Supplemental Material for Sampling Code Clones from Program Dependence Graphs with GRAPLE

Tim A. D. Henderson* and Andy Podgurski†

Department of Electrical Engineering and Computer Science
Case Western Reserve University
Cleveland, Ohio

This document contains a proof of Theorem 1 from the paper and a worked example of the mathematical technique. A few definitions and figures are restated to add clarity to the proof along with some expanded explanation.

0.1 Formal Definitions

A *directed labeled graph* (labeled digraph) G is a set of vertices V, a set of edges $E = V \times V$, and a labeling function which maps vertices (or edges) to labels $l : V|E \to L$. E can be represented by a matrix $\mathbf{E}$. $\mathbf{E}_{i,j} = 1$ if and only if there is an edge from vertex v_i to vertex v_j, otherwise it is 0.

H is a subgraph of G ($H \sqsubseteq G$), if and only if an injective mapping $m : V_H \to V_G$ exists such that:

1. All vertices in H map vertices in G with the same label:
   $\forall\ v \in V_H\ [l_H(v) = l_G(m(v))]$
2. All edges in H are in G:
   $\forall\ (u, v) \in E_H\ [(m(u), m(v)) \in E_G]$
3. All edge labels match:
   $\forall\ (u, v) \in E_H\ [l_H(u, v) = l_G(m(u), m(v))]$

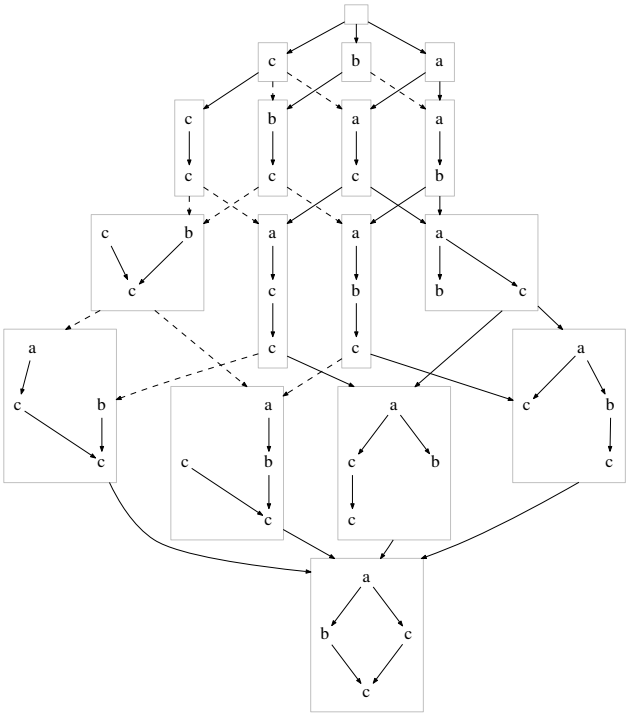
Such a mapping m is known as an *embedding*. A digraph A is *isomorphic* to another digraph B, $A \cong B$, if $A \sqsubseteq B$ and $B \sqsubseteq A$. The *isomorphism class* of a subgraph H is the set of all of the subgraphs of G isomorphic to H with distinct mappings, denoted $[\![H]\!] = \{H' \sqsubseteq G : H' \cong H \wedge m_{H'} \neq m_H\}$.

The subgraph relation $\cdot \sqsubseteq \cdot$ induces a *connected subgraph lattice* $\mathcal{L}_G$ representing all possible ways of constructing G (see Figure 1). $\mathcal{L}_G$ can itself be viewed as a directed graph where each vertex u represents a unique connected[1] subgraph of G. An edge exists between u and v if adding one edge to u creates

*tadh@case.edu
†podgurski@case.edu

[1] In this paper, *connected* ignores edge direction (see Fig. 2).

Figure 1: A connected subgraph lattice. The bottom node of the lattice contains the graph all other graphs are a subgraph of.



