Maximal k-Edge-Connected Subgraphs in Almost-Linear Time for Small k

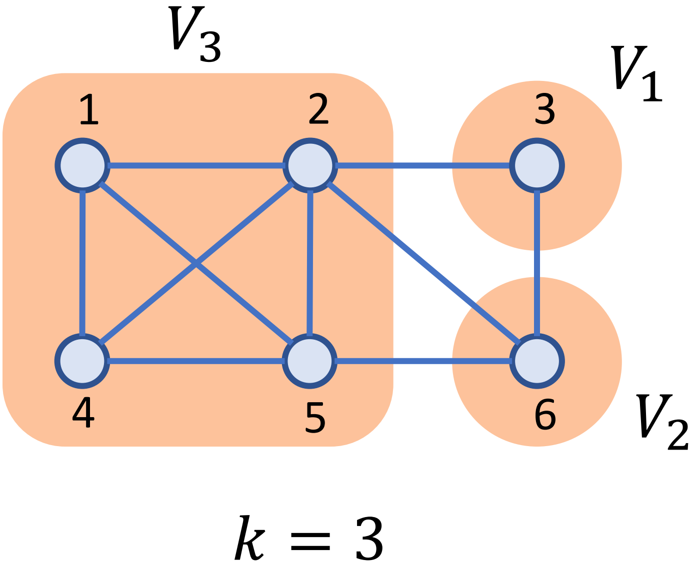
Thatchaphol Saranurak
University of Michigan

Wuwei Yuan
IIIS, Tsinghua University

ESA'23

Maximal k-Edge-Connected Subgraphs Problem

- Graph G is k-(edge-)connected if one needs to delete at least k edges to disconnect G
- Input: undirected unweighted graph $G = (V, E)$ with $n = |V|$ and $m = |E|$ and number k
- Output: unique vertex partition $\{V_1, \ldots, V_z\}$ such that,
  - $G[V_i]$ is k-connected, and
  - there is no $V_i' \supset V_i$ where $G[V_i']$ is k-connected
- Dynamic k-connected subgraphs problem: maintain k-connected subgraphs under updates

