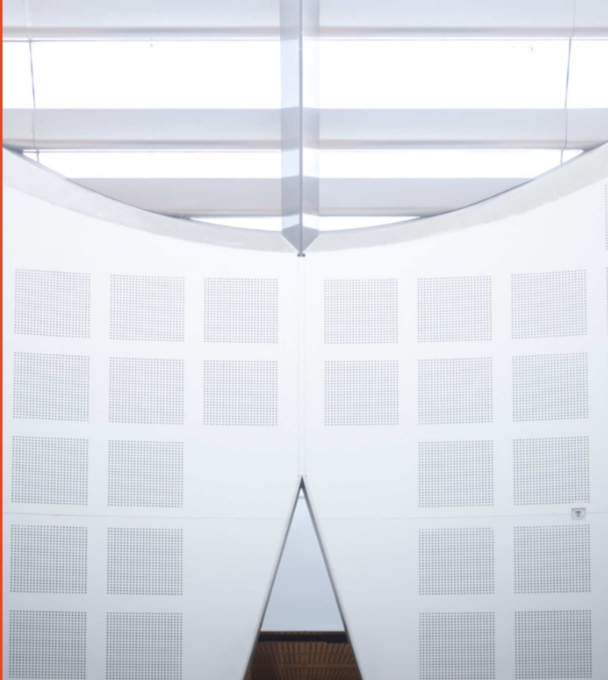
Efficient Maximum k-Defective Clique Computation with Improved Time Complexity

Lijun Chang

School of Computer Science
The University of Sydney

June 11, 2024

THE UNIVERSITY OF SYDNEY



Graphs are Everywhere

- A graph $G = (V, E)$ consists of a set V of vertices and a set E of edges

Figure: Social networks

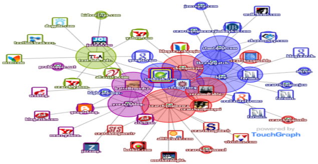


Figure: Web graphs

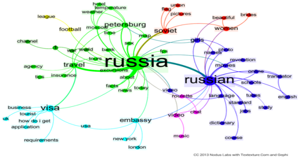


Figure: Graph of texts

Figure: Internet of things

Real Graphs are usually Globally Sparse but Locally Dense

- The entire graph is sparse, but there are groups of vertices with high concentration of edges within them.

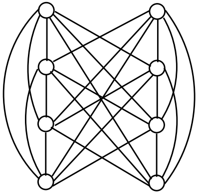
Graphs	n	m	$d_{avg}(G)$	$d_{max}(G)$	$\omega(G)$
as-Skitter	$1,694,616$	$11,094,209$	13.09	$35,455$	67
soc-LiveJournal1	$4,843,953$	$42,845,684$	17.69	$20,333$	321
uk-2005	$39,252,879$	$781,439,892$	39.82	$1,776,858$	589
it-2004	$41,290,577$	$1,027,474,895$	49.77	$1,326,744$	$3,222$

Table: Statistics of some real graphs ($\omega(G)$ is the clique number of G)

- Finding dense subgraphs is a fundamental problem with many applications.
  - community detection in social networks
  - anomaly detection in financial networks
  - protein complexes detection in biological networks
  - ...

k-Defective Clique

- The clique model, requiring all vertices to be connected to each other, represents the most dense subgraph model.
  - Clique-related problems have been extensively studied.
  - E.g., enumerate all maximal cliques, find a maximum clique.
- However, the clique model is often too restrictive for applications
  - Various clique relaxations have been formulated in the literature, such as quasi-clique, k-plex, k-club, and k-defective clique.
- k-defective clique allows the subgraph to miss up-to k edges (in total)
  - For the example graph below, the maximum clique size is 4, while the maximum k-defective clique size for any $k \leq 4$ is $4 + k$.



State of the Art of Maximum k-Defective Clique Computation

- It is NP-hard to compute the maximum (vertex) k-defective clique
- The state-of-the-art time complexity is achieved by the MADEC$^+$ algorithm proposed in[1], which runs in $\mathcal{O}^*(\sigma_k^n)$ time.
  - $\sigma_k < 2$ is the largest real root of the equation $x^{2k+3} - 2x^{2k+2} + 1 = 0$.
- KDBB proposed in[2] is practically faster than MADEC$^+$
  - KDBB is still inefficient in practice.
  - The time complexity of KDBB is the trivial $\mathcal{O}^*(2^n)$.

