ELSEVIER

Discrete Applied Mathematics 115 (2001) 177–186

DISCRETE
APPLIED
MATHEMATICS

The edge-disjoint paths problem is NP-complete for series–parallel graphs

Takao Nishizeki[a], Jens Vygen[b,*], Xiao Zhou[c]

[a] *Graduate School of Information Sciences, Tohoku University, Aoba-yama 05, Sendai 980-8579, Japan*
[b] *Research Institute for Discrete Mathematics, University of Bonn, Lennéstrasse 2, 53113 Bonn, Germany*
[c] *Graduate School of Information Sciences, Tohoku University, Aoba-yama 05, Sendai 980-8579, Japan*

Received 23 March 1999; received in revised form 4 April 2000; accepted 30 March 2001

Abstract

Many combinatorial problems are NP-complete for general graphs. However, when restricted to series–parallel graphs or partial k-trees, many of these problems can be solved in polynomial time, mostly in linear time. On the other hand, very few problems are known to be NP-complete for series–parallel graphs or partial k-trees. These include the subgraph isomorphism problem and the bandwidth problem. However, these problems are NP-complete even for trees. In this paper, we show that the edge-disjoint paths problem is NP-complete for series–parallel graphs and for partial 2-trees although the problem is trivial for trees and can be solved for outerplanar graphs in polynomial time. © 2001 Elsevier Science B.V. All rights reserved.

1. Introduction

Many combinatorial problems are NP-complete for general graphs, and are unlikely to be solvable in polynomial time. However, many "natural" problems defined on unweighted graphs can be efficiently solved for series–parallel graphs or partial k-trees (graphs of treewidth bounded by a constant k) in polynomial time or mostly in linear time [1–4,23,24,29]. On the other hand, very few problems are known to be NP-complete for series–parallel graphs or partial k-trees. These include the subgraph isomorphism problem and the bandwidth problem [5,11,15,22]. However, these problems are NP-complete even for ordinary trees [8].

The edge-disjoint paths problem asks whether there exist p pairwise edge-disjoint paths P_i, $1 \leqslant i \leqslant p$, connecting terminals s_i and t_i in a given graph G with p terminal

[*] Corresponding author.

E-mail addresses: nishi@ecei.tohoku.ac.jp (T. Nishizeki), vygen@or.uni-bonn.de (J. Vygen), zhou@ecei.tohoku.ac.jp (X. Zhou).

0166-218X/01/$ - see front matter © 2001 Elsevier Science B.V. All rights reserved.
PII: S0166-218X(01)00223-2

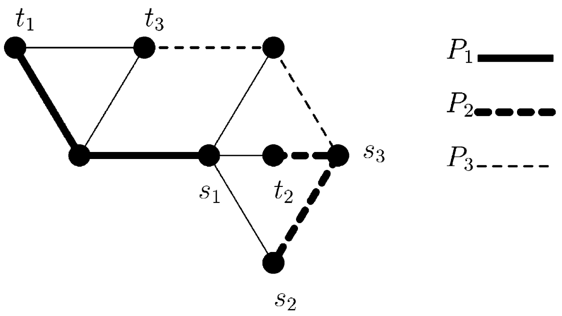


Fig. 1. Three edge-disjoint paths P_1, P_2 and P_3 in a series–parallel graph.

pairs (s_i,t_i), $1 \leqslant i \leqslant p$, assigned to vertices of G. Fig. 1 illustrates three edge-disjoint paths P_1, P_2 and P_3 in a series–parallel graph. The vertex-disjoint paths problem is similarly defined. These problems come up naturally when analyzing connectivity questions or generalizing (integral) network flow problems. Another reason for the growing interest is the variety of applications, e.g. in VLSI-design and communication [12,13,16,20,21,28]. If $p=O(1)$, then the vertex-disjoint paths problem can be solved in polynomial time for any graph by Robertson and Seymour's algorithm based on their series of papers on graph minor theory [17,18]. The edge-disjoint paths problem on a graph G can be reduced in polynomial time to the vertex-disjoint paths problem on a new graph similar to the line graph of G. Therefore, the edge-disjoint paths problem can also be solved in polynomial time for general graphs if $p=O(1)$. However, if p is not bounded, then both the edge-disjoint and vertex-disjoint paths problems are NP-complete even for planar graphs [14,26]. For a survey, see [7,25].

