

Algorithms for Finding an Induced Cycle in Planar Graphs and Bounded Genus Graphs

Yusuke KOBAYASHI *

Ken-ichi KAWARABAYASHI †

Abstract

In this paper, we consider the problem of 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 [13, 14, 16].

In particular, an $O(n^2)$ time algorithm is given for the case $k = 2$ by McDiarmid, Reed, Schrijver and Shepherd [18], 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 $k = o\left(\left(\frac{\log n}{\log \log n}\right)^{\frac{2}{3}}\right)$. More precisely, when $k = o\left(\left(\frac{\log n}{\log \log n}\right)^{\frac{2}{3}}\right)$, then the ICP in planar graphs can be solved in $O(n^{2+\varepsilon})$ time for any $\varepsilon > 0$.
2. The time complexity is linear if the given graph is planar and k is fixed.
3. The above results are extended to graphs embedded in a fixed surface.

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

Let us point out that we give the first polynomial time algorithm for the problem for the bounded genus case. In fact, our proof gives a short proof of a result announced in [20] (without complete proof) which gives a linear time algorithm for the disjoint paths problem for fixed k for the bounded genus case. We also extend this result to the induced disjoint paths problem.

*University of Tokyo, Tokyo 113-8656, Japan (E-mail: Yusuke.Kobayashi@mist.i.u-tokyo.ac.jp, Telephone and Fax: +81-3-5841-6924) Supported by JSPS Research Fellowships for Young Scientists. This study was partially supported by the GCOE "The research and training center for new development in mathematics".

†National Institute of Informatics, Tokyo 101-8430, Japan (E-mail: k.keniti@nii.ac.jp) Research partly supported by JSPS Postdoctoral Fellowship for Research Abroad.

Let us observe that if k is as a part of the input, then the problem is still NP-complete, and so 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 cycle problem, in which $X = V$. It is well-known that the Hamiltonian cycle 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 [24].

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 of length more than 3 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 odd holes 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 [11, 12].

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 , Bienstock [2] showed that the ICP in a general graph is NP-complete even if $k = 2$. See also [17] for more results on the ICP in a general graph. However if the given graph is constrained to be planar, McDiarmid, Reed, Schrijver and Shepherd [18] gave the following result.

THEOREM 1.1. *Suppose an input graph is planar. Then there is an $O(n^2)$ 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 [18] 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) ?
4. What if the graph is not planar, but embedded in a fixed surface ?

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 [13, 14, 16].

THEOREM 1.2. ([13, 14, 16]) *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 (DPP) and introduced in [13, 14, 16]. Let G be a

graph and P_1, \dots, P_k be connected subgraphs in G . We say that P_1, \dots, 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, \dots, 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, \dots, P_k such that P_i connects given vertices s_i and t_i for $i = 1, \dots, k$. 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 $O(n^{2k})$ 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.

Concerning the fourth question, we note that all algorithms in [13, 14, 16, 18] use a lot of nice properties of planar graphs, and they do not deal with graphs embedded in a fixed surface. An algorithm for the (non-induced) disjoint paths problem in a fixed surface is considered in [20].

1.2 Main Results

Our first result is the following:

THEOREM 1.3. *If $k = o((\frac{\log n}{\log \log n})^{\frac{2}{3}})$, then the ICP in planar graphs can be solved in $O(n^{2+\varepsilon})$ 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 $O(h(k)n^2)$ time and $h(k) = O(n^\varepsilon)$ for any $\varepsilon > 0$ when $k = o((\frac{\log n}{\log \log n})^{\frac{2}{3}})$. Therefore, when k is fixed this algorithm runs in $O(n^2)$ time, which generalizes Theorem 1.1 and answers the second question.

COROLLARY 1.1. *If k is fixed, then the time complexity is $O(n^2)$.*

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 1.4. *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.

We also show that the above results can be extended to graphs embedded in a fixed surface. These results generalize Theorems 1.3 and 1.4, and answer the fourth question.

