MATHEMATICAL ENGINEERING
TECHNICAL REPORTS

Algorithms for Finding an Induced Cycle in Planar Graphs

Ken-ichi KAWARABAYASHI and Yusuke KOBAYASHI

(Communicated by Kazuo MUROTA)

METR 2008–23 May 2008

DEPARTMENT OF MATHEMATICAL INFORMATICS
GRADUATE SCHOOL OF INFORMATION SCIENCE AND TECHNOLOGY
THE UNIVERSITY OF TOKYO
BUNKYO-KU, TOKYO 113-8656, JAPAN

WWW page: http://www.keisu.t.u-tokyo.ac.jp/research/techrep/index.html

Algorithms for Finding an Induced Cycle in Planar Graphs

Ken-ichi KAWARABAYASHI *   Yusuke KOBAYASHI †

May 2008

Abstract

In this paper, we consider the problem for finding an induced cycle passing through k given vertices, which we call the *induced cycle problem*. The significance of finding induced cycles stems from the fact that precise characterization of perfect graphs would require structures of graphs without an odd induced cycle, and its complement. There has been huge progress in the recent years, especially, the Strong Perfect Graph Conjecture was solved in [6]. Concerning recognition of perfect graphs, there had been a long-standing open problem for detecting an odd hole and its complement, and finally this was solved in [4].

Unfortunately, the problem of finding an induced cycle passing through two given vertices is NP-complete in a general graph [2]. However, if the input graph is constrained to be planar and k is fixed, then the induced cycle problem can be solved in polynomial time [11, 12, 13].

In particular, an $\mathrm{O}(n^2)$ time algorithm is given for the case $k = 2$ by McDiarmid, Reed, Schrijver and Shepherd [14], where n is the number of vertices of the input graph.

Our main results in this paper are to improve their result in the following sense.

1. The number of vertices k is allowed to be non-trivially super constant number, up to $k = \mathrm{o}((\frac{\log n}{\log \log n})^{\frac{2}{3}})$. More precisely, when $k = \mathrm{o}((\frac{\log n}{\log \log n})^{\frac{2}{3}})$, then the ICP can be solved in $\mathrm{O}(n^{2+\varepsilon})$ time for any $\varepsilon > 0$.
2. The time complexity is linear if k is fixed.

We note that the linear time algorithm (the second result) is independent from the first result.

Let us observe that if k is as a part of the input, then the problem is still NP-complete. We need to impose some condition on k.

1 Introduction

1.1 The Induced Cycle Problem

For a graph $G = (V, E)$ and a vertex set $X \subseteq V$ we consider the problem of finding a cycle passing through all vertices in X. The most famous problem of this type is the Hamiltonian path problem, in which $X = V$. It is well-known that the Hamiltonian path problem is NP-complete, and it remains NP-complete even if G is constrained to be planar. On the other hand, if the number of vertices in X is fixed, this problem can be reduced to the disjoint paths problem and is solvable in polynomial time with the aid of the seminal result of Robertson and Seymour's algorithm for the disjoint paths problem [20].

*National Institute of Informatics, Tokyo 101-8430, Japan (E-mail: k_keniti@nii.ac.jp)

†University of Tokyo, Tokyo 113-8656, Japan (E-mail: Yusuke_Kobayashi@mist.i.u-tokyo.ac.jp, Telephone and Fax: +81-3-5841-6924)

In this paper, we focus on the problem for finding an induced cycle through all given vertices, which we call the *induced cycle problem (ICP)*. Here we say that a subgraph $H = (V_H, E_H)$ of G is *induced* if E_H is a set of all edges in E with both ends in V_H.

Induced cycle problem (ICP)

Input: A graph $G = (V, E)$ and a vertex set $X \subseteq V$ with $|X| = k$, whose elements are called *terminals*.

Output: An induced cycle C passing through all vertices in X.

Finding induced cycles has been studied for many years by many researchers, because precise characterization of perfect graphs would require structure of graphs without odd holes and their complements, where induced cycles are called *holes* in this literature. Therefore, detecting an induced cycle is significant in the context of the recognition of the perfect graphs. In particular, it had been a long-standing open question whether or not there is a polynomial time algorithm to test if a graph is perfect. There has been huge progress in recent years. Beside the solution of the Strong Perfect Graph Conjecture [6], detecting either an odd hole or its complement can be done in polynomial time [4], thereby it gives rise to a recognition of perfect graphs. On the other hand, it is unknown whether we can detect induced cycles of odd length in a given graph in polynomial time or not, while detection of induced cycles of even length can be done in polynomial time [5, 7, 8]. In fact, this motivation creates some of work for a similar concept "induced minor", see [9, 10].

Let us now discuss the complexity issues for the ICP. Although we can find a cycle through k specified vertices in polynomial time for fixed k, the ICP in a general graph is NP-complete even if $k = 2$ [2]. However if the given graph is constrained to be planar, McDiarmid, Reed, Schrijver and Shepherd [14] gave the following result.

Theorem 1.1. *Suppose an input graph is planar. Then there is an* $\mathrm{O}(n^2)$ *time algorithm for the ICP with* $k = 2$*, where* n *is the number of vertices of the given graph.*

Our research is motivated by Theorem 1.1. Natural questions arising from the result by McDiarmid, Reed, Schrijver and Shepherd [14] are the following:

1. What if k is as a part of input ?
2. What about general case (namely, fixed constant k) ?
3. Can we get a faster algorithm for $k = 2$ or even general case (for fixed k) ?

Concerning the first question, it is still NP-complete. To see this, suppose G is a planar graph, and $X = V(G)$. We now subdivide each edge once (that is, add a vertex of degree 2 to each edge). Let G' be the resulting graph. Then finding an induced cycle through all the vertices of X in G' corresponds to finding a Hamiltonian cycle in G, which is still NP-complete. Therefore, in order to get a polynomial time algorithm for the first question, we need to impose some condition on the input k.

Concerning the second question, it is known that the ICP in a planar graph can be solved in polynomial time [11, 12, 13].

Theorem 1.2 ([11, 12, 13])**.** *The ICP is solvable in polynomial time when k is fixed and the input graph is planar.*

This theorem comes from polynomial time algorithms for the *induced disjoint paths problem (IDPP)*, which is the induced version of the disjoint paths problem and introduced in [11, 12, 13]. If the number of terminals k is fixed, there exists a polynomial-time reduction from the ICP to the IDPP, and hence the ICP is solvable in polynomial time. However, by a naive reduction algorithm, an instance of the ICP is reduced to $\mathrm{O}(n^{2k})$ instances of the IDPP, and so the time complexity of the algorithm for the ICP seems too expensive, even for $k = 2$.

