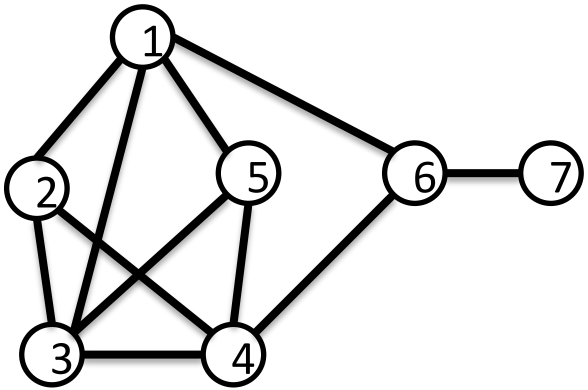
- Maintaining clusters of core vertices
  - Use the *disjoint-set data structure* to maintain the CCs of the connectivity graph
- For a core vertex *u*
  - First exam every neighbor *v* such that, *(i)* *v* is a core vertex, and *(ii)* *u* is structure-similar to *v*
    - *union(u,v)*
  - Then exam every neighbor *v* such that *(i)* *v* is a core vertex, and *(ii)* the structural similarity between *u* and *v* have not been computed
    - If *u* and *v* are in different CCs, check whether *u* is structure-similar to *v*, and *union(u,v)* if it is.
    - That is, if they are already in the same CC, we do not compute the structural similarity

Analysis of pSCAN

- Time complexity is $O(\alpha(G) \times m)$
  - $\alpha(G)$ is the arboricity of G.
- Space complexity is $O(m)$

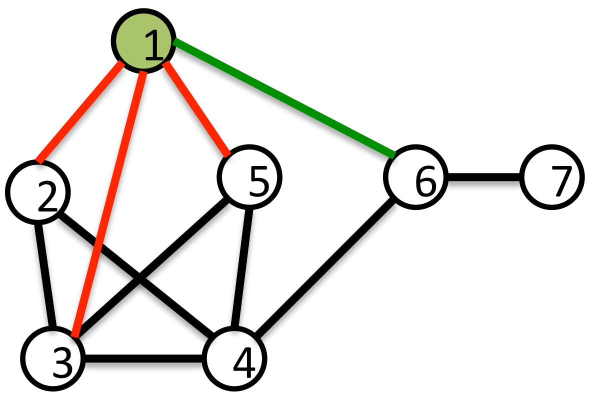
Theorem:
pSCAN is worst-case optimal

Running Example (ε=0.6, μ=3)



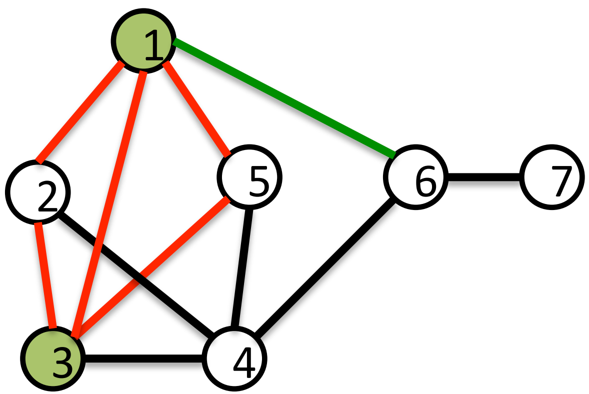
u	sd(u)	ed(u)
1	0	4
3	0	4
4	0	4
2	0	3
5	0	3
6	0	3
7	0	2

