OLAK: An Efficient Algorithm to Prevent Unraveling in Social Networks

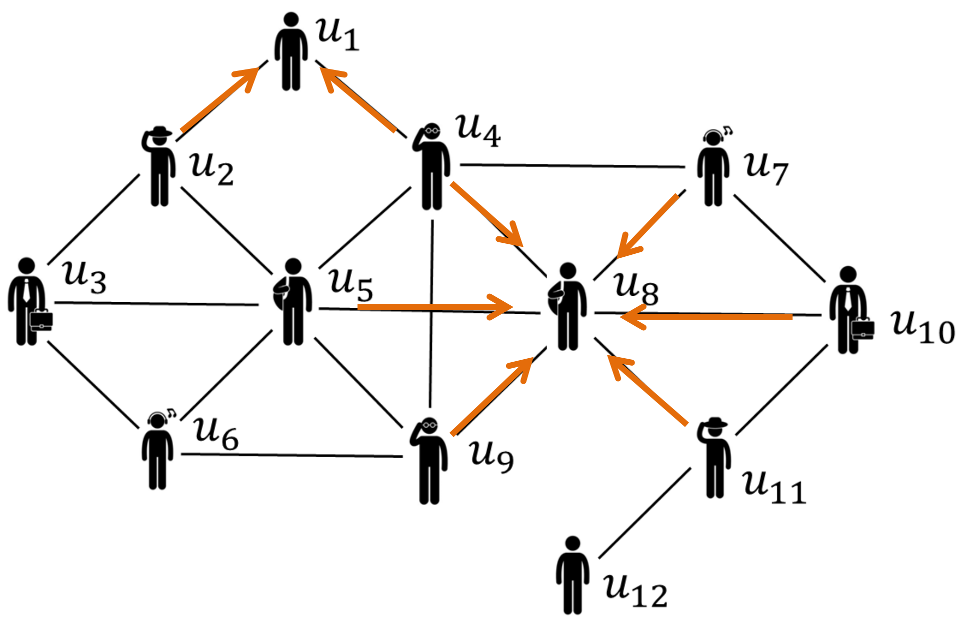
Fan Zhang[1], Wenjie Zhang[2], Ying Zhang[1], Lu Qin[1], Xuemin Lin[2]

1 *University of Technology Sydney*, 2 *University of New South Wales*

Network Unraveling

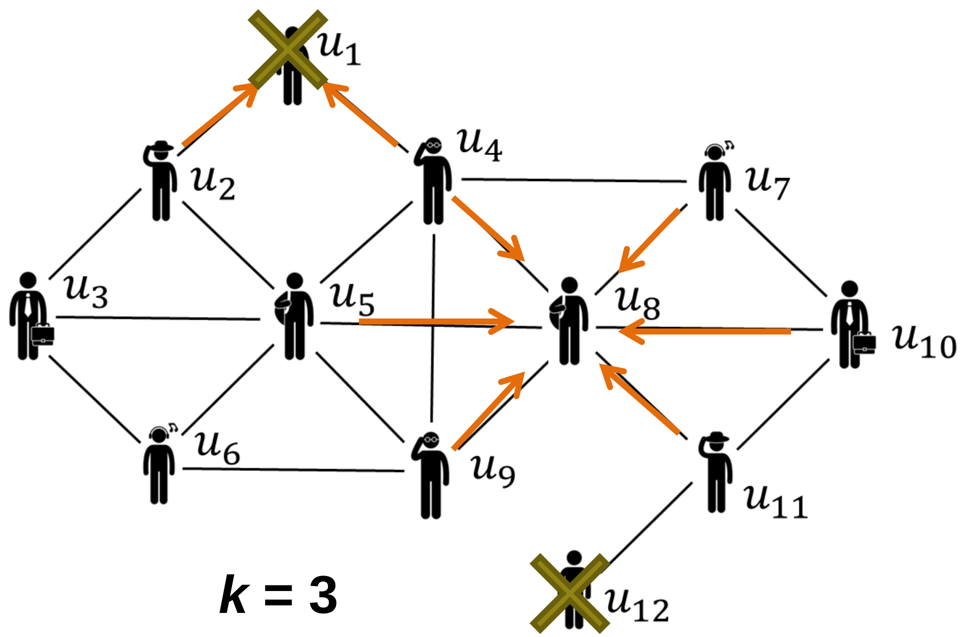
Kshipra Bhawalkar, Jon Kleinberg, Kevin Lewi, Tim Roughgarden, and Aneesh Sharma. "Preventing unraveling in social networks: the anchored k-core problem." SIAM Journal on Discrete Mathematics 29, no. 3 (2015): 1452-1475.

The engagement of a user is influenced by the number of her friends.



Network Unraveling

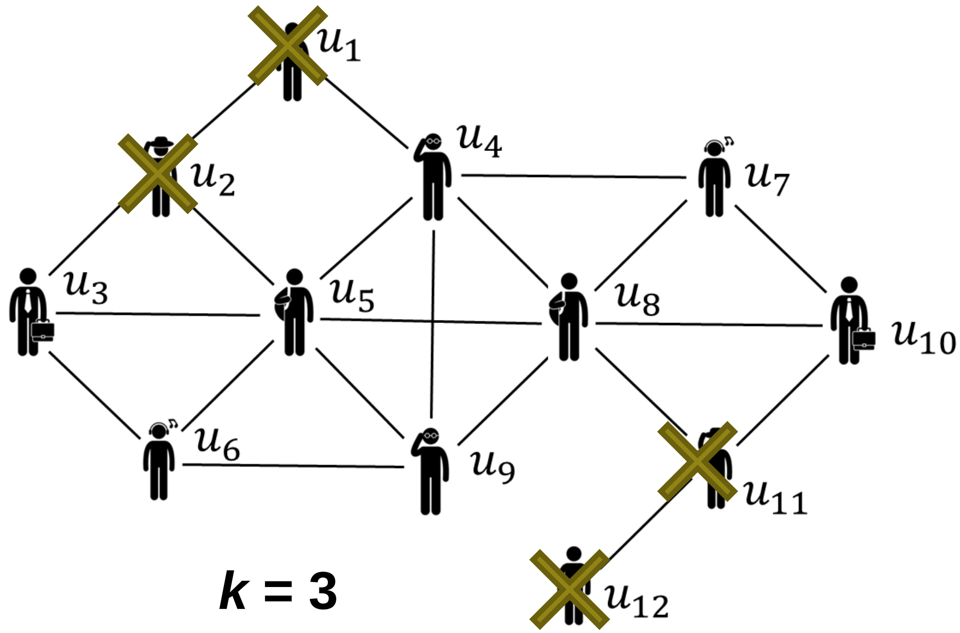
An equilibrium: a group has the minimum degree of *k*



Kshipra Bhawalkar, Jon Kleinberg, Kevin Lewi, Tim Roughgarden, and Aneesh Sharma. "Preventing unraveling in social networks: the anchored k-core problem." SIAM Journal on Discrete Mathematics 29, no. 3 (2015): 1452-1475.

Network Unraveling

An equilibrium: a group has the minimum degree of *k*.

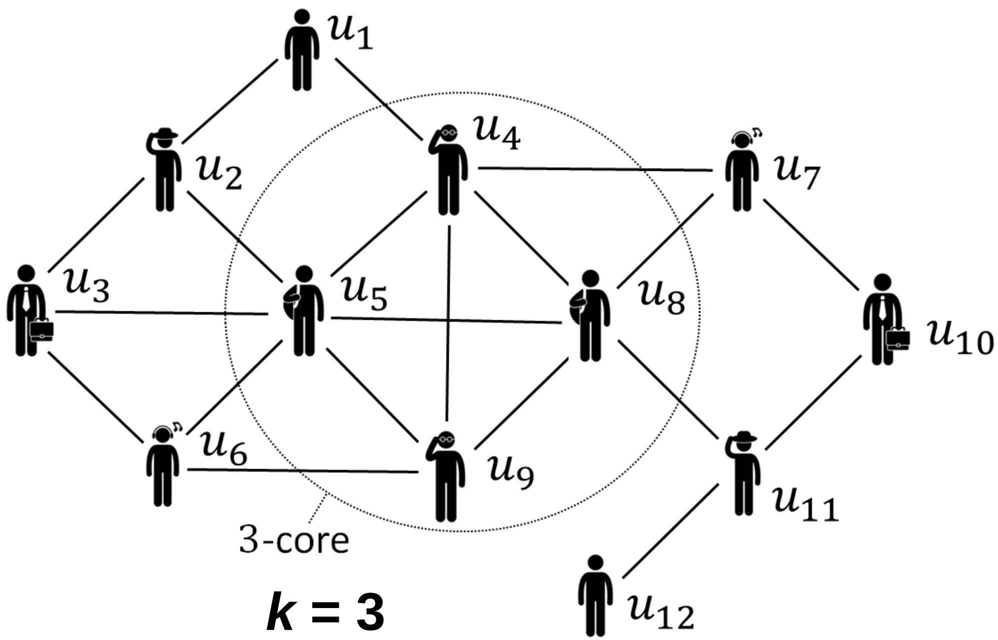


***k* = 3**

Kshipra Bhawalkar, Jon Kleinberg, Kevin Lewi, Tim Roughgarden, and Aneesh Sharma. "Preventing unraveling in social networks: the anchored k-core problem." SIAM Journal on Discrete Mathematics 29, no. 3 (2015): 1452-1475.

Network Unraveling

A social group tends to be a *k*-core in the network.

