

## 7 APPENDIX

PROOF OF COROLLARY 1. For DkS, we set  $|S| = k$ , it has an upper bound as follows:

$$\begin{aligned} \rho(S) &= \frac{\sum_{e \in \mathcal{E}(S)} w_e}{|S|} = \frac{\sum_{e=(u,v), u,v \in S} f_e(u) + f_e(v)}{|S|} \\ &\leq \frac{\sum_{e=(u,v), u,v \in S} (f_e(u) + f_e(v)) + \sum_{e=(u,v), u \in S, v \notin S} f_e(u)}{|S|} \\ &= \frac{\sum_{u \in S} \sum_{e \ni u} f_e(u)}{|S|} = \frac{\sum_{u \in S} l_u}{|S|} \\ &= \frac{\sum_{i=0}^{j-1} \lambda_i * |B_i| + (k - |B_{j-1}|) * \lambda_j}{k} \end{aligned}$$

Therefore, result 2 holds up. As for result 1, if  $k = |B_j|$ , according to property 3 the equivalency condition in line 2 holds up because  $f_e(u) = 0$  if  $u \in S$  and  $v \notin S$ , then result 1 holds up. These results also hold up for DalkS because if  $|S| > k$ , it will add more nodes into  $S$  with lower upper bounds on their loads.  $\square$

Before proofs of Lemma 2, we introduce the definition of  $k$ -core from [43]:  $k$ -core is the maximal subgraph  $G_k$  in graph  $G$ , the degree of where any vertex  $v$  in  $G_k$  is satisfied with  $d_{G_k}(v) \geq k$ .

PROOF OF LEMMA 2. The Greedy algorithm just moves any node whose degree is the lowest in the remaining graph  $\mathcal{H}$ . Let us remark  $A(k)$  as the nodeset of  $k$ -core for specific  $k$ . We claim that when deleting a node  $u \in A(k)$ , there mustn't be any node  $v \notin A(k)$  in the remaining graph  $\mathcal{H}$ . We prove it by way of contradiction, w.l.o.g, we set  $u$  as the first node to be deleted in  $A(k)$  in Greedy, therefore  $u$  has the lowest degree in  $\mathcal{H}$  and  $d_{\mathcal{H}}(u) \geq d_{A(k)}(u) \geq k$  for any node  $v$  in  $\mathcal{H}$ , which produces a  $k$ -core subgraph with a larger size, and it leads to a contradiction.  $\square$

PROOF OF THEOREM 5. We set  $\mathcal{H}_k (k > 0)$  is the remaining graph after  $k$  iterations in Algorithm 2 and  $\mathcal{H}_0$  is the initial whole graph,  $\mathcal{H}'_{k+1}$  is the nodeset to be deleted in  $k+1$  iteration. Therefore,  $\mathcal{H}_{k+1} = \mathcal{H}_k \setminus \mathcal{H}'_{k+1}$ ,

$$\begin{aligned} \rho(\mathcal{H}_{k+1}) &= \rho(\mathcal{H}_k \setminus \mathcal{H}'_{k+1}) \\ &= \frac{\mathcal{W}(\mathcal{E}(\mathcal{H}_k)) - \mathcal{W}(\mathcal{E}(\mathcal{H}'_{k+1}))}{|\mathcal{H}_k| - |\mathcal{H}'_{k+1}|} \\ &\geq \frac{\rho(\mathcal{H}_k) \cdot |\mathcal{H}_k| - \sum_{v \in \mathcal{H}'_{k+1}} d_{\mathcal{H}_k}(v)}{|\mathcal{H}_k| - |\mathcal{H}'_{k+1}|} \\ &> \frac{\rho(\mathcal{H}_k) \cdot |\mathcal{H}_k| - \sum_{v \in \mathcal{H}'_{k+1}} \rho(\mathcal{H}_k)}{|\mathcal{H}_k| - |\mathcal{H}'_{k+1}|} \\ &= \frac{\rho(\mathcal{H}_k) \cdot |\mathcal{H}_k| - \rho(\mathcal{H}_k) \cdot |\mathcal{H}'_{k+1}|}{|\mathcal{H}_k| - |\mathcal{H}'_{k+1}|} \\ &= \rho(\mathcal{H}_k) \end{aligned}$$

That means the density of graph  $\mathcal{H}$  monotonically increases in iterations, then any deleted node has a lower degree (when it is being deleted) than the final density, i.e.,  $\delta$ . Therefore, the remaining graph  $\mathcal{H}$  is a  $\delta$ -core and it is the graph of some time of the greedy search according to Lemma 2.