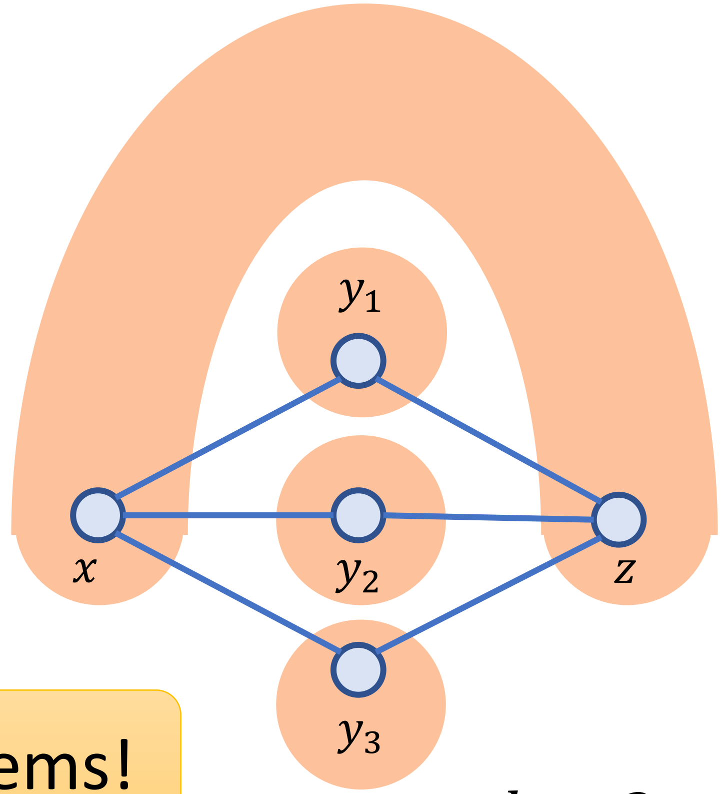


k-Connected Components

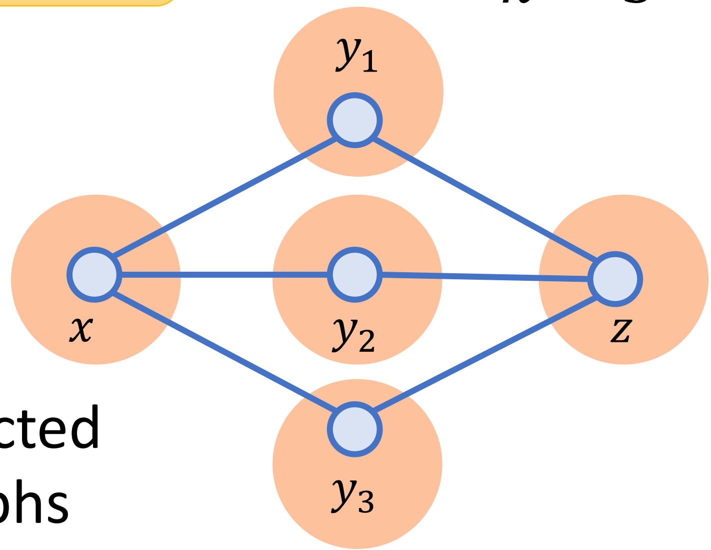
- Two vertices s and t are k-connected in G if one needs to delete at least k edges to disconnect s and t in G
- Set of vertices S is k-connected if every pair of vertices in S is k-connected
- k-connected component: maximal k-connected subset

[Abboud+'22]: find all k-connected components in $O\left(m^{1+o(1)}\right)$ time

Different problems!



$k = 3$



k-connected subgraphs

Applications

(Dynamic) k-connected subgraphs

Edge strength

k-certificate

Cut sparsifier
[Benczur+'96]

Decremental k-connectivity in non-sparse graphs
[Aamand+'23]

Previous Works

All above algorithms
require $\Omega(n^{3/2})$ time
when $m = O(n)$ and
$k = 3$

Reference	Time	Constraints
Folklore	$\tilde{O}(mn)$	Randomized
Chechik+ [SODA'17]	$\tilde{O}\left(m\sqrt{n}k^{O(k)}\right)$	
Forster+ [SODA'20]	$\tilde{O}\left(mk + n^{3/2}k^3\right)$	Randomized
Georgiadis+ ['22]	$\tilde{O}\left(m + n^{3/2}k^8\right)$	

$\tilde{O}(\cdot)$ hides polylog factors

Previous Works

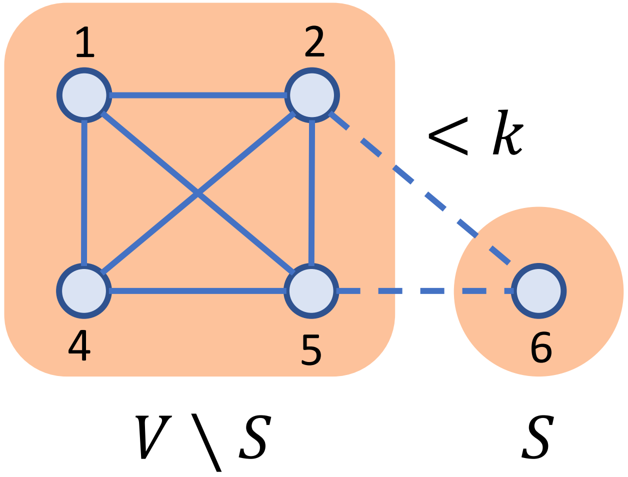
All above algorithms require $\Omega(n^{3/2})$ time when $m = O(n)$ and $k = 3$

Reference	Time	Constraints
Folklore	$\tilde{O}(mn)$	Randomized
Chechik+ [SODA'17]	$\tilde{O}\left(m\sqrt{n}k^{O(k)}\right)$	
Forster+ [SODA'20]	$\tilde{O}\left(mk + n^{3/2}k^3\right)$	Randomized
Georgiadis+ ['22]	$\tilde{O}\left(m + n^{3/2}k^8\right)$	
This work	$O\left(m + n^{1+o(1)}\right)$	$k = \log^{o(1)} n$