[1] Xiaoyu Chen et al. "Computing maximum k-defective cliques in massive graphs". In: *Comput. Oper. Res.* 127 (2021), p. 105131.

[2] Jian Gao et al. "An Exact Algorithm with New Upper Bounds for the Maximum k-Defective Clique Problem in Massive Sparse Graphs". In: *Proc. of AAAI'22*. 2022, pp. 10174–10183.

Our Contribution: Improve the Time Complexity

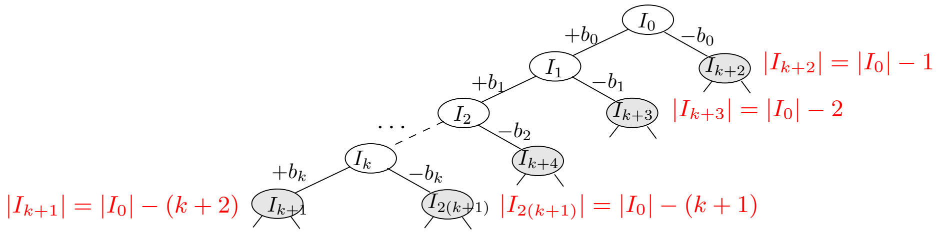
Algorithm 1: kDC(G, k)

$C^* \leftarrow \emptyset$;
Branch&Bound$(G, \emptyset)$;
return C^*;
Procedure Branch&Bound(g, S)
$(g', S') \leftarrow$ apply reduction rules **RR1** and **RR2** to (g, S);
if g' *is a k-defective clique* **then** update C^* by $V(g')$ and **return**;
$b \leftarrow$ a vertex of $V(g') \setminus S'$ that has at least one non-neighbor in S'; `/* If no such vertex,` b `is an arbitrary vertex of` $V(g') \setminus S'$ `*/`;
Branch&Bound$(g', S' \cup b)$; `/* Left branch includes` b `*/`;
Branch&Bound$(g' \setminus b, S')$; `/* Right branch excludes` b `*/`;

RR1. Given an instance (g, S), for a vertex $u \in V(g) \setminus S$ satisfying $|\overline{E}(S \cup u)| > k$, we remove u from g.

RR2. Given an instance (g, S), for a vertex $u \in V(g) \setminus S$ satisfying $|\overline{E}(S \cup u)| \leq k$ and $d_g(u) \geq |V(g)| - 2$, we greedily add u to S.

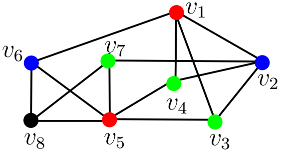
Our Contribution: Improve the Time Complexity



- $I = (g, S)$ and $|I| = |V(g) \setminus S|$.
- After exhaustively applying **RR1** and **RR2**, the resulting instance (g, S) satisfies the condition that all vertices of $V(g) \setminus S$ have at least two non-neighbors in g.
- Thus, there exists a sequence of vertices $\{b_0, \ldots, b_{k-1}, b_k\}$ such that after adding them to S, we can remove at least one vertex by **RR1**.
- The time complexity is $\mathcal{O}^*(\gamma_k^n)$ where γ_k is the largest real root of $x^{|I|} = x^{|I|-1} + \cdots + x^{|I|-(k+1)} + x^{|I|-(k+2)}$, equivalent to $x^{k+3} - 2x^{k+2} + 1 = 0$.

Our Contribution: Improve the Practical Performance

- A coloring of a graph is assigning each vertex a color such that for every edge in the graph, its two end-points have different colors.



- Given an instance (g, S) and a coloring of $V(g) \setminus S$ with c colors $\{1, \ldots, c\}$, let $\pi_1, \pi_2, \ldots, \pi_c$ be the partitioning of $V(g) \setminus S$ based on their colors.
  - Each π_i consists of all vertices with color i and thus is an independent set.
- The existing graph coloring-based upper bound is

$$|S| + \sum_{i=1}^{c} \min\left(\left\lfloor \frac{1+\sqrt{8k+1}}{2} \right\rfloor, |\pi_i|\right)$$

  - An independent set with $> \lfloor \frac{1+\sqrt{8k+1}}{2} \rfloor$ vertices will induce $> k$ missing edges

