氏名（本籍地）	伊藤　健洋（いとう　たけひろ）
学 位 の 種 類	博　士（情報科学）
学 位 記 番 号	情　博　第　354　号
学位授与年月日	平成18年３月24日
学位授与の要件	学位規則第４条第１項該当
研 究 科 、専 攻	東北大学大学院情報科学研究科（博士課程）システム情報科学専攻
学 位 論 文 題 目	Algorithms for Partitioning and Coloring Graphs（グラフの分割と彩色に関する研究）
論 文 審 査 委 員	（主査）東北大学教授　西関　隆夫
	東北大学教授　徳山　豪　東北大学教授　篠原　歩
	オレゴン大学教授　Andrzej Proskurowski（アメリカ）
	東北大学助教授　周　暁

論 文 内 容 の 要 旨

1 Introduction

In this thesis we deal with four topics of graph partitionings and colorings, and hence our results can be divided into the following four parts: Chapter 3 deals with partitioning graphs of supply and demand; Chapter 4 focuses on uniform partitions of vertex-weighted graphs; Chapter 5 deals with distance-edge-colorings; and Chapter 6 focuses on multicolorings.

A partial k-tree is a graph with tree-width bounded by a fixed constant k [2, 3]. Graph partitioning problems in Chapters 3 and 4 are NP-hard even for partial k-trees, although a number of combinatorial problems including many NP-hard problems on general graphs can be solved very efficiently for partial k-trees. Telle and Proskurowski present a theory of algorithm design for a large class of vertex partitioning problems restricted to partial k-trees [3], but our problems do not belong to their class. Similarly, coloring problems in Chapters 5 and 6 do not belong to any known class of problems which are solvable in polynomial time for partial k-trees.

We give several efficient algorithms for the four topics on partial k-trees. Our algorithms are based on a clever and detailed formation of dynamic programming.

2 Preliminaries

We assume that k is a bounded positive integer. A *k-tree* is defined recursively as follows [2]:

(1) A complete graph with $k+1$ vertices is a k-tree.

(2) If G is a k-tree and k vertices induce a complete subgraph of G, then a graph obtained from G by adding a new vertex and joining it with each of the k vertices is a k-tree.

Every subgraph of a k-tree is called a *partial k-tree*.

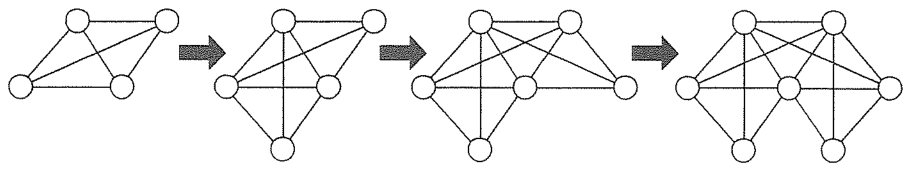

Figure 1: A process of generating 3-trees.

Figure 1 illustrates a process of generating 3-trees. The graphs in Figs. 4 and 5 are indeed partial 3-trees since they are subgraphs of the last 3-tree in Fig. 1.

A partial k-tree can be represented by a binary tree, called a *tree-decomposition* [2]. Since a tree-decomposition of a partial k-tree can be found in linear time [2], we may assume that a partial k-tree and its tree-decomposition are given. We solve many problems in this thesis by a dynamic programming approach based on a tree-decomposition.

3 Partitioning graphs of supply and demand

Let $G = (V, E)$ be a graph with vertex set V and edge set E. The set V is partitioned into two sets V_s and V_d. Let $|V| = n$, $|V_s| = n_s$ and $|V_d| = n_d$, then $n = n_s + n_d$. Each vertex $u \in V_s$ is called a *supply vertex* and is assigned a positive real number sup(u), called a *supply of* u, while each vertex $v \in V_d$ is called a *demand vertex* and is assigned a positive real number dem(v), called a *demand of* v. Each demand vertex can receive "power" from at most one supply vertex through edges in G. One thus wishes to partition G into exactly n_s connected components by deleting edges from G so that each component C has exactly one supply vertex whose supply is no less than the sum of demands of all demand vertices in C. For example, the graph in Fig. 2(a) has such a partition, where each supply vertex is drawn as a rectangle and each demand vertex as a circle, the supply or demand is written inside, the deleted edges are drawn by thick dotted lines, and each connected component is indicated by a thin dotted line. However, such a partition does not always exist. The *partition problem* is a decision problem to ask whether G has such a partition. If G has no such partition, then we wish to partition G into connected components so that each component C either has no supply vertex or has exactly one supply vertex whose supply is no less than the sum of demands of all demand vertices in C, and wish to maximize the "fulfillment," that is, the sum of demands of the demand vertices in all components with supply vertices. We call the problem the *maximum partition problem*. Clearly, the maximum partition problem is a generalization of the partition problem. Figure 2(b) illustrates a solution of the maximum partition problem for a tree, whose