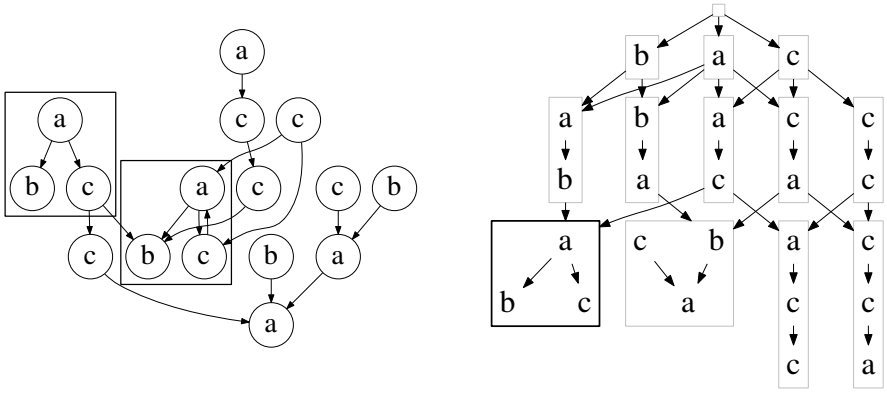
a subgraph $u + \epsilon$ which is isomorphic to v, $v \cong u + \epsilon$. A *k-frequent connected subgraph lattice* k-$\mathcal{L}_G$ contains only those subgraphs which have at least k embeddings in G, see Figure 2. Finally, a *pattern* refers to the isomorphism class $[\![H]\!]$ of a subgraph H.

0.2 Computing the Probability of Selecting a Maximal Subgraph

In order to use the HT estimator outlined in the paper, it is necessary to determine the probability p_i that the i^{th} maximal frequent pattern $[\![H_i]\!]$ is selected on a random walk of the k-frequent connected subgraph lattice (k-$\mathcal{L}_G$). We compute these probabilities using the theory of Markov chains.

A *finite-state Markov chain* [1] consists of a finite set of states, $S = \{s_1, \ldots, s_n\}$, and a matrix $\mathbf{P}$, called the *transition matrix*, where $\mathbf{P}_{i,j}$ gives the probability of a state transition from s_i to s_j. A Markov chain moves from state to state according to the probabilities in the transition matrix. A random walk in a graph G can be viewed as a Markov chain whose set of states S corresponds to the vertex set V_G. An *absorbing Markov chain* [1] is a special type of Markov

Figure 2: Figure 2b is a connected subgraph lattice of Figure 2a but only includes subgraphs with 2 or more embeddings in Figure 2a. The boxed nodes in the graph show the embeddings of the boxed subgraph in the lattice. The lattice places subgraphs in a partial order where a subgraph A is less than B if A is a subgraph of B. (See Section 0.1)



(a) A labeled directed graph

(b) 2-frequent connected subgraph lattice

chain which always ends in a state that cannot be exited, called an *absorbing state*.

To construct an absorbing Markov chain from the lattice k-$\mathcal{L}_G$, let the states of the chain be the vertices of the lattice (i.e., the frequent patterns $[\![H_i]\!]$). To model how the algorithm in Listing 1 transitions from one lattice node to the next by uniformly selecting a neighboring node, let the transition probability for an edge $v_i \to v_j$ be the reciprocal of the out-degree of v_i:

$$\mathbf{P}_{i,j} = \begin{cases} \frac{1}{\sum_k \mathbf{E}_{i,k}} & \text{if } \mathbf{E}_{i,j} = 1 \\ 1 & \text{if } i = j \wedge v_i \text{ is maximal} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

The selection probability p_i of $[\![H_i]\!]$ is the probability that state s_i absorbs the Markov process starting at the bottom lattice node. To compute p_i, arrange the transition matrix $\mathbf{P}$ into *canonical form* such that the transient states come before the absorbing states:

$$\mathbf{P} = \begin{array}{c} \\ \text{TR.} \\ \text{ABS.} \end{array} \begin{array}{c} \begin{array}{cc} \text{TR.} & \text{ABS.} \end{array} \\ \begin{bmatrix} \mathbf{Q} & \mathbf{R} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \end{array} \tag{2}$$