THEOREM 1.5. *Let G be a graph with n vertices embedded in a fixed surface. If $k = o((\frac{\log n}{\log \log n})^{\frac{2}{3}})$, then the ICP in G can be solved in $O(n^{2+\varepsilon})$ time for any $\varepsilon > 0$.*

THEOREM 1.6. *The ICP is solvable in linear time when k is fixed and the input graph is embedded in a fixed surface.*

Let us point out that we give the first polynomial time algorithm for the ICP for the bounded genus case.

In fact, our proof of Theorem 1.6 has several appealing features. It gives the following linear time algorithm:

COROLLARY 1.2. *The induced disjoint paths problem is solvable in linear time when k is fixed and the input graph is embedded in a fixed surface.*

Reed [20] announced without complete proof that a linear time algorithm for the disjoint paths problem when k is fixed and the input graph is planar [20, 21], can be extended to the bounded genus case. Corollary 1.2 certainly implies this result. In addition, our proof is short with the aid of the recent result in [15]. For details, we refer the reader to Section 5.

Our proofs of Theorems 1.3, 1.4, 1.5, and 1.6 basically follow the same lines as the proof of the disjoint paths problem by Robertson and Seymour [24], together with some arguments in [20, 21], which improves the time complexity of the algorithm of the disjoint paths by Robertson and Seymour ($O(n^3)$ time algorithm) to linear time when an input graph is planar. Since we only consider planar graphs and graphs embedded in a fixed surface, we do not need the full power of Robertson and Seymour's proof [24] (but we still need a deep topological result in Graph Minors XI [23]). On the other hand, our cycle must be induced, so some of arguments in [24] 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 [24] and Reed, Robertson, Schrijver, Seymour's proof [21] do not care much about sharpening the hidden constant of k . These proofs just guarantee 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 and graphs embedded in a fixed surface more closely.

Let us emphasize that, previously, there is no known polynomial time algorithm for the problems when the input graph is a bounded genus graph. Our proof requires recent progress in this area [15]. In fact, in order to get a linear time algorithm for Corollary 1.2, we need to extend the methods in [20, 21] to the bounded genus case, which needs deep topological understandings of graphs on a fixed surface [10, 19, 23]. Therefore, the bounded genus case is a non-trivial extension of the planar case.

The rest of this paper is devoted to proofs for Theorems 1.3, 1.4, 1.5, and 1.6. In Sections 2 and 3, we give proofs for Theorems 1.3 and 1.5, respectively, and proofs for Theorems 1.4 and 1.6 are given in Sections 4 and 5.

Most of notations and terminologies used in this paper for graphs embedded in a fixed surface are described in [19].

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

In this section, we give a proof for Theorem 1.3.

2.1 Deletable Vertex for the ICP Suppose we are given an instance of the ICP in a planar graph $G = (V, E)$ with terminal set $X \subseteq V$. A vertex $v \in V \setminus X$ is *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, \dots, C_l and disks $\Delta_1, \Delta_2, \dots, \Delta_l$ such that C_i bounds Δ_i for $i = 1, \dots, l$, $v \in \Delta_1 - C_1$, $\Delta_1 \subseteq \Delta_2 \subseteq \dots \subseteq \Delta_l$, and Δ_l does not intersect X . We say that such C_1, C_2, \dots, 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 in a planar graph with k terminals every $(22k + 2)$ -isolated vertex is deletable.*

This is one of our main contributions, and we give a sketch of the proof.

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

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

CONDITION 2.1. *The sum over all t of the number of crossing of R with C_t is as small as possible.*

Each connected component B of $R - (\Delta_{11k+1} - C_{11k+1})$ such that $B \cap X = \emptyset$ and $B \not\subseteq C_{11k+1}$ is called a *bridge* of Δ_{11k+1} in R . A *curve* is a subset of the plane which is the 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-self-intersecting 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.

Then, we can show the following lemma, but we omit the proof.