Running Example (ε=0.6, μ=3)



u	sd(u)	ed(u)
3	1	4
4	0	4
2	1	3
5	1	3
6	0	2
7	0	2

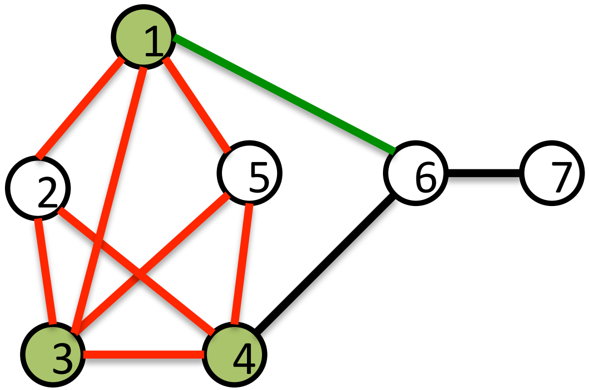
Running Example (ε=0.6, μ=3)



union(1,3)

u	sd(u)	ed(u)
4	0	4
2	2	3
5	2	3
6	0	2
7	0	2

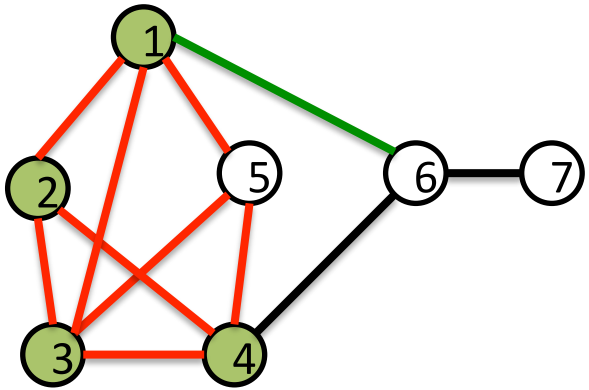
Running Example (ε=0.6, μ=3)



u	sd(u)	ed(u)
2	3	3
5	3	3
6	0	2
7	0	2

union(1,3)
union(3,4)

Running Example (ε=0.6, μ=3)



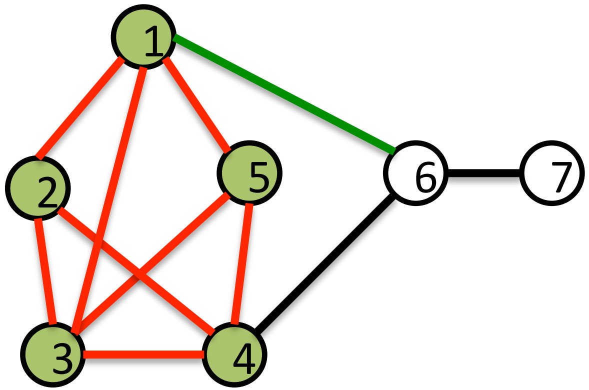
u	sd(u)	ed(u)
5	3	3
6	0	2
7	0	2

union(1,3)

union(3,4)

union(2,1)

Running Example (ε=0.6, μ=3)



u	sd(u)	ed(u)
6	0	2
7	0	2

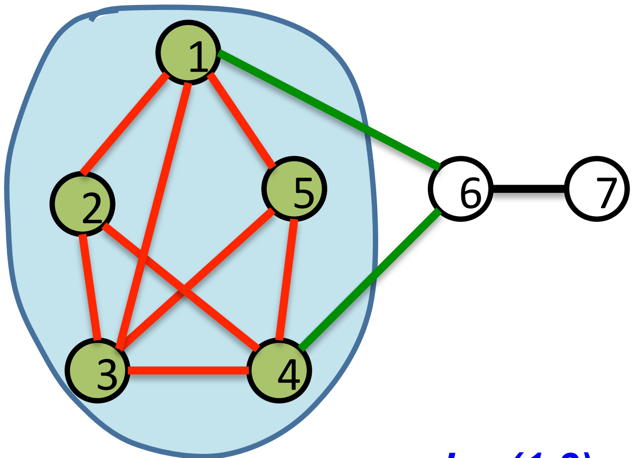
union(1,3)

union(3,4)

union(2,1)

union(5,1)

Running Example (ε=0.6, μ=3)



u	sd(u)	ed(u)
6	0	2
7	0	2

union(1,3)
union(3,4)
union(2,1)
union(5,1)

