Journal of Combinatorial Theory, Series B ••• (••••) •••–•••

Contents lists available at ScienceDirect

Journal of Combinatorial Theory, Series B

www.elsevier.com/locate/jctb

The disjoint paths problem in quadratic time

Ken-ichi Kawarabayashi [a], Yusuke Kobayashi [b], Bruce Reed [c,d]

[a] National Institute of Informatics, 2-1-2, Hitotsubashi, Chiyoda-ku, Tokyo, Japan
[b] Department of Mathematical Informatics, Graduate School of Information Science and Technology, University of Tokyo, Tokyo 113-8656, Japan
[c] Canada Research Chair in Graph Theory, McGill University, Montreal, Canada
[d] Project Mascotte, INRIA, Laboratoire I3S, CNRS, Sophia-Antipolis, France

ARTICLE INFO

Article history:
Received 6 July 2009
Available online xxxx

Keywords:
Disjoint paths
Quadratic time
Graph minor
Tree-width

ABSTRACT

We consider the following well-known problem, which is called the *disjoint paths problem*. For a given graph G and a set of k pairs of terminals in G, the objective is to find k vertex-disjoint paths connecting given pairs of terminals or to conclude that such paths do not exist. We present an $O(n^2)$ time algorithm for this problem for fixed k. This improves the time complexity of the seminal result by Robertson and Seymour, who gave an $O(n^3)$ time algorithm for the disjoint paths problem for fixed k. Note that Perković and Reed (2000) announced in [24] (without proofs) that this problem can be solved in $O(n^2)$ time. Our algorithm implies that there is an $O(n^2)$ time algorithm for the k edge-disjoint paths problem, the minor containment problem, and the labeled minor containment problem. In fact, the time complexity of all the algorithms with the most expensive part depending on Robertson and Seymour's algorithm can be improved to $O(n^2)$, for example, the membership testing for minor-closed class of graphs.

1. Introduction

1.1. Background of the disjoint paths problem

In the vertex-(edge-)disjoint paths problem, we are given a graph G and a set of k pairs of vertices $(s_1, t_1), \ldots, (s_k, t_k)$ in G (which are sometimes called *terminals*), and we have to decide whether or not G has k vertex-(edge-)disjoint paths $P_1, \ldots, P_k$ in G such that P_i joins s_i and t_i for $i = 1, 2, \ldots, k$.

E-mail addresses: k_keniti@nii.ac.jp (K. Kawarabayashi), kobayashi@mist.i.u-tokyo.ac.jp (Y. Kobayashi), breed@cs.mcgill.ca (B. Reed).

0095-8956/$ – see front matter
doi:10.1016/j.jctb.2011.07.004

Please cite this article in press as: K. Kawarabayashi et al., The disjoint paths problem in quadratic time, J. Combin. Theory Ser. B (2011), doi:10.1016/j.jctb.2011.07.004

Furthermore, we find such paths if they exist. This is certainly a central problem in algorithmic graph theory and combinatorial optimization. See the surveys [9,32]. It has attracted attention in the contexts of transportation networks, VLSI layout and virtual circuit routing in high-speed networks or Internet. A basic technical problem here is to interconnect certain prescribed "channels" on the chip such that wires belonging to different pins do not touch each other. In this simplest form, the problem mathematically amounts to finding disjoint trees in a graph or disjoint paths in a graph, each connecting a given set of vertices.

Let us give previous known results on the disjoint paths problem. If k is a part of the input of the problem, then this is one of Karp's original NP-complete problems [12], and it remains NP-complete even if G is constrained to be planar (Lynch [21]). The seminal work of Robertson and Seymour says that there is a polynomial time algorithm (actually an $O(n^3)$ time algorithm) for the disjoint paths problem when the number of terminals, k, is fixed (in this paper, we shall refer to this problem as "the k disjoint paths problem"). Actually, this algorithm is one of the spin-offs of their groundbreaking work on the graph minor project spanning 23 papers, and giving several deep and profound results and techniques in discrete mathematics.

The disjoint paths problem is a special case of finding a multi-commodity flow. In the multi-commodity flow question, the commodities at the sources $s_1, s_2, \ldots, s_k$ are different and the demand at each t_i is for a specific commodity. This is the type of question we need to resolve when sending information through the information highway network and so has become increasingly of interest to computer scientists (see, for example the work of Chekuri et al. [3–6] and of Tardos and Kleinberg [15–18]). One special case which is of great interest is that all demands are at most $1/2$. This problem setting behaves very different from the disjoint paths problem. Indeed there are many such flow type problems for which the half integral version can be at least approximately solved although the integral version is intractable (see [20,22]). A similar situation holds with respect to the k disjoint path problem. The proof of correctness of Robertson and Seymour's algorithm requires almost all of the graph minors project papers and more than 500 pages.[1] On the other hand, Kawarabayashi and Reed [13] gave a nearly linear time algorithm for the half integral version, which improves the previous known result by Kleinberg [16] who gave an $O(n^3)$ time algorithm. In addition, the correctness of this algorithm is much simpler than that of Robertson and Seymour's.

1.2. Motivation and main results

Our motivation is that it seems that the time complexity $O(n^3)$ of Robertson and Seymour's algorithm is too expensive. Reed announced in [27] (see also [24]) that he proved that this problem can be solved in $O(n^2)$ time. However a detailed description of the algorithm was not fully written down. We now come to know tangles and brambles, and their corresponding grid minors more closely, so we think that we should be able to improve the time complexity. In [13], two of us gave a "nearly" linear time algorithm for the half integral version of the k disjoint paths problem. Namely, the time complexity is $O(m + n \log n)$. This greatly improves the result by Kleinberg [16]. We now try to improve the time complexity of Robertson and Seymour's algorithm. Let us remark that the results in [26,29] show that there is a linear time algorithm for the k disjoint paths problem when an input graph is planar. Also, there is a linear time algorithm for the k disjoint paths problem when an input graph is a bounded genus graph [8,19]. So, it would be conceivable that the time complexity of Robertson and Seymour's algorithm can be improved to linear or nearly linear. However, there are a lot of technical difficulties; as we mentioned, the half integral version of the k disjoint paths problem is much easier. In this paper, we managed to prove the following theorem, which improves the time complexity of Robertson and Seymour's algorithm to quadratic.