LEMMA 2.1. *No bridge of Δ_{11k+1} in R is null-homotopic, and at most 5 bridges of Δ_{11k+1} in R have a same homotopy type.*

By this lemma we show Theorem 2.1.

Proof for Theorem 2.1. Since the graph is planar, the number of homotopy types of bridges is at most $2k$. Thus, by Lemma 2.1, $R - (\Delta_{11k+1} - C_{11k+1})$ 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 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 [24], Robertson and Seymour gave an $O(n^2)$ 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 [24] 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 [24]. Here $\text{poly}(x)$ is a polynomial of x and n is the number of vertices of the input graph. Since the proof is the same as in [24], we omit it.

THEOREM 2.2. *If the tree-width of a graph G is at most w , then the ICP with k terminals in G is solvable in $O(\text{poly}((k+w)^{(k+w)}n^2))$ 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 $\{(i, j) \mid 1 \leq i, j \leq t\}$, 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 2.3. ([26]) *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 2.2. *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 2.4. *The ICP with k terminals in a planar graph can be solved in $O(\text{poly}(w^w)n^2)$ time, where $w = O(k^{\frac{3}{2}})$.*

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 2.3 and Lemma 2.2, there exists a constant c_1 such that the tree-width of G is at most $w \leq c_1 k^{\frac{3}{2}}$. Then, solved the ICP as shown in Theorem 2.2.

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 $O(n^2)$ time in total. Step 2 can be done in $O(\text{poly}(w^w)n^2)$ time by Theorem 2.2, 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 $w \leq c_1 k^{\frac{3}{2}}$ as in Theorem 2.4. When $k = o(\left(\frac{\log n}{\log \log n}\right)^{\frac{2}{3}})$, it holds that

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

Since

$$\log(w^w) = w \log w = o(\log n),$$

we have that $\log(\text{poly}(w^w)) = o(\log n)$. Thus, for any $\varepsilon > 0$ there exists $N > 0$ such that $\log(\text{poly}(w^w)) < \varepsilon \log n$ for any $n > N$. Then, $\text{poly}(w^w) = O(n^\varepsilon)$, and so the ICP is solvable in $O(n^{2+\varepsilon})$ time by Theorem 2.4.

3 Extension to a Fixed Surface

In this section, we give an algorithm for the ICP for a graph embedded in a fixed surface which is an extension of the algorithm in the previous section, and give a proof of Theorem 1.5. See [19] for notations and basic tools for graphs on surfaces, which will be used in this section.

A *surface* is a compact connected 2-manifold without boundary. We define the Euler genus of a surface Σ as $2 - \chi(\Sigma)$, where $\chi(\Sigma)$ is the Euler characteristic of Σ . Let G be a graph that is embedded in a surface Σ . To simplify the notation we do not distinguish between a vertex of G and the point of Σ used to represent the vertex, and we do not distinguish between an edge of G and the non-self-intersecting curve of Σ representing it.

Let Σ be a surface of Euler genus $g \geq 1$. Then, it is known that Σ can be obtained from a 2-sphere by adding $\frac{g}{2}$ handles or g cross-caps. For a graph embedded in a surface, we can define *l-isolated* vertices and *nested cycles surrounding* a vertex v in the same way as planar graphs.

To give an algorithm for the ICP in a graph embedded in a fixed surface, we use the following theorem, which is an extension of Theorem 2.1.

THEOREM 3.1. *Let G be a graph embedded in a surface of Euler genus g . For any instance of the ICP with k terminals in G every $2f(k, g)$ -isolated vertex is deletable, where $f(k, g) = 5(8g + 2k) + k + 1$.*

In order to prove this theorem, suppose v is a $2f(k, g)$ -isolated vertex. Since there exist $2f(k, g)$ disjoint nested cycles, there exist $f(k, g)$ mutually induced nested cycles surrounding v . We take mutually induced nested cycles $C_1, C_2, \dots, C_{f(k, g)}$ such that each disk Δ_t bounded by C_t is as small as possible for $t = 1, \dots, f(k, g)$. Let R be an induced cycle through all vertices in X satisfying Condition 2.1.