$\mathbf{Q}_{i,j}$ is the probability of transitioning from a transient state s_i to transient state s_j. $\mathbf{R}_{i,j}$ is the probability of transitioning from transient state s_i to absorbing state s_{t+j} where t is the number of transient states. $\mathbf{I}$ is the identity matrix

Listing 1: GRAPPLE's sampling procedure

```
# param G : the graph being mined
# param bottom : lattice node for the empty subgraph
# param min_support : int, minimum number of embeddings
# returns : leaf node of the frequent connected subgraph
#           lattice which is a maximal frequent subgraph
def walk(G, bottom, min_support):
  v = u = bottom
  while v is not None:
    u = v
    v = rand_select(get_children(G, u, min_support))
  return u

# param u : a lattice node
# returns : a list of lattice nodes which are 1 edge
#           extensions of u
def get_children(G, u, min_support):
  exts = list()
  for emb in u.embeddings:
    for a in embedding.V:
      for e in G.edges_to_and_from(a):
        if not emb.has_edge(e):
          exts.append(emb.extend_with_edge(e))
  groups = group_isomorphs(exts)
  return [ LatticeNode(lbl, group)
           for lbl, group in groups.iteritems()
           if len(group) >= min_support ]

# param subgraphs : a list of subgraphs of G
# returns : map label -> list of isomorphic subgraphs.
def group_isomorphs(subgraphs):
  isomorphs = dict()
  for sg in subgraphs:
    label = bliss.canonical_label(sg)
    if label not in isomorphs:
      isomorphs[label] = list()
    isomorphs[label].append(sg)
  return { label: minimum_image_supported(group)
           for label, group in isomorphs.iteritems() }
```

and $\mathbf{0}$ is the zero matrix, as once a Markov process enters an absorbing state it never leaves. The probability of a process starting at the bottom of the lattice s_0 and being absorbed by state s_i with zero or more transitions ($\xrightarrow{\star}$) is [1]:

$$p_i = \Pr[s_0 \xrightarrow{\star} s_i] = (\mathbf{P}^{\infty})_{0,i} = ((\mathbf{I} - \mathbf{Q})^{-1}\mathbf{R})_{0,(i-t)} = (\mathbf{NR})_{0,(i-t)} \tag{3}$$

Note, $\mathbf{N} = (\mathbf{I} - \mathbf{Q})^{-1}$ is the fundamental matrix for absorbing Markov chains. It is equivalent to summation of the power series of $\mathbf{Q}$ [1].

$$\mathbf{N} = (\mathbf{I} - \mathbf{Q})^{-1} = \mathbf{I} + \mathbf{Q} + \mathbf{Q}^2 + \mathbf{Q}^3 + ... \tag{4}$$

The term $\mathbf{N}_{i,j}$ is the probability of transitioning from state i to state j after 0 or more steps.

0.2.1 Computing p_i with a submatrix of P

Lemma 1. *Let s_i be an absorbing state in a Markov chain formed from a k-frequent connected subgraph lattice of a graph G. The selection probability $p_i = (\mathbf{NR})_{0,(i-t)}$ can be computed from a sub-matrix of the transition matrix $\mathbf{P}$ containing only those states from which s_i can be reached.*

Proof. If there does not exist a path $v_j \xrightarrow{\star} v_i$ in the k-frequent connected subgraph lattice k-$\mathcal{L}_G$ then the product of its adjacency matrix entries corresponding to any sequence of edges possibly connecting v_j to v_i must be zero. Therefore, summing over all such edge sequences, we have:

$$\left(\sum_{n=1}^{\infty}\sum_{k_1=1}^{n}\dots\sum_{k_n=1}^{n}\mathbf{E}_{j,k_1}\left(\prod_{i=1}^{n-1}\mathbf{E}_{k_i,k_{i+1}}\right)\mathbf{E}_{k_n,i}\right)=0 \tag{5}$$