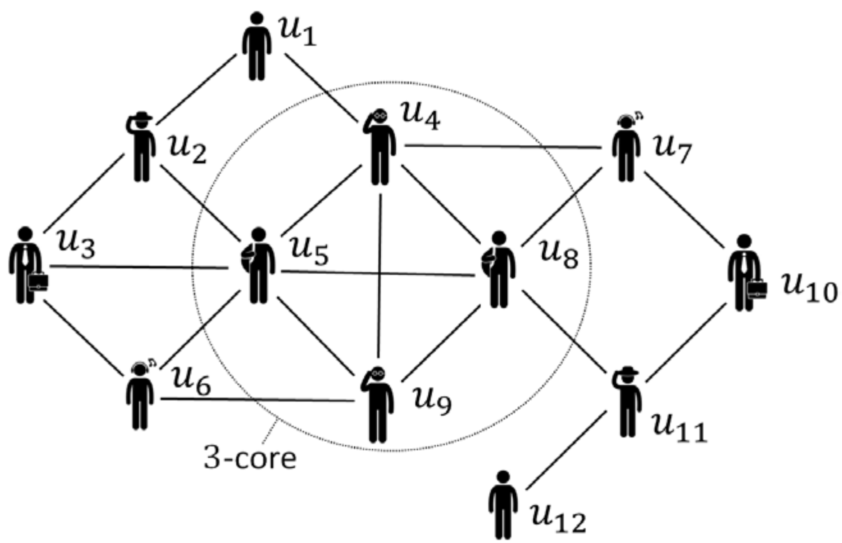


***k* = 3**

Kshipra Bhawalkar, Jon Kleinberg, Kevin Lewi, Tim Roughgarden, and Aneesh Sharma. "Preventing unraveling in social networks: the anchored k-core problem." SIAM Journal on Discrete Mathematics 29, no. 3 (2015): 1452-1475.

k-Core

- Given a graph *G*, the *k*-core of *G* is a maximal subgraph where each node has at least *k* neighbors (i.e., *k* adjacent nodes, or a degree of *k*).

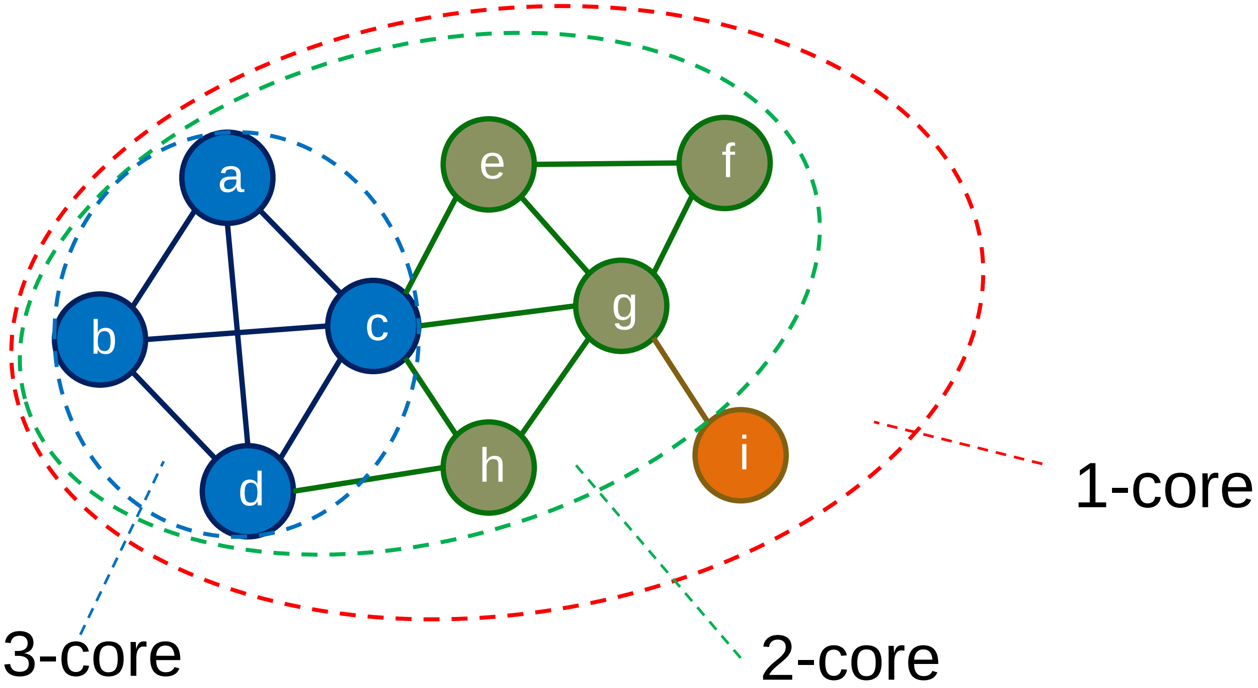


Applications: community detection, social contagion, user engagement, event detection, ……

S. B. Seidman. Network structure and minimum degree. Social networks, 5(3):269–287, 1983.

k-Core Decomposition

- **Core number** of a node v: the largest value of *k* such that there is a *k*-core containing *v.*
- Core decomposition: compute the **core number** of each node in *G.*



S. B. Seidman. Network structure and minimum degree. Social networks, 5(3):269–287, 1983.

The Collapse of Friendster

- Founded in 2002.
- Popular at early 21st century, over 115 million users in 2011.
- Suspended in 2015 for lack of engagement by the online community.

D. Garcia, P. Mavrodiev, and F. Schweitzer. Social resilience in online communities: the autopsy of friendster. In COSN, pages 39–50, 2013.

The core number threshold steadily increased.

The Collapse of Friendster

- Founded in 2002.
- Popular at early 21st century, over 115 million users in 2011.
- Suspended in 2015 for lack of engagement by the online community.

K. Seki and M. Nakamura. The collapse of the friendster network started from the center of the core. In ASONAM, pages 477–484, 2016.

The collapse started from the center of the core.

User Engagement

- Founded in 2002.
- Popular at early 21st century, over 115 million users in 2011.
- Suspended in 2015 for lack of engagement by the online community.

J. Ugander, L. Backstrom, C. Marlow, and J. Kleinberg. Structural diversity in social contagion. PNAS, 109(16):5962–5966, 2012.

Social influence is tightly controlled by the number of friends in current subgraph, like *k*-core.

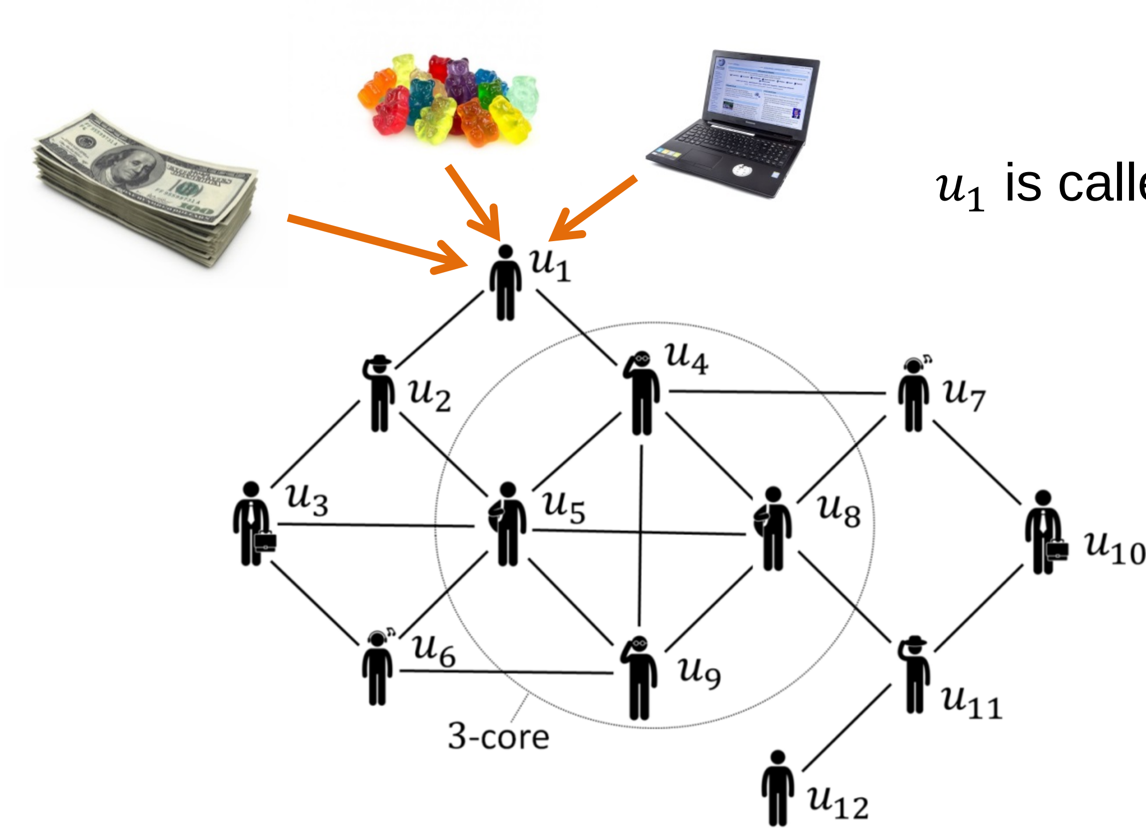
User Engagement

- Founded in 2002.
- Popular at early 21st century, over 115 million users in 2011.
- Suspended in 2015 for lack of engagement by the online community.

F. D. Malliaros and M. Vazirgiannis. To stay or not to stay: modeling engagement dynamics in social graphs. In CIKM, pages 469–478, 2013.

The degeneration property of k-core can be used to quantify engagement dynamics.