Therefore, in order to answer the second and the third questions, we need to find a faster and more efficient algorithm.

1.2 Main Results

First we show the following theorem:

Theorem 1.3. *If $k = \mathrm{o}((\frac{\log n}{\log\log n})^{\frac{2}{3}})$, then the ICP can be solved in $\mathrm{O}(n^{2+\varepsilon})$ time for any $\varepsilon > 0$, where n is the number of vertices of the input graph.*

Obviously, Theorem 1.3 generalizes Theorem 1.2 and answers the first and second questions. In fact, we show that the ICP is solvable in $\mathrm{O}(h(k)n^2)$ time and $h(k) = \mathrm{O}(n^{\varepsilon})$ for any $\varepsilon > 0$ when $k = \mathrm{o}((\frac{\log n}{\log\log n})^{\frac{2}{3}})$. Therefore, when k is fixed this algorithm runs in $\mathrm{O}(n^2)$ time, which generalizes Theorem 1.1 and answers the second question.

Corollary 1.4. *If k is fixed, then the time complexity is $\mathrm{O}(n^2)$.*

On the other hand, since the IDPP in planar graphs is solvable in linear time, we expect that the ICP in a planar graph is also solvable in linear time when k is fixed. By this motivation, we give a more efficient algorithm for the ICP when k is fixed:

Theorem 1.5. *The ICP is solvable in linear time when k is fixed and the input graph is planar.*

This theorem is a generalization of Theorem 1.2 and gives the best answer for the third question.

Our proofs of Theorems 1.2 and 1.5 basically follow the same line of the proof of the disjoint paths problem by Robertson and Seymour [20], together with some arguments in [16, 17], which improves the time complexity of the algorithm of the disjoint paths by Robertson and Seymour ($\mathrm{O}(n^3)$ time algorithm) to linear time when an input graph is planar. Since we only consider planar graphs, we do not need the full power of Robertson and Seymour's proof [20] (but we still need a deep topological result in Graph Minors XI [19]). On the other hand, our cycle must be induced, so some of arguments in [20] must be extended to induced paths, which needs much more involved arguments. In addition, we are also interested in the case when k is as a part of the input. Therefore, we need to sharpen the function of k. This needs nontrivial amount of work, since both Robertson-Seymour's proof [20] and Reed, Robertson, Schrijver, Seymour's proof [17] do not care much about sharpening the hidden constant of k. These proofs just guarantees the existence of the function of k, therefore, it is highly expensive, and nonpractical. Our proofs, though, give fairly small function of k. Therefore, we believe that this result may be viewed as a much more practical result. Price to pay is to need to analyze the structure of planar graphs more closely.

The rest of this paper is devoted to proofs for Theorems 1.3 and 1.5. In Sections 2 and 3, we give proofs for Theorems 1.3 and 1.5, respectively.

Most of notations and terminologies used in this paper for planar graphs are described in [15].

2 Polynomial Time Algorithm for the Case $k = o((\frac{\log n}{\log \log n})^{\frac{2}{3}})$

2.1 Deletable Vertex for the ICP

Suppose we are given an instance of the ICP with terminal set $X \subseteq V$. A vertex $v \in V \setminus X$ is called *deletable* if G has a desired induced cycle, then $G - v$ also has one. A vertex $v \in V \setminus X$ is *l-isolated* if there exist l disjoint cycles $C_1, C_2, \ldots, C_l$ and disks $\Delta_1, \Delta_2, \ldots, \Delta_l$ such that C_i bounds Δ_i for $i = 1, \ldots, l$, $v \in \Delta_1 - C_1$, $\Delta_1 \subseteq \Delta_2 \subseteq \cdots \subseteq \Delta_l$, and Δ_l does not intersect X. We say that such $C_1, C_2, \ldots, C_l$ are *nested cycles surrounding v*.

The following theorem gives a sufficient condition for a vertex to be deletable, and plays an important role in our first algorithm for the ICP.

Theorem 2.1. *For any instance of the ICP with k terminals every $(22k + 2)$-isolated vertex is deletable.*

The rest of this subsection is devoted to the proof of Theorem 2.1.

Suppose v is a $(22k+2)$-isolated vertex. Since there exist $22k+2$ disjoint nested cycles, there exist $11k+1$ induced nested cycles surrounding v. We take induced nested cycles $C_1, C_2, \ldots, C_{11k+1}$ such that each disk Δ_t bounded by C_t is as small as possible for $t = 1, \ldots, 11k+1$. More precisely, we assume the following:

Assumption 1. For $t = 1, 2, \ldots, 11k + 1$, there is no cycle $C'_t \neq C_t$ contained in $\Delta_t - \Delta_{t-1}$ such that C'_t does not pass through a vertex adjacent to C_{t-1}, where $\Delta_0 = C_0 = \{v\}$.

By this assumption, we can easily see the following.

Claim 2.2. *Let P be a path whose end vertices are on the outside of Δ_{11k+1}. For $t = 1, 2, \ldots, 11k+1$, each connected component of $P \cap (\Delta_t - C_t)$ intersects C_{t-1} or passes through a vertex adjacent to C_{t-1}, where $C_0 = \{v\}$.*

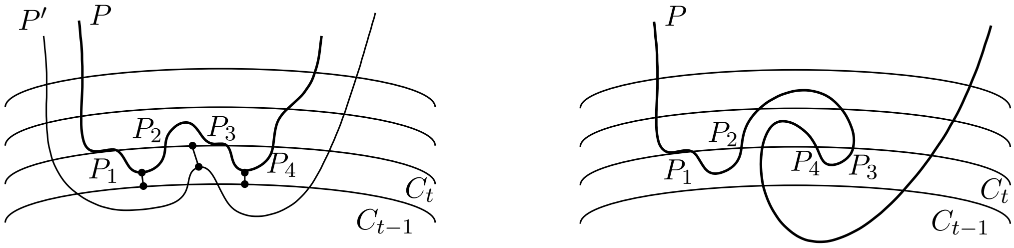

Figure 1: Paths crossing C_t more than twice.

For $t = 1, 2, \ldots, 11k + 1$, we say that a path or a cycle $P = (v_0, e_1, v_1, e_2, \ldots, v_l)$ *crosses* C_t if there exist integers q, r with $0 < q \leq r < l$ such that a subpath $P' = (v_q, e_{q+1}, \ldots, v_r)$ is contained in C_t, e_q and e_{r+1} are not in C_t, and exactly one of e_q and e_{r+1} is in Δ_t. In this case, we say that P crosses C_t at P'.

Let R be an induced cycle through all vertices in X satisfying the following assumption:

Assumption 2. The sum over all t of the number of crossing of R with C_t is as small as possible.

