Efficient Enumeration Algorithms for Connected Induced Subgraphs with Large Girth

Kazuhiro Kurita[1], Alessio Conte[2], Kunihiro Wasa[3], Takeaki Uno[3], and Hiroki Arimura[1]

[1]IST, Hokkaido University, Sapporo, Japan, {k-kurita, arim}@ist.hokudai.ac.jp
[2]Universit'a di Pisa, Pisa, Italy, conte@di.unipi.it
[3]National Institute of Informatics, Tokyo, Japan, {wasa, uno}@nii.ac.jp

Abstract

The girth is the length of shortest cycle in a graph. Recently, the problem for computing the girth of a graph have been studied well. Itai and Rodeh developed the first non trivial algorithm for computing girth in general graphs. Their algorithm runs in $O(nm)$ time in the worst case and it runs in $O(n^2)$ time on average case. For unweighted planer graphs, Djidjev gave a $O\left(n^{5/4}\log n\right)$ time algorithm for computing girth. Chang *et al.* improved Djidjev's algorithm to a linear time algorithm. To the best of our knowledge. there are no known results for enumeration of induced connected subgraphs with large girth. In this paper, we propose two algorithms to solve this problem. The former algorithm enumerates all solutions in $O(n)$ time using $O\left(n^3\right)$ space. The latter algorithm enumerates all solutions in $O(m)$ time using $O(m)$ space.

1 Introduction

A *subgraph enumeration problem* is to output all subgraphs in an input graph G satisfying given constraint $\mathcal{R}$ without duplication. In this paper, we address the subgraph enumeration of all connected induced subgraphs with girth at least k. The girth of a graph is the length of its shortest cycle. In general, the efficiency of an algorithm is measured by the number of operations performed and the amount of space used with respect to the input size. However, the efficiency of enumeration algorithms is often measured by both the input size and the output size, i.e., the number of solutions. An enumeration algorithm is called *output polynomial time algorithm* [11]. if the algorithm runs in $poly(N, M)$ time, where N is the input size and M is the number of solutions; this is also referred to as *polynomial total time*. Moreover, if an enumeration algorithm runs in $Mpoly(N)$ time, it is called an *amortized polynomial time algorithm*. Another evaluation indicator for an enumeration algorithm is the *delay*. The delay is the maximum time between the following: (i) the time for finding the first solution, (ii) the time for finding the $i+1$th solution after finding the ith, and (iii) time between finding the last solution and the termination of the algorithm. In this paper, we present two amortized polynomial time algorithms. The former presented algorithm is faster than the second, however its space requirement is higher. The second presented algorithm uses less space, but it is less efficient compared to the first algorithm. Our problem is similar to the shortest path problem since length of the minimum cycle including v is equal to the shortest path from v to v. Hence, the girth of a graph can be computed using an *all pairs shortest path* algorithm, e.g., the Floyed-Warshall algorithm [4]. However, using an all pairs shortest algorithm are not efficient because our problem computes girth of a connected induced subgraph many times. Thus, rather than using an all pairs shortest path algorithm as a subroutine, we use distance matrices or breath first search to compute the distances between vertices.

1.1 Related work

The girth of a graph is related to many fundamental graph properties. Chandran and Subramanian showed that an upper bound of the treewidth can be obtained from the girth and average degree of a graph [1]. It is known that girth is related to another graph properties, e.g. average and minimum degree, chromatic number, diameter, and maximum genus [3, 5]. Furthermore, Raman [12] showed that for several W[1]-hard or W[2]-hard problems, if the girth of an input graph is large enough the problems

become FPT. Example problems are dominating set, independent set, and induced matching [10, 12]. The problem of finding the girth of a graph is still studied in recent work. Itai and Rodeh [8] showed the first non trivial result of finding the girth. They showed that it can find the minimum length cycle in $O(nm)$ time. Moreover, Djidjev [6] improved this result for a planar graph, which runs in $O\left(n^{5/4}\log n\right)$ time. With the latest result, the girth of a planer graph is computable in linear time [2].

If a graph G has girth ∞, then clearly G is acyclic, i.e., a tree; the same applies if the girth is larger than n. Consequently, if we enumerate all connected induced subgraphs with girth at least $n+1$ where n is the number of vertices, then our problem is equal to enumerating induced subtrees. Wasa *et al.* developed an efficient algorithm for the induced subtree enumeration problem [15]. An enumeration of tree structure is studied for a long time. Read and Tarjan provided an enumeration algorithm for spanning tree in $O(n+m+ns)$ [13] where n is the number of vertices, m is the number of edges, and s is the number of solutions. Kapoor and Ramesh give a more space efficient algorithm [9] that uses $O(nm)$ space. Shioura and Uno give an optimal algorithm for spanning tree enumeration [14]. This algorithm enumerates all spanning trees in $O(s+n+m)$ time using $O(n+m)$ space. Other than spanning trees, Ferreira *et al.* gave an output-sensitive algorithm for enumerating bounded-size trees in a graph in $O(sk)$ where k is the size of tree [7]. Moreover, Wasa *et al.* proposed constant amortized time algorithm for enumerating bounded-size trees in a tree [16].