A natural question is whether the vertex-disjoint and edge-disjoint paths problems can be efficiently solved for a restricted class of graphs, say series–parallel graphs or partial k-trees. Indeed Scheffler showed that the vertex-disjoint paths problem can be solved in linear time for partial k-trees even if p is not bounded [19]. Frank obtained a necessary and sufficient condition for the existence of edge-disjoint paths in a class of planar graphs [6]. His result together with the algorithms in [13,28] yields a polynomial-time algorithm for the edge-disjoint paths problem on outerplanar graphs. Note that outerplanar graphs are series–parallel.

On the other hand, Zhou et al. showed that the edge-disjoint paths problem can be solved in polynomial time for partial k-trees if either $p=O(\log n)$ or the location of terminals satisfies some condition, where n denotes the number of vertices in a given partial k-tree [32]. However, it has not been known whether the edge-disjoint paths problem is NP-complete for series–parallel graphs or partial k-trees if there is no restriction on the number of terminal pairs or the location of terminals.

In this paper we show that the edge-disjoint paths problem is NP-complete for series–parallel graphs and for partial 2-trees. To the best of our knowledge this is the first example of problems which are efficiently solvable for outer-planar graphs but NP-complete for series–parallel graphs or partial 2-trees.

2. Main theorem

In this section, we first give some definitions. The paper deals with *undirected* graphs without self-loops. A graph is called a *simple graph* if it has no parallel edges. Let $G=(V,E)$ denote a graph with vertex set V and edge set E. We often denote by $V(G)$ and $E(G)$ the vertex set and the edge set of G, respectively. For two disjoint subsets U and W of $V(G)$, we denote by $E_G(U,W)$ a set of edges e in G such that an end of e is in U and the other end is in W. An edge joining vertices u and v is denoted by $\{u,v\}$. It is well-known that a graph is *series–parallel* if and only if it can be constructed from a single edge by two operations: doubling an edge and subdividing an edge [23]. Furthermore, a simple graph is a *partial* 2*-tree* (or, equivalently, has tree-width at most 2) if and only if each biconnected component is series–parallel. Thus, a series–parallel graph having no parallel edges is a partial 2-tree.

Our main result is the following theorem.

Theorem 2.1. *The edge-disjoint paths problem is NP-complete for series–parallel graphs.*

In the remainder of this section we will give a proof of Theorem 2.1. The Minimum Graph Bisection problem is defined as follows: Given an undirected graph H with $2n$ vertices and a positive integer c, does there exist a partition $V(H)=U\dot{\cup}W$ with $|U|=|W|=n$ and $|E_H(U,W)|\leqslant c$? This problem was shown to be NP-complete by Garey et al. [9]. Clearly the edge-disjoint paths problem is in NP [8]. Therefore it suffices to show that the Minimum Graph Bisection problem can be transformed in polynomial time to the edge-disjoint paths problem for series–parallel graphs.

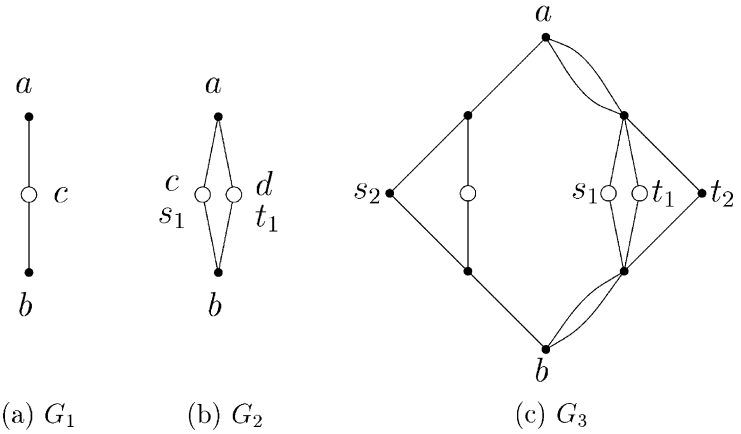
Given a graph H and a positive integer c (i.e. an instance of Minimum Graph Bisection), we construct an instance of the edge-disjoint paths problem. It will consist of a series–parallel graph G, and some number of terminal pairs, say p pairs $(s_1,t_1),(s_2,t_2),\dots,(s_p,t_p)$. We shall prove that there are edge-disjoint paths $P_1,P_2,\dots,P_p$ from s_1 to t_1, from s_2 to t_2 and so on if and only if an instance (H,c) of Minimum Graph Bisection is an yes-instance, i.e. H has a bisection with at most c edges.

