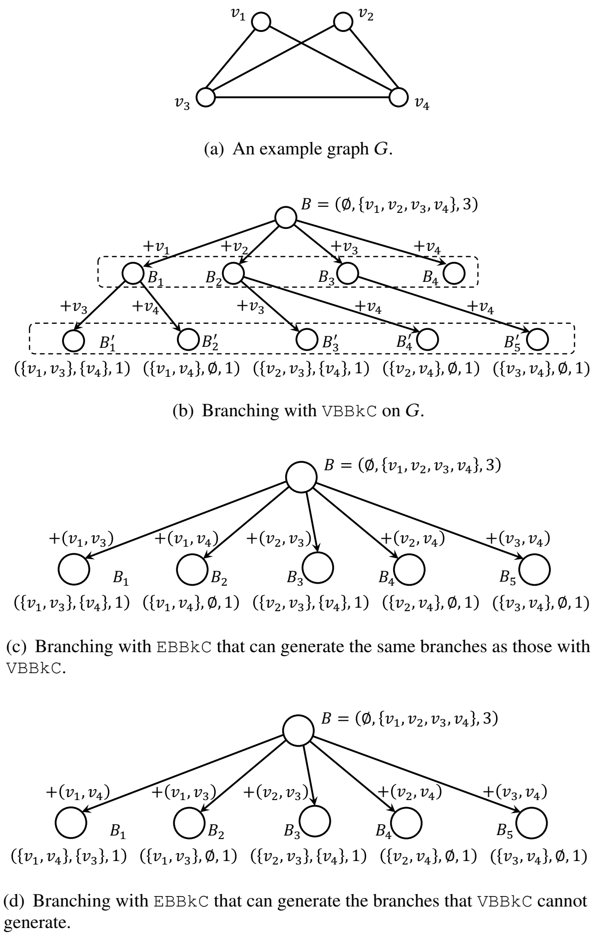


Figure 13: Illustration of the advantages of EBBkC over BBkC.

A PROOF OF THEOREM 4.2

The running time of EBBkC−T is dominated by the recursive listing procedure (lines 6-10). Given a branch $B = (S, g, l)$, we denote by $T(g, l)$ the upper bound of time cost of listing l-cliques under such branch. When $k \geq 3$, with different values of l, we have the following recurrences.

$$T(g,l) \leq \begin{cases} O(k \cdot |V(g)|) & l = 1 \\ O(k \cdot |E(g)|) & l = 2 \\ \sum_{e_i \in E(g)} \Big(T(g_i, l-2) + T'(g_i) \Big) & 3 \leq l \leq k-2 \end{cases} \tag{10}$$

where $T'(g_i)$ is the time for constructing g_i given $B = (S, g, l)$ (line 9 of Algorithm 3). We first show the a lemma which builds the relationship between g_i and g, then we present the complexity of $T'(g_i)$.

Lemma A.1. *Given a branch $B = (S, g, l)$ and the sub-branches $B_i = (S_i, g_i, l_i)$ produced at B. We have (1) when $l < k$,*

$$\sum_{e_i \in E(g)} |E(g_i)| < \frac{\tau^2}{4} \cdot |E(g)| \tag{11}$$

and (2) when $l = k$ (we note that the branch B corresponds to the universal branch $B = (\emptyset, G, k)$),

$$\sum_{e_i \in E(G)} |E(g_i)| < \frac{\tau^2}{2} \cdot |E| \tag{12}$$

Proof. **Case $l < k$.** $E(g_i)$ can be obtained by checking for each edge in $\{e_{i+1}, \cdots, e_{|E(g)|}\}$ whether it is in $E[e_i]$. Clearly, we have $|E(g_i)| \leq |E(g)| - i$. Then

$$\sum_{e_i \in E(g)} |E(g_i)| \leq \sum_{i=1}^{|E(g)|} (|E(g)| - i) = \frac{|E(g)|(|E(g) - 1|)}{2} \tag{13}$$

According to Lemma 4.1, each branch contains at most τ vertices, which indicates that $|E(g)|$ is at most $\tau(\tau-1)/2$. Therefore,

$$\frac{|E(g)|(|E(g) - 1|)}{2} \leq \frac{\tau(\tau-1)}{4} \cdot (|E(g) - 1|) < \frac{\tau^2}{4} \cdot |E(g)| \tag{14}$$

Case $l = k$. According Lemma 4.1, $V[e] \leq \tau$. Then $E[e]$ contains at most $\tau(\tau-1)/2$ edges. Thus,

$$\sum_{e_i \in E} |E(g_i)| \leq \sum_{e_i \in E} \frac{\tau(\tau-1)}{2} < \frac{\tau^2}{2} \cdot |E| \tag{15}$$

which completes the proof. □

Consider $T'(g_i)$. When $l = 3$, for each edge $e_i \in E(g)$, we just need to compute $V(g_i)$ by checking for each vertex in $V(g)$ whether it is in $V[e_i]$, which costs at most $O(\tau)$ time since $|V(g)| \leq \tau$. When $l > 3$, for each edge $e_i \in E(g)$, we need to compute both $V(g_i)$ and $E(g_i)$. Therefore,

$$\sum_{e_i \in E(g)} T'(g_i) = \begin{cases} O(\tau \cdot |E(g)|) & l = 3 \\ O(\tau^2 \cdot |E(g)|) & l > 3 \end{cases} \tag{16}$$

Given the above analyses, we prove the Theorem 4.2 as follows.