Prevent Network Unraveling



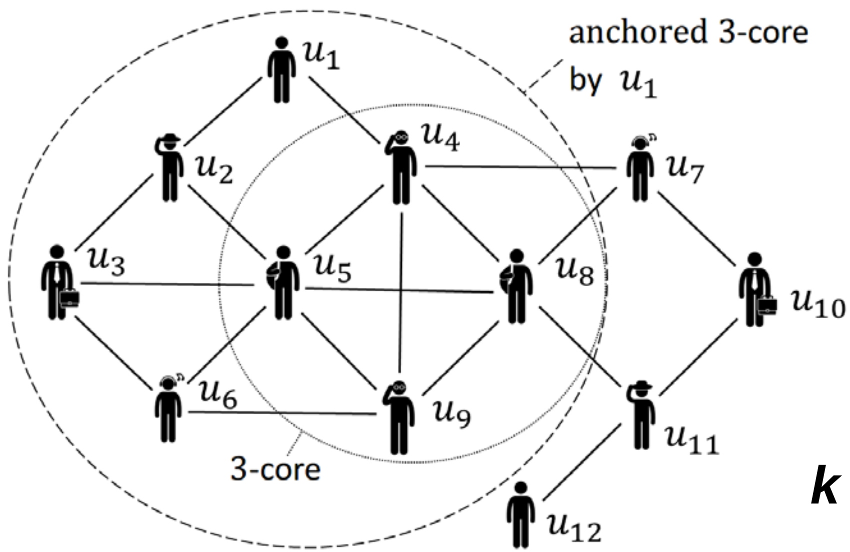
u_1 is called an anchor.

Kshipra Bhawalkar, Jon Kleinberg, Kevin Lewi, Tim Roughgarden, and Aneesh Sharma. "Preventing unraveling in social networks: the anchored k-core problem." SIAM Journal on Discrete Mathematics 29, no. 3 (2015): 1452-1475.

Prevent Network Unraveling

Anchor: if a node *u* is an anchor, *u* will never leave the *k*-core community (i.e., the degree of *u* is always $+\infty$).

Anchored *k*-Core: the *k*-core with some anchors.



***k* = 3**

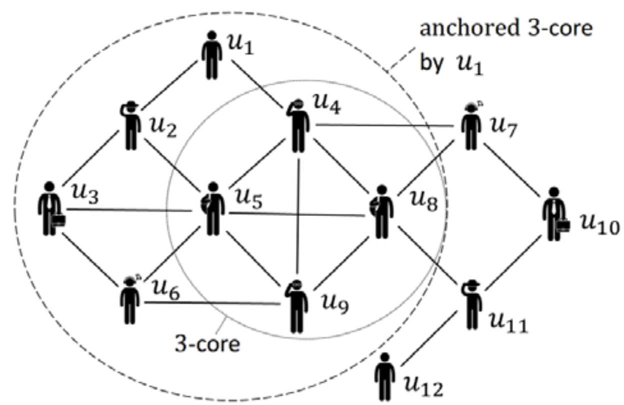
Prevent Network Unraveling

Follower: a node *v* is a follower of an anchor *u,* if *v* is not in k-core but belongs to anchored k-core by anchoring *u.*

Anchored k-Core Problem: Given two integers *k* and *b*, find *b* anchors to maximize the number of followers (i.e., maximize the number of nodes in anchored *k*-core).

NP-Hard

When *k* = 3 and *b* = 1, u_1 is a best anchor with 3 followers for the anchored k-core problem.



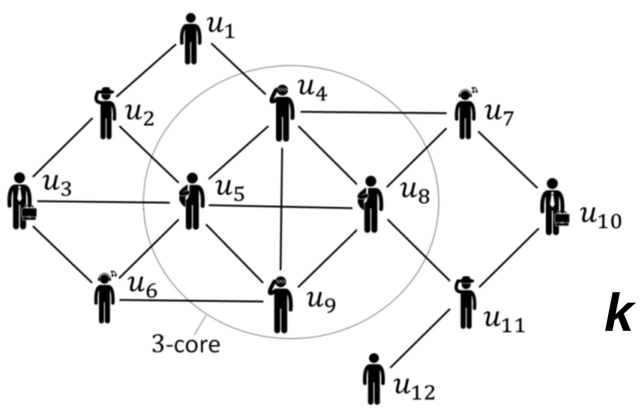
K. Bhawalkar, J. M. Kleinberg, K. Lewi, T. Roughgarden, and A. Sharma. Preventing unraveling in social networks: the anchored k-core problem. *SIAM J. Discrete Math.*, 29(3):1452–1475, 2015.

Theorems for Anchoring One Node

***k*-Shell:** the nodes in k-core but not in (k+1)-core.

•**Theorem 1:** if *v* is a follower of *u*, *v* belongs to (*k*-1)-shell.

•**Theorem 2:** if *u* has at least 1 follower, *u* belongs to (*k*-1)-shell or *u* is a neighbor of a node in (*k*-1)-shell.



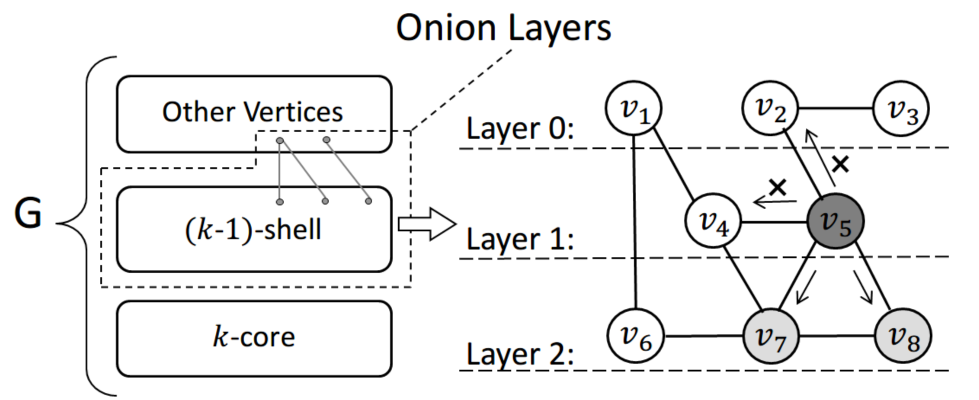
***k* = 3**

OLAK Algorithm for Anchored *k*-Core Problem

A greedy algorithm: Computing anchored *k*-core for every candidate anchor node to find a best anchor (the one with most followers) in each iteration.

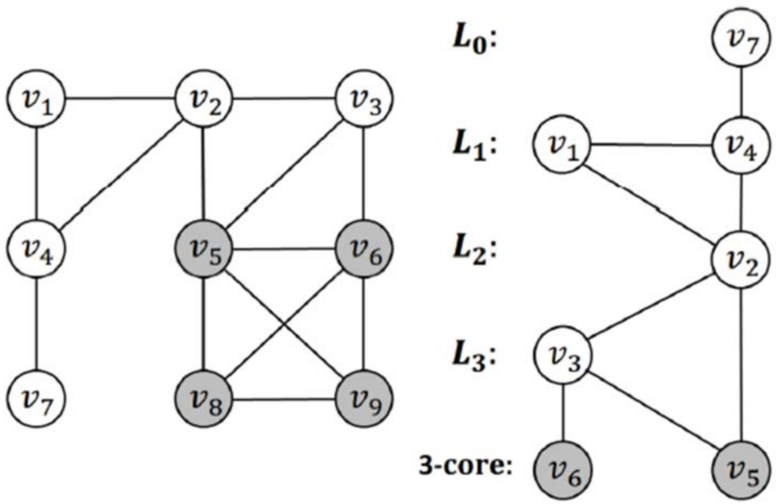
Onion Layers: a structure based on (*k*-1)-shell and the neighbors of (*k*-1)-shell nodes according to deletion order of these nodes in *k*-core computation.

We only need to explore a small portion of the Onion Layers to find all followers for an anchor.



Onion Layers in OLAK Algorithm

k = 3 in the following example



Onion Layer Structure (L_0^3)

$C_k(G)$ is the k-core of G,

$deg(u, N)$ is the degree of u in N,

$NB(L, G)$ is the neighbor set of L in G

Algorithm : OnionPeeling(G, k)

Input : G : a social network, k : degree constraint

Output : *onion layers* $\mathcal{L}$ (i.e., L_0^s)

$N := C_{k-1}(G)$; $i := 0$;

$P := \{u \mid deg(u, N) < k \ \& \ u \in N\}$;

while $P \neq \emptyset$ **do**

  $i := i + 1$; $L_i := P$;

  $N := N \setminus P$;

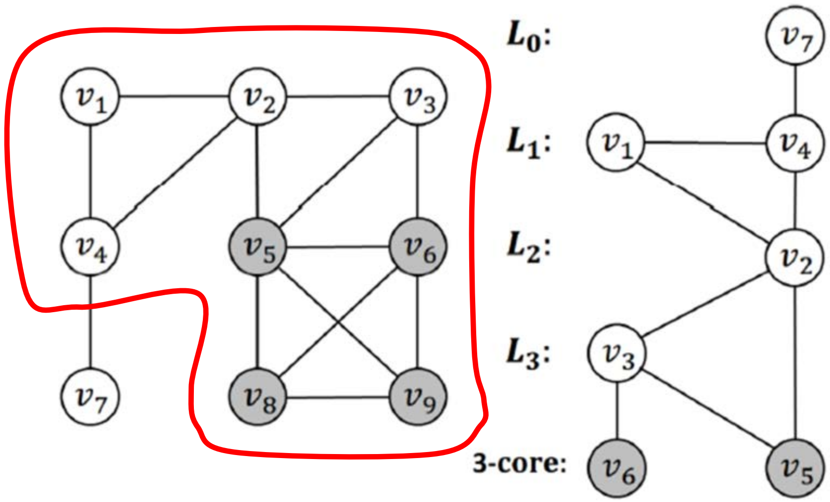
  $P := \{u \mid deg(u, N) < k \ \& \ u \in N\}$;