Our Contribution: Improve the Practical Performance

- Drawbacks of the existing upper bound $|S| + \sum_{i=1}^{c} \min\left(\left\lfloor \frac{1+\sqrt{8k+1}}{2} \right\rfloor, |\pi_i|\right)$
  - It considers $\pi_1, \ldots, \pi_c$ independently, includes much more vertices than necessary.
    - Suppose $|\pi_i| \geq \lfloor \frac{1+\sqrt{8k+1}}{2} \rfloor$, $\forall 1 \leq i \leq c$, then the upper bound is $|S| + c \cdot \lfloor \frac{1+\sqrt{8k+1}}{2} \rfloor$.
    - But obviously $|S| + c + k$ is a much smaller upper bound (*e.g.*, when c is large)
  - It does not consider the non-edges in S, and the non-edges between S and $V(g) \setminus S$.
- Our upper bound
  - For each π_i, sort its vertices into non-decreasing order regarding $|\overline{N}_S(\cdot)|$, and define the weight of the j-th vertex in the sorted order, denoted v_{i_j}, to be $\mathsf{w}(v_{i_j}) = |\overline{N}_S(v_{i_j})| + j - 1$, where the index j starts from 1.
  - Let $v_1, v_2, \ldots,$ be an ordering of $V(g) \setminus S$ in non-decreasing order regarding their weights $\mathsf{w}(\cdot)$.
  - The maximum k-defective clique in the instance (g, S) is of size at most $|S|$ plus the largest i such that $|\overline{E}(S)| + \sum_{j=1}^{i} \mathsf{w}(v_j) \leq k$.
- We also propose two reductioin rules for practical performance. See our paper.

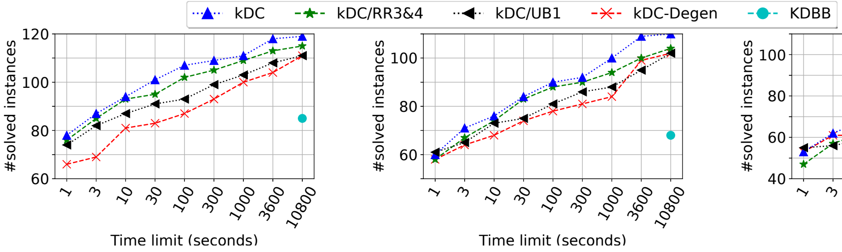
Performance Study

	Real-world graphs			Facebook graphs			DIMACS10&SNAP		
	kDC	KDBB	MADEC_p^+	kDC	KDBB	MADEC_p^+	kDC	KDBB	MADEC_p^+
$k = 1$	**133**	117	115	**114**	110	110	**37**	36	36
$k = 3$	**130**	107	94	**114**	110	104	**37**	35	31
$k = 5$	**127**	104	81	**114**	108	78	**37**	34	28
$k = 10$	**119**	85	36	**111**	109	9	**36**	30	15
$k = 15$	**110**	68	26	101	**103**	0	**29**	25	10
$k = 20$	**104**	56	20	**88**	80	0	**27**	22	6

Table: Number of solved instances by the algorithms kDC, KDBB and MADEC_p^+ with a time limit of 3 hours (best performers are highlighted in bold)

- The **real-world graphs** collection contains **139** real-world graphs from the Network Data Repository with up to 5.87×10^7 vertices and 1.06×10^8 undirected edges.
- The **Facebook graphs** collection contains **114** Facebook social networks from the Network Data Repository with up to 5.92×10^7 vertices and 9.25×10^7 undirected edges.
- The **DIMACS10&SNAP graphs** collection contains **37** graphs with up to 1.04×10^6 vertices and 6.89×10^6 undirected edges.

Performance Study



(a) $k = 10$ (b) $k = 15$ (c) $k = 20$

Figure: Number of solved instances for real-world graphs (vary time limit)

- kDC/UB1 is kDC without our new upper bound.
- kDC/RR3&4 is kDC without our new practical reduction rules.
- kDC-Degen: kDC with the initial solution computed by Degen.

Conclusion

- We improved the time complexity of maximum k-defective clique computation from $\mathcal{O}^*(\gamma_{2k}^n)$ to $\mathcal{O}^*(\gamma_k^n)$.
- We also significantly improved the practical performance.
- The source code is available at `https://lijunchang.github.io/Maximum-kDC/`

- We recently futher improved the time complexity to $\mathcal{O}^*(\gamma_{k-1}^n)$ in[3]

[3] Lijun Chang. "Maximum Defective Clique Computation: Improved Time Complexities and Practical Performance". In: *CoRR* abs/2403.07561 (2024). arXiv: 2403.07561.