Technical Report: On Extension for Structure-Preserving Subgraph Isomorphism Query

Zhe Fan Byron Choi
Department of Computer Science, Hong Kong Baptist University, Hong Kong, China
{zfan, bchoi}@comp.hkbu.edu.hk

ABSTRACT

This technical report briefly shows some ideas of extensions for *structure-preserving subgraph isomorphism* SPsubIso in [1, 2]. In particular, in Section 1, we show how SPsubIso supports *edge-labeled graphs*. In section 2, we report how our proposed encryption method CGBE and encoding scheme support *induced subgraph isomorphism*.

In these two sections, the system setting follows that of [1], *i.e.,* we aim to protect both query graph Q and data graph G. We can easily extend it to the setting of [2], as that of [1] subsumes the cases in [2].

1. EXTENSION FOR EDGE-LABELED GRAPHS

Violation. Suppose there are n types of edge labels for both Q and G. For simplicity, we let each type τ of the edge label is in $[1, n]$, *i.e.,* $\tau \in [1, n]$. By definition, the *violations* for each mapping between Q and G are:

(1) $\mathbf{M_Q}(u_i, u_j) = \tau$ but $\mathbf{M_G}(v_i, v_j) = 0$; or

(2) $\mathbf{M_Q}(u_i, u_j) = \tau$ but $\mathbf{M_G}(v_i, v_j) = \tau'$, $\tau \neq \tau'$

where (u_i, u_j) of Q is *mapped* to (v_i, v_j) of G in the given mapping.

Intuitively, there are two cases of violation: (1) Q has an edge (u_i, u_j) but there is *no* mapped edge (v_i, v_j) in G; or (2) Q has an edge (u_i, u_j) with label τ while that of the mapped edge (v_i, v_j) in G is τ', where $\tau \neq \tau'$.

Encoding. Based on the violations defined above, we denote each type τ as a large *prime* number q_τ and let $q = \Pi_{\tau \in [1,n]} q_\tau$. Then, given a query graph Q and its adjacency matrix $\mathbf{M_Q}$, the encoding of Q is defined as follows.

Definition 1: The *encoding* of the entries of $\mathbf{M_Q}$ are: $\forall u_i, u_j \in V(Q)$,

$$\begin{cases} \mathbf{M_Q}(u_i, u_j) = q & \text{if } \mathbf{M_Q}(u_i, u_j) = 0 \\ \mathbf{M_Q}(u_i, u_j) = q/q_\tau & \text{if } \mathbf{M_Q}(u_i, u_j) = \tau \end{cases}$$

□

Similarly, for data graph G, its encoding is defined as below:

Definition 2: The *encoding* of the entries of $\mathbf{M_G}$ are: $\forall v_i, v_j \in V(G)$,

$$\begin{cases} \mathbf{M_Q}(v_i, v_j) = 1 & \text{if } \mathbf{M_G}(v_i, v_j) = 0 \\ \mathbf{M_Q}(v_i, v_j) = q_\tau & \text{if } \mathbf{M_Q}(v_i, v_j) = \tau \end{cases}$$

□

It is worth remarking that the encoding is to detect the violations for subgraph isomorphism checking.

Encryption. We directly adopt CGBE [1] to encrypt all entries of encoded $\mathbf{M_Q}$ and $\mathbf{M_G}$.

Violation detection. Under the well-defined violations, the encodings and the encryption method, we are ready to accordingly transform the detection of *violation* for each mapping as follows:

(1) $\mathbf{M_Q}(u_i, u_j) \times \mathbf{M_G}(v_i, v_j) = q/q_\tau \pmod q$; or

(2) $\mathbf{M_Q}(u_i, u_j) \times \mathbf{M_G}(v_i, v_j) = qq_{\tau'}/q_\tau \pmod q$

where (u_i, u_j) of Q is *mapped* to (v_i, v_j) of G in the given mapping; Otherwise, the product *must* be 0 (mod q).