2 Preliminary

Let $G=(V(G),E(G))$ be a simple undirected graph with no self-loops, with vertex set $V(G)$ and edge set $E(G)\subseteq V(G)\times V(G)$. If G is clear from the context, we denote $V(G)=V$ and $E(G)=E$. An edge e with vertex u and v is denoted by $e=\{u,v\}$. Two vertices $u,v\in V$ are *adjacent* if there is an edge $\{u,v\}$ in E. Let $N_G(u)$ be the set of vertices that are adjacent to u. We say v is a *neighbor* of u if $v\in N_G(u)$. Let $N_G[u]=N_G(u)\cup\{u\}$ be the set of closed neighbors of u. The *set of neighbor* of U is defined as follows: $N(U)=\bigcup_{u\in U}N_G(u)\setminus U$. Similarly, let $N[U]$ be $\bigcup_{u\in U}N_G(u)\cup U$. Let $d_G(u)=|N_G(u)|$ be the *degree* of u in G. $\Delta(G)=\max_{x\in V}d(x)$ denotes the maximum degree of G. For any vertex subset $V'\subseteq V$, we call $G[V']=(V',E[V'])$ an *induced subgraph*, where $E[V']=\{\{u,v\}\in E(G)\mid u,v\in V'\}$. Since $G[V']$ is uniquely determined by V', we identify $G[V']$ with V'. We define $G\setminus\{e\}=(V,E\setminus\{e\})$ and $G\setminus\{v\}=G[V\setminus\{v\}]$. For simplicity, we will use $v\in G$ and $e\in G$ to refer to $v\in V(G)$ and $e\in E(G)$, respectively.

An alternating sequence $\pi=(v_1,e_1,\ldots,e_k,v_k)$ of vertices and edges is a *path* if each edge and vertex in π appears at most once. We also call π an v_0-v_n *path*. An alternating sequence $C=(v_1,e_1,\ldots,e_k,v_k)$ of vertices and edges is a *cycle* if $(v_1,e_1,\ldots,v_{k-1})$ is a v_0-v_{k-1} path and $v_k=v_1$. The length of a path and a cycle is defined by the number of its edges. The *shortest path* between v_0 and v_n is the path π with minimum length in v_0-v_n path. The *girth* of a graph G is the length of a shortest cycle in G. The girth of a graph G is denoted by $g(G)$. We say that a graph $G=(V,E)$ is *connected* if any two vertices $u,v\in V$ have a u-v path.

Now, we define the k-girth connected induced subgraph enumeration problem as follows:

Problem 1 (k-girth connected induced subgraph enumeration problem)**.** *Enumerate all connected induced subgraphs in a given graph G with girth at least k without duplicates.*

As a main result, we propose the algorithms EKG 1 and EKG 2. EKG 1 solves this problem in $O(n)$ time using $O\left(n^3\right)$ space. EKG 2 solves this problem in $O(m)$ time using $O(m)$ space.

3 Enumeration by the binary partition method

In this section, we propose a basic algorithm based on the binary partition method. The proposed algorithm solves the k-girth connected induced subgraph enumeration problem in $O\left(n^2m\right)$ amortized time with $O(m)$ space. We develop the more efficient proposed algorithms EKG 1 and EKG 2 based on the basic algorithm.

We first introduce the binary partition method, which is one of the fundamental frameworks for constructing enumeration algorithms. The *binary partition method* is an algorithm $\mathcal{A}$ which enumerates all solutions by recursively dividing a search space into disjoint search spaces. We call a dividing step an *iteration*. Let G, $\mathcal{S}(G)$, and X be an input graph, the set of solutions for G, and an iteration of the algorithm, respectively. Let $\mathcal{S}(X)$ be the set of solutions included in a search space of X. Note that, for the initial iteration I, $\mathcal{S}(I)=\mathcal{S}(G)$ holds. At the initial iteration, $\mathcal{A}$ picks a vertex v in G and then divides $\mathcal{S}(I)$ into $\mathcal{S}_0(I),\ldots,\mathcal{S}_k(I)$ such that $\mathcal{S}(I)=\mathcal{S}_0(I)\cup\cdots\cup\mathcal{S}_k(I)$ and $\mathcal{S}_0(I),\ldots,\mathcal{S}_k(I)$ are pairwise

Algorithm 1: Enumerate all connected induced subgraphs with girth at least k.

```
1 Procedure EKG (G, k)
    //G: an input graph, k: positive integer
2   RecEKG (∅, V(G));
3 Procedure RecEKG (S, C(S))
    //S: the current solution, C(S): the candidate set of S
4   Output S ;
5   for v ∈ C(S) do
6     RecEKG (S ∪ {v}, C(S ∪ {v}));
7     G ← G \ {v};
8   return;
```

disjoint. For any iteration X, if $|\mathcal{S}(X)| = 1$ then $\mathcal{A}$ outputs the element. $\mathcal{A}$ recursively applies this procedure until all vertices are selected.