We may assume that R passes through v, because otherwise v is deletable. Now we observe some properties of R.

Claim 2.3. *For $t = 1, 2, \ldots, 11k$, every path of $R \cap \Delta_{11k+1}$ crosses C_t at most twice.*

Proof. We prove the claim by induction on t.

If a path P of $R \cap \Delta_{11k+1}$ crosses C_1 more than twice, $P \cap \Delta_1$ has at least two components. Then, at least one of such components Q does not passes through v, and Q passes through a vertex adjacent to v by Claim 2.2. This contradicts that R is induced and passes through v.

Suppose that a path $P = (v_0, e_1, v_1, e_2, \ldots, v_l)$ of $R \cap \Delta_{11k+1}$ crosses C_t more than twice for $t \geq 2$. Then, P crosses C_t at least four times. We may assume that P crosses C_t at $P_1, P_2, \ldots, P_j$, where $P_i = (v_{q_i}, e_{q_i+1}, \ldots, v_{r_i})$ for $i = 1, \ldots, j$ and $q_1 \leq r_1 < q_2 \leq r_2 < \cdots < q_j \leq r_j$.

Now we consider the positional relationship among P_1, P_2, P_3, and P_4. Since P_3 and P_4 are contained in a same component of $C_t - P_1 - P_2$, there are the following two cases: P_1, P_2, P_3, and P_4 lie on C_t clockwise or counterclockwise in this order, or they lie on C_t clockwise or counterclockwise in the order P_1, P_2, P_4, P_3 (see Fig. 1).

In either case, let Q_1 and Q_2 be subpaths of P between P_1 and P_2, and between P_3 and P_4, respectively. Let Q be the subpath of C_t connecting P_2 and P_3 which does not intersect with P_1. By the assumption that R crosses $C_1, \ldots, C_{11k+1}$ as few as possible, there exists a path $P' \subseteq R \cap \Delta_{11k+1}$ different from P which intersects Q or passes through a vertex adjacent to Q. On the other hand, by Claim 2.2, each of Q_1 and Q_2 intersects C_{t-1} or passes through a vertex adjacent to C_{t-1}. Thus, P' crosses C_{t-1} more than twice by the planarity of G, which completes the proof by induction on t. $\square$

For a path or a cycle L, each connected component B of $L - (\Delta_{11k+1} - C_{11k+1})$ such that $B \cap X = \emptyset$ and $B \not\subseteq C_{11k+1}$ is called a *bridge* of Δ_{11k+1} in L. A *curve* is a subset of Σ which

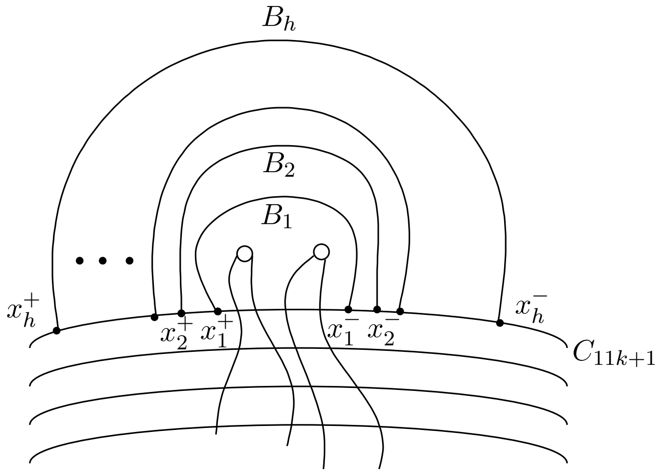

Figure 2: Bridges with the same homotopy type.

is an image of some continuous mapping defined on $[0,1]$. Let B be a bridge of Δ_{11k+1} with end vertices x and y, and J_B be a non-selfintersecting curve connecting x and y whose interior is contained in $\Delta_{11k+1} - C_{11k+1}$. Since $B \cup J_B$ forms a closed curve, the plane is divided into two parts Σ_B^+ and Σ_B^-, which are inside and outside $B \cup J_B$, respectively. Thus, B defines a partition $(X \cap \Sigma_B^+, X \cap \Sigma_B^-)$ of terminals. If $X \cap \Sigma_B^+ = \emptyset$ or $X \cap \Sigma_B^- = \emptyset$, B is said to be *null-homotopic*. We say that two bridges B_1 and B_2 have a same *homotopy type* if they define a same partition of terminals.

Claim 2.4. *No bridge of Δ_{11k+1} in R is null-homotopic.*

Proof. If there exists a null-homotopic bridge, then there exists a bridge B such that B is null-homotopic and Σ_B^+ or Σ_B^- does not contain any edges in R by the planarity of G. Since B cannot be rerouted using edges in C_{11k+1}, by a similar argument as the proof for Claim 2.3, some path of $R \cap \Delta_{11k+1}$ crosses C_{11k} more than twice. This contradicts Claim 2.3. □

Claim 2.5. *At most 5 bridges of Δ_{11k+1} in R have a same homotopy type.*

Proof. Suppose that there exist more than 5 bridges $B_1, B_2, \ldots, B_h$ which have a same homotopy type. We may assume that these bridges $B_1, B_2, \ldots, B_h$ in order from "inside" to "outside", that is, these bridges satisfy that for $j = 1, 2 \ldots, h$ one of $\Sigma_{B_j}^+$ and $\Sigma_{B_j}^-$ contains $B_1, B_2, \ldots, B_{j-1}$ and the other contains $B_{j+1}, B_{j+2}, \ldots, B_h$ (see Fig. 2). Let x_j^+ and x_j^- be end vertices of B_j for $j = 1, \ldots, h$ such that vertices $x_h^+, x_{h-1}^+, \ldots, x_2^+, x_1^+, x_1^-, x_2^-, \ldots, x_h^-$ lie on C_{11k+1} clockwise in this order.

By the planarity of G, there exists subpaths P^+ and P^- of R from x_3^+ to x_4^+ and from x_3^- to x_4^-, respectively. Without loss of generality, we may assume that P^+ does not pass through B_2.