Theorem 1.1. *There is an $O(n^2)$ time algorithm for the k disjoint paths problem for fixed k.*

[1] Recently, a much shorter proof for the correctness of the graph minor algorithm is obtained in [14]. The proof hinges upon a significantly shorter proof of the "unique linkage theorem" [35].

Please cite this article in press as: K. Kawarabayashi et al., The disjoint paths problem in quadratic time, J. Combin. Theory Ser. B (2011), doi:10.1016/j.jctb.2011.07.004

1.3. Overview

Our algorithm follows Robertson–Seymour's algorithm [33]. So, let us first sketch the Robertson–Seymour's algorithm on the disjoint paths problem.

At a high level, it is based on the following two cases: either a given graph G has bounded tree-width or else it has large tree-width. In the first case, one can apply dynamic programming to a tree-decomposition of bounded tree-width, see [1,2,33]. The second case again breaks into two cases depending on whether G has a large clique minor or not. Suppose that G has a large clique minor. If we can link up the terminals to the minor, then we can use this clique minor to link up the terminals in any desired way. Otherwise, there is a small separation such that the large clique minor is cut off from the terminals by this separation. In this case, we can prove that there is a vertex v in the clique minor which is irrelevant, i.e., the k disjoint paths problem is feasible in G if and only if it is in $G - v$. So, suppose G does not have a large clique minor. Then one can prove that, after deleting bounded number of vertices, there is a large wall which is essentially planar. This makes it possible to prove that the middle vertex v of this wall is irrelevant. This requires the whole graph minor papers, and the structure theorem of graph minors [34].

Robertson and Seymour could only give an $O(n^2)$ time algorithm to find such an irrelevant vertex v. Then the algorithm recurses in the graph $G - v$. Thus if we could give an $O(n)$ algorithm to find such an irrelevant vertex, we could prove Theorem 1.1. This is our main task. We need to consider the two cases, namely, a graph with or without a large clique minor. In both cases, we need to find an irrelevant vertex in time $O(n)$. This will be proved in Sections 4 and 5.

The technical difficulties we have to overcome are the following two points:

1. When there is a large clique minor, we have to find an irrelevant vertex in $O(n)$ time, rather than in $O(m)$ time. This means that we cannot use the standard flow algorithm (since the input graph could have $\Theta(n^2)$ edges). We shall use the algorithm by Nagamochi and Ibaraki [23] to find such a vertex. Roughly, at the beginning of our algorithm, we have to construct a "spanner" G' of the input graph G with at most $O(n)$ edges, which maintains connectivity between each vertex in G and the terminals in $\{s_1, \ldots, s_k, t_1, \ldots, t_k\}$. This allows us to find a small separation in $O(n)$ time, and hence we are able to find an irrelevant vertex in the large clique minor in $O(n)$ time. In addition, after deleting the irrelevant vertex, we have to update this "spanner" in $O(n)$ time.
2. When there is no large clique minor, in $O(n)$ time, we have to find a large wall which is closest to a "leaf" in the seminal graph minor decomposition theorem. Then we have to find a "nearly" planar embedding induced by this wall in $O(n)$ time. Note that at the moment, there are $O(n)$ edges in G.

2. Preliminary

In this paper, n and m always mean the number of vertices of a given graph and the number of edges of a given graph, respectively. Sometimes we say "disjoint paths", which mean "vertex-disjoint paths". Let $G = (V, E)$ be a graph. For a subgraph H of G, the vertex set and the edge set of H are denoted by $V(H)$ and $E(H)$, respectively. A *separation* (A, B) of G is a pair of edge-disjoint subgraphs such that $G = A \cup B$. The order of the separation (A, B) is $|V(A) \cap V(B)|$. For $X \subseteq V$, let $N(X)$ denote the set of vertices in $V \setminus X$ that are adjacent to X, and let $G[X]$ be the subgraph induced by X. We now look at definitions of the tree-width and walls.

2.1. Tree-width

Tree-width was introduced by Halin in [10], but it went unnoticed until it was rediscovered by Robertson and Seymour [30] and, independently, by Arnborg and Proskurowski [1].

A *tree-decomposition* of a graph G consists of a tree T and a subtree S_v of T for each vertex v of G such that if uv is an edge of G then S_u and S_v intersect. For each node t of the tree, we let W_t be the set of vertices v of G such that $t \in S_v$. The *width* of a tree-decomposition is the maximum of

Please cite this article in press as: K. Kawarabayashi et al., The disjoint paths problem in quadratic time, J. Combin. Theory Ser. B (2011), doi:10.1016/j.jctb.2011.07.004

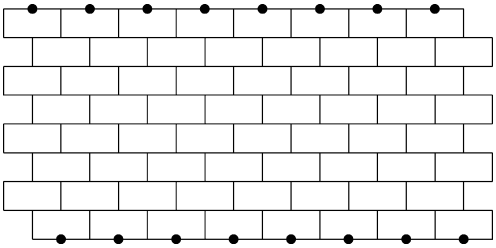

Fig. 1. An elementary wall of height 8.

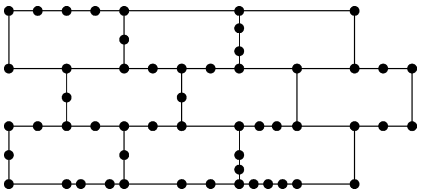

Fig. 2. A wall of height 3.

$|W_t| - 1$ over the nodes t of T. The *tree-width* of a graph is the minimum width among all possible tree-decompositions of the graph.

We can apply dynamic programming to solve problems on graphs of bounded tree-width, in the same way that we apply it to trees (see e.g. [1]), provided that we are given a bounded width tree-decomposition.

Robertson and Seymour developed the first polynomial time algorithm for constructing a tree-decomposition of a graph of bounded width [33], and eventually came up with an algorithm which runs in $O(n^2)$ time, for this problem. Reed [25] developed an algorithm for the problem which runs in $O(n \log n)$ time, and then Bodlaender [2] developed a linear time algorithm. This algorithm was further improved in [24].