Next, we introduce the *enumeration tree* $\mathcal{T}(G) = \mathcal{T} = (\mathcal{V}, \mathcal{E})$. Here, $\mathcal{V}$ is the set of iterations of $\mathcal{A}$ for G and $\mathcal{E}$ is a subset of $\mathcal{V} \times \mathcal{V}$. For any iteration X, we define the vertex set V_X as $\bigcap_{S \in \mathcal{S}(X)} S$. That is, V_X is the set of vertices that appear in the all solutions of $\mathcal{S}(X)$. For any iterations X and Y, Y is a *child* of X if $V_Y \subset V_X$ and $|V_X \setminus V_Y| = 1$ hold. We call X the *parent* of Y. An iteration Z that is a child of Y is called a *grandchild* of X. In the binary enumeration tree $\mathcal{T}(G)$, we call an iteration an *internal iteration* if the iteration has children, and a *leaf iteration* otherwise. Moreover, an iteration X is the *root iteration* if there is no iteration that has X as a child. That is, X is the first iteration called by $\mathcal{A}$.

Now, we propose the algorithm EKG (See Algorithm 1). EKG is based on the binary partition method, but in EKG , each iteration outputs a solution. This algorithm enumerates all connected induced subgraph with girth at least k in $O\left(n^2 m\right)$ time with $O\left(m\right)$ space. Let $S(X)$ be the solution that outputted on an iteration X. Let $C(S(X)) = \{v \in N(S(X)) \mid g(G[S(X) \cup \{v\}]) \geq k\}$. That is, $C(S(X))$ is the set of vertices v such that $S(X) \cup \{v\}$ is also a solution. We call $C(S(X))$ the candidate set of $S(X)$.

Lemma 1. *Let $\mathcal{T}$ be an enumeration tree consisted of EKG , and X be an iteration on $\mathcal{T}$. Then, $G[S(X)]$ is connected and $g(G[S(X)]) \geq k$ holds.*

Proof. When X is the root iteration, $S(X) = \emptyset$. Then, $g(G[X)) \geq k$ in X since $g(G[S(X)]) = \infty$ by definition of the girth. By definition of connectivity, $G[X]$ is connected. For any iteration X such that $|S(X)| \leq k$, suppose that $G[S(X)]$ is connected and $g(G[S(X)]) \leq i$. From the definition of $C(S(X))$, for any vertex $v' \in C(S(X))$, at least one neighbor of v' belongs to S and $g(G[S \cup \{v\}]) \geq k$ holds. Hence, every output induces a subgraph that is connected and girth at least k. □

Thus, from the definition of $C(S(X))$ and Lemma 1, we can easily obtain the correctness of EKG .

Lemma 2. *EKG enumerates all solutions in an input graph G without duplication.*

Proof. From Lemma 1, EKG outputs only solutions. In addition, EKG does not output each solution more than once. We show that EKG outputs all solutions. Let i be the size of solution. When $i = 0$, EKG obviously outputs the empty subgraph. Next, EKG outputs all solution with size i. Let S' be a vertex set such that $|S'| = i + 1$, $G[S']$ is connected, and $g(G[S']) \geq k$. Note that every connected subgraphs of a graph with girth at least k is also girth at least k. $G[S']$ has at least one vertex v. There is a vertex v in S' such that $G[S' \setminus \{v\}]$ is also a solution and for any iteration Y on the path from the root I to X'' on $\mathcal{T}$ EKG does not pick v from the candidate set of Y, where $S(X'') = S' \setminus \{v\}$. From the definition of $C(S(X''))$, $v \in C(S(X''))$. Hence, the statement holds. □

By adopting the finding girth algorithm [8] on Line 6, we can obtain the following theorem.

Theorem 1. *EKG solves the k-girth connected induced subgraph enumeration problem in $O\left(n^2 m\right)$ amortized time per solution.*

Proof. By Lemma 2, EKG enumerates all solutions without duplication. Next, we show the time complexity of EKG . The bottleneck of RecEKG is Line 6. Let X be an iteration. EKG computes the girth of subgraph $G[S(X) \cup \{u, v\}]$ for each $u \in C(S(X)) \setminus \{v\}$ in Line 6. Line 6 needs $O\left(n^2 m\right)$ time when we adopt a $O\left(nm\right)$ time algorithm for computing the girth $g(G[S(X) \cup \{u, v\}])$. In addition, this line executes $|C(S(X))|$ times for each iteration. Thus, the total computation time of

EKG is $O\left(\sum_{X\in\mathcal{T}} |C(S(X))|\, n^2m\right)$ time. Since $O\left(\sum_{X\in\mathcal{T}} |C(S(X))|\right) = O\left(|\mathcal{V}|\right)$, the total time of EKG is $O\left(|\mathcal{V}|\, n^2m\right)$ time. Note that the number of solution is $O\left(|\mathcal{V}|\right)$. Therefore, EKG runs $O\left(n^2m\right)$ amortized time per solution. □

4 Proposed Algorithms

As mentioned in the previous section, the bottleneck of the basic algorithm is updating the candidate set. In this section, we present two efficient update algorithms for the candidate sets.