Optimizations

- Adaptive structure-similar checking
  - Compute the minimum number of common neighbors, *cn(u,v)*, required for the two vertices to be similar
  - Terminate early if *(i)* the number of computed common neighbors is ≥*cn(u,v)*, or *(ii)* the upper bound number of common neighbors is smaller than *cn(u,v)*
- Pruning rule
  - For two vertices to be structure-similar, their degrees must satisfy a certain condition
- Cross link
  - $\sigma(u,v)=\sigma(v,u)$

Outline

- Structural Graph Clustering
- A Two-Step Framework
- Our pSCAN Approach and Optimizations
- **Experimental Studies**
- Conclusion

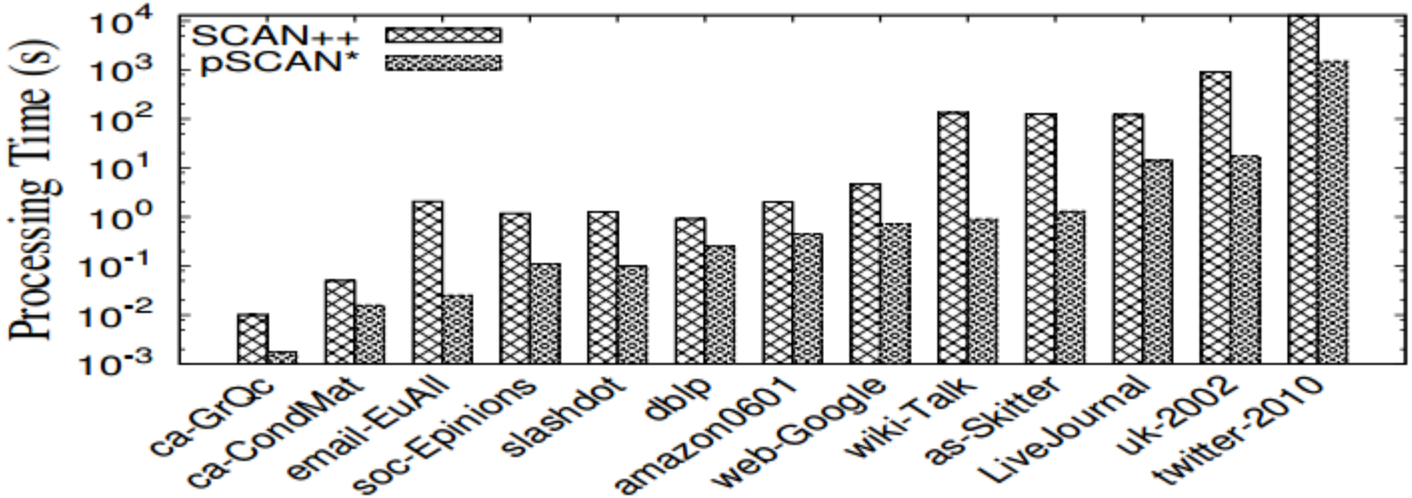
Experimental Results

- Datasets

Graph	#Edges	#Vertices	$\bar{d}$	c
ca-GrQc	13,422	4,158	6.46	0.557
ca-CondMat	91,286	21,363	8.55	0.642
email-EuAll	339,925	224,832	3.02	0.079
soc-Epinions	405,739	75,877	10.69	0.138
slashdot	468,554	77,350	11.12	0.055
dblp	1,049,866	317,080	6.62	0.632
amazon0601	2,443,311	403,364	12.11	0.418
web-Google	3,074,322	665,957	9.23	0.459
wiki-Talk	4,656,682	2,388,953	3.90	0.053
as-Skitter	11,094,209	1,694,616	13.09	0.258
LiveJournal	42,845,684	4,843,953	17.69	0.274
uk-2002	261,556,721	18,459,128	28.34	0.603
twitter-2010	1,202,513,344	41,652,230	57.7	0.073