Theorem 2.1. *(See Bodlaender [2].) For any fixed integer w, there exists an $O(n)$ time algorithm that, given a graph G, either finds a tree-decomposition of G of width w or concludes that the tree-width of G is more than w.*

2.2. Wall

An elementary wall of height eight is depicted in Fig. 1. An *elementary wall* of height h for $h \geqslant 2$ is similar. It consists of h levels each containing h bricks, where a brick is a cycle of length six. A *wall* of height h is obtained from an elementary wall of height h by subdividing some of the edges, i.e. replacing the edges with internally vertex-disjoint paths with the same endpoints (see Fig. 2). The *nails* of a wall are the vertices of degree three within it. Any wall has a unique planar embedding. The *perimeter* of a wall W, denoted $\mathrm{per}(W)$ is the boundary of the unique face in this embedding which contains $4(h-1)$ nails.

One of the most important results concerning the tree-width is the main result of Graph Minors V [31] which says the following:

Theorem 2.2. *(See Robertson and Seymour [31].) For any r, there exists a constant $f_1(r)$ such that if G has tree-width at least $f_1(r)$, then G contains a wall W of height r.*

The best known upper bound for $f_1(r)$ is given in [7,27,36]. It is 20^{2r^5}. The best known lower bound is $\Theta(r^2 \log r)$, see [36]. Furthermore, such a wall can be found in linear time.

Please cite this article in press as: K. Kawarabayashi et al., The disjoint paths problem in quadratic time, J. Combin. Theory Ser. B (2011), doi:10.1016/j.jctb.2011.07.004

Theorem 2.3. *(Follows from [2,24,31].) For fixed integer r, in a graph G with tree-width at least $f_1(r)$, we can find a wall W of height r in linear time.*

Here we give an outline of the linear time algorithm. By the algorithm in [24], we can find in linear time a subgraph G' of G of tree-width at least $f_1(r)$ and a tree-decomposition of G' of width at most $2f_1(r)$. Then, since G' has a wall of height r by Theorem 2.2, it can be found in linear time by the dynamic programming method [2].

3. Finding a flat large wall in K_t-minor-free graphs

In this section, we apply a structural result of Robertson and Seymour concerning graphs which have a large tree-width but no large clique minor. To state this result we will need a few definitions.

Recall that the *nails* of a wall are the vertices of degree three within it. Any wall has a unique planar embedding. For any wall W in a given graph G, there is a unique component U of $G - \mathrm{per}(W)$ containing $W - \mathrm{per}(W)$. The *compass* of W, denoted $\mathrm{comp}(W)$, is the subgraph of G induced by $V(U) \cup V(\mathrm{per}(W))$. A *subwall* of a wall W is a wall which is a subgraph of W. A subwall of W of height h is *proper* if it consists of h consecutive bricks from each of h consecutive rows of W. The *exterior* of a proper subwall W' of a wall W is $W - W'$. We say a proper subwall W' is *dividing* in G if the compass of W' in G is disjoint from $W - W'$.

A wall is *flat* if its compass does not contain two vertex-disjoint paths connecting the diagonally opposite corners. Note that if the compass of W has a planar embedding whose infinite face is bounded by the perimeter of W then W is clearly flat. In order to characterize flat walls, we use the result of Seymour [37], Thomassen [41], and others on the 2 disjoint paths problem.

We now mention the characterization of a flat wall. By the characterization of graphs containing 2 disjoint paths (see [37, Theorem 4.1] for example), a wall W is flat if and only if there are pairwise disjoint sets $A_1, \ldots, A_l \subseteq V(\mathrm{comp}(W))$ containing no corners of W such that

(1) for $1 \leqslant i, j \leqslant l$ with $i \neq j$, $N(A_i) \cap A_j = \emptyset$,
(2) for $1 \leqslant i \leqslant l$, $|N(A_i)| \leqslant 3$, and
(3) if W' is the graph obtained from $\mathrm{comp}(W)$ by deleting A_i and adding new edges joining every pair of distinct vertices in $N(A_i)$ for each i, then W' can be drawn in a plane so that all corners of W are on the outer face boundary.

We call $A_1, \ldots, A_l$ with these conditions *attached vertex sets*. If such $A_1, \ldots, A_l$ exist, we say that $\mathrm{comp}(W)$ *can be embedded into a plane up to 3-separations*, and an embedding as in (3) is called a *flat embedding*. It is easy to see that any proper subwall of a flat wall must be both flat and dividing. Furthermore, if x and y are two vertices of a flat wall W and there is a path between them which is internally disjoint from W then either x and y are both on $\mathrm{per}(W)$ or some brick contains both of them.

Finally, we state the main result in this section. Robertson and Seymour (Theorem (9.6) in [33]) proved the following algorithmic result:

Theorem 3.1. *(See Robertson and Seymour [33] and Kapadia et al. [11].) For any fixed integers $t \geqslant 2$ and $h \geqslant 2$, there is a computable constant $f_2(t,h)$ such that the following can be done in $O(m)$ time, where m is the number of edges of a given graph G.*

Input: *A graph G, a wall H of height at least $f_2(t,h)$.*

Output: *Either*

1. *a K_t-minor, or*
2. *a subset $X \subseteq V(G)$ of order at most $\binom{t}{2}$ and t^2 disjoint proper subwalls $H_1, \ldots, H_{t^2}$ of height h such that each H_i is dividing and flat in $G - X$ for $i = 1, \ldots, t^2$. In addition, a flat embedding of H_i is also given for $i = 1, \ldots, t^2$.*

Please cite this article in press as: K. Kawarabayashi et al., The disjoint paths problem in quadratic time, J. Combin. Theory Ser. B (2011), doi:10.1016/j.jctb.2011.07.004