Proof. We prove by induction on l to show that

$$T(g,l) \leq \lambda \cdot (k + 2l) \cdot |E(g)| \cdot \left(\frac{\tau}{2}\right)^{l-2} \tag{17}$$

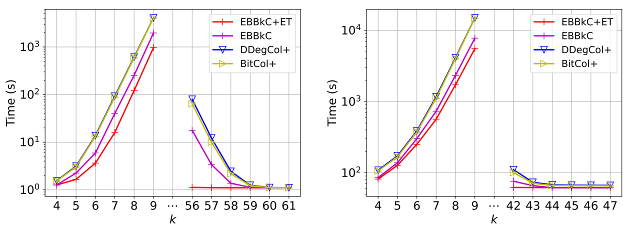
where λ is positive constant. Since the integer l decreases by 2 when branching, then when k is odd (resp. even), l would always be odd (resp. even) in all branches. Thus, we need to consider both cases.

Base Case ($l = 2$ and $l = 3$). When $l = 2$, it is easy to verify that there exists λ such that $T(g, 2) \leq \lambda \cdot k \cdot |E(g)|$ satisfies Eq. (17). When $l = 3$, we put Eq. (16) into Eq. (10), and it is easy to verify that $T(g, 3) \leq \lambda \cdot k \cdot |E(g)| \cdot \tau$, which also satisfies Eq. (17).

Induction Step. We first consider the case when $3 < l < k$. Assume the induction hypothesis, i.e., Eq. (17), is true for $l = p$ $(p + 2 < k)$. When $l = p + 2$,

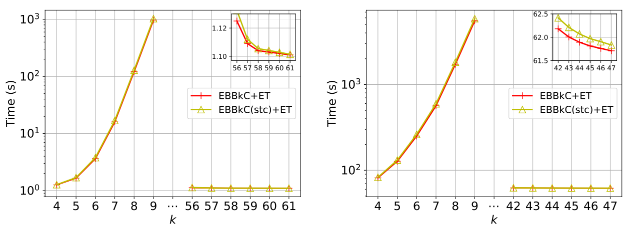
$$\begin{aligned} T(g, p+2) &\leq \sum_{e_i \in E(g)} \Big(T(g_i, p) + T'(g_i) \Big) \\ &\leq \lambda \cdot \tau^2 \cdot |E(g)| + \sum_{e_i \in E(g)} \lambda \cdot (k + 2p) \cdot |E(g_i)| \cdot \left(\frac{\tau}{2}\right)^{p-2} \\ &< \lambda \cdot \tau^2 \cdot |E(g)| + \lambda \cdot (k + 2p) \cdot |E(g)| \cdot \left(\frac{\tau}{2}\right)^{p} \\ &\leq \lambda \cdot (k + 2p + 4) \cdot |E(g)| \cdot \left(\frac{\tau}{2}\right)^{p} \end{aligned} \tag{18}$$

The first inequality derives from recurrence. The second inequality derives from Eq. (16) and the induction hypothesis. The third inequality derives from Lemma A.1 (case $l < k$). The fourth inequality derives from the fact that $\tau^2 \leq 4 \cdot (\tau/2)^p$ when $p \geq 2$.



(a) ST (b) OR

Figure 14: Ablation studies.



(a) ST (b) OR

Figure 15: Effects of the color-based pruning rules (comparison between the algorithms with and without the Rule (2)).

Then consider the cases when $l = k$.

$$
\begin{aligned}
T(G,k) &\le \sum_{e_i \in E(G)} \Big(T(g_i, k-2) + T'(g_i)\Big) \\
&\le \lambda \cdot \tau^2 \cdot E(G) + \sum_{e_i \in E(G)} \lambda \cdot (3k-4) \cdot |E(g_i)| \cdot \left(\frac{\tau}{2}\right)^{k-4} \\
&< \lambda \cdot \tau^2 \cdot E(G) + 2\lambda \cdot (3k-4) \cdot |E(G)| \cdot \left(\frac{\tau}{2}\right)^{k-2} \\
&< 6\lambda \cdot k \cdot |E(G)| \cdot \left(\frac{\tau}{2}\right)^{k-2}
\end{aligned}
\tag{19}
$$

The first inequality derives from recurrence. The second inequality derives from Eq. (16) and the induction hypothesis. The third inequality derives from Lemma A.1 (case $l = k$). The fourth inequality derives from the fact that $\tau^2 \le 4 \cdot (\tau/2)^{k-2}$ when $k \ge 4$.

