Fast Maintenance of 2-hop Labels for Shortest Distance Queries on Fully Dynamic Graphs — Supplemental Materials

The road map of this supplement is as follows.
- In Section S1, we provide detailed discussions on time complexities.
- In Section S2, we prove that CLN can eliminate all redundant indexes.

S1. THE DISCUSSIONS ON TIME COMPLEXITIES

The time complexity of FastDeM**:** FastDeM has a cost of

$$O\Big(\delta^2 + \Upsilon \cdot (\log \Upsilon + d_a \cdot \delta)\Big).$$

The reason is that populating CL takes $O(\delta^2)$ time, and $DIFFUSE$ takes $O(\Upsilon \cdot (\log \Upsilon + d_a \cdot \delta))$ time, assumed that labels are stored in hashes. Particularly, $DIFFUSE$ pops $O(\Upsilon)$ elements out of the priority queue. Each pop operation takes $O(\log \Upsilon)$ times. After each pop operation, it searches $O(d_a)$ neighbors, and a distance query that costs $O(\delta)$ may be conducted in each of these searches.

The time complexity of FastInM**:** FastInM has a cost of

$$O\Big(\Upsilon \cdot \Big(\log \Upsilon + d_a \cdot \delta + \kappa \cdot (d_a + \delta)\Big)\Big),$$

where κ is the average number of PPR elements of each vertex-hub pair. The details are as follows. First, FastInM pushes labels into AL_1 in $O(\delta)$ time. Then, it performs $SPREAD_1$ in $O(\Upsilon \cdot d_a)$ time, since there are $O(\Upsilon)$ labels deactivated in Line 13, and each deactivation is followed by $O(d_a)$ neighbor searches. Subsequently, it performs $SPREAD_2$ in $O(\Upsilon \cdot \kappa \cdot (d_a + \delta))$ time, since for each of $O(\Upsilon)$ tuples in AL_2 in Line 17, it checks $O(\kappa)$ PPR elements in Line 18, while checking each PPR element takes $O(d_a + \delta)$ time, *e.g.,* it compares $O(d_a)$ values in Line 20 and performs the distance query in $O(\delta)$ time in Line 21 [1]. After that, it performs $SPREAD_3$ in $O(\Upsilon \cdot (\log \Upsilon + d_a \cdot \delta))$ time, just like $DIFFUSE$ in FastDeM.

The time complexity of CLN**:** We eliminate redundant indexes from time to time such that redundant indexes do not become orders of magnitude more than non-redundant ones. Then, we ignore the change of δ in the following analyses. Initially, we explain that a re-generation process, *i.e.,* Algorithm 1, has a time complexity of

$$O\Big(|E| \cdot \delta^2 + |E| \cdot \delta \cdot \log |V|\Big)$$

as follows. For each generated label associated with a vertex $v \in V$, Algorithm 1 inserts $O(deg(v))$ elements into Q. There are $O(|E| \cdot \delta)$ elements in Q in total. For each element in Q, Algorithm 1 takes $O(\log |V|)$ time to pop it out, and also takes $O(\delta)$ time to query a distance. In comparison, CLN has a time complexity of

$$O\Big(|V| \cdot \delta^2 + |E| \cdot \delta \cdot \log |V|\Big).$$

The details are as follows. First, cleaning L in Lines 2-6 takes $O(|V| \cdot \delta^2)$, since there are $O(|V| \cdot \delta)$ labels, and checking whether a label is redundant or not in Line 4 takes $O(\delta)$. Second, generating PPR in Lines 7-16 takes $O(|E| \cdot \delta \cdot \log |V| + |PPR| \cdot \delta)$, since this process is similar to the process of Algorithm 1 with $O(|PPR|)$ distance queries. Given that we generally have $|PPR| \ll |V| \cdot \delta$ in practice, *i.e.,* the number of PPR elements is generally much smaller than the number of 2-hop labels, $O(|PPR| \cdot \delta)$ is covered by $O(|V| \cdot \delta^2)$. As we often have $|E| \gg |V|$ and $\delta \gg \log |V|$, it can be considered that CLN has a smaller time complexity than Algorithm 1.

S2. THE EFFECTIVENESS OF CLN

We show that CLN can eliminate all redundant indexes as follows. First, we present the canonical constraint [2–4] below. For a given rank of vertices, there is only one set of 2-hop labels that satisfies the canonical constraint, *e.g.*, L in Figure 1 in the main contents. We refer to a set of 2-hop labels that satisfies the canonical constraint as a canonical set of 2-hop labels, which is minimal in that deleting any label from this set induces that it does not satisfy the 2-hop cover constraint. PLL is a widely-used algorithm for generating a canonical set of 2-hop labels [1, 5].