The algorithm in [33] takes $O(mn)$ time because Robertson and Seymour used an $O(mn)$ time algorithm to test the flatness (i.e., give a flat embedding in the compass of H_i in the second conclusion), and this part is the most expensive. More precisely, Robertson and Seymour first proved that there is an $O(m)$ time algorithm to get the conclusion of Theorem 3.1, but without the conclusion "flat" (this step corresponds to (9.4) in [33] that runs in $O(m)$ time). Then for each of dividing subwalls, test whether or not it is flat (this step corresponds to (8.1) in [33] that runs in $O(mn)$ time). There is now an $O(m)$ time algorithm to test the flatness by Kapadia, Li and Reed [11], which improves the previous best known result by Tholey [39,40] which gives an $O(m + n\alpha(n, n))$ time algorithm, where the function α is the inverse of the Ackermann function (see text by Tarjan [38]). Note that the algorithms in [11,39,40] also find a flat embedding of a wall (if it is flat) in the same running time. Actually, in (nearly) linear time, these algorithms find attached vertex sets $A_1, \ldots, A_l$ and reduce a given instance of the 2 disjoint paths problem to an instance in a graph $G' = (V', E')$ that has no separation (A, B) of order at most three such that A contains all the terminals (see Property (P) in [40]). After this reduction a flat embedding is equivalent to a planar embedding which can be found in linear time. Thus if we use the algorithm in [11] for testing the flatness, we can get an $O(m)$ time algorithm, as claimed in Theorem 3.1.

Note that [11] is a paper under submission. If we use the result in [39,40] instead of that in [11], the running time in Theorem 3.1 becomes $O(m + n\alpha(n, n))$, and consequently the running time in Theorem 1.1 becomes $O(n^2\alpha(n, n))$, which is slightly larger than $O(n^2)$.

In either case, with the help of Theorem 3.1, we find desired k disjoint paths or an irrelevant vertex in $O(n)$ time. We describe the procedure in the next section.

4. Irrelevant vertices

Let us recall that a vertex v is irrelevant if the k disjoint paths problem is feasible in G if and only if it is in $G - v$. In this section, we give some theorems concerning irrelevant vertices.

4.1. When the graph has a large clique minor

Let us first give theorems concerning a graph with a large clique minor.

Theorem 4.1. *(See Robertson and Seymour [33, Theorem (5.4)].) Let $s_1, \ldots, s_k$, $t_1, \ldots, t_k$ be terminals in a given graph G. If there is a clique minor of order at least $3k$ in G, and there is no separation (A, B) of order at most $2k - 1$ in G such that A contains all the terminals and $B - A$ contains at least one node of the clique minor, then there are disjoint paths P_i with two ends in s_i, t_i for $i = 1, \ldots, k$. Furthermore, such paths can be found in $O(m)$ time.*

If there is a separation (A, B) of order at most $2k - 1$ such that A contains all the terminals and $B - A$ contains at least one vertex of the clique minor, we can find a vertex of the clique minor which is irrelevant in $O(m)$ time (see arguments in [33, Theorem (6.2)]).

In order to improve the running time from $O(m)$ to $O(n)$, we need the result by Nagamochi and Ibaraki [23]. They gave an algorithm to reduce the number of edges from m to $O(n)$ keeping the connectivity of the graph. More precisely, for a graph $G = (V, E)$ and an integer t, their algorithm finds a subgraph $G_t = (V, E_t)$ of G such that $|E_t| = O(|V|)$ and $\kappa(u, v; G_t) \geqslant \min\{\kappa(u, v; G), t\}$, where $\kappa(u, v; G)$ denotes the vertex connectivity between u and v in G. Such a graph G_t is said to be a *t-certificate* of G. This procedure takes $O(m)$ time, so if we newly construct a t-certificate each time after deleting an irrelevant node, the running time becomes expensive. In order to avoid this, we show the following:

Theorem 4.2. *Let k be a fixed integer. Suppose we are given a graph $G = (V, E)$, terminals $s_1, \ldots, s_k, t_1, \ldots, t_k$, a clique minor K of order $3k$, and a $2k$-certificate $G_{2k} = (V, E_{2k})$ of G with $|E_{2k}| = O(|V|)$. We can find either*

1. *desired k disjoint paths P_i with two ends in s_i, t_i for $i = 1, \ldots, k$ in $O(|V|)$ time or*

Please cite this article in press as: K. Kawarabayashi et al., The disjoint paths problem in quadratic time, J. Combin. Theory Ser. B (2011), doi:10.1016/j.jctb.2011.07.004

2. *a graph* $G' = (V', E')$ *with* $|V'| \leqslant |V| - 1$ *and a* $2k$*-certificate* $G'_{2k} = (V', E'_{2k})$ *such that* $|E'_{2k}| = O(|V'|)$, V' *contains all terminals, and the* k *disjoint paths problem in* G' *is equivalent to the original problem in* $O(|V| + |E \setminus E_{2k}|)$ *time.*

Proof. First, we find in G a smallest separation (A, B) of order at most $2k-1$ such that A contains all the terminals, $B - A$ contains at least one node of the clique minor. Note that if the smallest order of such a separation is at least $2k$, we can find desired k disjoint paths in $G_{2k} + K$ by Theorem 4.1, and the running time is $O(|V|)$. If the smallest order is at most $2k-1$, we can find such a separation in $O(|V|)$ time by using a simple flow algorithm in $G_{2k} + K$. Then, we obtain G' by deleting all vertices in $B - A$ and adding new edges joining every pair of distinct vertices in $A \cap B$. One can see that this procedure does not affect the solution by Theorem 4.1. Furthermore, by executing the same procedure to G_{2k}, we obtain a graph G'' with $\kappa(u, v; G'') \geqslant \min\{\kappa(u, v; G'), 2k\}$ for any u, v in G'. Since G'' has at most $|E_{2k}| + \binom{2k-1}{2} = O(|V|)$ edges, we can apply the algorithm of Nagamochi and Ibaraki to G'' in $O(|V|)$ time. Then we obtain a $2k$-certificate $G'_{2k} = (V', E'_{2k})$ of G' with $|E'_{2k}| = O(|V'|)$. □

In Theorem 4.2, we assumed that a clique minor of order $3k$ is given. But how do we find such a clique minor in linear time? Here is one way.