We claim that the process before getting the  $\delta$ -core is a monotonic increasing phase of density in Greedy. We can confirm two facts:

1. During Greedy, if there is a node  $u \in \mathcal{H}'_1$  existing in the remaining graph, the deletion in Greedy will increase the density of the remaining graph. That's because if Greedy deletes a node  $v \notin \mathcal{H}'_1$ , then  $d_{\mathcal{H}}(v) \leq d_{\mathcal{H}}(u) < \rho(\mathcal{G}) \leq \rho(\mathcal{H})$ .  $\rho(\mathcal{H})$  will increase and  $\rho(\mathcal{G}) \leq \rho(\mathcal{H})$  still holds up. If Greedy deletes the node  $u$ , now that  $d_{\mathcal{H}}(u) \leq \rho(\mathcal{H})$ , then  $\rho(\mathcal{H})$  will also increase and  $\rho(\mathcal{G}) \leq \rho(\mathcal{H})$  holds up.
2. During Greedy, if there is a node  $u \in \mathcal{H}'_{k+1}$  existing in the remaining graph  $\mathcal{H} > \mathcal{H}_k$ , and there isn't any node belonging to  $\mathcal{H}'_k$ . Then:  $\rho(\mathcal{H}) \geq \rho(\mathcal{H}_k)$  because we delete more nodes with lower degrees. Therefore, when we deletes a node  $v \notin \mathcal{H}'_{k+1}$ , then  $d_{\mathcal{H}}(v) \leq d_{\mathcal{H}}(u) < \rho(\mathcal{G}) \leq \rho(\mathcal{H}_k) \leq \rho(\mathcal{H})$ , then  $\rho(\mathcal{H})$  will increase and  $\rho(\mathcal{G}) \leq \rho(\mathcal{H})$  holds up. If Greedy deletes the node  $u$ , now that  $d_{\mathcal{H}}(v) \leq \rho(\mathcal{H})$ ,  $\rho(\mathcal{H})$  will also increase and  $\rho(\mathcal{G}) \leq \rho(\mathcal{H})$  holds up.

Therefore, the density monotonically increases in Greedy before getting  $\delta$ -core.  $\square$

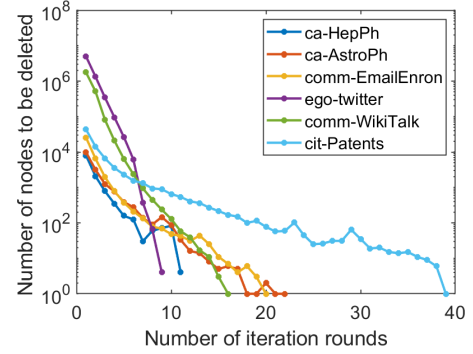


Figure 7: Exponential decrease in the number of deleted nodes.

---

### Algorithm 3: GREEDY DSPSOLVER

---

**Input:** Undirected graph  $\mathcal{G}$ ; density metric  $\rho(\cdot)$

**Output:**  $\mathcal{S}^*$ : the nodeset of the densest subgraph of  $\mathcal{G}$ .

---

```

1  $\mathcal{S}, \mathcal{S}^* \leftarrow \mathcal{V}$ 
2 while  $\mathcal{S} \neq \emptyset$  do
3    $\triangleright$  find the vertex  $u^*$  with the lowest degree in  $\mathcal{S}$ 
4    $u^* \leftarrow \arg \min_{u \in \mathcal{S}} d_{\mathcal{S}}(u)$ 
5   Remove  $u^*$  and all its adjacent edges from  $\mathcal{G}$ .
6    $\triangleright \mathcal{S} \setminus \{u\}$ : the remaining nodeset without  $u$ 
7    $\mathcal{S} \leftarrow \mathcal{S} \setminus \{u\}$ 
8   if  $\rho(\mathcal{S}) > \rho(\mathcal{S}^*)$  then
9      $\mathcal{S}^* \leftarrow \mathcal{S}$ 
10 return  $\mathcal{S}^*$ .