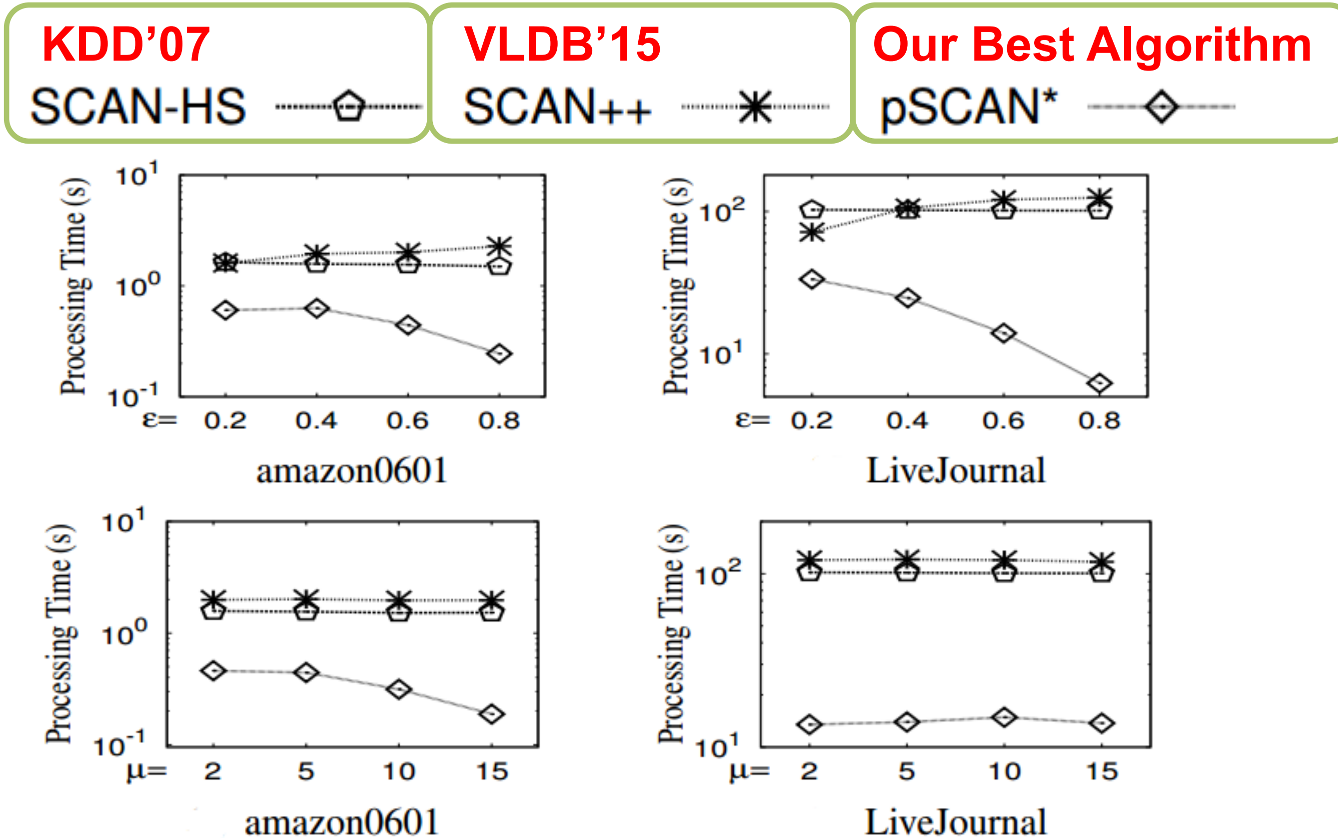
- Environments
  - Intel Xeon Processor 2.9GHz CPU and 32GB memory
  - All algorithms are implemented in C++