To describe our construction, we need a gadget G_j defined for each $j\geqslant 1$ as follows: G_j consists of a graph, a subset T of j vertices in the graph, and $j-1$ terminal pairs $(s_1,t_1),(s_2,t_2),\dots,(s_{j-1},t_{j-1})$. The graph in each gadget can be constructed from a single edge $\{a,b\}$ by doubling and subdividing edges, so it will be series–parallel.

For $j=1$, we just subdivide an edge $\{a,b\}$, obtain a new vertex c, and set $T{:=}\{c\}$. The resulting gadget is G_1. (See Fig. 2(a) where the vertex c in T is drawn by a white circle.)

For $j=2$, we double an edge $\{a,b\}$, and subdivide each of the two resulting edges; let c and d be the two new vertices. We set $T{:=}\{c,d\}$, $s_1{:=}c$, and $t_1{:=}d$. The resulting gadget is G_2. (See Fig. 2(b).)

For $j>2$, we start with the graph depicted in Fig. 3, with a terminal pair (s_{j-1},t_{j-1}). The numbers at the edges $\{a,c\}$, $\{d,b\}$, $\{a,e\}$ and $\{f,b\}$ indicate that many parallel



(a) G_1 (b) G_2 (c) G_3

Fig. 2. Gadgets G_1, G_2 and G_3 where the vertices in T are drawn by white circles.

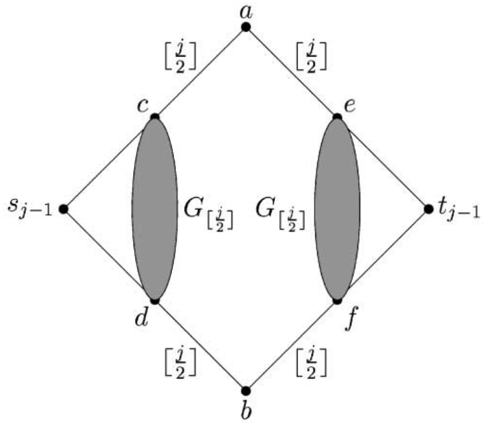


Fig. 3. Construction of G_j.

edges. Now we replace the edges $\{c,d\}$ and $\{e,f\}$ by the gadgets $G_{\lfloor j/2\rfloor}$ and $G_{\lceil j/2\rceil}$, respectively. We get a total of $(\lfloor j/2\rfloor - 1) + (\lceil j/2\rceil - 1) + 1 = j - 1$ terminal pairs and a set T of $\lfloor j/2\rfloor + \lceil j/2\rceil = j$ vertices. Clearly G_j is obtained from a single edge $\{a,b\}$ by two operations: doubling an edge and subdividing an edge. Thus G_j is a series–parallel graph. One can easily observe that G_j has $O(j)$ vertices.

This completes the description of the gadgets. The gadgets G_3 and G_7 are illustrated in Figs. 2(c) and 4, respectively.