Theorem 4.3. *(See Reed and Wood [28, Theorem 2.1].) For any fixed integer* t*, if* G *has at least* $2^{t-3}|V(G)|$ *edges, then there is an* $O(n)$ *time algorithm to find a* K_t*-minor.*

This theorem improves Algorithm (6.4) in [33]. Note that although the number of the edges in G is not necessarily $O(n)$, we only need to focus on $2^{t-3}|V(G)|$ edges to find a K_t-minor. Thus, the running time is not $O(m)$ but $O(n)$. Our algorithm will keep applying Theorem 4.3 to get a clique minor of order $3k$. After applying Theorem 4.3, we may assume that G has at most $2^{3k-3}|V(G)|$ edges. Hence, in what follows in this section, we assume that $m = O(n)$ and describe how we find an irrelevant vertex in a graph with no K_{3k}-minor.

4.2. When the graph has no large clique minor

Next, we discuss the case when the graph has no large clique minor. In order to state the theorem, we need to define *realizable partitions*. Let $G = (V, E)$ be a graph and $Z \subseteq V$ be a vertex set. A partition $\mathcal{P} = \{Z_1, \ldots, Z_p\}$ of Z is *realizable* if there are disjoint trees $T_1, \ldots, T_p$ in G such that $Z_i \subseteq V(T_i)$ for $i = 1, \ldots, p$. Robertson and Seymour [33] showed the following theorem.

Theorem 4.4. *(See Robertson and Seymour [33, Theorem (4.1)].) Let* G *be a graph with a tree-decomposition of width at most* w *(for fixed* w*), and let* Z *be a vertex set of order at most* $2k$ *(for fixed* k*). Then there is an* $O(f(k, w)n)$ *time algorithm to enumerate all realizable partitions of* Z *in* G *for some function* f *of* k, w*.*

Note that the problem of enumerating all realizable partitions corresponds to the problem called 0-*folio* in [33]. It is generalized to the problem called *l*-*folio* in [33], which plays a central role in the proof of the seminal result of Robertson and Seymour.

We also note that although the result in [33] is stated in terms of "branch-width," it does not cause any problems because branch-width differs only by a constant factor from tree-width.

Now we are ready to state the result concerning a graph without a large clique minor.

Theorem 4.5. *(See Robertson and Seymour [33, Theorem (10.3)].) For any fixed integer* k*, there is a computable constant* $f(k)$ *satisfying the following: if there is a subset* $X \subseteq V(G)$ *of order at most* $\binom{3k}{2}$ *such that there is a flat wall* W *of height* $f(k)$ *in* $G - X$*, then there is a vertex* v *in* W *such that* v *is irrelevant. Furthermore, if we have*

(W1) *a flat embedding of* comp(W) *with attached vertex sets* $A_1, \ldots, A_l$ *and*

Please cite this article in press as: K. Kawarabayashi et al., The disjoint paths problem in quadratic time, J. Combin. Theory Ser. B (2011), doi:10.1016/j.jctb.2011.07.004

(W2) *all realizable partitions of Z_i in A'_i for each i, where $A'_i = A_i \cup X \cup (N(A_i) \cap V(\mathrm{comp}(W)))$ and $Z_i = X \cup (N(A_i) \cap V(\mathrm{comp}(W)))$,*

then we can find an irrelevant vertex v in linear time.

We now give some remarks on differences between this theorem and the original statement of Theorem (10.3) in [33].

The original statement of Theorem (10.3) in [33] is a combination of Theorem 4.5 above and Theorem (10.1) in [33], and so the condition (W2) does not appear in the original statement explicitly.

In Theorem (10.3) in [33], they use a *rural division* instead of the attached vertex sets $A_1, \ldots, A_l$. For $A_1, \ldots, A_l$, the corresponding rural division can be obtained as follows. For each pair of sets A_i and A_j with $N(A_i) \cap V(\mathrm{comp}(W)) \subseteq N(A_j) \cap V(\mathrm{comp}(W))$, replace them by their union. For the resulting sets $B_1, \ldots, B_{l'}$, let G_i be the subgraph of $G - X$ induced by $B_i \cup N(B_i)$ for $i = 1, \ldots, l'$. Then, the rural division consists of the graphs $G_1, \ldots, G_{l'}$ and the graphs each consisting of a single edge (together with its end vertices) not contained in one of $G_1, \ldots, G_{l'}$.

Although the original statement of Theorem (10.3) in [33] says that it requires quadratic time in the above statement, the precise running time is linear. This is because the number of iterations (denoted by l) in the proof of Theorem (10.3) of [33] is bounded by a constant which only depends on k.

We also note that the original statement requires a *vision* of each A_i, which consists of (W2) and the following information:

(W3) the clockwise ordering of $N(A_i)$ in the flat embedding of $\mathrm{comp}(W)$,
(W4) for each vertex x in $N(A_i) \cap V(\mathrm{comp}(W))$, a vertex y of W that can be an endpoint of a path P from x such that P does not hit W, except for y.

Since the flat embedding of $\mathrm{comp}(W)$ gives us (W3) and (W4) in linear time, we omit them in the statement. In the original statement of [33], they consider flat walls but do not use their flat embeddings, and so they mention the information (W3) and (W4) explicitly.

Theorem (10.1) in [33] shows that (W2) can be computed in quadratic time if all the components $G[A_1], \ldots, G[A_l]$ have tree-width bounded by a fixed constant $g(k)$. By improving the running time of this statement to linear time, we show the following theorem.

Theorem 4.6. *Let G, X, and W be as in the first part of Theorem* 4.5. *If we have a flat embedding of* $\mathrm{comp}(W)$ *with attached vertex sets $A_1, \ldots, A_l$ such that all the components $G[A_1], \ldots, G[A_l]$ have tree-width bounded by a fixed constant $g(k)$, we can find in $O(n)$ time an irrelevant vertex v.*

Proof. By Theorem 4.5, it suffices to show that (W2) can be computed in linear time when all the components $G[A_1], \ldots, G[A_l]$ have tree-width bounded by a fixed constant $g(k)$. Let H be the compass of W in $G - X$.