Scalability Testing

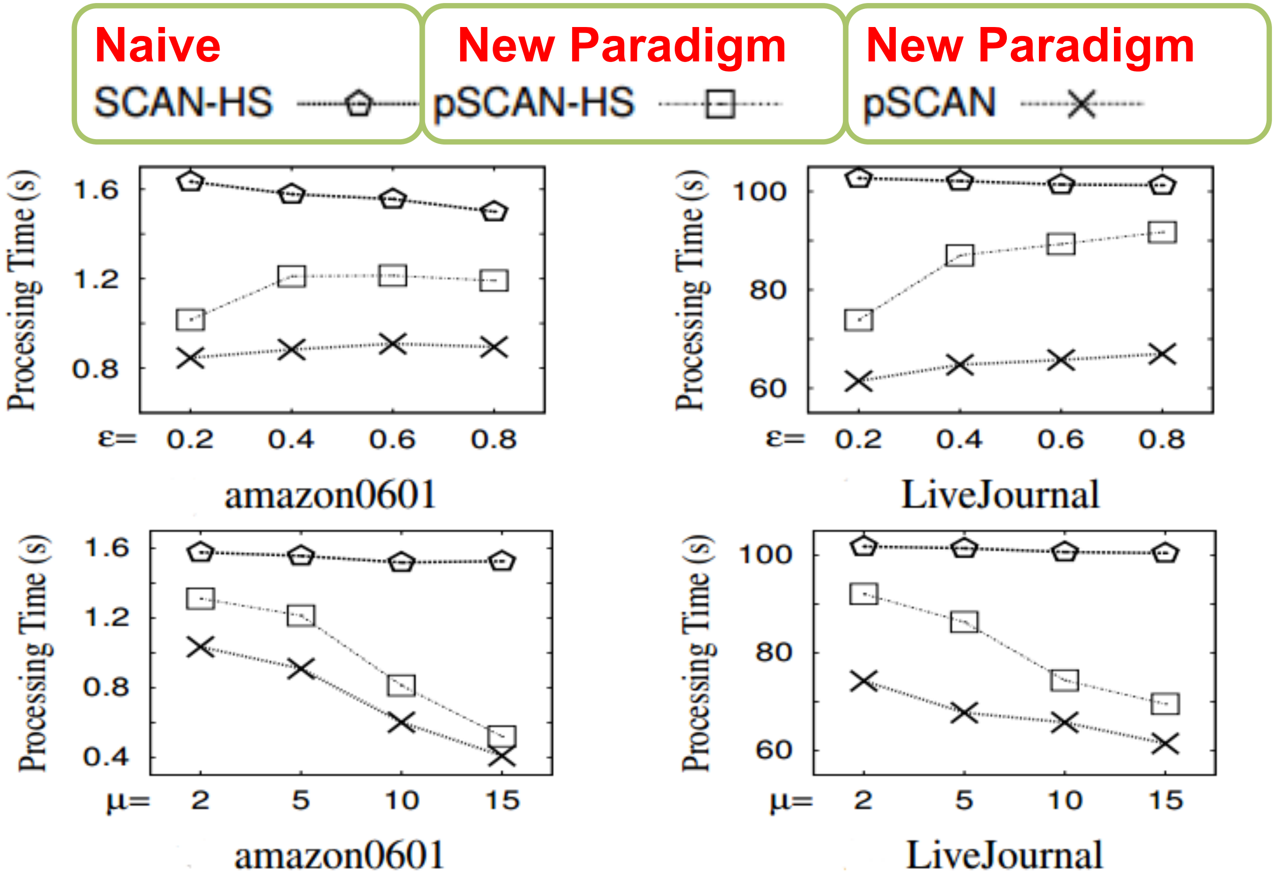


Scalability Testing on Real Graphs ($\epsilon = 0.6, \mu = 5$)