Now the graph G is constructed as follows: We start from a cycle C_4 of length 4, with vertices A, B, D, C in this order. (See Fig. 5.) Then we replace the edge $\{A,C\}$ of C_4 by $2n$ parallel edges, each of which corresponds to one of the $2n$ vertices in H. Now, for each vertex $v \in V(H)$, the edge $\{A,C\}$ corresponding to v is replaced

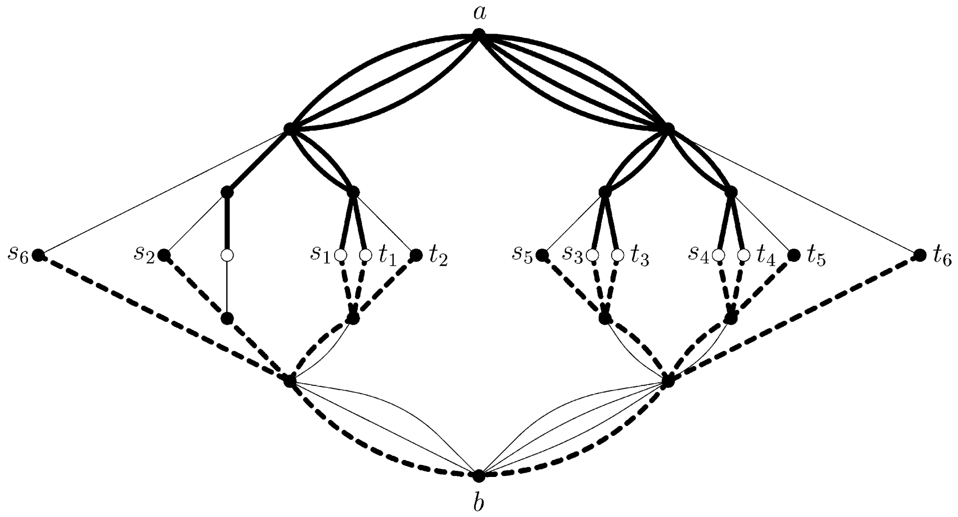


Fig. 4. The gadget G_7.

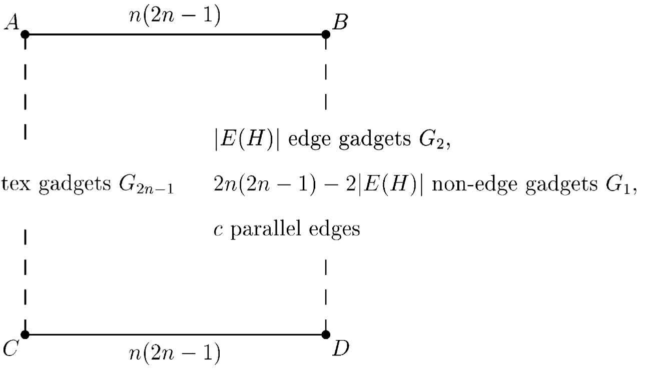

A
$n(2n-1)$
B
$|E(H)|$ edge gadgets G_2,
$2n$ vertex gadgets G_{2n-1}
$2n(2n-1)-2|E(H)|$ non-edge gadgets G_1,
c parallel edges
C
$n(2n-1)$
D

Fig. 5. Construction of graph G.

by the gadget G_{2n-1}, and the $2n-1$ vertices in T of G_{2n-1} are marked by $s_{(v,w)}$, $w \in V(H) \setminus \{v\}$ (in addition to the marks these vertices have already received by the gadget construction). These $2n$ gadgets G_{2n-1} are called *vertex gadgets*. (See Fig. 6.)

On the other hand, we replace the edge $\{B,D\}$ of C_4 by $2n(2n-1)-|E(H)|+c$ parallel edges. For each edge $e=\{v,w\} \in E(H)$, we take one of these edges $\{B,D\}$ and replace it by the gadget G_2, marking the two vertices in T of G_2 by $t_{(v,w)}$ and $t_{(w,v)}$. These $|E(H)|$ gadgets G_2 are called *edge gadgets*. Moreover, for each ordered pair (v,w) of vertices $v,w \in V(H)$ such that $v \neq w$ and $\{v,w\} \notin E(H)$, we take one of these edges $\{B,D\}$ and replace it by G_1 (i.e. subdivide it) and mark the vertex in T of G_1 by $t_{(v,w)}$. These $2n(2n-1)-2|E(H)|$ gadgets G_1 are called *non-edge gadgets*. The remaining c parallel edges joining B and D are left unchanged.