$\tilde{O}(\cdot)$ hides polylog factors

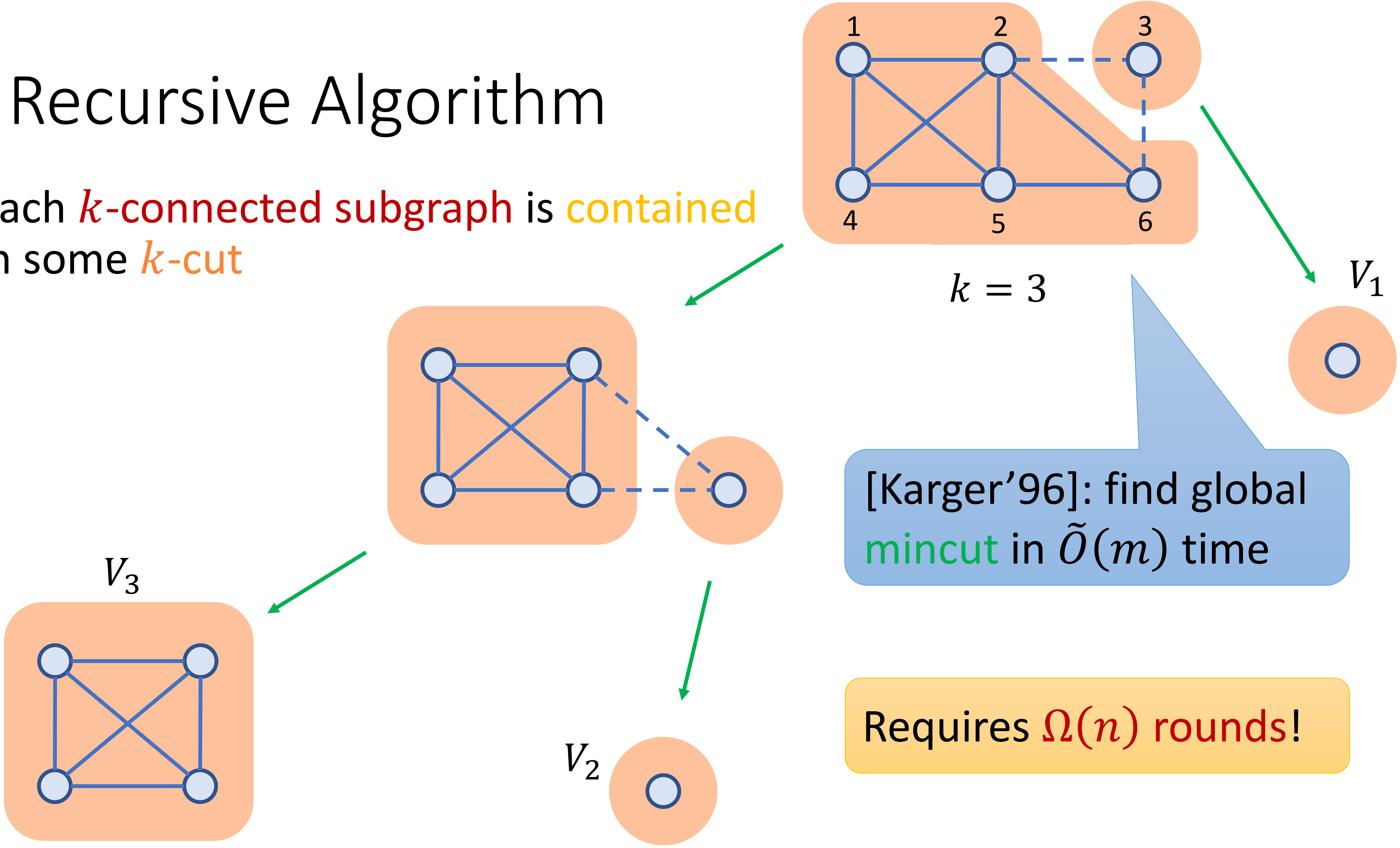
Definitions

- $E(S, T)$: set of edges between S and T
- Vertex set S is a k-cut if $|E(S, V \setminus S)| < k$



Recursive Algorithm

- Each k-connected subgraph is contained in some k-cut



[Karger'96]: find global mincut in $\tilde{O}(m)$ time

Requires $\Omega(n)$ rounds!