Let B_1, \dots, B_M be bridges of $\Delta_{f(k, g)}$ in R . Now we define concepts of “null-homotopic bridges” and “homotopy types of bridges” when the graph is not planar but is embedded in a fixed surface. Note that the definitions below are consistent with those for planar graphs described in Section 2.1. We say that a closed curve C is *null-homotopic* if it is contractible and the disk bounded by it contains no terminals. Let B be a bridge of $\Delta_{f(k, g)}$ with end vertices x and y , and J_B be a non-self-intersecting curve connecting x and y whose interior is contained in $\Delta_{f(k, g)} - C_{f(k, g)}$. We say that B is *contractible* if a closed curve $B \cup J_B$ is contractible, and B is said to be *null-homotopic* if $B \cup J_B$ is null-homotopic. Let B_1 and B_2 be bridges of $\Delta_{f(k, g)}$ with end vertices x_1, y_1 and x_2, y_2 , and let J_1 and J_2 be disjoint non-self-intersecting curves from $\{x_1, y_1\}$ to $\{x_2, y_2\}$ whose interiors are contained in $\Delta_{f(k, g)} - C_{f(k, g)}$. We say that B_1 and B_2 have a same *homotopy type* if $B_1 \cup B_2 \cup J_1 \cup J_2$ forms a null-homotopic closed curve. Note that this relation is an equivalence relation.

Then, we can show the following lemmas, whose proofs are omitted.

LEMMA 3.1. *No bridge of $\Delta_{f(k, g)}$ in R is null-homotopic, and at most 5 bridges of $\Delta_{f(k, g)}$ in R have a same homotopy type.*

LEMMA 3.2. *The number of homotopy types of bridges is at most $8g + 2k$.*

By the same argument for Theorem 2.1, we obtain Theorem 3.1 with the aid of these lemmas.

For a graph embedded in a fixed surface, the following relationship between the tree-width and sizes of grid minors is known.

THEOREM 3.2. ([9]) *Let t be a positive integer and G be a graph embedded in a surface of Euler genus g . If G has no $t \times t$ grid minor, then the tree-width of G is at most $4t(g + 1)$.*

In order to see the relationship between sizes of grid minors and isolated vertices, we use the concept of *good* subgraphs. Let H be a subgraph of G . An *H*-component is a subgraph of G which is either an edge not in H but with both ends in H (and its ends in H also belong to the component), or a connected component Q' of $G - V(H)$ together with all edges from Q' to H (and their ends in H). Let C_H be the outer cycle of H . We say that H is *good* in G if the union of H and all those H -components which intersect $H - V(C_H)$ is planar. Then we can derive the following theorem from the result in [27], which is stated in terms of “hexagonal grid” instead of the grid.

THEOREM 3.3. ([27]) *Let t be a positive integer and G be a graph embedded in a surface of Euler genus g . If G has no good subgraph which is a subdivision of the $t \times t$ grid, then G has no $400t\sqrt{g} \times 400t\sqrt{g}$ grid minor.*

By combining Theorems 3.2 and 3.3, we obtain the following theorem.

THEOREM 3.4. *Let t be a positive integer and G be a graph embedded in a surface of Euler genus g . If G has no good subgraph which is a subdivision of the $t \times t$ grid, then the tree-width of G is at most $O(tg^{3/2})$.*

By Theorems 3.1 and 3.4, using the same arguments in Section 2.2, we obtain Theorem 1.5.

4 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 [13, 14] for the IDPP, which is based on the algorithms of [20, 21] for the disjoint paths problem in a planar graph.

4.1 The c -embedded Induced k -linkage-realizations The ICP is closely related to a problem called *induced disjoint paths problem (IDPP)*. As described in Section 1.1, the induced disjoint paths problem is to find k mutually induced paths P_1, \dots, P_k such that P_i connects given vertices s_i and t_i for $i = 1, \dots, k$.