For $t = 1, 2, \ldots, 11k + 1$ and for $x, y \in C_t$, we denote by $C_t[x, y]$ a path traveling from x to y clockwise along C_t. For $i = 1, 2, \ldots, 6$, let T_i be a connected component of $R \cap \Delta_{11k+1}$ containing x_i^-. Since $h \geq 6$, T_2 and T_5 cross C_{11k}, and T_3 and T_4 cross C_{11k-1} by Claim 2.2. For

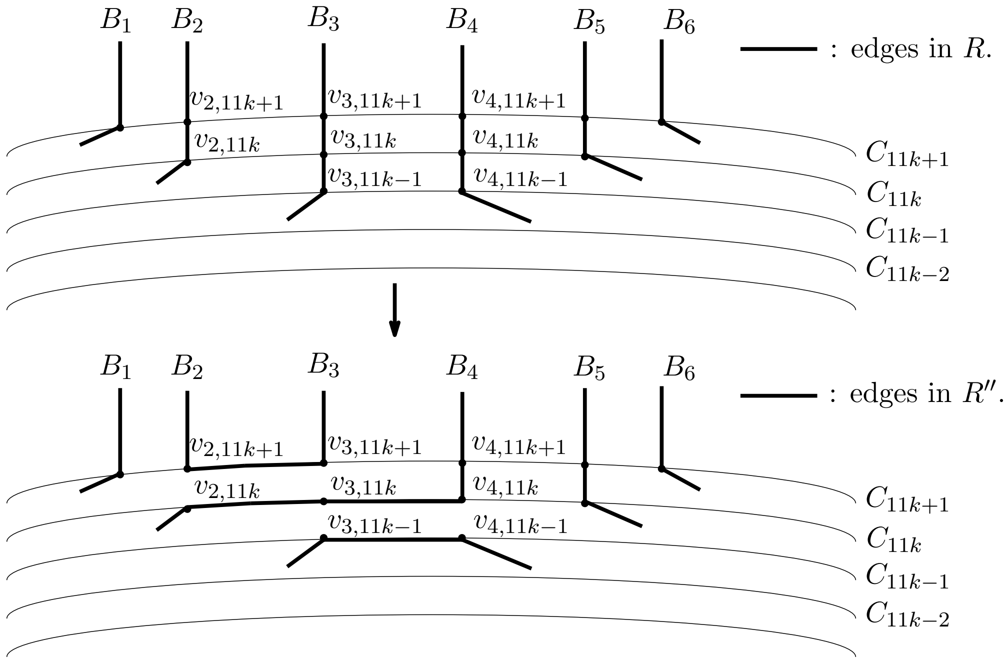

Figure 3: Construction of R''.

$i = 1, 2, \dots, 6$ and for $u, v \in T_i$, let $T_i[u, v]$ denote the subpath of T_i from u to v. For a concise description, we only discuss the case when T_i and C_t intersects only two vertices $u_{i,t}$ and $v_{i,t}$ for $(i, t) = (2, 11k), (2, 11k+1), (3, 11k-1), (3, 11k), (3, 11k+1), (4, 11k-1), (4, 11k), (4, 11k+1)$, where $v_{i,t}$ is nearer to x_i^- than $u_{i,t}$. We note that other cases can be dealt with in a similar way by Claim 2.3.

As shown in Fig. 3, define subgraphs R' and R'' by

$$
\begin{aligned}
R' &= (R - T_3[v_{3,11k-1}, v_{3,11k}] - T_4[v_{4,11k-1}, v_{4,11k}]) \\
&\quad \cup C_{11k}[v_{3,11k}, v_{4,11k}] \cup C_{11k-1}[v_{3,11k-1}, v_{4,11k-1}], \\
R'' &= (R' - T_2[v_{2,11k}, v_{2,11k+1}] - T_3[v_{3,11k}, v_{3,11k+1}]) \\
&\quad \cup C_{11k+1}[v_{2,11k+1}, v_{3,11k+1}] \cup C_{11k}[v_{2,11k}, v_{3,11k}].
\end{aligned}
$$

Then, R' consists of two cycles: one contains $T_2[v_{2,11k}, v_{2,11k+1}]$ and the other contains $T_3[v_{3,11k}, v_{3,11k+1}]$, because P^+ does not contain B_2. Thus, R'' forms a cycle.

If R'' is not induced, i.e. there exist $v_1, v_2 \in R''$ with $(v_1, v_2) \in E - R''$, then we replace a subpath of R'' from v_1 to v_2 containing no terminals by an edge (v_1, v_2). While R'' is not induced, we execute this procedure repeatedly. Then the obtained induced cycle passes through all terminals and crosses $C_1, C_2, \dots, C_{11k+1}$ less than R, which contradicts Assumption 2. □

Proof for Theorem 2.1. Since the graph is planar, the number of homotopy types of bridges is at most $2k$. Thus, by Claims 2.4 and 2.5, $R - (\Delta_{11k+1} - C_{11k+1})$ has at most $5 \cdot 2k + k = 11k$ connected components not contained in C_{11k+1}. Then, $R \cap \Delta_{11k+1}$ has at most $11k$ components not contained in C_{11k+1}. Therefore, $R \cap \Delta_{11k+1}$ can be rerouted using edges in $11k$ cycles $C_1, C_2, \dots, C_{11k}$ so that R passes through all vertices in X and does not pass through v, which means that v is deletable. □

2.2 Tree Width and Algorithm

A *tree-decomposition* of a graph $G = (V, E)$ is a pair $(T, \mathcal{W})$, where $T = (V_T, E_T)$ is a tree and $\mathcal{W} = \{W_t \mid t \in V_T\}$ is a family of subsets of V satisfying the followings:

1. $\bigcup_{t \in V_T} W_t = V$.

2. For every edge $e \in E$ there exists $t \in V_T$ such that W_t contains both ends of e.

3. If $t, t', t'' \in V_T$ and t' lies on the path of T between t and t'', then $W_t \cap W_{t''} \subseteq W_{t'}$.

The *width* of $(T, \mathcal{W})$ is defined as $\max_{t \in V_T}(|W_t| - 1)$, and the *tree-width* of G is the minimum width of a tree-decomposition of G.

Tree-width is a good measure of algorithmic tractability of graphs. It is known that a number of hard problems on graphs, such as “Hamiltonian cycle” and “chromatic number”, can be solved efficiently when the given graph has small tree-width [1].

In [20], Robertson and Seymour gave an $\mathrm{O}(n^2)$ time algorithm for a generalization of the disjoint paths problem called *folio* when the tree-width of the input graph is bounded, where n is the number of vertices of the input graph. Note that although the result in [20] is stated in terms of “branch-width”, it does not cause any problems because branch-width differs only a constant factor from tree-width. An improvement of the running time is shown in [3].

For the ICP, the following theorem holds by using almost the same algorithm as (3.3) and (4.1) in [20]. Here $\mathrm{poly}(x)$ is a polynomial of x and n is the number of vertices of the input graph. Since the proof is the same as in [20], we omit it.