Our Approach

- Key idea: maintain list of vertices L, which contains at least one vertex from each k-cut
- Initially, $L \leftarrow V$
- While $|L| > 1$
  - Choose arbitrary $u, v \in L$
  - Check if u and v are k-connected

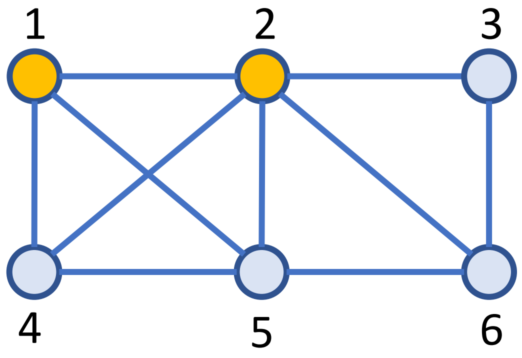
[JS'21]: Fully dynamic pairwise k-connectivity for $k = \log^{o(1)} n$ in $O\left(n^{o(1)}\right)$ time

Our Approach (Cont.)

L contains at least one vertex from each k-cut

- If u and v are k-connected
  - Remove v from L

u and v are in the same k-connected component, so they are in the same k-cut



$k = 3$

$L = \{1,2,3,4,5,6\}$

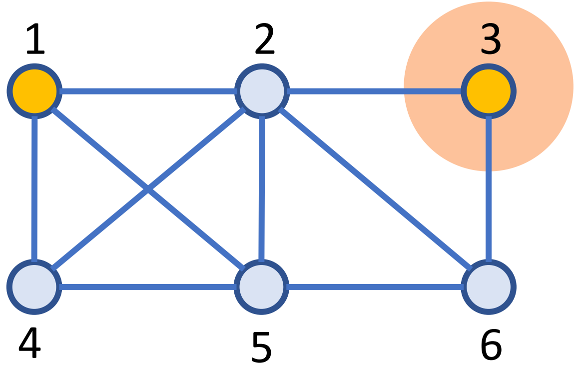
Our Approach (Cont.)

L contains at least one vertex from each k-cut

- Otherwise, u and v are not k-connected
  - We can find two k-connected components
  - Remove smaller one U and recurse on U
  - Add neighbours of U to L

$|U| < |V(G)|/2$

[CHILP'17]: Each k-cut in $G \setminus U$ either
(1) is a k-cut in G, or
(2) contains some neighbour of U

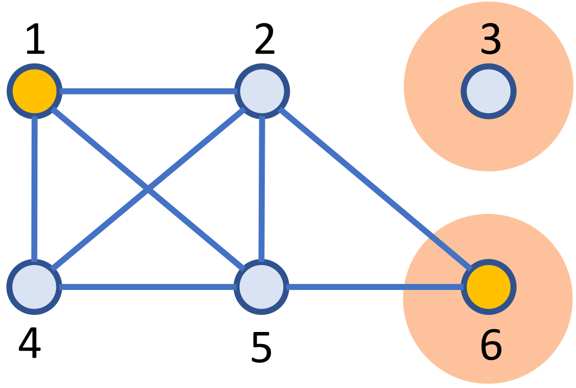


$k = 3$

$L = \{1,3,4,5,6\}$

Our Approach (Cont.)

L contains at least one vertex from each k-cut

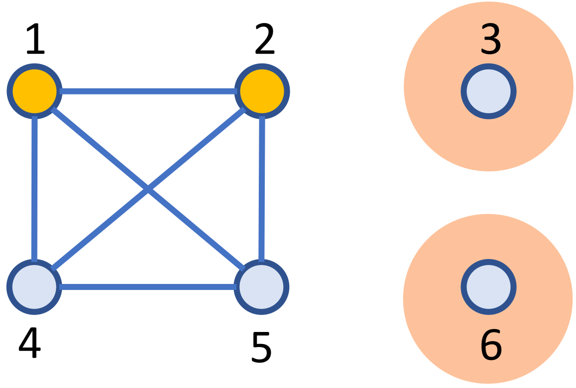


$k = 3$

$L = \{1,2,4,5,6\}$

Our Approach (Cont.)