4.1 $O\left(n\right)$ time and $O\left(n^3\right)$ space algorithm

In this subsection, we show EKG 1 using Algorithm 2. EKG 1 enumerates all solutions in $O\left(n\right)$ time using $O\left(n^3\right)$ space by using distance matrices.

We first introduce some notations used in this subsection. Let $\mathcal{T}$, X, X', and X'' be an enumeration tree, any iteration, a child of X, and a child of X', respectively. Path $\pi(u,v) = (u = u_1, e_1, \ldots, e_i, u_i = v)$ is a *head disjoint path* for path $\pi'(u,v) = (u = u'_1, e'_1, \ldots, e'_j, u'_j = v)$ if $e_1 \neq f_1$ holds. We say that $\pi(u,v)$ includes vertex u' or edge e' if u' or e' is on $\pi_1(u,v)$, respectively. Similarly, we say that cycle C includes vertex v or edge e if v or e is on C, respectively. Let $\pi(u,v)$ be a shortest path between vertices u and v. We call $\pi'(u,v)$ a *second shortest path* if $\pi'(u,v)$ is a head disjoint path for $\pi(u,v)$ and for any head disjoint path $\pi''(u,v)$ for $\pi(u,v)$, the length of $\pi'(u,v)$ is less than or equal to that of $\pi''(u,v)$. The *second distance* between u and v is defined by the length of a second shortest path between u and v.

EKG 1 uses the two distance matrices $M(S)$ and $M^{h_2}(S)$, where S is the set of vertices. $M(S)$ is a matrix with $C(S) \cup S$ columns and $C(S)$ rows. Each entry (i,j) of $M(S)$ corresponds to the length of a shortest path between the ith vertex in $C(S) \cup S$ and the jth vertex in $C(S)$ such that the set of vertices in the shortest path other than the source and target vertices is a subset of S. Note that the source and target vertex may belong to S, respectively. $M^{h_2}(S)$ is a matrix with $C(S)$ columns and $C(S)$ rows. Each entry (i,j) of $M^{h_2}(S)$ corresponds to the length of a second shortest path between the ith vertex in $C(S)$ and the jth vertex in $C(S)$ such that the set of vertices in the second shortest path other than the source and target vertices is a subset of S. We denote the distance and the second distance by between u and v $M_{uv}(S)$ and $M^{h_2}_{uv}(S)$, respectively. Moreover, we write $M_{uwv}(S)$ and $M^{h_2}_{uwv}(S)$ for the distance and the second distance from u to v via w, respectively. That is, $M_{uwv}(S) = M_{uw}(S) + M_{wv}(S)$ and $M^{h_2}_{uwv}(S) = M^{h_2}_{uw}(S) + M^{h_2}_{wv}(S)$. In EKG 1, we compute the candidate set by using two distance matrices. In the following lemma, we show a good relation of a shortest path and a second shortest path between u and v for computing $M^{h_2}(S)$ efficiently.

Lemma 3. *Let u and v be two distinct vertices in $C(S(X))$ such that $G[S(X)]$ is a solution. Let $\pi(u,v)$ and $\pi'(u,v)$ be two distinct paths from u to v in $G[S(X) \cup \{u,v\}]$. If $|\pi(u,v)| + |\pi'(u,v)| < k$, then $\pi(u,v)$ and $\pi'(u,v)$ share no vertices other than u and v.*

Proof. Proof by contradiction. Suppose that $\pi(u,v)$ and $\pi'(u,v)$ share vertex w such that $w \neq u$ and $w \neq v$. Without loss of generality, we can assume that there is cycle C that has u and w. From $|\pi(u,v)| + |\pi(u,v)| < k$, $|C| < k$. This contradicts $u \in C(S(X))$. Hence, the statement holds. □

To compute the candidate set, we show the following lemmas.

Lemma 4. *Let u and v be two distinct vertices in $C(S(X))$. Then, $g(G[S(X) \cup \{u,v\}]) \geq k$ if and only if $M_{uv}(S(X)) + M^{h_2}_{uv}(S(X)) \geq k$.*

Proof. It is obvious that if $g(G[S(X) \cup \{u,v\}]) \geq k$, then $M_{uv}(S(X)) + M^{h_2}_{uv}(S(X)) \geq k$ holds by definition. Next, we show that if $M_{uv}(S(X)) + M^{h_2}_{uv}(S(X)) \geq k$, then $g(G[S(X) \cup \{u,v\}]) \geq k$. From the assumption, for any cycle C in $G[S(X) \cup u, v])$ such that does not have both u and v, the length of C is at least k. Thus, in the following, we only consider the length of a cycle that includes both u and v in $G[S(X) \cup \{u,v\}])$. Let $\pi(u,v)$ and $\pi'(u,v)$ be two distinct paths between u and v in $G[S(X) \cup \{u,v\}])$, respectively. From Lemma 3, if there is vertex w that shared by $\pi(u,v)$ and $\pi'(u,v)$, then the length of the cycle consisting of $\pi(u,v)$ and $\pi'(u,v)$ is at least k. Thus, we assume that $\pi(u,v)$ and $\pi'(u,v)$ share no vertices other than u and v. Without loss of generality, we can assume that $\pi(u,v)$ is a shortest path. Since $\pi'(u,v)$ is a head disjoint path for $\pi(u,v)$, the length of $\pi'(u,v)$ is at least the length of a second shortest path between u and v. Thus, if $M_{uv}(S(X)) + M^{h_2}_{uv}(S(X)) \geq k$, then the length of any cycle including both u and v is at least k. Hence, the statement holds. □

From above discussion, we can see that the candidate set can be computed by using the two distance matrices. The next lemma shows that we can compute the candidate set in $O(n)$ time by using the property of Lemma 4.