In [13, 14], 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 $\mathcal{X} =$

$\{X_1, X_2, \dots, X_p\}$ of X (i.e. a partition of a subset of X) is *induced-realizable* if there are mutually induced trees T_1, \dots, T_p in G such that $X_i \subseteq T_i$ and $(X \setminus X_i) \cap T_i = \emptyset$ for $i = 1, \dots, p$. In this case, we say that a subgraph T consisting of T_1, \dots, T_p *induced-realizes* \mathcal{X} . A *punctured plane* is a 2-manifold obtained by removing from the plane c open disks whose closures are disjoint, 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:

c -embedded induced k -realizations

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 [13, 14], 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 $\mathcal{X} = \{(X_1, s_1, t_1), (X_2, s_2, t_2), \dots, (X_p, s_p, t_p)\}$ be a set of triples, where X_1, X_2, \dots, 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 $i = 1, 2, \dots, p$. We say that \mathcal{X} is *induced-linkage-realizable* if there are paths or cycles L_1, \dots, L_p in G such that $X \cap L_i = X_i$, end vertices of L_i are s_i and t_i for $i = 1, 2, \dots, p$, where $s_i = t_i$ if L_i is a cycle or a single vertex, and a subgraph L consisting of L_1, \dots, 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:

c -embedded induced k -linkage-realizations

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 4.1. *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 1.4 is immediately derived from Theorem 4.1.

4.2 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 [25] and used in the algorithms for the disjoint paths problem [20, 21, 24] (see also [22]).

THEOREM 4.2. ([13, 14]) *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 4.2 and will be used in our algorithm. Our proof for Theorem 4.3 is based on Theorem 4.2, in which the c -embedded induced k -linkage-realizations is reduced to the c -embedded induced k -realizations.

THEOREM 4.3. *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 4.2. Let $v \in V \setminus X$ be a $g(k)$ -isolated vertex.

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' .

Since $|X'| \leq 2k$ and v is an $f(2k)$ -isolated vertex in G' , by Theorem 4.2, \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.

4.3 Algorithm Building on the ideas in [13, 14] (see also [20, 21]), 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. Note that the solution of the original instance can be obtained from the solutions of the new instances with fewer cuffs.

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 of 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. 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 of 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 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 of 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. 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.

PROPOSITION 4.1. *Algorithm Cuff_Reduction runs correctly in linear time.*

We can show this proposition by the same arguments in [13, 14, 20, 21], and so we omit the detail. We only mention that in order to see the correctness of Step 3-2, we use the following characterization of l -isolated vertices.

THEOREM 4.4. ([20, 23]) *Suppose that G has no O-arc of length at most $2l$ for some positive integer l . Then, a vertex v is l -isolated if and only if $d_G(v) \geq l$.*

Here $d_G(v)$ is 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 $bd(\Sigma)$.

When $c = 1$, we can determine whether $\mathcal{X} = \{(X_1, s_1, t_1), (X_2, s_2, t_2), \dots, (X_p, s_p, t_p)\}$ 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 .

As a consequence of the above arguments, we can solve the c -embedded induced k -linkage-realizations in linear time, which completes Theorem 4.1.

5 Extension to a Fixed Surface

In this section, we give a linear time algorithm for the ICP in a fixed surface, and prove Theorem 1.6, which is an extension of Theorem 1.4. To show Theorem 1.6, we introduce an extension of the c -embedded induced k -linkage-realizations. In the same way as a punctured

plane, a *punctured surface* is a 2-manifold obtained by removing from a surface c open disks whose closures are disjoint, and each disk is called a *cuff*.

(c, g) -embedded induced k -linkage-realizations

Input: A graph $G = (V, E)$ embedded in a punctured surface Σ which is obtained from a surface of Euler genus at most g by removing 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.