$L_0 := \{u \mid u \in NB(L_1^i, G) \setminus \{N \cup L_1^i\}\}$;

return L_0^i

Onion Layers in OLAK Algorithm

k = 3 in the following example



Onion Layer Structure (L_0^3)

P N

$C_k(G)$ is the k-core of G,
$deg(u, N)$ is the degree of u in N,
$NB(L, G)$ is the neighbor set of L in G

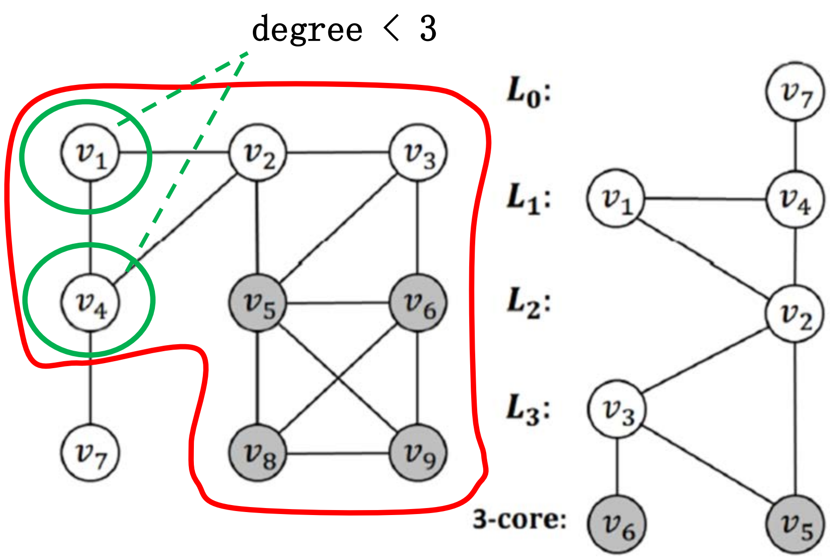
Algorithm : OnionPeeling$(G,\ k)$

Input : G : a social network, k : degree constraint
Output : *onion layers* $\mathcal{L}$ (i.e., L_0^s)

$N := C_{k-1}(G)$; $i := 0$;
$P := \{u \mid deg(u, N) < k \ \&\ u \in N\}$;
while $P \neq \emptyset$ **do**
  $i := i + 1$; $L_i := P$;
  $N := N \setminus P$;
  $P := \{u \mid deg(u, N) < k \ \&\ u \in N\}$;
$L_0 := \{u \mid u \in NB(L_1^i, G) \setminus \{N \cup L_1^i\}\}$;
return L_0^i

Onion Layers in OLAK Algorithm

k = 3 in the following example



Onion Layer Structure (L_0^3)

$C_k(G)$ is the k-core of G,
$deg(u, N)$ is the degree of u in N,
$NB(L, G)$ is the neighbor set of L in G

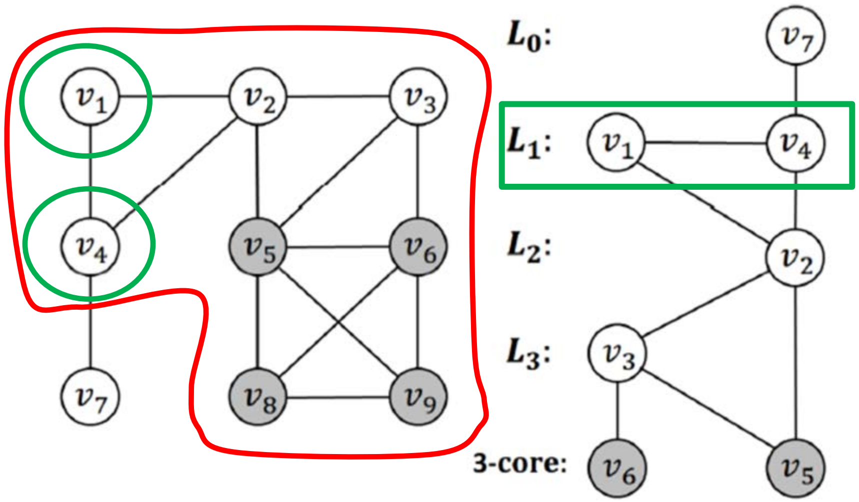
Algorithm : OnionPeeling(G, k)

Input : G : a social network, k : degree constraint
Output : *onion layers* $\mathcal{L}$ (i.e., L_0^s)

$N := C_{k-1}(G)$; $i := 0$;
$P := \{u \mid deg(u, N) < k \;\&\; u \in N\}$;
while $P \neq \emptyset$ **do**
  $i := i + 1$; $L_i := P$;
  $N := N \setminus P$;
  $P := \{u \mid deg(u, N) < k \;\&\; u \in N\}$;
$L_0 := \{u \mid u \in NB(L_1^i, G) \setminus \{N \cup L_1^i\}\}$;
return L_0^i

Onion Layers in OLAK Algorithm

k = 3 in the following example



Onion Layer Structure (L_0^3)

P N

$C_k(G)$ is the k-core of G,
$deg(u, N)$ is the degree of u in N,
$NB(L, G)$ is the neighbor set of L in G

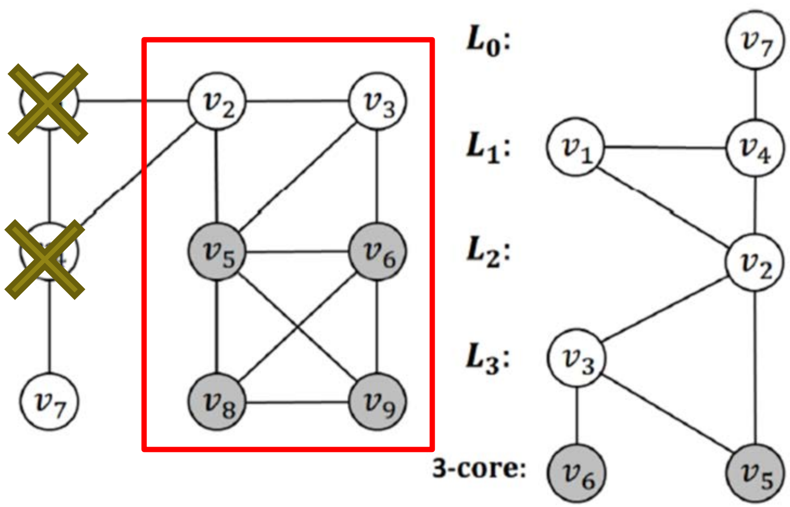
Algorithm : OnionPeeling(G, k)

Input : G : a social network, k : degree constraint
Output : *onion layers* $\mathcal{L}$ (i.e., L_0^s)

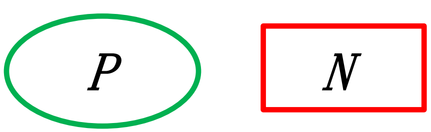
$N := C_{k-1}(G)$; $i := 0$;
$P := \{u \mid deg(u, N) < k \;\&\; u \in N\}$;
while $P \neq \emptyset$ **do**
  $i := i + 1$; $L_i := P$;
  $N := N \setminus P$;
  $P := \{u \mid deg(u, N) < k \;\&\; u \in N\}$;
$L_0 := \{u \mid u \in NB(L_1^i, G) \setminus \{N \cup L_1^i\}\}$;
return L_0^i

Onion Layers in OLAK Algorithm

k = 3 in the following example



Onion Layer Structure (L_0^3)



$C_k(G)$ is the k-core of G,
$deg(u, N)$ is the degree of u in N,
$NB(L, G)$ is the neighbor set of L in G

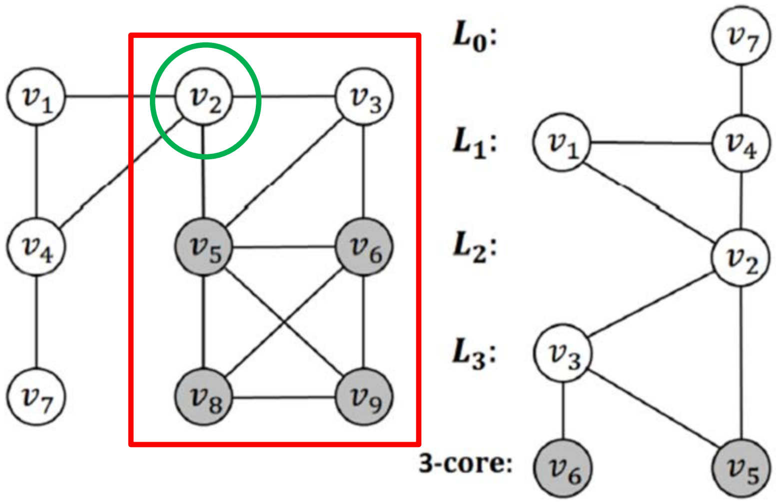
Algorithm : OnionPeeling(G, k)

Input : G : a social network, k : degree constraint
Output : *onion layers* $\mathcal{L}$ (i.e., L_0^s)

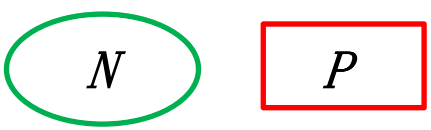
```
N := C_{k-1}(G); i := 0;
P := {u | deg(u, N) < k & u ∈ N};
while P ≠ ∅ do
    i := i + 1; L_i := P;
    N := N \ P;
    P := {u | deg(u, N) < k & u ∈ N};
L_0 := {u | u ∈ NB(L_1^i, G) \ {N ∪ L_1^i}};
return L_0^i
```