L contains at least one vertex from each k-cut

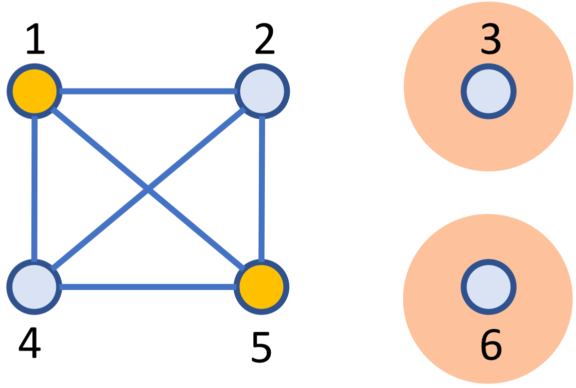


$k = 3$

$L = \{1,2,4,5\}$

Our Approach (Cont.)

L contains at least one vertex from each k-cut

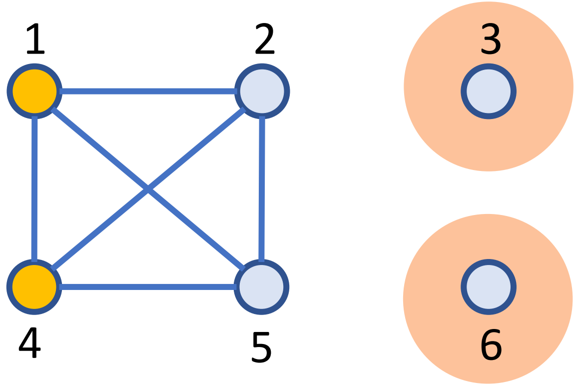


$k = 3$

$L = \{1,4,5\}$

Our Approach (Cont.)

L contains at least one vertex from each k-cut



$k = 3$

$L = \{1,4\}$

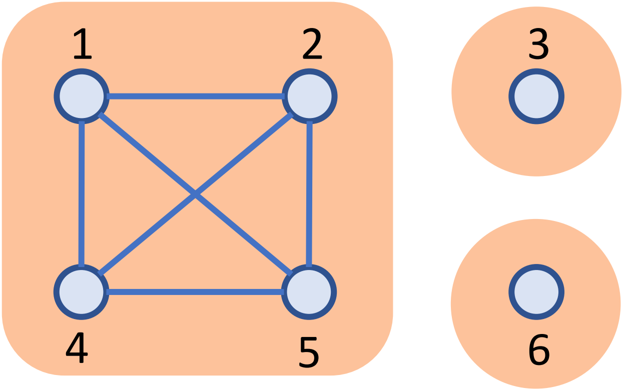
Our Approach (Cont.)

L contains at least one vertex from each k-cut

- Finally: the remaining graph is k-connected
- #vertices ever added into L in each recursion step: $O(m)$
- #recursion levels: $O(\log n)$
- Total running time: $O\left(m^{1+o(1)}\right)$

$|U| < |V(G)|/2$

[GIKP'22]: Can be improved to $O\left(m + n^{1+o(1)}\right)$ using sparsification techniques



$k = 3$

$L = \{1\}$

Conclusion

- Our results (for $k = \log^{o(1)} n$)
  - Maximal k-edge-connected subgraphs in $O\left(m + n^{1+o(1)}\right)$ time
  - Decremental maximal k-connected subgraphs in $O\left(m^{1+o(1)}\right)$ time
- Open problems
  - Remove constraint on k
  - Improve $n^{o(1)}$ to $\mathrm{polylog}(n)$
  - Weighted / directed graphs
  - Incremental / fully dynamic updates
  - Maximal k-vertex-connected subgraphs

Thank you!