Theorem 2.6. *If the tree-width of a graph G is at most w, then the ICP with k terminals in G is solvable in* $\mathrm{O}(\mathrm{poly}((k+w)^{(k+w)})n^2)$ *time.*

It is known that tree-width of a planar graph is closely related with the size of a grid minor. A $t \times t$ *grid* is a simple graph with t^2 vertices $\{(i, j) \mid 1 \le i, j \le t\}$, where (i, j) and (i', j') are adjacent if $|i - i'| + |j - j'| = 1$. A graph H is a *minor* of a graph G if H can be obtained from a subgraph of G by contracting edges.

Theorem 2.7 ([22])**.** *Let t be a positive integer. If a planar graph G has no $t \times t$ grid minor, then the tree-width of G is at most* $6t - 5$.

On the other hand, the following relation holds between the size of a grid minor and l-isolated vertices.

Lemma 2.8. *Let l be a positive integer. If a planar graph G has k terminals and a $(2l+1)\lfloor\sqrt{k}+1\rfloor \times (2l+1)\lfloor\sqrt{k}+1\rfloor$ grid minor, there exists an l-isolated vertex in G.*

Proof. Suppose that G has a $(2l+1)\lfloor\sqrt{k}+1\rfloor \times (2l+1)\lfloor\sqrt{k}+1\rfloor$ grid minor. Since the number of terminals is k, G has a $(2l+1)\times(2l+1)$ grid minor H enclosing no terminals. More precisely, H is obtained from a subgraph H' of G by contracting edges, and all terminals are contained in the unbounded outer face of H'. Then, the central vertex of H' is an l-isolated vertex in G. □

By combining these results, we obtain the following theorem.

Theorem 2.9. *The ICP with k terminals in a planar graph can be solved in* $\mathrm{O}(\mathrm{poly}(w^w)n^2)$ *time, where* $w=\mathrm{O}(k^{\frac{3}{2}})$.

Proof. We consider the following algorithm which consists of two steps.

Step 1. For a given planar graph G, if G has a $(22k+2)$-isolated vertex $v \in V\setminus X$, then remove v from G. Execute this procedure repeatedly while G has a $(22k+2)$-isolated vertex.

Step 2. We may assume that G has no $(22k+2)$-isolated vertex. By Theorem 2.7 and Lemma 2.8, there exists a constant c_1 such that the tree-width of G is at most $w \le c_1 k^{\frac{3}{2}}$. Then, solved the ICP as shown in Theorem 2.6.

It is obvious that Step 1 does not affect the solution of the ICP by Theorem 2.1. For each vertex $v \in V\setminus X$, we can determine whether there are $22k+2$ nested cycles surrounding v or not in linear time. Thus, Step 1 can be done in $\mathrm{O}(n^2)$ time in total. Step 2 can be done in $\mathrm{O}(\mathrm{poly}(w^w)n^2)$ time by Theorem 2.6, which shows the theorem. □

Now, we are ready to prove Theorem 1.3.

Proof for Theorem 1.3. Let c_1 be an integer such that $w \le c_1 k^{\frac{3}{2}}$ as in Theorem 2.9. When $k=\mathrm{o}((\frac{\log n}{\log\log n})^{\frac{2}{3}})$, it holds that

$$w \le c_1 k^{\frac{3}{2}} = \mathrm{o}\left(\frac{\log n}{\log\log n}\right).$$

Since

$$\log(w^w) = w\log w = \mathrm{o}\left(\frac{\log n}{\log\log n}\log\log n\right) = \mathrm{o}(\log n),$$

we have that $\log(\mathrm{poly}(w^w)) = \mathrm{o}(\log n)$. Thus, for any $\varepsilon > 0$ there exists $N > 0$ such that $\log(\mathrm{poly}(w^w)) < \varepsilon \log n$ for any $n > N$. Then, $\mathrm{poly}(w^w) = \mathrm{O}(n^\varepsilon)$, and so the ICP is solvable in $\mathrm{O}(n^{2+\varepsilon})$ time by Theorem 2.9. □

3 Linear Time Algorithm for the ICP in a Planar Graph

In this section, we give a linear time algorithm that solves the ICP in a planar graph when k is fixed. Our algorithm is inspired by the algorithm in [11, 12] for the IDPP, which is based on the algorithms of [16, 17] for the disjoint paths problem in a planar graph.

3.1 The c-embedded Induced k-linkage-realizations

The ICP is closely related to a problem called *induced disjoint paths problem (IDPP)*. Let G be a graph and $P_1, \ldots, P_k$ be connected subgraphs in G. We say that $P_1, \ldots, P_k$ are *mutually induced* if P_i and P_j have neither common vertices nor adjacent vertices for any distinct i, j. We note that even if $P_1, \ldots, P_k$ are mutually induced, each P_i is not necessarily induced by some vertex set. The induced disjoint paths problem is to find k mutually induced paths $P_1, \ldots, P_k$ such that P_i connects given vertices s_i and t_i for $i = 1, \ldots, k$.

In [11, 12], a new problem called *c-embedded induced k-realizations* is introduced, which is a generalization of the IDPP in a planar graph. Let $G = (V, E)$ be a graph and $X \subseteq V$ be a terminal set. A subpartition $\mathcal{X} = \{X_1, X_2, \ldots, X_p\}$ of X (i.e. a partition of a subset of X) is *induced-realizable* if there are mutually induced trees $T_1, \ldots, T_p$ in G such that $X_i \subseteq T_i$ and $(X \setminus X_i) \cap T_i = \emptyset$ for $i = 1, \ldots, p$. In this case, we say that a subgraph T consisting of $T_1, \ldots, T_p$ *induced-realizes* $\mathcal{X}$. Let Σ be a surface obtained by removing from the plane c open disks whose closures are disjoint. Such a surface is called a *punctured plane* and each disk is called a *cuff*. The boundary of Σ is denoted by $bd(\Sigma)$. The c-embedded induced k-realizations is the following problem:

<u>c-embedded induced k-realizations</u>

Input: A graph $G = (V, E)$ embedded on a punctured plane Σ with at most c cuffs, and a terminal set $X \subseteq V \cap bd(\Sigma)$ with $|X| = k$.

Output: All induced-realizable partitions of X in G.

This problem is known to be solvable in linear time [11, 12], which leads a linear time algorithm for the IDPP in a planar graph when k is fixed.