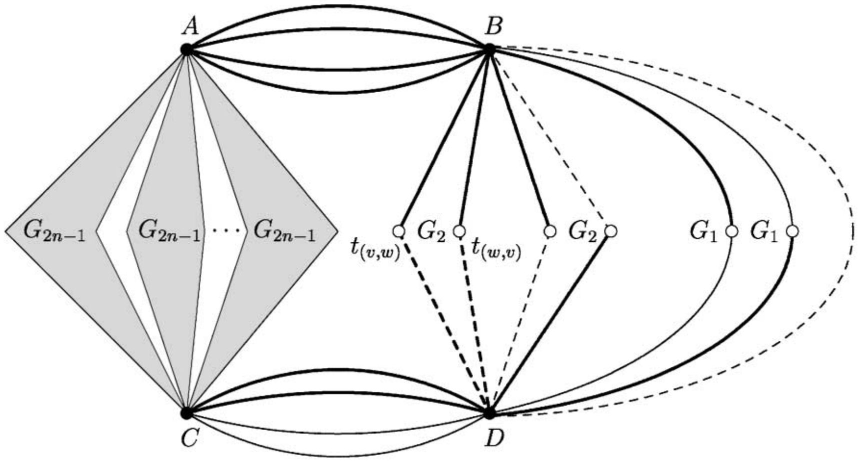


Fig. 6. Paths in G for terminals in edge gadgets and non-edge gadgets.

Finally we replace the edge $\{A,B\}$ of C_4 by $n(2n-1)$ parallel edges, and we replace $\{C,D\}$ by $n\ (2n-1)$ parallel edges. This completes the construction of graph G; see Figs. 5 and 6. Clearly G is obtained from C_4 by two operations: doubling an edge and subdividing an edge. Hence G is a series–parallel graph. Furthermore, both the number of vertices in G and the number p of terminal pairs are clearly bounded by a polynomial in n; more precisely, $|V(G)| = O(n^2)$ and $p = O(n^2)$. Thus one can construct G from H in polynomial time.

The deletion of the $2n\ (2n-1)$ edges $\{A,B\}$ and $\{C,D\}$ from G separates exactly $2n\ (2n-1)$ terminal pairs $(s_{(v,w)}, t_{(v,w)})$, each consisting of two terminals in different gadgets, where $v \in V(H)$ and $w \in V(H) \setminus \{v\}$. Therefore, all the edges $\{A,B\}$ and $\{C,D\}$ in G must be used by the paths for these terminal pairs. It is easy to see that all terminal pairs introduced within the same vertex gadget can be realized completely within that gadget, with shortest paths (of length 2 or 4); either all with shortest paths around the bottom or all with shortest paths around the top. In Fig. 4 the dotted lines indicate the shortest paths around the bottom.

Suppose that there is a bisection of H into n red vertices in U and n blue vertices in W such that $|E_H(U,W)| \leqslant c$. Then we can find edge-disjoint paths in G as follows.

For each red vertex $v \in U$, we realize all the $2n-2$ terminal pairs introduced with the vertex gadget G_{2n-1} for v by shortest paths within the gadget, all around the bottom, and we let each of the $2n-1$ paths for $s_{(v,w)}$, $w \in V(H) \setminus \{v\}$, pass the gadgets in which it is contained towards the top, then use an edge $\{A,B\}$ and finally use an edge $\{B, t_{(v,w)}\}$ in an edge gadget G_2 or a non-edge gadget G_1. (In Fig. 4 the paths starting at $s_{(v,w)}$ toward the top in G_7 are drawn by thick solid lines. In Fig. 6 the paths for terminals $t_{(v,w)}$ are drawn by thick solid lines.) For blue vertices in W we do the opposite: we realize all the $2n-2$ terminal pairs in the vertex gadget by shortest paths around the top, and we let each of the $2n-1$ paths for $s_{(v,w)}$ pass the gadget towards the bottom and use edges $\{C,D\}$ and $\{D,t_{(v,w)}\}$. This can be done because H has n red vertices and n blue vertices, and G has exactly $2n(2n-1)$ terminal pairs $(s_{(v,w)},t_{(v,w)})$ separated by the $2n(2n-1)$ edges $\{A,B\}$ and $\{C,D\}$ where $v\in V(H)$ and $w\in V(H)\setminus\{v\}$.