Onion Layers in OLAK Algorithm

k = 3 in the following example



Onion Layer Structure (L_0^3)



$C_k(G)$ is the k-core of G,

$deg(u, N)$ is the degree of u in N,

$NB(L, G)$ is the neighbor set of L in G

Algorithm : OnionPeeling(G, k)

Input : G : a social network, k : degree constraint

Output : *onion layers* $\mathcal{L}$ (i.e., L_0^s)

$N := C_{k-1}(G)$; $i := 0$;

$P := \{u \mid deg(u, N) < k \;\&\; u \in N\}$;

while $P \neq \emptyset$ **do**

  $i := i + 1$; $L_i := P$;

  $N := N \setminus P$;

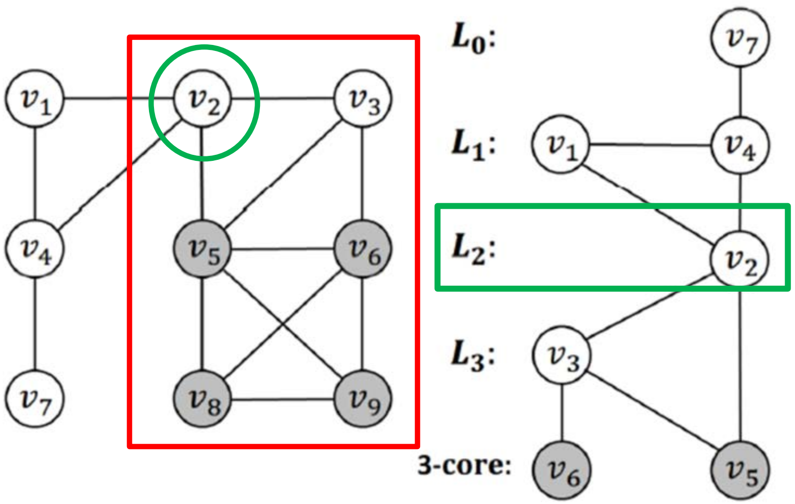
  $P := \{u \mid deg(u, N) < k \;\&\; u \in N\}$;

$L_0 := \{u \mid u \in NB(L_1^i, G) \setminus \{N \cup L_1^i\}\}$;

return L_0^i

Onion Layers in OLAK Algorithm

k = 3 in the following example



Onion Layer Structure (L_0^3)

N P

$C_k(G)$ is the k-core of G,
$deg(u,N)$ is the degree of u in N,
$NB(L,G)$ is the neighbor set of L in G

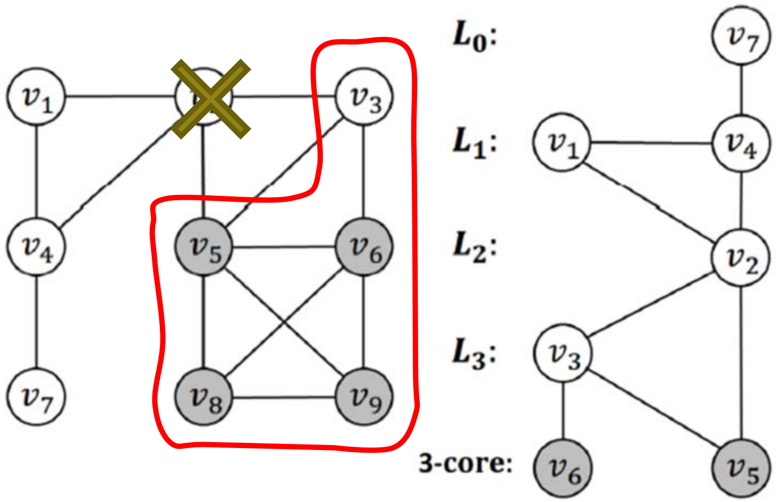
Algorithm : OnionPeeling$(G,\ k)$

Input : G : a social network, k : degree constraint
Output : *onion layers* $\mathcal{L}$ (i.e., L_0^s)

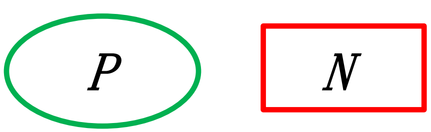
$N := C_{k-1}(G)$; $i := 0$;
$P := \{u \mid deg(u,N) < k\ \&\ u \in N\}$;
while $P \neq \emptyset$ **do**
  $i := i+1$; $L_i := P$;
  $N := N \setminus P$;
  $P := \{u \mid deg(u,N) < k\ \&\ u \in N\}$;
$L_0 := \{u \mid u \in NB(L_1^i, G) \setminus \{N \cup L_1^i\}\}$;
return L_0^i

Onion Layers in OLAK Algorithm

k = 3 in the following example



Onion Layer Structure (L_0^3)



$C_k(G)$ is the k-core of G,

$deg(u, N)$ is the degree of u in N,

$NB(L, G)$ is the neighbor set of L in G

Algorithm : OnionPeeling(G, k)

Input : G : a social network, k : degree constraint

Output : *onion layers* $\mathcal{L}$ (i.e., L_0^s)

$N := C_{k-1}(G)$; $i := 0$;

$P := \{u \mid deg(u, N) < k \;\&\; u \in N\}$;

while $P \neq \emptyset$ **do**

  $i := i + 1$; $L_i := P$;

  $N := N \setminus P$;

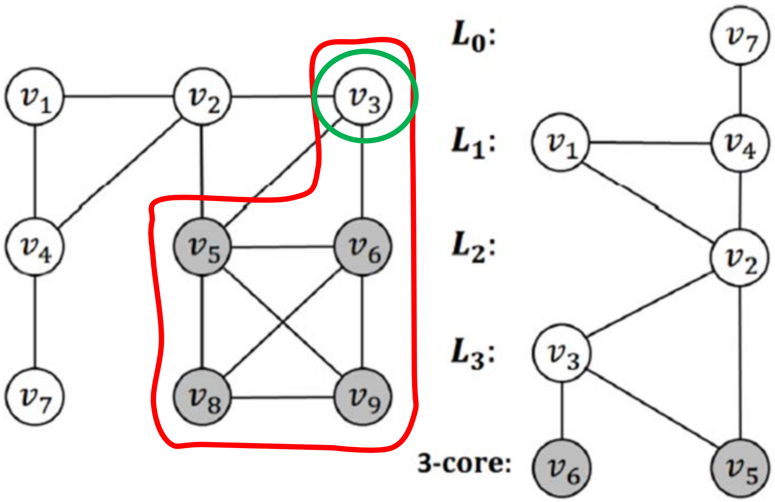
  $P := \{u \mid deg(u, N) < k \;\&\; u \in N\}$;

$L_0 := \{u \mid u \in NB(L_1^i, G) \setminus \{N \cup L_1^i\}\}$;

return L_0^i

Onion Layers in OLAK Algorithm

k = 3 in the following example



Onion Layer Structure (L_0^3)

P N

$C_k(G)$ is the k-core of G,

$deg(u, N)$ is the degree of u in N,

$NB(L, G)$ is the neighbor set of L in G

Algorithm : OnionPeeling(G, k)

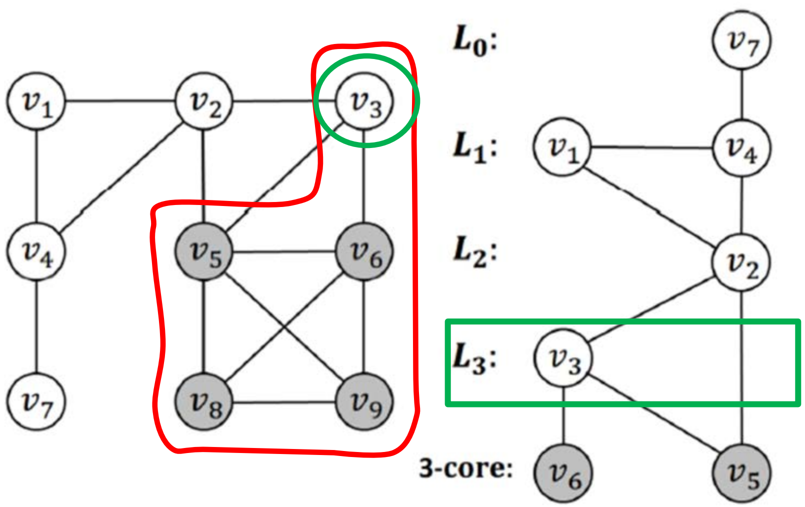
Input : G : a social network, k : degree constraint

Output : *onion layers* $\mathcal{L}$ (i.e., L_0^s)

```
N := C_{k-1}(G); i := 0;
P := {u | deg(u, N) < k & u ∈ N};
while P ≠ ∅ do
    i := i + 1; L_i := P;
    N := N \ P;
    P := {u | deg(u, N) < k & u ∈ N};
L_0 := {u | u ∈ NB(L_1^i, G) \ {N ∪ L_1^i}};
return L_0^i
```

Onion Layers in OLAK Algorithm

k = 3 in the following example



Onion Layer Structure (L_0^3)

P N

$C_k(G)$ is the k-core of G,
$deg(u, N)$ is the degree of u in N,
$NB(L, G)$ is the neighbor set of L in G