Intuitively, it follows those two cases of violation defined above: (1) Q has an edge (u_i, u_j) but there is *no* mapped edge (v_i, v_j) in G, *i.e.,* the product is $q/q_\tau \pmod q$; or (2) Q has an edge (u_i, u_j) with label τ while that of the mapped edge (v_i, v_j) in G is τ', $\tau \neq \tau'$, *i.e.,* the product is $qq_{\tau'}/q_\tau \pmod q$.

SPsubIso. For the SPsubIso, the mathematical computations are the same to those three steps in [1], based on the re-defined violation detection as above.

False positiveness. One can easily verify that there are still *bounded* false positiveness in the SPsubIso, which is negligible, similar to that of [1, 2].

2. EXTENSION FOR INDUCED SUBGRAPH ISOMORPHISM

For simplicity, we only discuss the graphs *without* edge labels in this section. One can verify that the extensions in Section 1 can be directly adopted in a similar way.

Violation. By definition, the *violation* of *induced subgraph isomorphism* for each mapping is as below:

(1) $\mathbf{M_Q}(u_i, u_j) = 0$ but $\mathbf{M_G}(v_i, v_j) = 1$; or

(2) $\mathbf{M_Q}(u_i, u_j) = 1$ but $\mathbf{M_G}(v_i, v_j) = 0$,

where (u_i, u_j) of Q is *mapped* to (v_i, v_j) of G in the given mapping.

Based on these, we observe that the semantics of violation for induced subgraph isomorphism is *XOR* between Q and G, as opposed to *AND* between Q and G for that of subgraph isomorphism.

Encoding. Under the violation, we re-define the encodings for Q and G as follows. We denote two large prime number q_1 and q_2, and let $q = q_1 q_2$.

Definition 3: The *encoding* of the entries of $\mathbf{M_Q}$ are: $\forall u_i, u_j \in V(Q)$,

$$\begin{cases} \mathbf{M_Q}(u_i, u_j) = q_1 & \text{if } \mathbf{M_Q}(u_i, u_j) = 0 \\ \mathbf{M_Q}(u_i, u_j) = q_2 & \text{if } \mathbf{M_Q}(u_i, u_j) = 1 \end{cases}$$

□

Definition 4: The *encoding* of the entries of $\mathbf{M_G}$ are: $\forall u_i, u_j \in V(G)$,

$$\begin{cases} \mathbf{M_G}(v_i, v_j) = q_2 & \text{if } \mathbf{M_G}(v_i, v_j) = 0 \\ \mathbf{M_G}(v_i, v_j) = q_1 & \text{if } \mathbf{M_G}(v_i, v_j) = 1 \end{cases}$$

□

Violation detection. The detection of violation for each mapping becomes as below.

(1) $\mathbf{M_Q}(u_i, u_j) \times \mathbf{M_G}(v_i, v_j) = q_1^2 \pmod q$; or
(2) $\mathbf{M_Q}(u_i, u_j) \times \mathbf{M_G}(v_i, v_j) = q_2^2 \pmod q$

where (u_i, u_j) of Q is *mapped* to (v_i, v_j) of G in the given mapping; Otherwise, the product *must* be 0 (mod q).

In this case, the property for the above encoding of induced subgraph isomorphism is the same to that of the subgraph isomorphism, *i.e.,* it can be transformed into a sequence of mathematical computations which are identical to that of subgraph isomorphism. CGBE and SPsubIso can be applied directly. Negligible false positiveness is still preserved, as expected.

3. REFERENCES

[1] Z. Fan, B. Choi, Q. Chen, J. Xu, H. Hu, and S. S. Bhowmick. Structure-preserving subgraph query services. *TKDE*, 27(8):2275–2290, 2015.

[2] Z. Fan, B. Choi, J. Xu, and S. S. Bhowmick. Asymmetric structure-preserving subgraph queries for large graphs. In *ICDE*, pages 339–350, 2015.