Lemma 5. *Let X be an iteration, X' be a child of X, and v be a vertex in $C(S(X))$ such that $S(X') = S(X) \cup \{v\}$. Then, EKG 1 computes $C(S(X'))$ from $C(S(X))$ in $O(n)$ time.*

Proof. From Lemma 4, vertex u in $C(S(X))$ belongs to $C(S(X'))$ if and only if $M_{uv}(S(X)) + M^{h_2}_{uv}(S(X)) \geq k$. This inequality is decidable using $M(S(X))$ and $M^{h_2}(S(X))$ in constant time. Since $|C(S(X))| \leq n$, the statement holds. □

Next, we consider an update of each distance matrices of a child iteration of the current iteration. The following lemma can be obtained from the properties of a shortest path and a second shortest path.

Lemma 6. *Let v be a vertex in $V \setminus S(X)$. $M_{uw}(S(X) \cup \{v\}) = \min\{M_{uw}(S(X)), M_{uvw}(S(X))\}$, where $u, w \in C(S(X)) \cup S(X)$.*

Proof. Let X' be a child iteration of X. We consider a shortest path $\pi(u, w)$ between u and w in $G[S(X')]$. If $\pi(u, w)$ does not include v, then $|\pi(u, w)| = M_{uw}(S)$. If $\pi(u, w)$ include v, then $|\pi(u, w)| = M_{uv}(S) + M_{vw}(S)$. Hence, $M_{uw}(S(X')$ is equal to $\min(M_{uw}(S(X)), M_{uvw}(S(X)))$. The statement holds. □

From Lemma 6, EKG 1 updates $M(S(X) \cup \{v\})$ in $O(|M(S(X) \cup \{v\})|)$. In other words, EKG 1 updates the distance matrix in $O(|C(S(X')) \cup S(X')|\,|C(S(X'))|)$ time.

Lemma 7. *Let X' be a child of X. For any $u, w \in C(S(X'))$, $M^{h_2}_{uw}(S(X'))$ is equal to the second smallest length of paths in $\mathcal{P}(X, X', u, w)$, where $\mathcal{P}(X, X', u, w) = \{\pi(u, x, w) \mid x \in N(u) \cap S(X')\}$ and $\pi(u, x, w)$ is a shortest path from u to w via x.*

Proof. Let $\pi(u, w)$ be a shortest path between u and w in $\mathcal{P} = \mathcal{P}(X, X', u, w)$. Thus, since a shortest path $\pi'(u, w)$ in $\mathcal{P} \setminus \{\pi(u, w)\}$ is a head disjoint path for $\pi(u, w)$, $\pi'(u, w)$ is a second shortest path between u and w. Hence, the statement holds. □

Using $M(S(X'))$, the length of each path in $\mathcal{P}$ can be computed in constant time. Hence, from Lemma 7, the length of a second shortest path from u to w can be computed in $O(\Delta)$ time. The next lemma shows that if the sum of the distance and the second distance between u and v is less than k, then EKG 1 computes a second distance between u and w in constant time.

Lemma 8. *Let X' be child of X such that $S(X') = S(X) \cup \{v\}$ and u, w be two vertices in $C(S(X'))$. Let $\pi(u, w)$ and $\pi'(u, w)$ be a shortest and a second shortest path in $G[S(X')]$, respectively. If $|\pi(u, w)| + |\pi'(u, w)| < k$, then, $|\pi(u, w)|$ and $|\pi'(u, w)|$ are equal to any of $M_{uw}(S(X)), M_{uvw}(S(X')), M^{h_2}_{uw}(S(X))$, respectively.*