The probability of a Markov chain that starts in state s_j eventually reaching state s_i is

$$\Pr[s_j \xrightarrow{\star} s_i] = \sum_{n=1}^{\infty}\sum_{k_1=1}^{n}\dots\sum_{k_n=1}^{n}\mathbf{P}_{j,k_1}\left(\prod_{i=1}^{n-1}\mathbf{P}_{k_i,k_{i+1}}\right)\mathbf{P}_{k_n,i} \tag{6}$$

If there does not exist a path in the lattice from v_j to v_i then this probability is zero. The selection probability formula $p_i = (\mathbf{NR})_{j,(i-t)}$ can be rewritten, with t indicating the number of transient nodes, as shown in Equation 7.

$$p_i = \sum_{k=1}^{t}\mathbf{N}_{j,k}\mathbf{R}_{k,(i-t)} \tag{7}$$

Using the definition of the fundamental matrix this equation can be rewritten as follows obtaining Equation 11.

$$p_i = \sum_{k=1}^{t}\left(\lim_{n=1}^{\infty}\left(\mathbf{I}+\sum_{e=1}^{n}\mathbf{Q}^e\right)\right)_{j,k}\mathbf{R}_{k,(i-t)} \tag{8}$$

$$p_i = \sum_{k=1}^{t}\left(\sum_{n=1}^{\infty}\sum_{k_1=1}^{t}\dots\sum_{k_n=1}^{t}\mathbf{Q}_{j,k_1}\left(\prod_{x=1}^{n-1}\mathbf{Q}_{k_x,k_{x+1}}\right)\mathbf{Q}_{k_n,k}\mathbf{R}_{k,(i-t)}\right) \tag{9}$$

$$p_i = \sum_{k=1}^{t}\left(\sum_{n=1}^{\infty}\sum_{k_1=1}^{t}\dots\sum_{k_n=1}^{t}\mathbf{P}_{j,k_1}\left(\prod_{x=1}^{n-1}\mathbf{P}_{k_x,k_{x+1}}\right)\mathbf{P}_{k_n,k}\mathbf{P}_{k,i}\right) \tag{10}$$

$$p_i = \sum_{n=1}^{\infty}\sum_{k_1=1}^{t}\dots\sum_{k_n=1}^{t}\mathbf{P}_{j,k_1}\left(\prod_{i=1}^{n-1}\mathbf{P}_{k_i,k_{i+1}}\right)\mathbf{P}_{k_n,i} \tag{11}$$

Note, Equation 11 is equivalent to the right hand side of Equation 6. Since $\Pr[s_j \xrightarrow{\star} s_i] = 0$ if there is no path in the lattice from v_j to v_i, vertices from which s_i cannot be reached have no effect on the computation of p_i and can

be omitted. Omitting vertices v_j for all v_j where there does not exist a path in the lattice to v_i corresponds to removing the j^{th} row and column from $\mathbf{P}$. Therefore, only a sub-matrix of $\mathbf{P}$ containing those states from which s_i can be reached are needed. □

Lemma 2. *The states from which an absorbing state s_i can be reached in a Markov chain formed from a k-frequent connected subgraph lattice k-$\mathcal{L}_G$ correspond to the vertices of the connected subgraph lattice of the graph represented by state s_i.*

Proof. A vertex v of k-$\mathcal{L}_G$ represents a graph. Given two vertices u and v of k-$\mathcal{L}_G$, u reaches v if and only if a sequences of edges can be added to u such that u extended with those edges is isomorphic to v (e.g. $u + \epsilon_1 + \epsilon_2 + ... \cong v$). Thus, the statement u reaches v in k-$\mathcal{L}_G$ is equivalent to saying u is a subgraph of v. If u is a subgraph of v then it will be a vertex in v's connected subgraph lattice $\mathcal{L}_v$ by the definition of connected subgraph lattice. Therefore, all states in which can reach s_i must be correspond to subgraphs of s_i and are therefore in s_i's connected subgraph lattice. □