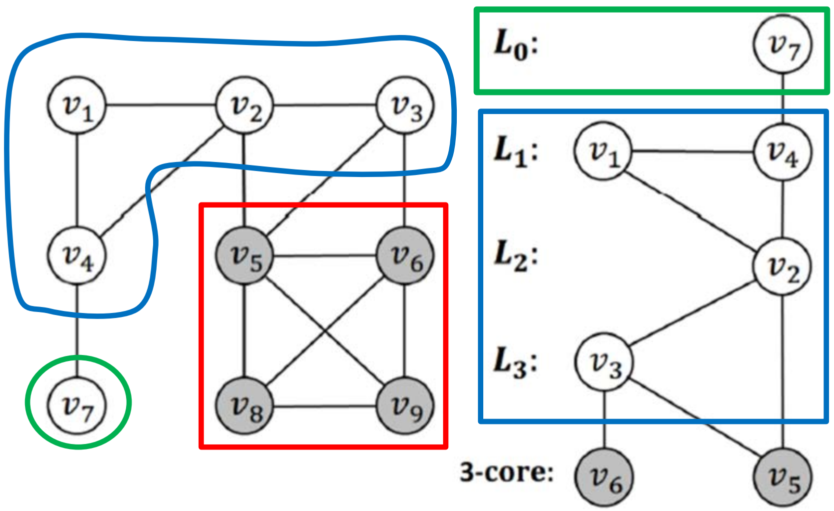
Algorithm : OnionPeeling(G, k)

Input : G : a social network, k : degree constraint
Output : *onion layers* $\mathcal{L}$ (i.e., L_0^s)

$N := C_{k-1}(G)$; $i := 0$;
$P := \{u \mid deg(u, N) < k \ \& \ u \in N\}$;
while $P \neq \emptyset$ **do**
  $i := i + 1$; $L_i := P$;
  $N := N \setminus P$;
  $P := \{u \mid deg(u, N) < k \ \& \ u \in N\}$;
$L_0 := \{u \mid u \in NB(L_1^i, G) \setminus \{N \cup L_1^i\}\}$;
return L_0^i

Onion Layers in OLAK Algorithm

k = 3 in the following example



Onion Layer Structure (L_0^3)

$C_k(G)$ is the k-core of G,
$deg(u, N)$ is the degree of u in N,
$NB(L, G)$ is the neighbor set of L in G

Algorithm : OnionPeeling(G, k)

Input : G : a social network, k : degree constraint
Output : *onion layers* $\mathcal{L}$ (i.e., L_0^s)

$N := C_{k-1}(G)$; $i := 0$;
$P := \{u \mid deg(u, N) < k \ \& \ u \in N\}$;
while $P \neq \emptyset$ **do**
  $i := i + 1$; $L_i := P$;
  $N := N \setminus P$;
  $P := \{u \mid deg(u, N) < k \ \& \ u \in N\}$;
$L_0 := \{u \mid u \in NB(L_1^i, G) \setminus \{N \cup L_1^i\}\}$;
return L_0^i

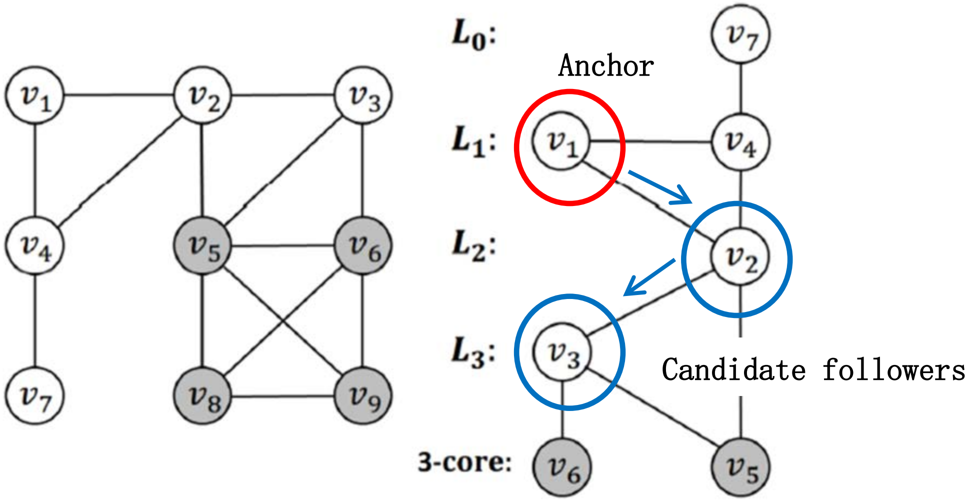
N

After **OnionPeeling** algorithm, N is the k-core of G.

Theorems for Anchored k-Core

Support Path: there is a support path from *u* to *v* if *u* can downward spread to *v* in Onion Layers through neighboring edges. Horizontal or upward spreads are NOT allowed.

Theorem 3: if *v* is a follower of *u*, there is a support path from *u* to *v*.

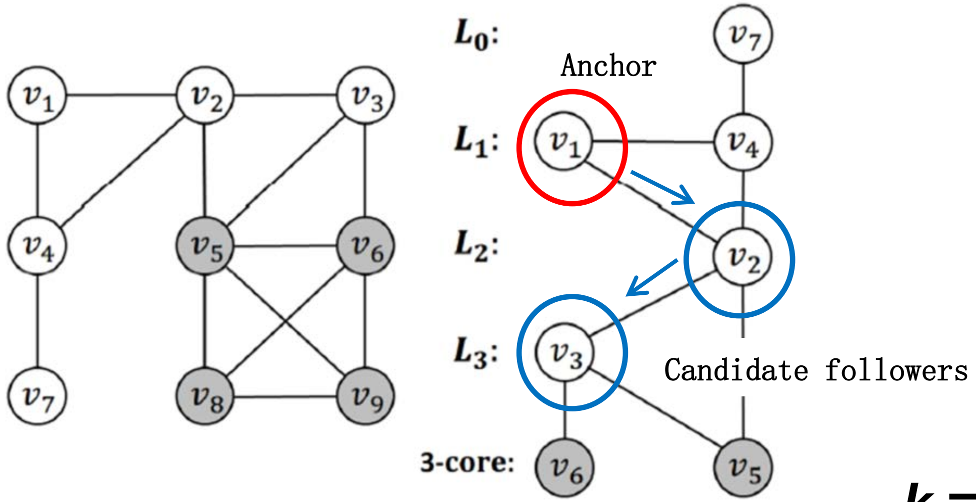


Onion Layer Structure (L_0^3)

Onion Layer Search to Find Followers

If we anchor the node v_1, only v_2 and v_3 become candidate followers, v_4 and v_7 cannot be followers of v_1.

Reason: v_4 and v_7 will still be deleted in the deletion order of producing onion layers (i.e., producing k-core), i.e., v_4 and v_7 cannot have larger degrees after anchoring v_1.



Onion Layer Structure (L_0^3)

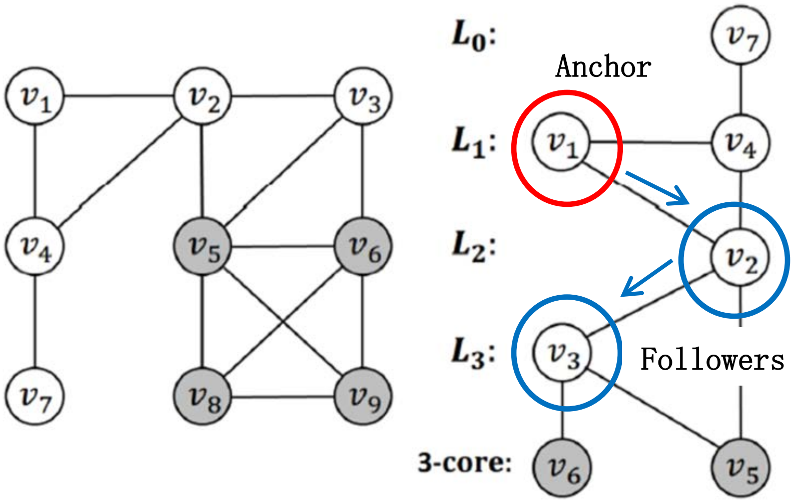
***k* = 3**

Theorems for Anchored k-Core

Theorem 4: if the degree upperbound of *u* is less than *k* in the Onion Layer Search, we can early terminate the spread on *u*.

Theorem 5: if *v* is a follower of *u*, *v* cannot have more followers than *u.*

v_2 or v_3 cannot have more followers than v_1.



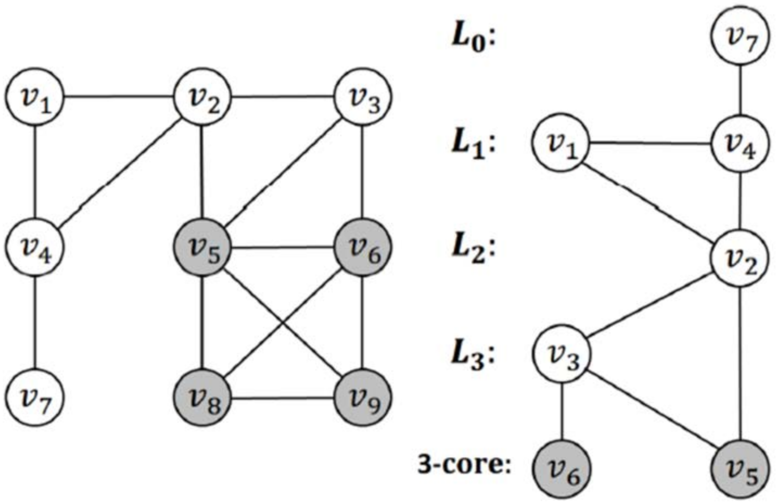
***k* = 3** Onion Layer Structure (L_0^3)

Follower Number Upper Bound

Let $W(x)$ denote the neighbors of a vertex x in lower layers, i.e., $W(x) = \{u \mid u \in NB(x) \cap \mathcal{L} \text{ and } l(u) > l(x)\}$. We use $UB(x)$ to denote the upper bound of $|\mathcal{F}(x)|$, where

$$UB(x) = \begin{cases} \sum_{u \in W(x)} (UB(u) + 1) & \textit{if } |W(x)| > 0; \\ 0 & \textit{otherwise.} \end{cases} \quad (1)$$

L is the *Onion Layers*,
$l(u)$ is the layer number of u,
$NB(u)$ is the neighbor set of u,
$F(x)$ is the follower set of x,



***k* = 3** Onion Layer Structure (L_0^3)

Theorem 6: An anchor *x* cannot have more followers than *UB(x)*.