Proof. Suppose that $|\pi(u, w)| + |\pi'(u, w)| < k$. Then, from Lemma 3, $\pi(u, w)$ and $\pi'(u, w)$ share no vertices other than u and w. Moreover, it is obvious that $\pi(u, w)$ is also a shortest path in $G[S(X') \cup \{u, w\}]$. From Lemma 3, either $\pi(u, w)$ or $\pi'(u, w)$ does not include v. (A) Suppose that $\pi(u, w)$ does not include v and $\pi'(u, w)$ include v. Then, a shortest path $\pi(u, w)$ in $G[S(X')]$ is also a shortest path in $G[S(X)]$. Hence, $|\pi(u, w)| = M_{uw}(S(X))$ holds. Next, we consider the length of $\pi'(u, w)$. Proof by contradiction. We assume that $|\pi'(u, w)| > M_{uvw}(S(X'))$. There is a path $\pi''(u, w)$ with length $M_{uvw}(S(X'))$ that has a common vertex with $\pi(u, w)$. However, this contradicts Lemma 3. Hence, $|\pi'(u, w)| = M_{uvw}(S(X'))$. (B) Suppose that $\pi(u, w)$ include v and $\pi'(u, w)$ does not include v. Then, $|\pi(u, w)|$ is equal to $M_{uvw}(S(X'))$. Let $\pi''(u, w)$ be a shortest path in $G[S(X)]$, $\pi'''(u, w)$ be a second shortest path in $G[S(X)]$. Since $\pi''(u, w)$ and $\pi'''(u, w)$ are head disjoint paths for each other, either $\pi''(u, w)$ or $\pi'''(u, w)$ is a head disjoint path for $\pi(u, w)$ in $G[S(X')]$. Thus, if $\pi''(u, w)$ is a head disjoint path for $\pi(u, w)$, then $|\pi'(u, w)|$ is equal to $|\pi''(u, w)| = M_{uw}(S(X))$. In addition, if $\pi'''(u, w)$ is head disjoint for $\pi(u, w)$, $|\pi'(u, w)|$ is equal to $|\pi'''(u, w)| = M^{h_2}_{uw}(S(X))$. (C) Suppose that both $\pi(u, w)$ and $\pi'(u, w)$ do not include v. Thus, a shortest path and a second shortest path in $G[S(X')]$ are equal to a shortest path and a second shortest path in $G[S(X)]$, respectively. Hence, $|\pi(u, w)| = M_{uw}(S(X))$ and $|\pi'(u, w)| = M^{h_2}_{uw}(S(X))$ holds. The statement holds. □

From Lemma 8, if $|\pi(u, w)| + |\pi'(u, w)| < k$, then the length of $\pi(u, w)$ is equal to the minimum value of $\{M_{uw}(S(X)), M_{uvw}(S(X')), M^{h_2}_{uw}(S(X))\}$ and the length of $\pi'(u, w)$ is equal to the second minimum value of $\{M_{uw}(S(X)), M_{uvw}(S(X')), M^{h_2}_{uw}(S(X))\}$ since $|\pi(u, w)| \geq |\pi(u, w)|$. Thus, Lemma 8 implies that we can obtain $M^{h_2}_{uw}(S(X'))$ in constant time by comparing values of $\{M_{uw}(S(X)), M_{uvw}(S(X')), M^{h_2}_{uw}(S(X))\}$.

Algorithm 2: Computing the candidate set and the distance matrices in EKG 1.

Procedure $NextC(v, C(S), M(S), M^{h_2}(S), S, k, G)$
  $S' \leftarrow S \cup \{v\}$;
  $C(S') \leftarrow N(v) \setminus (S \cup C(S))$;
  for $u \in C(S)$ **do**
    if $M_{uv}(S) + M^{h_2}_{uv}(S) \geq k$ **then**
      $C(S') \leftarrow C(S') \cup \{u\}$;
  for $u \in C(S')$ **do**
    for $w \in C(S')$ **do**
      $M_{uw}(S') \leftarrow \min(M_{uw}(S), M_{uv}(S) + M_{vw}(S))$;
  for $u \in C(S')$ **do**
    for $w \in C(S')$ **do**
      $p_1 \leftarrow \min(M_{uw}(S), M_{uvw}(S'), M^{h_2}_{uw}(S))$;
      $p_2 \leftarrow$ the second smallest length in $\{M_{uw}(S), M_{uvw}(S'), M^{h_2}_{uw}(S)\}$;
      if $p_1 + p_2 < k$ **then**
        $M^{h_2}_{uw}(S') \leftarrow p_2$;
      else
        $p_1, p_2 \leftarrow \infty, \infty$;
        for $x \in N(w) \cap S'$ **do**
          if $M_{wxu}(S') \leq p_1$ **then**
            $p_2 \leftarrow p_1$;
            $p_1 \leftarrow M_{wxu}(S')$;
        $M^{h_2}_{uw}(S') \leftarrow p_2$;
  return $C(S')$;

We show the detail of update the candidate set and the distance matrices in Algorithm 2. In following, we analyze the time complexity of EKG 1. Let X be any iteration in enumeration tree $\mathcal{T}$, and $T(X)$ be the time complexity of iteration X. In X, it is obvious that EKG 1 outputs a solution in $O(n)$ time. Also, EKG 1 computes the candidate set in $O(n)$ time from Lemma 5 and update $M(S)$ to $M(S')$ in $O(|C(S') \cup S'| |C(S')|)$ time from Lemma 6. The next lemma shows the time complexity for updating $M^{h_2}(S(X))$. Let $ch(X)$ be the set of children X.