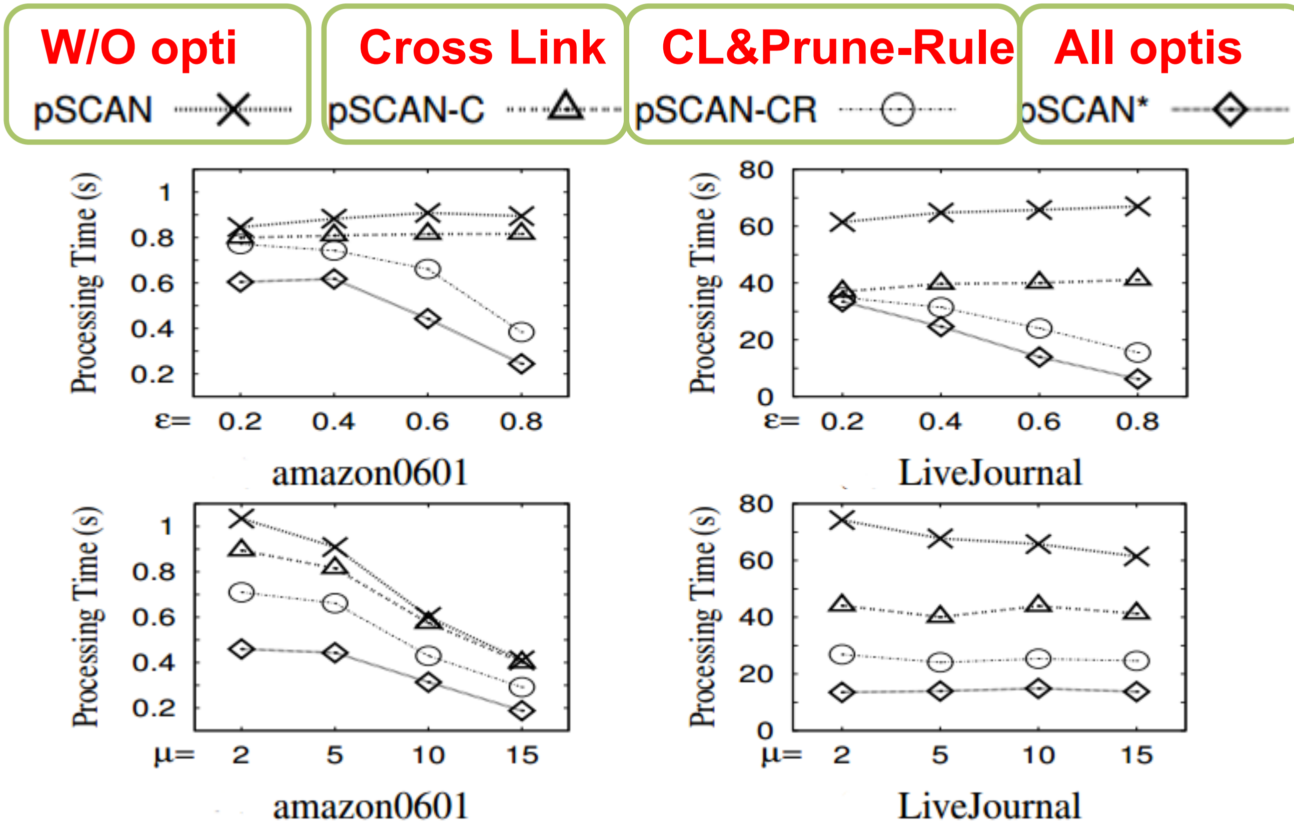
Comparing pSCAN* with SCAN-HS, SCAN++



Evaluating Our New Paradigm



Evaluating Optimization Techniques



Outline

- Structural Graph Clustering
- A Two-Step Framework
- Our pSCAN Approach and Optimizations
- Experimental Studies
- **Conclusion**

Conclusion

- A new paradigm for exact structural graph clustering
- A new approach aiming to reduce the number of structural similarity computations
- Prove that pSCAN is worst-case optimal
- three optimization techniques to speed up the checking of structure-similar between two vertices

Thanks!

Q&A

Never Stand Still