If the ends v and w of an edge $e=\{v,w\}\in E(H)$ have the same color, then the following (i) and (ii) hold:

(i) both paths, from $s_{(v,w)}$ to $t_{(v,w)}$ and from $s_{(w,v)}$ to $t_{(w,v)}$, enter the edge gadget G_2 for e at the same vertex; either both enter the gadget at B or both at D, depending on whether v is red or blue; and hence

(ii) we can realize the path connecting the two terminals in T of the edge gadget G_2 for e by a path of length 2 using none of the c parallel edges $\{D,B\}$ as indicated by thick dotted lines in Fig. 6.

On the other hand, if the ends v and w of e have different colors, then we realize the path connecting the two terminals in the edge gadget G_2 for e by a path of length 3 using one of the c edges $\{D,B\}$ as indicated by thin dotted lines in Fig. 6. This can be done because the cut of the bisection of H has at most c edges whose ends have different colors. (In Fig. 6 thick lines indicate (a) paths for terminals in an edge gadget for an edge with two red ends, (b) paths for terminals in an edge gadget for an edge with a red end and a blue end, and (c) paths for terminals in two non-edge gadgets, while dotted lines indicate paths for terminal pairs within edge-gadgets.)

Conversely, suppose that G has edge-disjoint paths connecting the terminal pairs. Clearly all the edges $\{A,B\}$ and $\{C,D\}$ in G are used by paths for terminal pairs $(s_{(v,w)},t_{(v,w)})$ where $v\in V(H)$ and $w\in V(H)\setminus\{v\}$. One can easily observe that

(i) all terminal pairs introduced within the same vertex gadget are realized completely within that gadget with shortest paths (of length 2 or 4); either all with shortest paths around the bottom or all with shortest paths around the top; and

(ii) all the $2n-2$ terminal pairs within the vertex gadget G_{2n-1} for a vertex $v\in V(H)$ are connected by shortest paths around the bottom if and only if all the $2n-1$ terminals $s_{(v,w)}$, $w\in V(H)\setminus\{v\}$, in T of the gadget G_{2n-1} are realized by paths towards the top.

Thus, for each vertex $v\in V(H)$, the $2n-1$ paths starting at $s_{(v,w)}$, $w\in V(H)\setminus\{v\}$, either all pass A and B or all pass C and D in G. In the first case we call v red, otherwise blue. (The vertex gadget in Fig. 4 corresponds to a red vertex.) Since there are $n(2n-1)$ edges $\{A,B\}$ and $n(2n-1)$ edges $\{C,D\}$ in G, we have n red and n blue vertices in H, inducing a bisection of H.

We claim that at most c edges are in the cut of this bisection of H. Consider an edge $e=\{v,w\}\in E(H)$ and the following three paths in the solution of the edge-disjoint paths problem: the path from $s_{(v,w)}$ to $t_{(v,w)}$, the path from $s_{(w,v)}$ to $t_{(w,v)}$, and the path connecting the two terminals in T of the edge gadget G_2 for e. If none of these three paths uses an edge $\{B,D\}$, then the third path has length 2 while the other two paths either both use an edge $\{A,B\}$ or both $\{C,D\}$, and hence v and w have the same color. Thus, if e is in the cut of H, that is, v and w have different colors, then at least one of the three paths uses one of the parallel edges $\{B, D\}$. Since there are only c parallel edges $\{B, D\}$ in G, there are at most c edges in the cut of H.

Thus we have proved Theorem 2.1.

