A Lemmas and Proofs

A.1 Lemma 1 (Monotonicity of Vertex Embeddings)

LEMMA 1. ***(Monotonicity of Vertex Embeddings).*** *Given a 1-hop subgraph $g_1(v_i)$ and one of its 1-hop substructures $s_1(v_i)$ with $s_1(v_i) \subseteq g_1(v_i)$, their embeddings satisfy $o(s_1(v_i)) \preceq o(g_1(v_i))$.* □

PROOF. Both $g_1(v_i)$ and $s_1(v_i)$ share the same center vertex v_i and thus have identical vertex label embeddings. Since $\mathcal{N}_1(s_1(v_i)) \subseteq \mathcal{N}_1(g_1(v_i))$, their structure embeddings satisfy

$$o_s(s_1(v_i)) = \sum_{v_j \in \mathcal{N}_1(s_1(v_i))} o_l(v_j) \preceq \sum_{v_j \in \mathcal{N}_1(g_1(v_i))} o_l(v_j) = o_s(g_1(v_i)).$$

Combining this with the non-negative weighted sum defining $o(v_i)$ yields $o(s_1(v_i)) \preceq o(g_1(v_i))$, completing the proof. □

A.2 Lemma 2 (Relationship Between Anti-Dominance Loss and Discrete Objective)

LEMMA 2. ***(Relationship Between Anti-Dominance Loss and Discrete Objective).*** *Let L_{cost} and L be defined in Eqs. 5 and 6, respectively. As the temperature parameter $\tau \to 0^+$, the anti-dominance loss L converges to the discrete anti-dominance cost L_{cost}. Moreover, for any finite $\tau > 0$, L serves as a smooth upper bound of L_{cost}.* □

PROOF. By definition, the dominance relation $o(v_i) \preceq o(v_j)$ holds if and only if $\min_k \left(o(v_i)[k] - o(v_j)[k]\right) \le 0$. Accordingly, the discrete indicator function $\mathbf{1}\{x \le 0\}$ can be approximated by the sigmoid function $\sigma(x/\tau)$, which converges pointwise to the indicator as $\tau \to 0^+$.

Therefore, for any vertex pair (v_i, v_j), we have:

$$\lim_{\tau \to 0^+} \sigma\left(\frac{\min_k \left(o(v_i)[k] - o(v_j)[k]\right)}{\tau}\right) = \mathbf{1}\left\{\min_k \left(o(v_i)[k] - o(v_j)[k]\right) \le 0\right\}.$$

Substituting this limit into the definition of L yields:

$$\lim_{\tau \to 0^+} L = \frac{1}{|V(G)|^2} \sum_{v_i, v_j} \mathbf{1}\{o(v_i) \preceq o(v_j)\} = L_{\text{cost}}.$$

Moreover, for any finite $\tau > 0$, the sigmoid function satisfies $\sigma(x/\tau) \ge \mathbf{1}\{x \le 0\}$ for all x. Thus, each term in L upper-bounds the corresponding indicator term in L_{cost}, which implies $L \ge L_{\text{cost}}$. Equality holds asymptotically as $\tau \to 0^+$, completing the proof. □

A.3 Lemma 3 (Dominance Preservation under One-dimensional Key Mapping)

LEMMA 3. ***(Dominance Preservation under One-dimensional Key Mapping).*** *Let v_i and v_j be two data vertices with monotonic vertex embeddings $o(v_i)$ and $o(v_j)$. If $o(v_i) \preceq o(v_j)$ holds, then their corresponding keys satisfy $key(v_i) \le key(v_j)$.* □

PROOF. By definition of dominance, $o(v_i) \preceq o(v_j)$ implies $o(v_i)[k] \le o(v_j)[k]$ for all dimensions k. Since both $o_l(\cdot)$ and $o_s(\cdot)$ are non-negative vectors by construction, their L_2 norms are monotonic with respect to coordinate-wise dominance. Thus, $\|o_l(v_i)\|_2 \le \|o_l(v_j)\|_2$ and $\|o_s(v_i)\|_2 \le \|o_s(v_j)\|_2$. Given $\alpha, \beta \ge 0$, it follows that $key(v_i) = \alpha\|o_l(v_i)\|_2 + \beta\|o_s(v_i)\|_2 \le \alpha\|o_l(v_j)\|_2 + \beta\|o_s(v_j)\|_2 = key(v_j)$, which completes the proof. □