To give a linear time algorithm for the ICP, we introduce a new problem which we call the *c-embedded induced k-linkage-realizations*. Let $G = (V, E)$ be a graph and $X \subseteq V$ be a terminal set. Let $\mathcal{X} = \{(X_1, s_1, t_1), (X_2, s_2, t_2), \ldots, (X_p, s_p, t_p)\}$ be a set of triples, where $X_1, X_2, \ldots, X_p$ are mutually disjoint subsets of X and s_i and t_i are vertices in X_i (possibly $s_i = t_i$) for $i = 1, 2, \ldots, p$. We say that $\mathcal{X}$ is *induced-linkage-realizable* if there are paths or cycles $L_1, \ldots, L_p$ in G such that $X \cap L_i = X_i$, end vertices of L_i are s_i and t_i for $i = 1, 2, \ldots, p$, where $s_i = t_i$ if L_i is a cycle or a single vertex, and a subgraph L consisting of $L_1, \ldots, L_p$ is induced. In this case, we say that L *induced-linkage-realizes* $\mathcal{X}$. The c-embedded induced k-linkage-realizations is described as follows:

<u>c-embedded induced k-linkage-realizations</u>

Input: A graph $G = (V, E)$ embedded on a punctured plane Σ with at most c cuffs, and a terminal set $X \subseteq V \cap bd(\Sigma)$ with $|X| = k$.

Output: All induced-linkage-realizable sets of triples.

In this section, we show that this problem is solvable in linear time when c and k are fixed.

Theorem 3.1. *The c-embedded induced k-linkage-realizations can be solved in linear time for any fixed c and k.*

When we are given an instance of the ICP in a graph G, there exists a desired induced cycle if and only if $\mathcal{X} = \{(X, s, s)\}$ is induced-linkage-realizable in G for a terminal $s \in X$. Thus, Theorem 1.5 is immediately derived from Theorem 3.1.

3.2 Deletable Vertex

Suppose we are given an instance of the c-embedded induced k-realizations (resp. c-embedded induced k-linkage-realizations). We say that a vertex $v \in V \setminus X$ is *deletable* if any partition (resp. set of triples) $\mathcal{X}$ is induced-realizable (resp. induced-linkage-realizable) in G if and only if $\mathcal{X}$ is induced-realizable (resp. induced-linkage-realizable) in $G - v$.

The following theorem plays an important role in algorithms for the c-embedded induced k-realizations, whereas non-induced version is a part of the main result of [21] and used in the algorithms for the disjoint paths problem [16, 17, 20] (see also [18]).

Theorem 3.2 ([11, 12])**.** *For any k, there exists an integer $f(k)$ such that for any instance of the c-embedded induced k-realizations every $f(k)$-isolated vertex is deletable.*

We present the following theorem for the c-embedded induced k-linkage-realizations, which is similar to Theorems 2.1 and 3.2 and will be used in our algorithm. Our proof for Theorem 3.3 is based on Theorem 3.2, in which the c-embedded induced k-linkage-realizations is reduced to the c-embedded induced k-realizations.

Theorem 3.3. *For any k, there exists an integer $g(k)$ such that for any instance of the c-embedded induced k-linkage-realizations every $g(k)$-isolated vertex is deletable.*

Proof. We show that $g(k) = f(2k) + 1$ satisfies the condition, where f is the function described in Theorem 3.2. Let $v \in V \setminus X$ be a $g(k)$-isolated vertex. It is obvious that if a set of triples $\mathcal{X}$ is induced-linkage-realizable in $G - v$, then $\mathcal{X}$ is induced-linkage-realizable in G.

Suppose that $\mathcal{X}$ is induced-linkage-realizable in G and a subgraph L consisting of p paths $L_1, \dots, L_p$ induced-linkage-realizes $\mathcal{X}$. For $i = 1, \dots, p$, let $(v_{i,1}, v_{i,2}, \dots, v_{i,l_i})$ be a sequence of terminals lying on L_i in this order from $v_{i,1} = s_i$ to $v_{i,l_i} = t_i$. Let $v_{i,j}^-$ (resp. $v_{i,j}^+$) be a vertex adjacent to $v_{i,j}$ in L_i which is between $v_{i,j-1}$ and $v_{i,j}$ (resp. $v_{i,j}$ and $v_{i,j+1}$) for $i = 1, \dots, p$ and for $j = 2, \dots, l_i$ (resp. $j = 1, \dots, l_i - 1$).

Define $X' = \{v_{i,j}^+, v_{i,j+1}^- \mid i = 1, \dots, p,\ j = 1, \dots, l_i - 1\}$ and $G' = G - X - (N(X) - X')$, where $N(X)$ is a set of all vertices adjacent to X. Then, $L - X$ induced-realizes a partition $\mathcal{X}'$ of X' consisting of $\{\{v_{i,j}^+, v_{i,j+1}^-\} \mid i = 1, \dots, p,\ j = 1, \dots, l_i - 1\}$ in G' (see Fig. 4).

Since $|X'| \leq 2k$ and v is an $f(2k)$-isolated vertex in G', by Theorem 3.2, $\mathcal{X}'$ is induced-realizable in $G' - v$. This means that $\mathcal{X}$ is induced-linkage-realizable in $G - v$. □

Note that $v_{i,j}^+$ and $v_{i,j}^-$ in the above proof are unknown when we are given an instance of the c-embedded induced k-linkage-realizations or the ICP. Thus, the above reduction does not lead an efficient algorithm for the c-embedded induced k-linkage-realizations or the ICP.

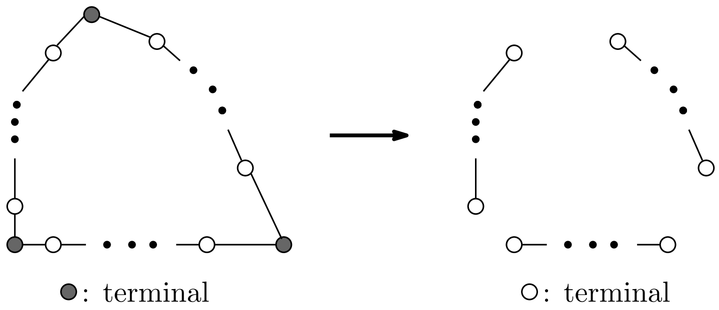

Figure 4: Reduction to c-embedded induced k-realizations.

3.3 Algorithm

Building on the ideas in [11, 12] (see also [16, 17]), we give a linear time algorithm for the c-embedded induced k-linkage-realizations.

For a description of our algorithm for the c-embedded induced k-linkage-realizations, we give some preliminaries. A curve $J \subseteq \Sigma$ is *proper* if $J \cap G \subseteq V$, and its *length* is defined as $|J \cap G|$. An *I-arc* is a proper non-self-intersecting curve in Σ. We say that $J \subseteq \Sigma$ is an *O-arc* if J is a proper non-self-intersecting (except for its end vertices) closed curve in Σ such that each component of $\Sigma - J$ contains a cuff.

When $c \geq 2$, we can transform an instance of the c-embedded induced k-linkage-realizations into some instances with fewer cuffs by executing Algorithm Cuff_Reduction described below.