For each A'_i, by Theorems 2.1 and 4.4, in time $O(|A'_i|)$, we can enumerate all realizable partitions of Z_i in A'_i. On the other hand, since $|N(A_i) \cap V(H)| \leqslant 3$ and $|X| \leqslant 9k^2$,

$$\sum_{i=1}^{l} |A'_i| \leqslant \sum_{i=1}^{l} |A_i| + \big(3 + |X|\big)l = O(n).$$

Thus, it can be done in $O(n)$ time, and Theorem 4.6 follows. □

5. Key theorems

The previous theorem, Theorem 4.6, is our main tool, but how do we find a subset $X \subseteq V(G)$ and a wall W such that

Please cite this article in press as: K. Kawarabayashi et al., The disjoint paths problem in quadratic time, J. Combin. Theory Ser. B (2011), doi:10.1016/j.jctb.2011.07.004

(C1) $|X| \leqslant \binom{3k}{2}$,
(C2) W is a flat wall of height $f(k)$ in $G - X$, and
(C3) all the components $G[A_1], \ldots, G[A_l]$ have tree-width at most $g(k)$, where $A_1, \ldots, A_l$ are attached vertex sets as in the flat embedding of comp(W)

in linear time? Robertson and Seymour [33] described the following algorithm, which takes $O(n^2)$ time.

Theorem 5.1. *(See Robertson and Seymour [33, Theorem (9.8)].) For any fixed integer k, there are computable constants $h(k)$ and $g(k)$ such that, if a given graph G has tree-width at least $h(k)$, then there is an $O(n^2)$ time algorithm to find either a K_{3k}-minor or a pair (X, W) satisfying the conditions* (C1)–(C3)*.*

Our main contribution in this paper is the following:

Theorem 5.2. *For any fixed integer k, there are computable constants $h(k)$ and $g(k)$ such that, if a given graph G has tree-width at least $h(k)$, then there is an $O(n)$ time algorithm to find either a K_{3k}-minor or a pair (X, W) satisfying the conditions* (C1)–(C3)*.*

Proof. We set $h(k) \geqslant f_1(f_2(3k, f(k)))$, where $f(k)$ is as in the condition (C2), f_1 is as in Theorem 2.3, and f_2 is as in Theorem 3.1. Set the constant $g(k)$ such that $g(k) \geqslant h(k)$.

For a given graph G with tree-width at least $h(k)$, we first apply Theorem 4.3. If there is a K_{3k}-minor, we are done. Thus we may assume that G has at most $2^{3k-3}|V(G)|$ edges.

By applying Theorem 2.3, we now have a wall of height $f_2(3k, f(k))$ in hand. We apply Theorem 3.1 with $t = 3k$ and $h = f(k)$ for the wall. Since G has at most $O(|V(G)|)$ edges, the time complexity of Theorem 3.1 is now $O(n)$. If we get a K_{3k}-minor, we are done. Thus by the definition of f_2, we may assume that there exist a set $X \subseteq V(G)$ with $|X| \leqslant \binom{3k}{2}$ and $(3k)^2$ dividing and flat proper subwalls of height $f(k)$ in $G - X$. We take such two subwalls W_1 and W_2, and let $A_{i,1}, A_{i,2}, \ldots, A_{i,l_i}$ be vertex sets as in the flat embedding of the compass of the wall W_i for $i = 1, 2$. For each i, j in turn, we test whether or not $A_{i,j}$ has tree-width at least $h(k)$. This can be done in linear time by Theorem 2.1.

If all of $G[A_{i,1}], G[A_{i,2}], \ldots, G[A_{i,l_i}]$ have tree-width at most $h(k)$, then W_i satisfies (C1)–(C3). Otherwise, we may assume that $G[A_{i,1}]$ has tree-width at least $h(k)$ for $i = 1, 2$. Furthermore, since $A_{1,1}$ and $A_{2,1}$ are disjoint, we may assume also that $|A_{1,1}| \leqslant |V|/2$. Let X' be at most three vertices in the compass that are adjacent to a vertex in $A_{1,1}$. Then, $X \cup X'$ is a cutset in G such that $G[A_{1,1}]$ is one of the components of $G - X - X'$. We repeat this algorithm with the input graph $G[A_{1,1} \cup X \cup X']$.

We only take care of how we choose two subwalls of height $f(k)$. In the graph $G[A_{1,1} \cup X \cup X']$, by applying Theorems 2.3 and 3.1, we can find $(3k)^2$ dividing and flat subwalls of height $f(k)$. Since $|X \cup X'| \leqslant \binom{3k}{2} + 3 < (3k)^2 - 2$, there exist two dividing and flat subwalls W_1' and W_2' of height $f(k)$ that do not contain any vertex of $X \cup X'$. Thus, when we execute the same procedure for W_1' and W_2' in the next iteration, we only need to look at the graph $G[A_{1,1}]$.

All the above operations can be done in $O(n)$ time, as described. Let us take $T(n)$ as the time complexity in the above operations with the input graph G of n vertices. When we repeat the algorithm, we throw away half of the vertices of the current graph. Thus, the time complexity will be $T(n) + O(n) + T(\frac{n}{2}) + T(\frac{n}{4}) + \cdots$, which is still $O(n)$. This completes the proof of Theorem 5.2. □

6. Algorithm

Let k be a positive integer. In this section, we describe our algorithm for the k disjoint paths problem.

Algorithm for the *k* disjoint paths problem

Input: A graph G with n vertices, and terminals $s_1, \ldots, s_k, t_1, \ldots, t_k$.

Please cite this article in press as: K. Kawarabayashi et al., The disjoint paths problem in quadratic time, J. Combin. Theory Ser. B (2011), doi:10.1016/j.jctb.2011.07.004

Output: k disjoint paths $P_1, \ldots, P_k$ such that P_i connects s_i and t_i for $i = 1, \ldots, k$, if they exist.

Description:

Step 1. Compute a $2k$-certificate G_{2k} of G in $O(m)$ time by the algorithm of [23], and go to Step 2.