3. Conclusion

In this paper, we showed that the edge-disjoint paths problem is NP-complete for series–parallel graphs G. Replace each set of parallel edges in G with the same number of parallel paths of length 2, then the resulting graph G' is a series–parallel graph having no parallel edges and hence is a partial 2-tree. Clearly G has edge-disjoint paths if and only if the resulting partial 2-tree G' has edge-disjoint paths. Thus the edge-disjoint paths problem is NP-complete for partial 2-trees, too.

The edge-disjoint paths problem is a decision problem, i.e., a YES-NO problem. The optimization version is the maximum edge-disjoint paths problem which asks to find a maximum number of edge-disjoint paths connecting terminal pairs in a given graph. Garg et al. [10] showed that the maximum multicommodity integral flow problem is NP-hard for a tree with edge-capacities 1 or 2. Replace each edge of capacity 2 in the tree with a pair of parallel paths of length 2, then the resulting graph is an outerplanar graph and hence is a partial 2-tree. Thus the result in [10] immediately implies that the maximum edge-disjoint paths problem is NP-hard for outerplanar graphs or partial 2-trees. However, it does not imply our result because the NP-hardness of an optimization problem does not always imply the NP-completeness of the decision problem.

Zhou et al. proved the following fact: the edge-disjoint paths problem can be solved in polynomial time for a partial k-tree G if the augmented graph G^+ obtained from G by adding an edge joining s_i and t_i for each pair of terminals (s_i, t_i) remains to be a partial k-tree [32]. Our result in this paper does not conflict with the fact above, because the augmented graph G^+ of the graph G constructed by our reduction is not always a partial k-tree with bounded k.

A class of *tractable* problems for series–parallel graphs or partial k-trees has been characterized [1–4,23]. It remains open to characterize a class of *intractable* problems, including the edge-disjoint paths problem, the subgraph isomorphism problem, and the bandwidth problem, for series–parallel graphs or partial k-trees.

András Frank observed that the f-factor problem reduces to the edge-disjoint paths problem in double-stars (graphs where two vertices cover all the edges). He raised the question of finding a common generalization of f-factors and edge-disjoint paths in outerplanar graphs. A natural graph family containing outerplanar graphs and double-stars are the partial 2-trees. However, our main result shows that this family is already too big to expect a good characterization.

Acknowledgements

Thanks to András Frank for giving us a chance to merge independent and simultaneous results [27,30,31] to the single paper.

References

[1] S. Arnborg, B. Courcelle, A. Proskurowski, D. Seese, An algebraic theory of graph reduction, J. Assoc. Comput. Mach. 40 (5) (1993) 1134–1164.

[2] S. Arnborg, J. Lagergren, D. Seese, Easy problems for tree-decomposable graphs, J. Algorithms 12 (2) (1991) 308–340.

[3] R.B. Borie, R.G. Parker, C.A. Tovey, Automatic generation of linear-time algorithms from predicate calculus descriptions of problems on recursively constructed graph families, Algorithmica 7 (1992) 555–581.

[4] B. Courcelle, The monadic second-order logic of graphs I: Recognizable sets of finite graphs, Inform. and Comput. 85 (1990) 12–75.

[5] A. Dessmark, A. Lingas, A. Proskurowski, Faster algorithms for subgraph isomorphism of k-connected partial k-trees. Proceedings of the Fourth European Symposium on Algorithms, Lecture Notes in Computer Science, Vol. 1136, Springer, Berlin, 1996, pp. 501–513.

[6] A. Frank, Edge-disjoint paths in planar graphs, J. Combin. Theory Ser. B 39 (1985) 164–178.

[7] A. Frank, Packing paths, circuits and cuts—a survey, in: B. Korte, L. Lovász, H.J. Prömel, A. Schrijver (Eds.), Paths, Flows, and VLSI-Layout, Springer, Berlin, 1990.

[8] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman, New York, 1979.

[9] M.R. Garey, D.S. Johnson, L. Stockmeyer, Some simplified NP-complete graph problems, Theoret. Comput. Sci. 1 (1976) 237–267.

[10] N. Garg, V.V. Vazirani, M. Yannakakis, Primal–dual approximation algorithms for integral flow and multicut in trees, Algorithmica 18 (1997) 3–20.