Definition 1 (Canonical Constraint). *Given a rank of vertices, a set L of 2-hop labels satisfies the canonical constraint if, a vertex v is a hub of $u \in V$, i.e., $v \in C(u)$, if and only if the rank of v is the highest among all vertices in all shortest paths between u and v.*

Suppose that the initial indexes are generated by Algorithm 1 in the main contents. To maintain 2-hop labels after edge weight changes, both the existing solution of InAsyn + RepairedDeAsyn and the proposed solution of FastInM + FastDeM generate a new label $L(u)[v]$ only when $r(u) < r(v)$. Let L_m be the maintained set of 2-hop labels by InAsyn + RepairedDeAsyn or FastInM + FastDeM after a number of edge weight changes. Further let L_r and PPR_r be the re-generated indexes by Algorithm 1 after these edge weight changes. L_r is a canonical set of 2-hop labels for the updated graph, and PPR_r is the record of the pruning information for generating L_r by PLL. We have the following lemma, which shows that L_m is a super set of the canonical set of 2-hop labels.

Lemma 1. $L_r \subseteq L_m$.

Proof. Consider an arbitrary label $(u', d'_{u's}) \in L_r(s)$. Since L_r is a canonical set of 2-hop labels for the updated graph, u' is the vertex with the highest rank in all shortest paths between s and u' in the updated graph, and $d'_{u's}$ is the distance between s and u' in the updated graph. The proofs of Theorems 1-2 show that the vertex with the highest rank in all shortest paths between s and another vertex in the updated graph is a hub of s after each edge weight decrease or increase maintenance. Thus, $(u', d'_{u's}) \in L_m(s)$, and this lemma holds. □

With the input of L_m, let L_c and PPR_c be the cleaned set of 2-hop labels and the cleaned PPR, respectively, by CLN. We further have the following theorem.

Theorem 1. $L_c = L_r, PPR_c = PPR_r$.

Proof. Consider an arbitrary label $(v, d_{uv}) \in L_r(u) \subseteq L_m(u)$. When CLN computes d'_{uv} in Line 4, if $d'_{uv} \le d_{uv}$, then there is a vertex $y \in C_{>r(v)}(u) \cap C(v)$ that is in a shortest path between u and v, and $r(y) > r(v)$. However, since L_c is canonical, this contradicts with the fact that the rank of v is the highest among all vertices in all shortest paths between u and v. Thus, $d'_{uv} > d_{uv}$, and CLN inserts (v, d_{uv}) into $L_c(u)$ in Line 6, *i.e.*, $(v, d_{uv}) \in L_c(u)$. On the other hand, consider an arbitrary label $(x, d_{ux}) \in L_m(u) \setminus L_r(u)$. Let $z \in V$ be the vertex with the highest rank among all vertices in all shortest paths between u and x. We have $r(z) > r(x)$, and $z \in C_{>r(x)}(u) \cap C(x)$. As a result, CLN computes $d'_{ux} = d_{ux} = d(u, z) + d(z, x)$, and does not insert (x, d_{ux}) into $L_c(u)$, *i.e.*, $(x, d_{ux}) \notin L_c(u)$. Thus, $L_c = L_r$. Moreover, since PPR_r is the record of the pruning information for generating L_r by PLL; and PPR_c is the record of the pruning information for generating L_c by PLL (notably, the process of generating PPR_c in CLN is essentially the same with the process of generating PPR in Algorithm 1), $PPR_c = PPR_r$. Hence, this theorem holds. □

Therefore, CLN is as effective as a re-generation process for eliminating all redundant indexes.

REFERENCES FOR THE SUPPLEMENT

1. Y. Li, L. H. U, M. L. Yiu, and N. M. Kou, "An experimental study on hub labeling based shortest path algorithms," Proc. VLDB Endow. **11**, 445–457 (2017).
2. I. Abraham, D. Delling, A. V. Goldberg, and R. F. Werneck, "Hierarchical hub labelings for shortest paths," in *European Symposium on Algorithms,* (Springer, 2012), pp. 24–35.
3. A. V. Goldberg, I. Razenshteyn, and R. Savchenko, "Separating hierarchical and general hub labelings," in *International Symposium on Mathematical Foundations of Computer Science,* (Springer, 2013), pp. 469–479.
4. M. Jiang, A. W.-C. Fu, R. C.-W. Wong, and Y. Xu, "Hop doubling label indexing for point-to-point distance querying on scale-free networks," Proc. VLDB Endow. **7** (2014).
5. T. Akiba, Y. Iwata, and Y. Yoshida, "Fast exact shortest-path distance queries on large networks by pruned landmark labeling," in *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data,* (2013), pp. 349–360.