Lemma 9. *EKG 1 updates $M^{h_2}(S(X))$ to $M^{h_2}(S(X'))$ in $O\left(\Delta \sum_{X'' \in ch(X')} |ch(X'')| + |C(S(X'))|^2\right)$ time.*

Proof. We consider the following two cases: (A) Suppose that vertices u and w in $C(S(X'))$ satisfies $M_{uw}(S(X')) + M^{h_2}_{uw}(S(X')) \geq k$. Then, from Lemma 7, then $M^{h_2}_{uw}(S(X'))$ can be computed in $O(\Delta)$ time. Moreover, $w \in C(S(X') \cup \{u\})$. Thus, the number of pairs (u, w) that satisfies the above condition is $O\left(\sum_{u \in C(S(X'))} |C(S(X') \cup \{u\}))|\right)$. Hence, EKG 1 needs $O\left(\Delta \sum_{X'' \in ch(X')} |ch(X'')|\right)$ time for computing this part. (B) Suppose that vertices u and w in $C(S(X'))$ satisfies $M_{uw}(S(X')) + M^{h_2}_{uw}(S(X')) < k$. Then, from Lemma 8, $M_{uw}(S(X'))$ and $M^{h_2}_{uw}(S(X'))$ can be computed in constant time. Here, the size of $M^{h_2}(S(X'))$ is equal to $|C(S(X'))|^2$. Hence, the statement holds. □

From Lemma 6, Lemma 7, and Lemma 9, we show that EKG 1 enumerates all solutions in $O(n)$ time per solutions.

Theorem 2. *EKG 1 enumerates all solutions in $O(n)$ time using $O(n^3)$ space.*

Proof. The correctness of EKG 1 can be easily obtained from Lemma 2. We first consider the space complexity of EKG 1. In each iteration, EKG 1 uses $O(n^2)$ space for two distance matrices. In addition, the height of $\mathcal{T}$ is at most n. Therefore, EKG 1 uses $O(n^3)$ space.

We consider the time complexity of EKG 1. Let X and X' be any iterations such that X' is a child of X, and $T(X, X')$ be the time complexity between X and X'. Let $c(X) = |C(S(X))|$ and $c_{sum}(X) =$

Algorithm 3: Computing the candidate set in EKG 2.

```
Procedure NextC(v, C(S), S, k, G)
    Compute all pairs distance between v and S \ {v} using breadth first search;
    S' ← S ∪ {v};
    C(S') ← N(v) \ N[S];
    for u ∈ C(S) \ {v} do
        N' ← N(u) ∩ S';
        p1, p2 ← ∞, ∞;
        for w ∈ N' do
            if dist_G(v, w) ≤ p1 then
                p2 ← p1;
                p1 ← dist_G(v, w);
        if k − 2 ≤ p1 + p2 then
            C(S') ← C(S') ∪ {u};
    return C(S');
```

$\sum_{X' \in ch(X)} |ch(X')|$. From Lemma 5, Lemma 6, and Lemma 7, $T(X) = O(n) + O(c(X')(c(X) + c(X'))) + O\left(\Delta c_{sum}(X') + c(X')^2\right) = O\left(n + c(X)c(X') + \Delta c_{sum}(X') + c(X')^2\right)$ time. Hence the total time of EKG 1 is $O\left(\sum_{(X,X') \in \mathcal{E}} (n + c(X)c(X') + \Delta c_{sum}(X') + c(X')^2)\right)$. The sum of children, grand children, and children of grand children for all iterations is $O(|\mathcal{V}|)$, that is, $O\left(\sum_{X \in \mathcal{V}} (c(X) + c_{sum}(X) + \sum_{X' \in ch(X)} c_{sum}(X')\right) = O(|\mathcal{V}|)$ since each iteration are counted at most three times on $\mathcal{T}$. In addition, for any iteration Y, $O(c(Y)) = O(n)$. Thus, $O\left(\sum_{(X,X') \in \mathcal{E}} (c(X)c(X'))\right) = O(|\mathcal{V}| n)$, and $O\left(\sum_{(X,X') \in \mathcal{E}} (\Delta c_{sum}(X') + c(X')^2)\right) = O(|\mathcal{V}| (\Delta + n))$. Hence, the total time is $O(|\mathcal{V}| n)$. Therefore, The time complexity of EKG 1 per solution is $O(|\mathcal{V}| n / |\mathcal{V}|) = O(n)$. □

4.2 $O(m)$ time $O(m)$ space algorithm

In section 4.1, the algorithm EKG 1 enumerates solutions using distance matrices. However, EKG 1 uses $O(n^3)$ space. We propose a small space algorithm EKG 2. EKG 2 reduces the space using not distance matrices but breadth first search. We show the detail of updating the candidate set in EKG 2 in Algorithm 3.

Let X' be a child of X such that $S(X') = S(X) \cup \{v\}$. From Lemma 4, for any vertices $u, v \in C(S(X))$, to compute the distance and the second distance between u and v, the algorithm tests whether $u \in C(S(X'))$ or not. Here, we consider breadth first search algorithm starting from v. Then, we show the following lemma.