A.4 Lemma 4 (Non-overlapping Condition for Label Clusters)

LEMMA 4. ***(Non-overlapping Condition for Label Clusters).*** *Let $M = \max_{v_i \in V(G)} \|o_s(v_i)\|_2$ be the maximum L_2 norm of vertex structure embeddings (VSEs) over all data vertices, and let $\Delta_{\min} = \min_i(\|o_l(l_{i+1})\|_2 - \|o_l(l_i)\|_2)$ be the minimum difference between the L_2 norms of vertex label embeddings (VLEs) of two adjacent labels, where labels are ordered by increasing $\|o_l(l)\|_2$. If the following holds:*

$$\frac{\alpha}{\beta} > \frac{M}{\Delta_{\min}},$$

then the key ranges of any two distinct label clusters in the iLabel index are strictly non-overlapping. □

PROOF. For any data vertex $v_i \in V(G)$ with label $l = L(v_i)$, since $\|o_s(v_i)\|_2 \in [0, M]$, its key value satisfies:
$key(v_i) = \alpha\|o_l(l)\|_2 + \beta\|o_s(v_i)\|_2 \in \left[\alpha\|o_l(l)\|_2,\ \alpha\|o_l(l)\|_2 + \beta M\right]$. Thus, the key range of the label cluster corresponding to l is bounded by the above interval.

Consider two adjacent labels l_i and l_{i+1} in ascending order of $\|o_l(l)\|_2$. The maximum key value in cluster C_{l_i} is at most $\alpha\|o_l(l_i)\|_2 + \beta M$, while the minimum key value in cluster $C_{l_{i+1}}$ is at least $\alpha\|o_l(l_{i+1})\|_2$. If $\alpha(\|o_l(l_{i+1})\|_2 - \|o_l(l_i)\|_2) > \beta M$, or equivalently Eq. 11 holds, then the following condition can be satisfied:

$$\max_{v \in C_{l_i}} key(v) < \min_{v \in C_{l_{i+1}}} key(v),$$

which implies that the two cluster key ranges are strictly non-overlapping. Since this condition holds for every adjacent pair of labels, all label clusters in the index are strictly separated. □

A.5 Lemma 5 (Key Lower-Bound Pruning)

LEMMA 5. ***(Key Lower-Bound Pruning).*** *Given a query vertex $q_i \in q$ with $key(q_i)$, for any data vertex $v_i \in G$ with $key(v_i)$ or index entry $N_i \in \mathcal{I}$ with key range $[N_i.min, N_i.max)$, if $key(v_i) < key(q_i)$ or $N_i.max < key(q_i)$ holds, then v_i or the entire subtree rooted at N_i can be safely pruned.* □

PROOF. By the construction of the key mapping (Eq. 10), key ordering is consistent with the dominance ordering of vertex embeddings. If $key(v_i) < key(q_i)$, then $o(q_i) \npreceq o(v_i)$, implying that v_i cannot be dominated by $o(q_i)$ and thus cannot be a valid candidate.

Similarly, if $N_i.\max < key(q_i)$, then all data vertices in the subtree rooted at N_i have keys smaller than $key(q_i)$ and therefore cannot be dominated by $o(q_i)$. Pruning under these conditions preserves all valid candidates and introduces no false dismissals. □

A.6 Lemma 6 (Key Upper-Bound Pruning)

LEMMA 6. ***(Key Upper-Bound Pruning).*** *Given a query vertex $q_i \in q$ with label $L(q_i)$, for any data vertex $v_i \in G$ with key $key(v_i)$ or index entry $N_i \in \mathcal{I}$ with key range $[N_i.min, N_i.max)$, if $key(v_i) \ge key(L(q_i)+1)$ or $N_i.min \ge key(L(q_i)+1)$ holds, then v_i or the entire subtree rooted at N_i can be safely pruned.* □

PROOF. By the *iLabel* key construction, all vertices with label $L(q_i)$ are indexed within the interval $[key(L(q_i)),\ key(L(q_i)+1))$, and key intervals of different labels are disjoint. Hence, any data vertex with $key(v_i) \geq key(L(q_i)+1)$ belongs to a different label cluster and cannot match q_i.