Theorem 1. *Let $[\![H_i]\!]$ be a maximal k-frequent pattern sampled from k-$\mathcal{L}_G$. Let s_i be the corresponding state in the Markov chain formed from k-$\mathcal{L}_G$. The selection probability of $[\![H_i]\!]$, $p_i = ((\mathbf{I} - \mathbf{Q})^{-1}\mathbf{R})_{0,(i-t)}$, can be computed from the submatrix of $\mathbf{P}$ that includes only the rows and columns that correspond to subgraphs of H_i.*

Proof. By Lemma 2 all states of the Markov chain which can reach s_i correspond to subgraphs of s_i. By Lemma 1 the only rows and columns of $\mathbf{P}$ which are necessary are the rows and columns for states which can reach s_i. Therefore, the sub-matrix of $\mathbf{P}$ that is needed only contains states which correspond to subgraphs of H_i and are in the connected subgraph lattice $\mathcal{L}_{H_i}$. □

As a consequence of Theorem 1, only a sub-matrix of $\mathbf{P}$ is needed to compute p_i for any given i, and that sub-matrix is exactly the one which corresponds to the connected subgraph lattice computed from the target subgraph.

Example 1. *The lattice in figure 2 corresponds to the following matrix* $\mathbf{P}$ *arranged in canonical form. Note: the empty spaces in the matrices represent the number 0.*

$$\begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \\ 11 \\ 12 \end{array}\!\!\begin{array}{c} \begin{array}{ccccccccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \end{array} \\ \left[\begin{array}{ccccccccccccc} & .33 & .33 & .33 & & & & & & & & & \\ & & & & .5 & .5 & & & & & & & \\ & & & & .25 & .25 & .25 & .25 & & & & & \\ & & & & & & .33 & .33 & .33 & & & & \\ & & & & & & & & & 1 & & & \\ & & & & & & & & & & 1 & & \\ & & & & & & & & & .5 & & .5 & \\ & & & & & & & & & & .5 & & .5 \\ & & & & & & & & & & & .5 & .5 \\ & & & & & & & & & 1 & & & \\ & & & & & & & & & & 1 & & \\ & & & & & & & & & & & 1 & \\ & & & & & & & & & & & & 1 \end{array}\right] \end{array}$$

If the subgraph $H_{11} = a \to c \to c$*, corresponding to state* s_{11}*, is sampled the following would be the submatrix of* $\mathbf{P}$ *needed to compute* p_{11}*.*

$$\mathbf{P}^{H_{11}} = \begin{array}{c} \\ 0 \\ 2 \\ 3 \\ 6 \\ 8 \\ 11 \end{array}\!\!\begin{array}{c} \begin{array}{cccccc} 0 & 2 & 3 & 6 & 8 & 11 \end{array} \\ \left[\begin{array}{cccccc} & .33 & .33 & & & \\ & & & .25 & & \\ & & & .33 & .33 & \\ & & & & & .5 \\ & & & & & .5 \\ & & & & & 1 \end{array}\right] \end{array}$$

The fundamental submatrix would be:

$$\mathbf{N}^{H_{11}} = (\mathbf{I} - \mathbf{Q}^{H_{11}})^{-1} = \begin{array}{c} \\ 0 \\ 2 \\ 3 \\ 6 \\ 8 \end{array}\!\!\begin{array}{c} \begin{array}{ccccc} 0 & 2 & 3 & 6 & 8 \end{array} \\ \left[\begin{array}{ccccc} 1 & .33 & .33 & .1914 & .1089 \\ & 1 & & .25 & \\ & & 1 & .33 & .33 \\ & & & 1 & \\ & & & & 1 \end{array}\right] \end{array}$$