Algorithm Cuff_Reduction

Input: An instance of the c-embedded induced k-linkage-realizations, where $c \geq 2$.

Output: Some instances of the c'-embedded induced k'-linkage-realizations, where $c' < c$ and k' is at most a constant depending on c and k.

Step 1. If there exists an O-arc J with length at most $4g(k) + 2$ such that each component of $\Sigma - J$ contains at least two cuffs, then consider the inside and the outside of J separately (see Fig. 5). More precisely, let D_1, D_2 be components of $\Sigma - J$, and consider the following two instances: one is in $D_1 \cup J$ with terminals $(X \cap D_1) \cup (J \cap V)$ and the other is in $D_2 \cup J$ with terminals $(X \cap D_2) \cup (J \cap V)$. Then, we can reduce the instance into two instances with fewer cuffs, and stop the algorithm. We note that the solution of the original instance is obtained by unifying the solutions of two small instances in constant time.

If such O-arc does not exist, go to Step 2.

Step 2. If there exists an O-arc J with length at most $4g(k) + 2$ such that one component of $\Sigma - J$ contains exactly one cuff C, then take the shortest one among such O-arcs. If there exist some shortest O-arcs, choose such an O-arc bounding a minimal disk. As the same way as Step 1, we reduce the instance into two instances: one is an instance with two cuffs and

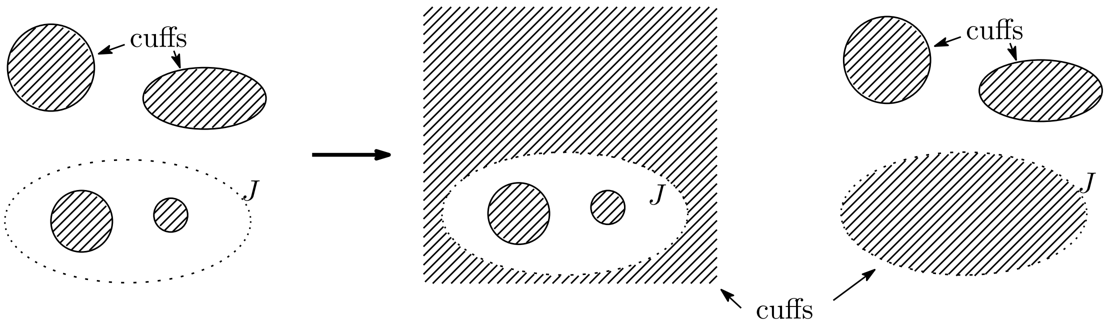

Figure 5: Reduction to instances with fewer cuffs.

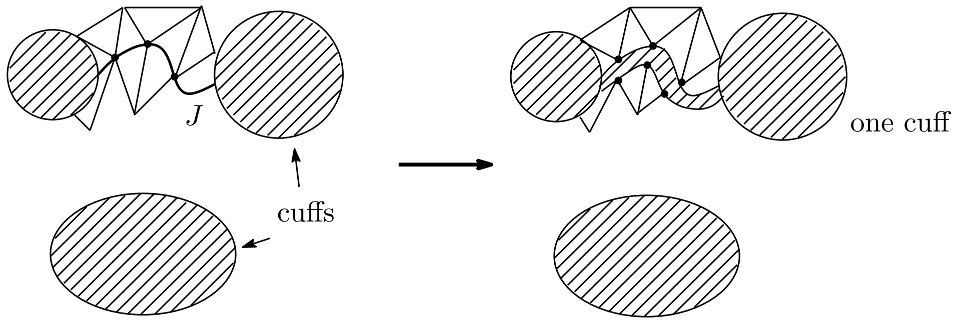

Figure 6: "Open" Σ along J.

the other is an instance with c cuffs in a smaller graph. For each obtained graph, execute Step 2 repeatedly, and if such O-arc does not exist in every graph, then execute Step 3 for each resulting graph.

Step 3. It suffices to consider the case when there is no O-arc with length at most $4g(k)+2$. Denote the cuffs by $C_1, \dots, C_c$, and find the shortest I-arc $J_{i,j}$ connecting C_i and C_j for distinct $1 \leq i, j \leq c$. Let J be the shortest I-arc among all $J_{i,j}$.

3-1. If the length of J is at most $2g(k)+2$, then "open" Σ along J and reduce the instance into an instance with $c-1$ cuffs (see Fig. 6). More precisely, for each vertex v on J, split v into two vertices v_1, v_2 and replace every edge vu incident to v by v_1u or v_2u so that J is contained in a new face. Furthermore, add all vertices in $\{v_1, v_2 \mid v \in J\}$ to terminals. Then, the instance is reduced into an instance with $c-1$ cuffs, and stop the algorithm.

3-2. If the length of J is more than $2g(k)+2$, delete all vertices of J except the first $g(k)+1$ and the last $g(k)+1$. Then, since the length of J becomes $2g(k)+2$, execute the same procedure as Step 3-1.

To see the correctness of Algorithm Cuff_Reduction, we prove that all vertices deleted in Step 3-2, say Q, are deletable.

Proposition 3.4. *Deleting Q does not affect induced-linkage-realizability of the original instance.*

For a proof of this proposition, we use the following characterization of l-isolated vertices. For a vertex v in a graph G, let $d_G(v)$ denote the minimum number of vertices of G on the interior of an I-arc J, where J is taken over all I-arcs of Σ with one endpoint v and the other in $bd(\Sigma)$. Then, the following theorem holds.

Theorem 3.5 ([16, 19])**.** *Suppose that G has no O-arc with length at most $2l$ for some positive integer l. Then, a vertex v is l-isolated if and only if $d_G(v) \geq l$.*

Now we are ready to show Proposition 3.4. Note that when we execute Step 3-2, we assume that the graph has no O-arc with length at most $4g(k) + 2$ and J is the shortest I-arc with its endpoints in $bd(\Sigma)$.

Proof for Proposition 3.4. By Theorem 3.3, it suffices to show that each vertex v is $g(k)$-isolated in $G - (Q \setminus \{v\})$.

First, we show that $G - Q$ has no O-arc with length at most $2g(k)$. Let C be an O-arc in $G - Q$ with minimum length. We may assume that C intersects G only in its vertices. Note that, by this assumption, C can also be regarded as an O-arc in G which might passes through some vertices in Q. Then, there exist $x, y \in Q$ such that two components K, K' of $C - \{x, y\}$ satisfy that $K \cap V \subseteq Q$ and $K' \cap Q = \{x, y\}$. Since J is the shortest I-arc, the length of K is less than or equal to that of K'. Thus, if the length of C is at most $2g(k)$ in $G - Q$, then C is an O-arc in G with length at most $4g(k) + 2$, which contradicts the assumption.