Similarly, if $N_i.\min \geq key(L(q_i)+1)$, then all vertices in the sub-tree rooted at N_i fall outside the label-consistent key range of q_i. Pruning under these conditions preserves all valid candidates and introduces no false dismissals. □

A.7 Lemma 7 (Embedding Dominance Pruning)

Lemma 7. ***(Embedding Dominance Pruning).*** *Given a query vertex q_i and a data vertex v_i, if their monotonic vertex embeddings satisfy $o(q_i) \npreceq o(v_i)$ (i.e., $o(v_i)$ is not dominated by $o(q_i)$), then v_i cannot be a valid candidate for q_i and can be safely pruned.* □

Proof. A data vertex v_i matches q_i only if the 1-hop subgraph $g_1(q_i)$ is a substructure of $g_1(v_i)$. Under the monotonic vertex embedding design (Section 3.2), this subgraph containment relationship implies $o(q_i) \preceq o(v_i)$ in the embedding space. If $o(q_i) \npreceq o(v_i)$ holds, then the necessary dominance condition is violated, and v_i cannot satisfy the required subgraph relationship. Therefore, pruning v_i does not introduce false dismissals. □

A.8 Lemma 8 (Hop-based Synopsis Pruning)

Lemma 8. ***(Hop-based Synopsis Pruning).*** *Given a query vertex q_i and a data vertex v_i, let $q_i.MBR_{hop}^{(t)}$ and $v_i.MBR_{hop}^{(t)}$ denote their hop-based embedding synopses at hop distance t. If there exists $t \in \{2,\ldots,k\}$ such that $q_i.MBR_{hop}^{(t)} \nsubseteq v_i.MBR_{hop}^{(t)}$, then v_i cannot be a valid candidate for q_i and can be safely pruned.* □

Proof. The synopsis $v_i.MBR_{hop}^{(t)}$ bounds the embeddings of all vertices in the t-hop neighborhood of v_i. If v_i were a valid candidate for q_i, the t-hop neighborhood of q_i would have to be realizable within that of v_i, which under monotonic embeddings requires $q_i.MBR_{hop}^{(t)} \subseteq v_i.MBR_{hop}^{(t)}$. Violation of this condition for any t implies infeasibility. Thus, v_i can be safely pruned. □

A.9 Lemma 9 (Degree-based Synopsis Pruning)

Lemma 9. ***(Degree-based Synopsis Pruning).*** *Given a query vertex q_i and a data vertex v_i, let $q_i.vse$ denote the vertex structure embedding of q_i. If $q_i.vse \notin v_i.MBR_{deg}^{(deg(q_i))}$, then v_i cannot be a valid candidate for matching q_i and can be safely pruned.* □

Proof. If v_i is a valid candidate for q_i, then the 1-hop subgraph $g_1(q_i)$ must correspond to one of the 1-hop substructures of v_i with degree $deg(q_i)$. By construction, $v_i.MBR_{deg}^{(\delta)}$ bounds the structural embeddings of all such substructures with degree δ.

Hence, $q_i.vse$ must lie within $v_i.MBR_{deg}^{(deg(q_i))}$. If this condition is violated, no feasible 1-hop substructure of v_i can match $g_1(q_i)$. Pruning v_i under this condition introduces no false dismissals. □

B Complexity Analysis

B.1 Time Complexity of *iLabel* Index Construction

Given a data graph G, for each vertex $v_i \in V(G)$, computing its monotonic embedding $o(v_i) = \alpha o_l(v_i) + \beta o_s(v_i)$ requires aggregating the label embeddings of its 1-hop neighbors. This incurs $O(d \cdot deg(v_i))$ time per vertex, where d is the embedding dimension. Since $\sum_{v_i \in V(G)} deg(v_i) = 2|E(G)|$, the total cost of embedding generation is $O(d \cdot |E(G)|)$.

For hop-based synopses, we compute t-hop neighborhoods using BFS expansion. For each hop level, all vertices and edges are visited at most once. Thus, constructing hop-based MBRs for all vertices takes $O(t \cdot (|V(G)| + |E(G)|))$ time, where t is a small constant.