Follower Number Upper Bound

Let $W(x)$ denote the neighbors of a vertex x in lower layers, i.e., $W(x) = \{u \mid u \in NB(x) \cap \mathcal{L} \text{ and } l(u) > l(x)\}$. We use $UB(x)$ to denote the upper bound of $|\mathcal{F}(x)|$, where

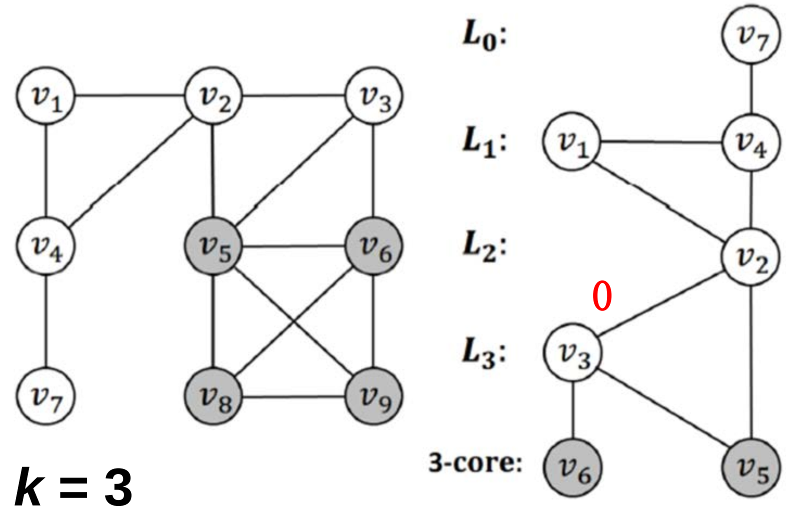
$$UB(x) = \begin{cases} \sum_{u \in W(x)} (UB(u) + 1) & \text{if } |W(x)| > 0; \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

L is the *Onion Layers*,

$l(u)$ is the layer number of u,

$NB(u)$ is the neighbor set of u,

$F(x)$ is the follower set of x,



***k* = 3**

Onion Layer Structure (L_0^3)

Theorem 6: An anchor *x* cannot have more followers than *UB(x)*.

Follower Number Upper Bound

Let $W(x)$ denote the neighbors of a vertex x in lower layers, i.e., $W(x) = \{u \mid u \in NB(x) \cap \mathcal{L} \text{ and } l(u) > l(x)\}$. We use $UB(x)$ to denote the upper bound of $|\mathcal{F}(x)|$, where

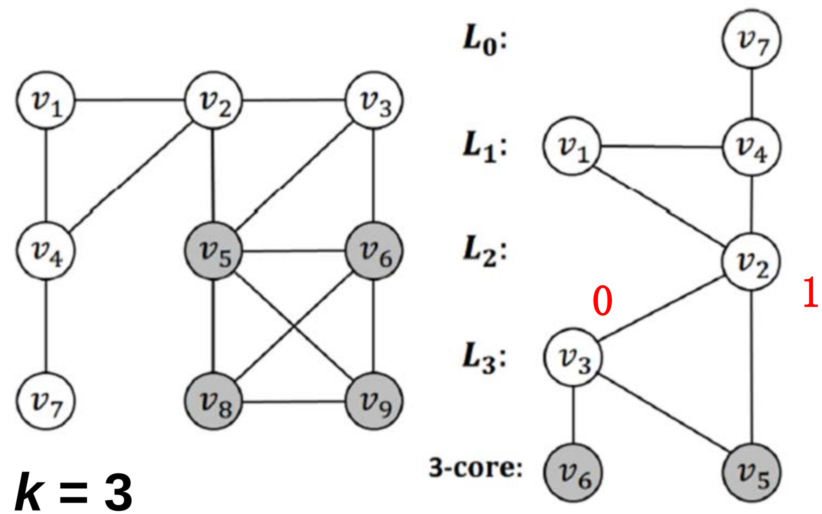
$$UB(x) = \begin{cases} \sum_{u \in W(x)} (UB(u) + 1) & \text{if } |W(x)| > 0; \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

L is the *Onion Layers*,

$l(u)$ is the layer number of u,

$NB(u)$ is the neighbor set of u,

$F(x)$ is the follower set of x,



***k* = 3**

Onion Layer Structure (L_0^3)

Theorem 6: An anchor *x* cannot have more followers than *UB(x)*.

Follower Number Upper Bound

Let $W(x)$ denote the neighbors of a vertex x in lower layers, i.e., $W(x) = \{u \mid u \in NB(x) \cap \mathcal{L} \text{ and } l(u) > l(x)\}$. We use $UB(x)$ to denote the upper bound of $|\mathcal{F}(x)|$, where

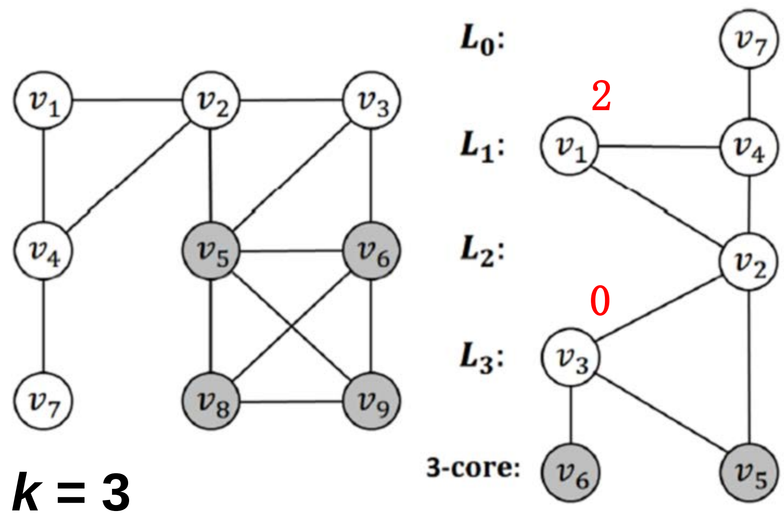
$$UB(x) = \begin{cases} \sum_{u \in W(x)} (UB(u) + 1) & \textit{if } \ |W(x)| > 0; \\ 0 & \textit{otherwise.} \end{cases} \quad (1)$$

L is the *Onion Layers*,

$l(u)$ is the layer number of u,

$NB(u)$ is the neighbor set of u,

$F(x)$ is the follower set of x,



Onion Layer Structure (L_0^3)

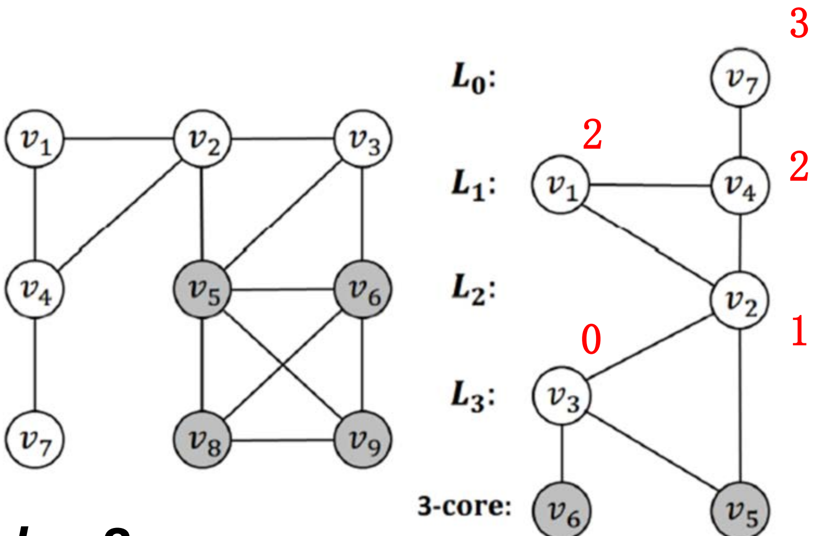
Theorem 6: An anchor *x* cannot have more followers than *UB(x)*.

Follower Number Upper Bound

Let $W(x)$ denote the neighbors of a vertex x in lower layers, i.e., $W(x) = \{u \mid u \in NB(x) \cap \mathcal{L} \text{ and } l(u) > l(x)\}$. We use $UB(x)$ to denote the upper bound of $|\mathcal{F}(x)|$, where

$$UB(x) = \begin{cases} \sum_{u \in W(x)} (UB(u) + 1) & \textit{if } \ |W(x)| > 0; \\ 0 & \textit{otherwise.} \end{cases} \quad (1)$$

L is the *Onion Layers*,
$l(u)$ is the layer number of u,
$NB(u)$ is the neighbor set of u,
$F(x)$ is the follower set of x,



***k* = 3**

Onion Layer Structure (L_0^3)

Theorem 6: An anchor *x* cannot have more followers than *UB(x)*.