We show that the (c, g) -embedded induced k -linkage-realizations can be solved in linear time, if c, g and k is fixed.

THEOREM 5.1. *The (c, g) -embedded induced k -linkage-realizations can be solved in linear time for any fixed c, g and k .*

This theorem is a generalization of Theorem 4.1, and Theorem 1.6 follows from this theorem. The rest of this section is devoted to a proof of Theorem 5.1. We use the following theorem which generalizes Theorem 4.3.

THEOREM 5.2. *For any k and g , there exists an integer $h(k, g)$ such that for any instance of the (c, g) -embedded induced k -linkage-realizations every $h(k, g)$ -isolated vertex is deletable.*

The proof is given by Lemma 3.2 and the arguments for Theorems 4.2 and 4.3, and we omit the detail.

Our algorithm for the (c, g) -embedded induced k -linkage-realizations is similar to that for the c -embedded induced k -linkage-realizations. However, since the graph is not planar we have to remove “short” closed curves which are not contractible. To state this precisely, we define some notations. When we deal with the (c, g) -embedded induced k -linkage-realizations in a punctured surface 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 $\Sigma - J$ consists of two components, each of them contains a cuff, and one of them forms a punctured plane. We say that $J \subseteq \Sigma$ is a *genus-reduction-curve* if J is a proper non-self-intersecting non-contractible closed curve in Σ which is not an O-arc.

In Algorithm Cuff_Reduction for the c -embedded induced k -linkage-realizations, we remove all short O-arcs before executing Step 3. On the other hand, when we deal with the (c, g) -embedded induced k -linkage-realizations, before removing short O-arcs we would like to transform the original graph into some graphs with no genus-reduction-curves of length at most $4h(k, g) + 2$.

If the graph G in the punctured surface Σ has a genus-reduction-curve of length at most $4h(k, g) + 2$, we take such a genus-reduction-curve J , and cut G and Σ along J (cf. [19]). In the resulting graph G' , the curve J corresponds to a curve J' with $2|J|$ vertices (if J is one-sided) or to two curves, which we call J' and J'' of order $|J|$, respectively, (if J is two-sided). Set $\Sigma := (\Sigma - J) \cup J'$ (if J is one-sided) or $\Sigma := (\Sigma - J) \cup J' \cup J''$ (if J is two-sided), and add all vertices of J' (and J'' if J is two-sided) to terminals. Then the resulting graph G' has smaller Euler genus, and there are one or two cuffs obtained from J , containing together $2|J|$ vertices. So this procedure simplifies the surface on the expense of adding more cuffs. Since a genus-reduction-curve of length at most $4h(k, g) + 2$ can be found in linear time [10] (see also Section 7 of [15]), this procedure can be done in linear time.

Now we modify Algorithm Cuff_Reduction so that it can be applied to the (c, g) -embedded induced k -linkage-realizations. The modified algorithm can be described as follows.

Algorithm Genus_Cuff_Reduction

Input: An instance of the (c, g) -embedded induced k -linkage-realizations, where $g \geq 1$ and $c \geq 2$.

Output: Some instances of the (c', g') -embedded induced k' -linkage-realizations, where

- $g' < g$ and c', k' is at most a constant depending on c, g and k , or
- $g' = g$, $c' < c$ and k' is at most a constant depending on c, g and k .

Step 1. If there exists a genus-reduction-curve J of length at most $4h(k, g) + 2$, then cut the punctured surface Σ along J . Then, we can reduce the instance into one or two instances with smaller genus, and stop the algorithm. Otherwise, go to Step 2.

Step 2. If there exists an O-arc J of length at most $4h(k, g) + 2$ such that each component of $\Sigma - J$ has genus at most $g - 1$ or contains at most $c - 1$ cuffs, then cut Σ along J . Then, we can reduce the instance into two instances with fewer cuffs, and stop the algorithm. Otherwise, go to Step 3.