```

---

Table 5: Dataset source and density of algorithms

Dataset	Source	Type	Pruning	w_app	exact	DLL	uw_Pruning+DLL
ca-HepPh	Stanford's SNAP database	scholar collaboration network	119	119	119	119	119
comm-EmailEnron	Stanford's SNAP database	communication	37.316	37.344	37.344	37.337	37.337
ca-AstroPh	Stanford's SNAP database	scholar collaboration network	28.481	29.616	32.11	29.552	29.552
PP-Pathways	Stanford's SNAP database	protein interaction network	74.159	77.995	77.995	77.995	77.995
soc-Twitter_ICWSM	konect	social network	25.678	25.683	25.69	25.686	25.685
soc-sign_slashdot	Stanford's SNAP database	social network	39.376	42.132	42.132	42.132	42.132
rating-StackOverflow	konect	social network	20.209	20.209	20.21	20.209	20.209
soc-sign_epinion	Stanford's SNAP database	social network	80.168	85.599	85.637	85.589	85.589
ego-twitter	Stanford's SNAP database	social network	59.281	68.414	69.622	68.414	68.414
soc-Youtube	Stanford's SNAP database	social network	45.545	45.58	45.599	45.576	45.577
comm-WikiTalk	Stanford's SNAP database	communication	114.139	114.139	114.139	114.139	114.139
nov_user_msg_time	We own it privately.	social network	278.815	278.815	278.815	278.815	278.815
cit-Patents	AMiner scholar datasets	scholar collaboration network	132.776	135.706	137.261	135.706	135.706
soc-Twitter_ASU	ASU	social network	593.847	593.847	593.847	593.847	593.847
soc-Livejournal	Livejournal	social network	104.596	104.601	104.609	104.603	104.603
soc-Orkut	Stanford's SNAP database	social network	227.861	227.872	227.874	227.872	227.872
soc-SinaWeibo	Network Repository	social network	164.967	165.193	165.415	165.196	165.191
wang-tripadvisor	konect	rating network	13.442	13.873	14.082	—	—
rec-YelpUserBusiness	Network Repository	rating network	87.825	87.912	87.921	—	—
bookcrossing	konect	rating network	92.148	92.322	92.374	—	—
librec-ciaodvd-review	konect	rating network	233.553	233.59	233.597	—	—
movielens-10m	konect	rating network	1351.35	1351.35	1351.35	—	—
epinions	konect	rating network	595.302	595.314	595.316	—	—
libimseti	konect	social network	1645.71	1645.73	1645.73	—	—
rec-movielens	Network Repository	rating network	1801.16	1801.16	1801.16	—	—
yahoo-song	konect	rating network	46725.2	46725.2	46725.2	—	—

**note:** : **Pruning** (w\_Pruning,uw\_Pruning). **w\_app**: approximation algorithms on weighted graph(Priority Tree,Pruning+Priority Tree,BBST). **exact**: exact algorithms(maxflow,w\_Pruning+maxflow). **DLL**:Doubly-linked list.

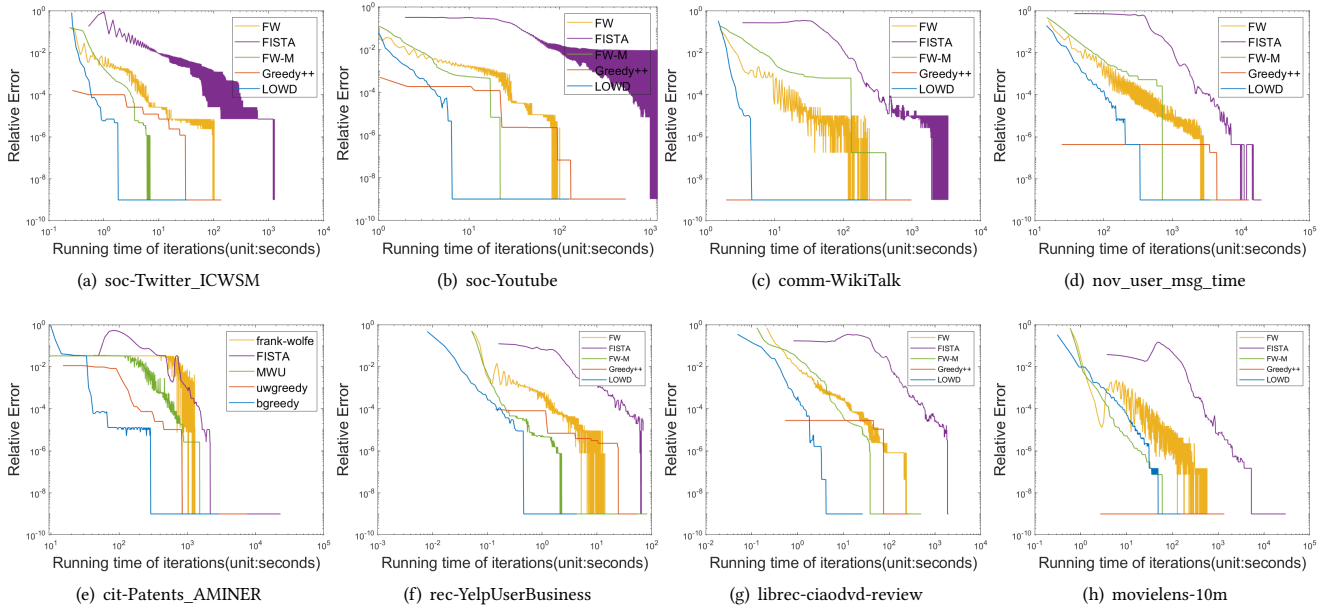


Figure 6: Comparison on detecting the densest subgraph without pruning.