Experimental Setting

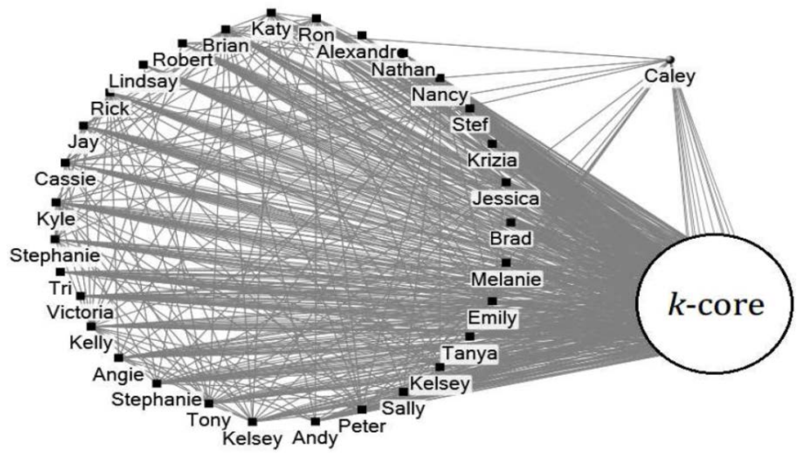
- Datasets:

Dataset	Nodes	Edges	d_{avg}	d_{max}
Facebook	4,039	88,234	43.7	1045
Brightkite	58,228	194,090	6.7	1098
Gowalla	196,591	456,830	4.7	9967
Yelp	552,339	1,781,908	6.5	3812
Flickr	105,938	2,316,948	43.7	5465
YouTube	1,134,890	2,987,624	5.3	28754
DBLP	1,566,919	6,461,300	8.3	2023
Pokec	1,632,803	8,320,605	10.2	7266
LiveJournal	3,997,962	34,681,189	17.4	14815
Orkut	3,072,441	117,185,083	76.3	33313

- Environments:
  - Intel Xeon 2.3GHz CPU and Redhat Linux System.
  - All algorithms are implemented in C++.

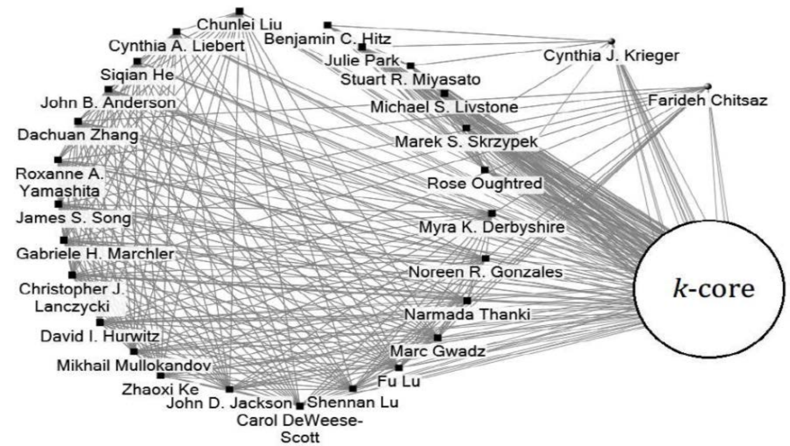
Case Studies

Yelp is a crowd-sourced local business review and social networking site.



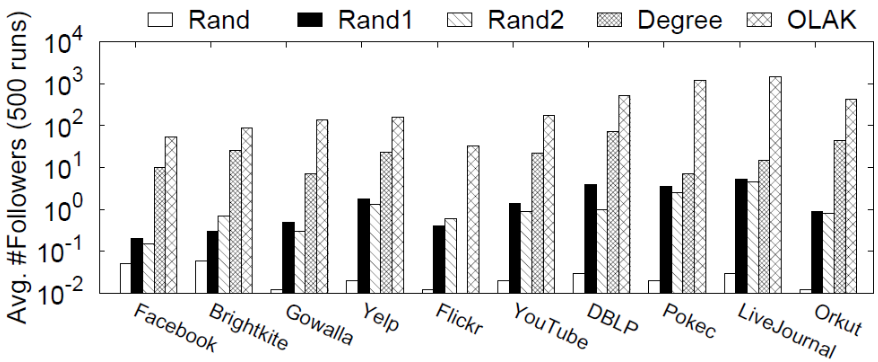
(a) Yelp, k=30, b=1

DBLP is a computer science bibliography website.

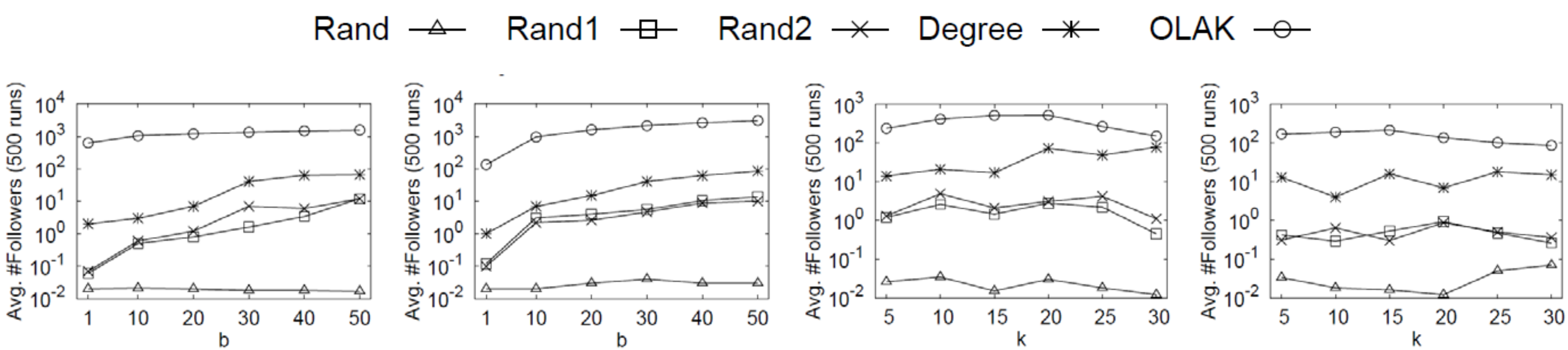


(b) DBLP, k=20, b=2

Number of Followers

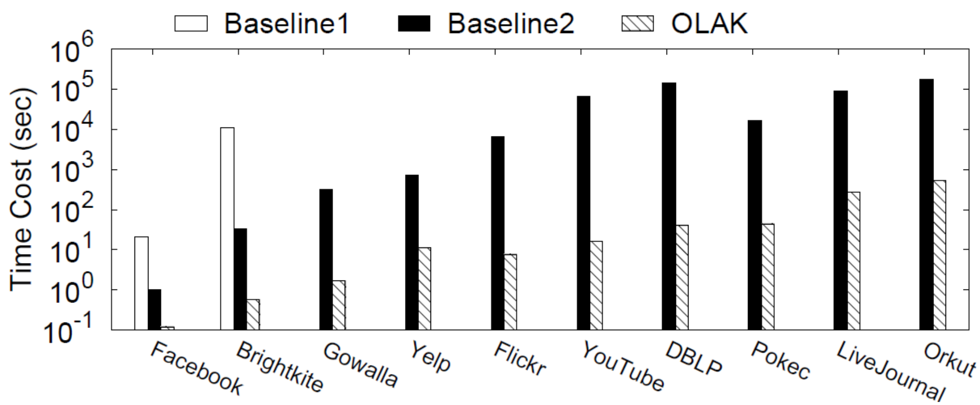


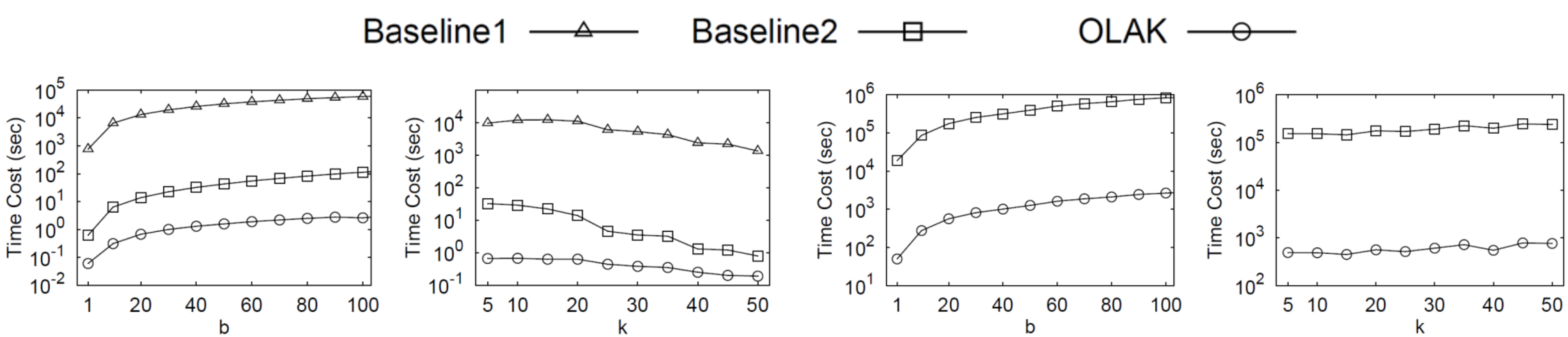
(a) 10 Datasets, k=20, b=20



(b) Pokec, k=20 (c) LiveJournal, k=20 (d) DBLP, b=20 (e) Gowalla, b=20

Efficiency





(a) Brightkite, k=20 (b) Brightkite, b=20 (c) Orkut, k=20 (d) Orkut, b=20

THANK YOU

Q&A