Step 2. While the graph G has at least $2^{3k-3}|V(G)|$ edges, we detect a K_{3k}-minor by Theorem 4.3, and remove some vertices as in Theorem 4.2. If the graph has less than $2^{3k-3}|V(G)|$ edges, then go to Step 3.

Step 3. Test whether or not a current graph has tree-width at most $h(k)$, where $h(k)$ is as in Theorem 5.2. If it has, then Theorem 2.1 gives rise to a tree-decomposition of width at most $h(k)$. We then use Theorem 4.4 to solve the problem. If the graph has tree-width more than $h(k)$, go to Step 4.

Step 4. Apply Theorem 5.2 to the graph. If we get a K_{3k}-minor, then apply Theorem 4.2 to construct a smaller instance in $O(n)$ time. If we get a wall as in Theorem 5.2, then apply Theorem 4.6 to find an irrelevant vertex in $O(n)$ time. Then, delete the irrelevant vertex and go to Step 3.

First, the total running time of Step 2 is $O(n^2 + m)$ by Theorem 4.2. When we go to Step 3 after executing Step 2, the number of edges is $O(n)$, and hence it takes $O(n)$ time in Step 4 to find one irrelevant vertex in the graph. Since we delete at most $O(n)$ vertices, the total running time of Step 4 is $O(n^2)$. This completes the proof of Theorem 1.1.

7. Concluding remarks

We conclude this paper by observing that other problems can also be solved in quadratic time.

Corollary 7.1 *(The k edge-disjoint paths problem). There is an $O(n^2)$ time algorithm for the edge-disjoint paths problem when k is a fixed constant, i.e., for the problem of constructing paths $P_1, \ldots, P_k$ that are not necessarily vertex-disjoint but edge-disjoint.*

The k edge-disjoint paths problem can be reduced to the k vertex-disjoint paths problem by constructing the line graph. However, since the number of vertices of the line graph is $O(m)$, by using this naive reduction, the running time becomes $O(m^2)$. In order to improve the running time to $O(n^2)$, we first reduce the number of edges to $O(n)$ by executing the same procedures as Theorems 4.2 and 4.3. With this preprocessing, we can reduce the k edge-disjoint paths problem to the k vertex-disjoint paths problem in a graph with $O(n)$ vertices, which yields Corollary 7.1.

The similar proof to Theorem 1.1 also works for the following problems:

Corollary 7.2 *(The labeled minor testing). There is an $O(n^2)$ time algorithm for the labeled minor containment problem. That is, given a graph G and non-null subsets $Z_1, \ldots, Z_t \subseteq V(G)$, with $\sum_i |Z_i| \leqslant k$, where k is a fixed constant, we can test in $O(n^2)$ time whether or not there exist mutually vertex-disjoint connected subgraphs $G_1, \ldots, G_t$ of G, with $Z_i \subseteq V(G_i)$ $(1 \leqslant i \leqslant t)$.*

Corollary 7.3 *(The minor testing). There is an $O(n^2)$ time algorithm for the minor containment problem. That is, for a fixed graph H and a given graph G, we can test whether or not G has a minor isomorphic to H in $O(n^2)$ time.*

Corollaries 7.2 and 7.3 are obtained by following the proof of Theorem 1.1 in the same way as in [33]. We have to use the concept of "l-folio" instead of "disjoint paths" and "realizable partitions," but the same argument also works.

Finally, we note that the time complexity of all the algorithms with the most expensive part depending on Robertson and Seymour's algorithm can also be improved to $O(n^2)$, for example, the membership testing for a minor-closed class of graphs. There are many, so we omit them.

Please cite this article in press as: K. Kawarabayashi et al., The disjoint paths problem in quadratic time, J. Combin. Theory Ser. B (2011), doi:10.1016/j.jctb.2011.07.004

Acknowledgments

The authors thank the anonymous referees for helpful comments. In particular, one of the referees gave a lot of insightful comments and suggestions that improved an earlier version of this paper. The first author is supported by Japan Society for the Promotion of Science, Grant-in-Aid for Scientific Research, by C&C Foundation, by Kayamori Foundation and by Inoue Research Award for Young Scientists. The second author is supported by Japan Society for the Promotion of Science, Grant-in-Aid for Scientific Research, and by Global COE Program "The research and training center for new development in mathematics," MEXT, Japan.