On the other hand, as J is the shortest I-arc, we can see that $d_{G-(Q\setminus\{v\})}(v) \geq g(k)$ holds for each vertex $v \in Q$.

Thus, by Theorem 3.5, each vertex $v \in Q$ is $g(k)$-isolated in $G - (Q \setminus \{v\})$. □

Proposition 3.6. *Algorithm Cuff_Reduction runs in linear time.*

Proof. In Steps 1 and 2, by using the augmenting path method of Ford and Fulkerson in G, we can find O-arcs with length at most $4g(k) + 2$ in linear time. We note that since the number of repetition of Step 2 is at most $|V|$, the total number of vertices increases by at most $2(4g(k)+2)|V|$.

In Step 3, by adding cuffs to Σ, we regard $G = (V, E)$ as a graph embedded on a plane. Let $\mathcal{F}$ be a face set of G and $F_1, F_2, \ldots, F_c \in \mathcal{F}$ be faces containing cuffs $C_1, C_2, \ldots, C_c$, respectively. We consider an auxiliary graph whose vertex set is $V \cup \mathcal{F}$ and whose edge set is

$$\{(v, F) \mid v \in V, F \in \mathcal{F}, v \text{ is on the boundary of } F\}.$$

Then, by finding the shortest path from F_i to F_j in the auxiliary graph for each $1 \leq i, j \leq c$, we can find J. Since the number of vertices in the auxiliary graph is $|V| + |\mathcal{F}| \leq 3|V| - 4$, it can be done in linear time. □

We note that since the number of vertices increases by at most $2(2g(k) + 2)$ in Step 3, by executing Algorithm Cuff_Reduction, the total number of vertices of obtained graphs is at most $|V| + (2(4g(k) + 2) + 2(2g(k) + 2))|V| = (12g(k) + 9)|V|$. Thus, by repeating Algorithm Cuff_Reduction, we can reduce the original instance into some instances with one cuff in linear time, and the total number of vertices is at most a constant multiple of $|V|$. Although the original instance may be reduced into $O(|V|)$ instances, the running time is $O(|V|)$ in total, because the total number of vertices is at most $O(|V|)$.

When $c = 1$, we can determine whether a given set of triples $\mathcal{X} = \{(X_1, s_1, t_1), (X_2, s_2, t_2), \ldots, (X_p, s_p, t_p)\}$ is induced-linkage-realizable or not in linear time. It can be done by finding a path from s_i to t_i passing through all vertices in X_i as close to the cuff as possible for each i, and we omit the detail of the algorithm.

As a consequence of the above arguments, we can solve the c-embedded induced k-linkage-realizations in linear time, which completes Theorem 3.1.

References

[1] S. Arnborg and A. Proskurowski: Linear time algorithms for NP-hard problems restricted to partial k-trees, *Discrete Applied Math.*, **23** (1989), pp. 11-24.

[2] D. Bienstock: On the complexity of testing for even holes and induced odd paths, *Discrete Math.*, **90** (1991), pp. 85–92.

[3] H. L. Bodlaender: A linear time algorithm for finding tree-decompositions of small treewidth, *SIAM Journal on Computing*, **25** (1996), pp. 1305–1317.

[4] M. Chudnovsky, G. Cornuéjols, X. Liu, P. D. Seymour and K. Vuskovic: Recognizing Berge graphs, *Combinatorica*, **25** (2005), pp. 143–186.

[5] M. Chudnovsky, K. Kawarabayashi and P. D. Seymour: Detecting even holes, *Journal of Graph Theory*, **48** (2005), pp. 85–111.

[6] M. Chudnovsky, N. Robertson, P. D. Seymour and R. Thomas: The strong perfect graph theorem, *Annals of Mathematics*, **64** (2006), pp. 51–219.

[7] G. Cornuéjols, M. Conforti, A. Kapoor and K. Vusksvic: Even-hole-free graphs. I. Decomposition theorem, *Journal of Graph Theory*, **39** (2002), pp. 6–49.

[8] G. Cornuéjols, M. Conforti, A. Kapoor and K. Vusksvic: Even-hole-free graphs. II. Recognition algorithm, *Journal of Graph Theory*, **40** (2002), pp. 238–266.

[9] M. R. Fellows: The Robertson-Seymour Theorems: a survey of applications, *Comtemporary Mathematics* 89, American Mathematical Society, 1987, pp. 1–18.

[10] M. R. Fellows, J. Kratochvil, M. Middendorf and F. Pfeiffer: The complexity of induced minors and related problems, *Algorithmica*, **13** (1995), pp. 266–282.

[11] K. Kawarabayashi and Y. Kobayashi: A linear time algorithm for the induced disjoint paths problem in planar graphs, manuscript.

[12] K. Kawarabayashi and Y. Kobayashi: The induced disjoint paths problem, *Proceedings of the 13th Integer Programming and Combinatorial Optimization Conference*, 2008, to appear.

[13] Y. Kobayashi: An extension of the disjoint paths problem, *METR 2007-14*, Department of Mathematical Informatics, University of Tokyo (2006).

[14] C. McDiarmid, B. Reed, A. Schrijver and B. Shepherd: Induced circuits in planar graphs. *Journal of Combinatorial Theory Ser. B*, **60** (1994), pp. 169–176.

[15] B. Mohar and C. Thomassen: *Graphs on Surfaces*, The Johns Hopkins University Press, Baltimore and London, 2001.

[16] B. Reed: Rooted routing in the plane, *Discrete Applied Math.*, **57** (1995), pp. 213–227.

[17] B. A. Reed, N. Robertson, A. Schrijver and P. D. Seymour: Finding disjoint trees in planar graphs in linear time, *Contemporary Mathematics* 147, American Mathematical Society, 1993, pp. 295–301.

[18] N. Robertson and P. D. Seymour: Graph minors. VII. Disjoint paths on a surface, *Journal of Combinatorial Theory Ser. B*, **45** (1988), pp. 212–254.

[19] N. Robertson and P. D. Seymour: Graph minors. XI. Circuits on a surface, *Journal of Combinatorial Theory Ser. B*, **60** (1994), pp. 72–106.

[20] N. Robertson and P. D. Seymour: Graph minors. XIII. The disjoint paths problem, *Journal of Combinatorial Theory Ser. B*, **63** (1995), pp. 65–110.

[21] N. Robertson and P. D. Seymour: Graph minors. XXII. Irrelevant vertices in linkage problems, manuscript.

[22] N. Robertson, P. D. Seymour, and R. Thomas: Quickly excluding a planar graph, *Journal of Combinatorial Theory Ser. B*, **62** (1994), pp. 323–348.