To computer degree-based synopses, for each vertex v_i, we sort the label embeddings of its 1-hop neighbors once per embedding dimension. This yields a per-vertex cost of $O(d \cdot deg(v_i) \log deg(v_i))$. Summing over all vertices, the total time complexity is $O(\sum_{v_i \in V(G)} d \cdot deg(v_i) \log deg(v_i))$.

After computing all key values, data vertices are sorted by $key(v_i)$, which takes $O(|V(G)| \log |V(G)|)$ time. Using bulk loading, B^+-tree can be constructed in linear time with respect to the number of sorted records, i.e., $O(|V(G)|)$.

Combining all steps, the overall time complexity of *iLabel* index construction is: $O(d \cdot |E(G)| + t \cdot (|V(G)| + |E(G)|) + \sum_{v_i \in V(G)} d \cdot deg(v_i) \log deg(v_i) + |V(G)| \log |V(G)|)$, which is near-linear in the graph size and small t.

B.2 Space Complexity of the *iLabel* Index

Given a data graph G and embedding dimension d, each data vertex v_i stores its d-dimensional monotonic embedding $o(v_i)$ and two types of auxiliary synopses. Storing vertex embeddings for all vertices requires $O(|V(G)| \cdot d)$ space.

For hop-based embedding synopses, each vertex v_i maintains t hop-based MBRs $v_i.MBR_{hop}^{(t)}$, each consisting of two d-dimensional boundary vectors. The total space for hop-based synopses is therefore $O(|V(G)| \cdot t \cdot d)$.

For degree-based embedding synopses, each vertex v_i stores one degree-based MBR $v_i.MBR_{deg}^{(\delta)}$ for each possible substructure degree $\delta \leq deg(v_i)$. Each MBR consists of two d-dimensional vectors, resulting in $O(d \cdot deg(v_i))$ space per vertex. Summed over all vertices, the total space requirement is $O(d \cdot |E(G)|)$.

The B^+-tree stores only one-dimensional keys and pointers. Both leaf and non-leaf nodes require $O(|V(G)|)$ space in total, which is negligible compared to embedding and synopsis storage.

Overall, the space complexity of the *iLabel* index is $O(|V(G)| \cdot d \cdot t + d \cdot |E(G)|)$, which scales linearly with the graph size and embedding dimension, and is suitable for large graphs.

B.3 Complexity Analysis of Algorithm 2

For each query vertex $q_i \in V(q)$, generating its monotonic vertex embedding requires aggregating embeddings of its 1-hop neighbors, which costs $O(deg(q_i) \cdot d)$ time (line 2), where d is the embedding dimension. Computing the hop-based synopsis $q_i.MBR_{hop}$ involves performing BFS expansions on the query graph up to k hops while maintaining per-dimension minimum and maximum values. This costs $O(|E(q)| + k \cdot |V(q)| \cdot d)$ per query vertex (line 3). Since query graphs are typically small, this cost is negligible compared to index traversal and refinement.

For each query vertex, descending the B$^+$-tree from the root to the first relevant leaf node incurs $O(h)$ time, where h is the height of the index (lines 4–9). Let R_i denote the number of data vertices whose keys fall within the query-specific range $[key(q_i), key(L(q_i)+1))$. During the sequential leaf scan, each such vertex is examined once. For each data vertex, embedding dominance pruning costs $O(d)$ (line 14), hop-based synopsis pruning costs $O((k-1)\cdot d)$ (line 15), and degree-based synopsis pruning costs $O(d)$ (lines 16–17). Thus, the total pruning cost per query vertex is $O(R_i \cdot k \cdot d)$ (lines 10–18).

After candidate sets are constructed, the greedy matching order generation procedure selects query vertices based on candidate set sizes. In the worst case, this procedure takes $O(|V(q)|^2)$ time (line 19).

Finally, the refinement phase performs a backtracking-based exact subgraph matching guided by the matching order and candidate sets (line 20). In the worst case, it enumerates all combinations of candidates, yielding $O(\prod_{q_i \in V(q)} |q_i.cand\_set|)$ time.

Therefore, the overall time complexity of Algorithm 2 is $O(\sum_{q_i \in V(q)} (deg(q_i)\cdot d + |E(q)| + k\cdot|V(q)|\cdot d + h + R_i\cdot k\cdot d) + |V(q)|^2 + \prod_{q_i \in V(q)} |q_i.cand\_set|)$.