[11] A. Gupta, N. Nishimura, The complexity of subgraph isomorphism for classes of partial k-trees, Theoret. Comput. Sci. 164 (1996) 287–297.

[12] J. Kleinberg, É. Tardos, Approximations for the disjoint paths problem in high-diameter planar networks, J. Comput. System Sci. 57 (1998) 61–73.

[13] K. Matsumoto, T. Nishizeki, N. Saito, An efficient algorithm for finding multi-commodity flows in planar networks, SIAM J. Comput. 14 (2) (1985) 289–302.

[14] M. Middendorf, F. Pfeiffer, On the complexity of disjoint paths problem, Combinatorica 13 (1) (1993) 97–107.

[15] J. Matoušek, R. Thomas, On the complexity of finding iso- and other morphisms for partial k-trees, Discrete Math. 108 (1992) 343–364.

[16] H. Ripphausen-Lipa, D. Wagner, K. Weihe, Efficient algorithms for disjoint paths in planar graphs, DIMACS Ser. Discrete Math. Theoret. Comput. Sci. 20 (1995) 295–354.

[17] N. Robertson, P.D. Seymour, Graph minors IX, Disjoint crossed paths, J. Combin. Theory Ser. B 49 (1) (1990) 40–47.

[18] N. Robertson, P.D. Seymour, Graph minors XIII, The disjoint paths problem, J. Combin. Theory Ser. B 63 (1) (1995) 65–110.

[19] P. Scheffler, A practical linear time algorithm for disjoint paths in graphs with bound tree-width, Technical Report 396, Department of Mathematics, Technische Universität Berlin, 1994.

[20] H. Suzuki, A. Ishiguro, T. Nishizeki, Edge-disjoint paths in a grid bounded by two nested rectangles, Discrete Appl. Math. 27 (1990) 157–178.

[21] H. Suzuki, T. Nishizeki, N. Saito, Algorithms for multicommodity flows in planar graphs, Algorithmica 4 (1989) 471–501.

[22] M. Sysło, NP-complete problems on some tree-structured graphs: a review, Proceedings of the WG'83 International Workshop on Graph Theoretic Concepts in Computer Science, University of Verlag Rudolf Trauner, Linz, West Germany, 1983, pp. 478–489.

[23] K. Takamizawa, T. Nishizeki, N. Saito, Linear-time computability of combinatorial problems on series–parallel graphs, J. ACM 29 (3) (1982) 623–641.

[24] J.A. Telle, A. Proskurowski, Algorithms for vertex partitioning problems on partial k-trees, SIAM J. Discrete Math. 10 (1997) 529–550.

[25] J. Vygen, Disjoint paths, Report No. 94816, Research Institute for Discrete Mathematics, University of Bonn, 1994.

[26] J. Vygen, NP-completeness of some edge-disjoint paths problems, Discrete Appl. Math. 61 (1995) 83–90.

[27] J. Vygen, Edge-disjoint paths in series–parallel graphs, Report No. 98871, Research Institute for Discrete Mathematics, University of Bonn, 1998.

[28] D. Wagner, K. Weihe, A linear-time algorithm for edge-disjoint paths in planar graphs, Combinatorica 15 (1995) 135–150.

[29] X. Zhou, S. Nakano, T. Nishizeki, Edge-coloring partial k-trees, J. Algorithms 21 (1996) 598–617.

[30] X. Zhou, T. Nishizeki, The edge-disjoint paths problem is NP-complete for partial k-trees, Proceedings of the Ninth International Symposium on Algorithms and Computation, Lecture Notes in Computer Science, Springer, Berlin, 1998.

[31] X. Zhou, T. Nishizeki, The edge-disjoint paths problem is NP-complete for series–parallel graphs, Technical Report 9801, Department of System Information Sciences, Graduate School of Information Sciences, Tohoku University, 1998.

[32] X. Zhou, S. Tamura, T. Nishizeki, Finding edge-disjoint paths in partial k-trees, Algorithmica 26 (2000) 3–30.