References

[1] S. Arnborg, A. Proskurowski, Linear time algorithms for NP-hard problems restricted to partial k-trees, Discrete Appl. Math. 23 (1989) 11–24.
[2] H.L. Bodlaender, A linear-time algorithm for finding tree-decompositions of small treewidth, SIAM J. Comput. 25 (1996) 1305–1317.
[3] C. Chekuri, S. Khanna, B. Shepherd, The all-or-nothing multicommodity flow problem, in: Proc. 36th ACM Symposium on Theory of Computing (STOC), 2004, pp. 156–165.
[4] C. Chekuri, S. Khanna, B. Shepherd, Edge-disjoint paths in planar graphs, in: Proc. 45th Annual IEEE Symposium on Foundations of Computer Science (FOCS), 2004, pp. 71–80.
[5] C. Chekuri, S. Khanna, B. Shepherd, Multicommodity flow, well-linked terminals, and routing problems, in: Proc. 37th ACM Symposium on Theory of Computing (STOC), 2005, pp. 183–192.
[6] C. Chekuri, S. Khanna, B. Shepherd, Edge-disjoint paths in planar graphs with constant congestion, in: Proc. 38th ACM Symposium on Theory of Computing (STOC), 2006, pp. 757–766.
[7] R. Diestel, K.Y. Gorbunov, T.R. Jensen, C. Thomassen, Highly connected sets and the excluded grid theorem, J. Combin. Theory Ser. B 75 (1999) 61–73.
[8] Z. Dvořák, D. Král', R. Thomas, Coloring triangle-free graphs on surfaces, in: ACM–SIAM Symposium on Discrete Algorithms (SODA), 2009, pp. 120–129.
[9] A. Frank, Packing paths, cuts and circuits – a survey, in: B. Korte, L. Lovász, H.J. Prömel, A. Schrijver (Eds.), Paths, Flows, VLSI-Layout, Springer-Verlag, Berlin, 1990, pp. 49–100.
[10] R. Halin, S-functions for graphs, J. Geom. 8 (1976) 171–186.
[11] R. Kapadia, Z. Li, B. Reed, Two disjoint rooted paths in linear time, preprint.
[12] R.M. Karp, On the computational complexity of combinatorial problems, Networks 5 (1975) 45–68.
[13] K. Kawarabayashi, B. Reed, A nearly linear time algorithm for the half integral disjoint paths packing, in: ACM–SIAM Symposium on Discrete Algorithms (SODA), 2008, pp. 446–454.
[14] K. Kawarabayashi, P. Wollan, A shorter proof of the graph minors algorithm – the unique linkage theorem, in: Proc. 42nd ACM Symposium on Theory of Computing (STOC), 2010, pp. 687–694. A detailed version of this paper is available at http://research.nii.ac.jp/~k_keniti/uniquelink.pdf.
[15] J. Kleinberg, Single-source unsplittable flow, in: Proc. 37th Annual IEEE Symposium on Foundations of Computer Science (FOCS), 1996, pp. 68–77.
[16] J. Kleinberg, Decision algorithms for unsplittable flow and the half-disjoint paths problem, in: Proc. 30th ACM Symposium on Theory of Computing (STOC), 1998, pp. 530–539.
[17] J. Kleinberg, É. Tardos, Disjoint paths in densely embedded graphs, in: Proc. 36th Annual IEEE Symposium on Foundations of Computer Science (FOCS), 1995, pp. 52–61.
[18] J. Kleinberg, É. Tardos, Approximations for the disjoint paths problem in high-diameter planar networks, J. Comput. System Sci. 57 (1998) 61–73.
[19] Y. Kobayashi, K. Kawarabayashi, Algorithms for finding an induced cycle in planar graphs and bounded genus graphs, in: ACM–SIAM Symposium on Discrete Algorithms (SODA), 2009, pp. 1146–1155.
[20] S. Kolliopoulos, C. Stein, Improved approximation algorithms for unsplittable flow problems, in: Proc. 38th Annual IEEE Symposium on Foundations of Computer Science (FOCS), 1997, pp. 426–435.
[21] J.F. Lynch, The equivalence of theorem proving and the interconnection problem, ACM SIGDA Newsletter 5 (1975) 31–65.
[22] M. Middendorf, F. Pfeiffer, On the complexity of the disjoint paths problem, Combinatorica 13 (1993) 97–107.
[23] H. Nagamochi, T. Ibaraki, A linear-time algorithm for finding a sparse k-connected spanning subgraph of a k-connected graph, Algorithmica 7 (1992) 583–596.
[24] L. Perković, B. Reed, An improved algorithm for finding tree decompositions of small width, Internat. J. Found. Comput. Sci. 11 (2000) 365–371.
[25] B. Reed, Finding approximate separators and computing tree width quickly, in: Proc. 24th ACM Symposium on Theory of Computing (STOC), 1992, pp. 221–228.
[26] B. Reed, Rooted routing in the plane, Discrete Appl. Math. 57 (1995) 213–227.
[27] B. Reed, Tree width and tangles: a new connectivity measure and some applications, in: Surveys in Combinatorics, in: London Math. Soc. Lecture Notes Series, vol. 241, Cambridge Univ. Press, Cambridge, 1997, pp. 87–162.
[28] B. Reed, D.R. Wood, A linear-time algorithm to find a separator in a graph excluding a minor, ACM Trans. Algorithms 5 (39) (2009).

Please cite this article in press as: K. Kawarabayashi et al., The disjoint paths problem in quadratic time, J. Combin. Theory Ser. B (2011), doi:10.1016/j.jctb.2011.07.004

[29] B. Reed, N. Robertson, A. Schrijver, P.D. Seymour, Finding disjoint trees in planar graphs in linear time, in: Contemp. Math., vol. 147, Amer. Math. Soc., Providence, RI, 1993, pp. 295–301.
[30] N. Robertson, P.D. Seymour, Graph minors. II. Algorithmic aspects of tree-width, J. Algorithms 7 (1986) 309–322.
[31] N. Robertson, P.D. Seymour, Graph minors. V. Excluding a planar graph, J. Combin. Theory Ser. B 41 (1986) 92–114.
[32] N. Robertson, P.D. Seymour, An outline of a disjoint paths algorithm, in: B. Korte, L. Lovász, H.J. Prömel, A. Schrijver (Eds.), Paths, Flows, and VLSI-Layout, Springer-Verlag, Berlin, 1990, pp. 267–292.
[33] N. Robertson, P.D. Seymour, Graph minors. XIII. The disjoint paths problem, J. Combin. Theory Ser. B 63 (1995) 65–110.
[34] N. Robertson, P.D. Seymour, Graph minors. XVI. Excluding a non-planar graph, J. Combin. Theory Ser. B 89 (2003) 43–76.
[35] N. Robertson, P. Seymour, Graph minors. XXI. Graphs with unique linkages, J. Combin. Theory Ser. B 99 (2009) 583–616.
[36] N. Robertson, P.D. Seymour, R. Thomas, Quickly excluding a planar graph, J. Combin. Theory Ser. B 62 (1994) 323–348.
[37] P.D. Seymour, Disjoint paths in graphs, Discrete Math. 29 (1980) 293–309.
[38] R.E. Tarjan, Data Structures and Network Algorithms, SIAM, Philadelphia, PA, 1983.
[39] T. Tholey, Solving the 2-disjoint paths problem in nearly linear time, Theory Comput. Syst. 39 (2006) 51–78.
[40] T. Tholey, Improved algorithms for the 2-vertex disjoint paths problem, in: Proc. 35th Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM), in: LNCS, vol. 5404, 2009, pp. 546–557.
[41] C. Thomassen, 2-Linked graphs, European J. Combin. 1 (1980) 371–378.

Please cite this article in press as: K. Kawarabayashi et al., The disjoint paths problem in quadratic time, J. Combin. Theory Ser. B (2011), doi:10.1016/j.jctb.2011.07.004