Lemma 10. *Let v and u be distinct vertices in $C(S(X))$, whre . Then, the distance and the second distance between u and v can be computed in in $O(m)$ time by using breath first search algorithm.*

Proof. Since breadth first search algorithm solves single-source shortest path problem [4], the distance between u and v can be computed in $O(m)$ time. We next consider the second distance from u to v. Without loss of generality, we assume that breadth first search starts from v. From Lemma 7, the second distance from u to v is equal to the second smallest length of paths in $\mathcal{P}(X, X', u, w)$, where $\mathcal{P}(X, u, w) = \{\pi(u, x, w) \mid x \in N(u) \cap S(X')\}$ and $\pi(u, x, w)$ is a shortest path from u to w via x. Using the result of breadth first search, the second smallest length in $\mathcal{P}$ can find in $O(d(u))$ time. The sum of all degrees in the graph is $O(m)$ because each edge is counted two. Hence, the statement holds. □

From Lemma 10 and Lemma 4, EKG 2 correctly computes the candidate set in $O(m)$ time. This implies that EKG 2 enumerates all solutions in G from Lemma 2. Finally, we show the following theorem.

Theorem 3. *EKG 2 enumerates all solutions in $O(m)$ time using $O(m)$ space.*

Proof. We can easily see the correctness of EKG 2 from Lemma 2, Lemma 10, and Lemma 4. We next consider the time complexity of EKG 2. In each iteration X, EKG 2 uses breath first search algorithm which runs $O(m)$ time. From Lemma 10, the time complexity of $T(X)$ is $O(m)$ time. Therefore, total

running time of EKG 2 is $O(|\mathcal{V}|\, m)$ time. Hence, EKG 2 outputs all solutions in $O(m)$ time per solution. Finally, we consider the space complexity of EKG 2. For each iteration, EKG 2 needs $O(m)$ space for breath first search algorithm and $O(n)$ space for storing the current solution $S(X)$. Hence, the statement holds. □

5 Conclusion

In this paper, we addressed the k-girth connected induced subgraph enumeration problems. We propose two algorithms EKG 1 and EKG 2. There are incomparable results, EKG 1 is a faster algorithm than EKG 2 but uses much space. On the other hands, EKG 2 uses less space but more time than EKG 1. As interesting future work, we develop more small space algorithm with the same time complexity as EKG 1. We are also interested in an enumeration problem for edge induced subgraphs with large girth. This problem is a generalization of a subtree enumeration problem since if the girth is ∞, then solutions have no cycle.

References

[1] L. S. Chandran and C. Subramanian. Girth and treewidth. *Journal of combinatorial theory, Series B*, 93(1):23–32, 2005.

[2] H.-C. Chang and H.-I. Lu. Computing the girth of a planar graph in linear time. *SIAM Journal on Computing*, 42(3):1077–1094, 2013.

[3] R. Cook. Chromatic number and girth. *Periodica Mathematica Hungarica*, 6(1):103–107, 1975.

[4] T. H. Cormen. *Introduction to algorithms*. MIT press, 2009.

[5] R. Diestel. *Graph Theory*. Grad. Texts in Math. Springer-Verlag, 4th edition, 2010.

[6] H. Djidjev. Computing the girth of a planar graph. *Automata, Languages and Programming*, pages 821–831, 2000.

[7] R. A. Ferreira, R. Grossi, and R. Rizzi. Output-sensitive listing of bounded-size trees in undirected graphs. In *Proc. ESA 2011*, volume 6942 of *LNCS*, pages 275–286. Springer, 2011.

[8] A. Itai and M. Rodeh. Finding a minimum circuit in a graph. *SIAM Journal on Computing*, 7(4):413–423, 1978.

[9] S. Kapoor and H. Ramesh. Algorithms for enumerating all spanning trees of undirected and weighted graphs. *SIAM Journal on Computing*, 24(2):247–265, 1995.

[10] H. Moser and S. Sikdar. The parameterized complexity of the induced matching problem. *Discrete Applied Mathematics*, 157(4):715–727, 2009.

[11] C. H. Papadimitriou. *Automata, Languages and Programming: 24th International Colloquium, ICALP '97 Bologna, Italy, July 7–11, 1997 Proceedings*, pages 2–6. Springer Berlin Heidelberg, 1997.

[12] V. Raman and S. Saurabh. Short cycles make w-hard problems hard: Fpt algorithms for w-hard problems in graphs with no short cycles. *Algorithmica*, 52(2):203–225, 2008.

[13] R. C. Read and R. E. Tarjan. Bounds on backtrack algorithms for listing cycles, paths, and spanning trees. *Networks*, 3(5):237–252, 1975.

[14] A. Shioura, A. Tamura, and T. Uno. An optimal algorithm for scanning all spanning trees of undirected graphs. *SIAM Journal on Computing*, 26(3):678–692, 1997.

[15] K. Wasa, H. Arimura, and T. Uno. Efficient enumeration of induced subtrees in a k-degenerate graph. In *Proc. ISAAC 2014*, volume 8889 of *LNCS*, pages 94–102. Springer, 2014.

[16] K. Wasa, Y. Kaneta, T. Uno, and H. Arimura. Constant time enumeration of bounded-size subtrees in trees and its application. In *Proc. COCOON 2012*, volume 7434 of *LNCS*, pages 347–359. Springer, 2012.