Computing $\mathbf{N}^{H_{11}}\mathbf{R}^{H_{11}}$ *yields:*

$$\mathbf{N}^{H_{11}}\mathbf{R}^{H_{11}} = \begin{array}{c} \\ 0 \\ 2 \\ 3 \\ 6 \\ 8 \end{array}\!\!\begin{array}{c} \begin{array}{ccccc} 0 & 2 & 3 & 6 & 8 \end{array} \\ \left[\begin{array}{ccccc} 1 & .33 & .33 & .1914 & .1089 \\ & 1 & & .25 & \\ & & 1 & .33 & .33 \\ & & & 1 & \\ & & & & 1 \end{array}\right] \end{array}\!\!\begin{array}{c} \begin{array}{c} 11 \end{array} \\ \left[\begin{array}{c} \\ \\ \\ .5 \\ .5 \end{array}\right] \end{array} = \begin{array}{c} \\ 0 \\ 2 \\ 3 \\ 6 \\ 8 \end{array}\!\!\begin{array}{c} \begin{array}{c} 11 \end{array} \\ \left[\begin{array}{c} .150 \\ .125 \\ .333 \\ .5 \\ .5 \end{array}\right] \end{array}$$

The probability for starting at the root node of the lattice (the empty subgraph H_0) and ending at H_{11} is:

$$\begin{aligned} p_{11} &= Pr[s_0 \xrightarrow{\star} s_{11}] \\ &= (\mathbf{P}^{\infty})_{0,11} \\ &= ((\mathbf{I} - \mathbf{Q})^{-1}\mathbf{R})_{0,11-9} \\ &= ((\mathbf{I} - \mathbf{Q}^{H_{11}})^{-1}\mathbf{R}^{H_{11}})_{0,0} \\ &= .150 \end{aligned}$$

0.3 Estimation for Sub-populations

Suppose that we wish to estimate the mean $\mu_{\mathcal{D}}$ of a study variable y_i for a *sub-population* or *domain* $\mathcal{D}$ of the population $\mathcal{U}$ of all (isomorphism classes of) maximal frequent connected subgraphs, where membership in $\mathcal{D}$ is not known beforehand but must be determined by examining members of a sample drawn from $\mathcal{U}$. This problem arises, for example, if we wish to estimate the average size of code clones having a particular property, e.g., ones involving security-sensitive code. We use this technique to examine the *proportion* of frequent patterns with more than 8 vertices which are considered to be code clones by the application developer.

Assume that the size $\mathcal{N}_{\mathcal{D}}$ of $\mathcal{D}$ is unknown. Let $\mathcal{S}$ be a sample drawn from $\mathcal{U}$, let $\mathcal{S}_{\mathcal{D}}$ be the part of this sample that belongs to $\mathcal{D}$, and let $n_{\mathcal{D}}$ be the size of $\mathcal{S}_{\mathcal{D}}$. (Note that $n_{\mathcal{D}}$ is a random variable.) To estimate the domain total $\tau_{\mathcal{D}}$ we can use the modified Horvitz Thompson estimator [2]:

$$\hat{\tau}_{\mathcal{D}} = \sum_{i \in \mathcal{S}_{\mathcal{D}}} \frac{y_i}{\pi_i} \tag{12}$$

where the sum is over $\mathcal{S}_{\mathcal{D}}$ rather than $\mathcal{S}$. To estimate the domain size $\mathcal{N}_{\mathcal{D}}$, we can use the simpler form $\hat{\mathcal{N}}_{\mathcal{D}} = \sum_{i \in \mathcal{S}_{\mathcal{D}}} \frac{1}{\pi_i}$. To estimate the domain mean $\mu_{\mathcal{D}}$, we can use the estimator [2]:

$$\hat{\mu}_{\mathcal{D}} = \frac{\hat{\tau}_{\mathcal{D}}}{\hat{\mathcal{N}}_{\mathcal{D}}} \tag{13}$$

References

[1] Grinstead, C. M., and Snell, J. L. *Introduction to Probability*, 2 ed. American Mathematical Society, Providence, RI, 1997.

[2] Särndal, C.-E., Swensson, B., and Wretman, J. *Model Assisted Survey Sampling*. Springer-Verlag, 1992.