Step 3. If there exists no O-arc of length at most $4h(k, g) + 2$, then go to Step 4. Otherwise, for an O-arc J of length at most $4h(k, g) + 2$, by Step 2, one component Σ_1 of $\Sigma - J$ is a disk containing exactly one cuff and the other component Σ_2 has genus g and contains c cuffs. We take such an O-arc J with maximal Σ_1 , cut Σ along J , and obtain two instances. Then, we execute Algorithm Genus_Cuff_Reduction for the instance in Σ_2 ,

and output some instances together with the instance in Σ_1 .

Step 4. It suffices to consider the case when the instance has no O-arc and no genus-reduction-curve of length at most $4h(k, g) + 2$. 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 $J_{i,j}$.

4-1. If the length of J is at most $2h(k, g) + 2$, then cut Σ along J and reduce the instance into an instance with $c - 1$ cuffs, and stop the algorithm.

4-2. If the length of J is more than $2h(k, g) + 2$, delete all vertices of J except the first $h(k, g) + 1$ and the last $h(k, g) + 1$. Then, since the length of J becomes $2h(k, g) + 2$, execute the same procedure as Step 4-1.

To see the correctness of this algorithm, the following theorem, which is an extension of Theorem 4.4, plays an important role. It can be derived from the main result of [23].

THEOREM 5.3. *Let G be a graph embedded in a punctured surface Σ , and l be a positive integer. Suppose that G has no proper closed curve of length at most $2l$ which is not contractible in Σ . Then, a vertex v is l -isolated if and only if $d_G(v) \geq l$.*

With the aid of this theorem, we can show the correctness of the algorithm. We also note that the solution of the input instance of the (c, g) -embedded induced k -linkage-realizations can be obtained from the solutions of the output instances.

Now we observe the running time of Algorithm Genus_Cuff_Reduction. We have already mentioned that Step 1 can be done in linear time, and the running time of the other steps are the same as Algorithm Cuff_Reduction. Therefore, it runs in linear time. We only note that O-arcs are found in Step 3 at most c times, because we take an O-arc such that Σ_1 is maximal. By executing Algorithm Genus_Cuff_Reduction at most c times, the original instance can be reduced to some instances of genus at most $g - 1$ and some instances of the $(1, g)$ -embedded induced k' -linkage-realizations. Repeating this, we reduce an instance of the (c, g) -embedded induced k -linkage-realizations to some instances of the c' -embedded induced k' -linkage-realizations and the $(1, g')$ -embedded induced k' -linkage-realizations. In the above reduction procedure, Algorithm Genus_Cuff_Reduction is executed at most constant times, and hence it can be done

in linear time. Thus the (c, g) -embedded induced k -linkage-realizations can be solved in linear time, which shows Theorem 5.1.

To the end, we now show how Theorem 5.1 implies Corollary 1.2. As with the proofs in [13, 21], we can easily reduce the IDPP to Theorem 5.1. In fact, Theorem 5.1 is a generalization of the IDPP. So, Corollary 1.2 follows from Theorem 5.1. Clearly the IDPP implies the DPP, because if we subdivide each edge once (that is, add a vertex of degree 2 to each edge), then finding the induced disjoint paths in the resulting graph corresponds to finding the disjoint paths in the original graph.