**Table 6: Comparison on detecting the densest subgraph without pruning.**

Dataset	Running time/s					iteration count $T$				
	LOWD	Greedy++	FW	FISTA	FW_M	LOWD	Greedy++	FW	FISTA	FW_M
ca-HepPh	0.0168	<b>0.0159</b>	0.0208	0.1109	<u>0.0161</u>	1	1	2	3	1
comm-EmailEnron	<u>0.1516</u>	<b>0.0714</b>	0.5551	27.5346	0.3012	19	2	88	567	46
ca-AstroPh	<b>0.3158</b>	0.8396	2.312	24.2470	<u>0.6593</u>	36	33	343	375	92
PP-Pathways	<u>0.1174</u>	<b>0.0333</b>	0.3214	12.6981	<u>1.7086</u>	8	1	18	131	167
soc-sign_slashdot	<u>0.2275</u>	<b>0.0704</b>	0.7881	11.6623	1.1051	11	1	44	81	64
soc-sign_epinion	<b>0.3213</b>	1.4706	1.1853	11.1319	<u>0.9581</u>	9	12	40	46	33
soc-Twitter_ICWSM	<b>1.8343</b>	30.6654	96.3405	1207.2601	<u>6.1467</u>	49	114	2816	4901	175
rating-StackOverflow	<u>2.1887</u>	<b>1.0537</b>	10.0728	567.1578	18.7819	32	2	178	1125	341
ego-twitter	<b>1.6001</b>	2.9768	22.0971	2567.2520	<u>1.7266</u>	34	21	550	8115	41
soc-Youtube	<b>6.5253</b>	133.3086	82.3855	1001.4894	<u>22.0866</u>	46	129	612	929	161
comm-WikiTalk	<u>4.6758</u>	<b>1.984</b>	118.307	1933.6041	414.9261	15	1	407	1151	1386
nov_user_msg_time	<b>341.1144</b>	4464.4998	2581.9086	9826.2465	<u>718.8805</u>	97	174	624	454	177
cit-Patents_AMINER	<b>291.4452</b>	846.3628	<u>837.8446</u>	27554.332	1523.2698	104	56	259	1913	471
soc-Twitter_ASU	<u>120.8906</u>	<b>19.2912</b>	511.3170	12174.952	2047.9598	40	1	156	1244	603
soc-Livejournal	–	–	–	–	–	–	–	–	–	–
soc-Orkut	–	–	–	–	–	–	–	–	–	–
soc-SinaWeibo_NETREP	–	–	–	–	–	–	–	–	–	–
wang-tripadvisor	<b>0.6686</b>	23.6761	30.209	198.1408	<u>15.6705</u>	93	187	3894	4512	2007
rec-YelpUserBusiness	<b>0.4621</b>	24.5879	5.1698	62.8514	<u>2.1192</u>	54	228	626	886	239
bookcrossing	<b>1.5177</b>	92.9406	27.4916	910.8345	<u>8.877</u>	78	259	1398	5510	448
librec-ciaodvd-review	<b>4.2248</b>	75.7599	231.5822	1882.2756	<u>38.6405</u>	79	143	4635	4339	773
movielens-10m	<u>49.2341</u>	<b>2.7029</b>	129.6344	5329.5159	60.719	159	1	450	1611	211
epinions	<b>30.4071</b>	3957.2229	4302.242	>37786.731	<u>743.6429</u>	64	688	8893	>10000	1582
libimseti	<b>25.7101</b>	5276.5399	751.3750	24832.745	<u>146.2627</u>	39	710	1196	4632	234
rec-movielens	<b>50.4427</b>	6928.9143	306.0513	24923.281	<u>148.2539</u>	54	733	355	3195	172
yahoo-song	–	–	–	–	–	–	–	–	–	–

**note:** We ignore some datasets which are very large.">10000" and ">37786.731" means we run 10000 iterations(running time: 37786.731s) and can't still detect the densest subgraph.