Conclusion. We conclude that given a graph $G = (V, E)$ and an integer $k \ge 3$, the time complexity of EBBkC can be upper bounded by $O(k \cdot |E(G)| \cdot (\tau/2)^{k-2})$. □

B A COUNTER-EXAMPLE FOR THE STATEMENT IN SECTION 4.2

Example. To illustrate the statement in Section 4.2, we consider the example in Figure 13. The graph G involves 4 vertices and 5 edges. Assume that we aim to list 3-cliques in G, i.e., $k = 3$. Consider the degeneracy ordering of the vertices $\pi_\delta = \langle v_1, v_2, v_3, v_4 \rangle$. Then VBBkC would generate 5 branches following the vertex ordering π_δ, and the illustration is shown in Figure 13(b). Correspondingly, we can construct an edge ordering $\pi_e = \langle (v_1,v_3), (v_1,v_4), (v_2,v_3), (v_2,v_4), (v_3,v_4) \rangle$ by following which we can produce the same branches as that of VBBkC. Consider the truss-based edge ordering $\pi_\tau = \langle (v_1,v_4), (v_1,v_3), (v_2,v_3), (v_2,v_4), (v_3,v_4) \rangle$. The produced branches are shown in Figure 13(d). We note that these produced branches cannot be generated with any vertex orderings. To see this, consider the branch B_1 and B_3 in Figure 13(d). If there exists a vertex ordering π_v that can generate B_1, then vertex v_3 must be ordered behind v_4 in π_v. Similarly, for branch B_3, vertex v_4 must be ordered behind v_3 in π_v. This derives a contradiction.

C TIME COMPLEXITY OF ALGORITHM 7

THEOREM C.1. *Given a branch $B = (S, g, l)$ with g being a t-plex ($t \ge 3$), kCtPlex lists all k-cliques within B in at most $O(|E(g)| + t \cdot \binom{|V(g)|}{l} + k \cdot c(g,l))$ time, where $c(g,l)$ is the number of l-cliques in g.*

PROOF. Let $T(g,l)$ be the total time complexity. Lines 1-2 of Algorithm 7 can be done in $O(E(g))$ time for partitioning $V(g)$ and constructing the inverse graph g_{inv}. Let $T'(|C|, l')$ be the time complexity to produce all sub-branches excluding the output part (lines 5-10). We have $T'(0,\cdot) = 0$ and $T'(\cdot, 0) = 0$.

$$T \le O(E(g) + k \cdot c(g,l)) + T'(|V(g) \setminus I|, l) \tag{20}$$

According to Eq. (9), for each $1 \le i \le |C|$, the size of the produced C_i is at most $|C| - i$. Thus, we have the following recurrence.

$$
\begin{aligned}
T'(|C|, l') &= \sum_{i=1}^{|C|} \Big(T'(|C_i|, l'-1) + O(t)\Big) \\
&= T'(|C|-1, l'-1) + O(t) + \sum_{j=0}^{|C|-2} \Big(T'(j, l'-1) + O(t)\Big) \\
&= T'(|C|-1, l'-1) + T'(|C|-1, l') + O(t)
\end{aligned}
\tag{21}
$$

We note that $O(t)$ is the time to filter out the vertices that are connected with v_i in g_{inv}, i.e., $N(v_i, g_{inv})$. Let $T^*(|C|, l') = T'(|C|, l') + O(t)$ and apply it to the above equation,

$$T^*(|C|, l') = T^*(|C|-1, l'-1) + T^*(|C|-1, l') \tag{22}$$

with the initial conditions $T^*(\cdot, 0) = O(t)$ and $T^*(0, \cdot) = O(t)$. Observe that Eq. (22) is similar to an identity equation $\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$. Then it is easy to verify that $T^*(|C|, l') = O(\binom{|C|}{l'} \cdot t)$. Therefore, $T'(|C|, l')$ can also be bounded by $O(\binom{|C|}{l'} \cdot t)$. Finally, since $|V(g) \setminus I| \le |V(g)|$, the time complexity of Algorithm 7 is at most $O(|E(g)| + t \cdot \binom{|V(g)|}{l} + k \cdot c(g,l))$. □

Remark. Recall that in Eq. (9), we need to filter out the vertices that are connected with v_i in g_{inv} to create the sub-branch. There are two possible ways to implement this procedure. One way is to use a shared array to store $V(g) \setminus I$, denoted by C. Every time we execute line 11 and 12, the pointer in C moves forward and marks those vertices in $N(v_i, g_{inv})$ as invalid, respectively. The benefit is that the procedure can be done in $O(t)$ in each round, as we show in Eq. (21), but will be hard to be parallelized. Another way is to use additional $O(C_i)$ to ensure that each sub-branch has its own copy of C_i, which makes the procedure easy to be parallelized. In our experiments, we implement the algorithm in the former way.

D ADDITIONAL EXPERIMENTAL RESULTS

The experimental results on ablation studies and the effects of the color-based pruning rules (comparison between the algorithms with and without the Rule (2)) on datasets ST and OR are shown in Figure 14 and Figure 15, respectively.