References

- [1] S. Arnborg and A. Proskurowski: Linear time algorithms for NP-hard problems restricted to partial k -trees, *Disc. Appl. Math.*, **23** (1989), pp. 11–24.
- [2] D. Bienstock: On the complexity of testing for even holes and induced odd paths, *Disc. Math.*, **90** (1991), pp. 85–92. See also Corrigendum by B. Reed, *Disc. Math.*, **102** (1992), p. 109.
- [3] H. L. Bodlaender: A linear time algorithm for finding tree-decompositions of small treewidth, *SIAM J. 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, *J. Graph Theory*, **48** (2005), pp. 85–111.
- [6] M. Chudnovsky, N. Robertson, P. D. Seymour and R. Thomas: The strong perfect graph theorem, *Ann. of Math.*, **64** (2006), pp. 51–219.
- [7] G. Cornuéjols, M. Conforti, A. Kapoor and K. Vuskovic: Even-hole-free graphs. I. Decomposition theorem, *J. Graph Theory*, **39** (2002), pp. 6–49.
- [8] G. Cornuéjols, M. Conforti, A. Kapoor and K. Vuskovic: Even-hole-free graphs. II. Recognition algorithm, *J. Graph Theory*, **40** (2002), pp. 238–266.
- [9] E. D. Demaine, F. V. Fomin, M. Hajiaghayi and D. M. Thilikos: Subexponential parameterized algorithms on bounded-genus graphs and H-minor-free graphs, *J. ACM*, **52** (2005), pp. 866–893.
- [10] J. Erickson and S. Har-Peled: Optimally cutting a surface into a disk, *Discrete and Computational Geometry*, **31** (2004), pp. 37–59. Conference version in *Proc. 18th Annual ACM Symposium on Computational Geometry*, 2002, pp. 244–253.
- [11] M. R. Fellows: The Robertson-Seymour Theorems: a survey of applications, *Contemporary Mathematics* 89, American Mathematical Society, 1987, pp. 1–18.
- [12] M. R. Fellows, J. Kratochvil, M. Middendorf and F. Pfeiffer: The complexity of induced minors and related problems, *Algorithmica*, **13** (1995), pp. 266–282.
- [13] K. Kawarabayashi and Y. Kobayashi: A linear time algorithm for the induced disjoint paths problem in planar graphs, manuscript.
- [14] K. Kawarabayashi and Y. Kobayashi: The induced disjoint paths problem, *Proc. the 13th Integer Programming and Combinatorial Optimization (LNCS 5035)*, 2008, pp. 47–61.
- [15] K. Kawarabayashi and B. Mohar: Graph and map isomorphism and all polyhedral embeddings in linear time, *Proc. the 40th Annual ACM Symposium on Theory of Computing*, 2008, pp. 471–480.
- [16] Y. Kobayashi: An extension of the disjoint paths problem, *METR 2007-14*, Department of Mathematical Informatics, University of Tokyo (2006).
- [17] B. Lévêque, D. Lin, F. Maffray and N. Trotignon: Detecting induced subgraphs, *Disc. Appl. Math.*, to appear.
- [18] C. McDiarmid, B. Reed, A. Schrijver and B. Shepherd: Induced circuits in planar graphs, *J. Combin. Theory Ser. B*, **60** (1994), pp. 169–176.
- [19] B. Mohar and C. Thomassen: *Graphs on Surfaces*, The Johns Hopkins University Press, Baltimore and London, 2001.
- [20] B. Reed: Rooted routing in the plane, *Disc. Appl. Math.*, **57** (1995), pp. 213–227.
- [21] 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.
- [22] N. Robertson and P. D. Seymour: Graph minors. VII. Disjoint paths on a surface, *J. Combin. Theory Ser. B*, **45** (1988), pp. 212–254.
- [23] N. Robertson and P. D. Seymour: Graph minors. XI. Circuits on a surface, *J. Combin. Theory Ser. B*, **60** (1994), pp. 72–106.
- [24] N. Robertson and P. D. Seymour: Graph minors. XIII. The disjoint paths problem, *J. Combin. Theory Ser. B*, **63** (1995), pp. 65–110.
- [25] N. Robertson and P. D. Seymour: Graph minors. XXII. Irrelevant vertices in linkage problems, manuscript.
- [26] N. Robertson, P. D. Seymour and R. Thomas: Quickly excluding a planar graph, *J. Combin. Theory Ser. B*, **62** (1994), pp. 323–348.
- [27] C. Thomassen: A simpler proof of the excluded minor theorem for higher surfaces, *J. Combin. Theory Ser. B*, **70** (1997), pp. 306–311.