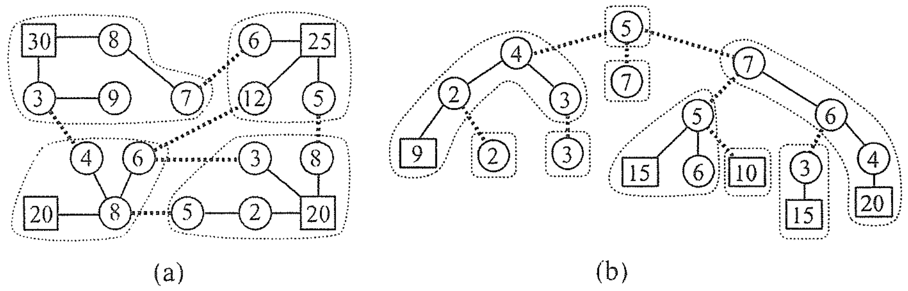

Figure 2: (a) Partition of a graph, and (b) partition of a tree with maximum fulfillment.

fulfillment is $(2+4+3)+(5+6)+3+(4+6+7)=40$.

The partition problem is NP-complete for series-parallel graphs, and the maximum partition problem is NP-hard even for trees. Hence it is very unlikely that these problems can be solved in polynomial time in general. Moreover, the partition problem is strongly NP-complete and the maximum partition problem is strongly NP-hard for general graphs. Therefore, there is no pseudo-polynomial-time algorithm for the two partition problems on general graphs unless P = NP. However, in this chapter, we give several efficient algorithms to solve the two partition problems for trees, series-parallel graphs and partial k-trees.

We first obtain the following theorems for trees.

Theorem 1 *The partition problem can be solved in linear time for trees.*

Theorem 2 *The maximum partition problem can be solved in time $O(F^2n)$ for a tree $T=(V,E)$ if the demands and supplies are integers and $F=\min\{\sum_{v\in V_d} dem(v), \sum_{v\in V_s} sup(v)\}$.*

The algorithm in Theorem 2 can be extended to an algorithm which finds a partition P of a tree T with $f(P)\geq(1-\varepsilon)f(T)$ in time polynomial in n and $1/\varepsilon$ for any real number ε, $0<\varepsilon<1$, where $f(P)$ is the fulfillment of P and $f(T)$ is the maximum fulfillment of T. Thus we have the following theorem.

Theorem 3 *There is a fully polynomial-time approximation scheme for the maximum partition problem on trees.*

We then obtain pseudo-polynomial-time algorithms to solve the partition problem and the maximum partition problem for series-parallel graphs and partial k-trees, as follows.

Theorem 4 *The partition problem can be solved in time $O(m_s^2n)$ and $O(m_s^{2k}n)$ for series-parallel graphs and partial k-trees, respectively, and the maximum partition problem can be solved in time $O(m_s^4n)$ and $O\bigl(m_s^{2(k+1)}n\bigr)$ for series-parallel graphs and partial k-trees, respectively, where m_s is the maximum supply.*

4 Uniform partitions

Let $G=(V,E)$ be a graph with vertex set V and edge set E, and let $|V|=n$. Assume that each vertex $v\in V$ is assigned a nonnegative integer $\omega(v)$, called the *weight* of v. Let l and u be nonnegative integers, called the *lower bound* and *upper bound* on component size, respectively. We wish to partition G into connected components by deleting edges from G so that the total weights of all components are almost uniform, that is, the sum of weights of all vertices in each component is at least l and at most u for some bounds l and u. We call such an almost uniform partition an *(l,u)-partition* of G. In this chapter we deal with the following three partition problems to find an (l,u)-partition of a given graph G: the *minimum (l,u)-partition problem* is to find an (l,u)-partition of G with the minimum number of components; the *maximum (l,u)-partition problem* is defined similarly; and the *(p,l,u)-partition problem* is to find an (l,u)-partition of G with a fixed number p of components.

Figures 3(a) and (b) illustrate two $(10,20)$-partitions of the same graph, where each vertex is drawn as a circle, the weight of each vertex is written inside the circle, and the deleted edges are drawn by dotted lines. The $(10,20)$-partition with four components in Fig. 3(a) is a solution for the minimum $(10,20)$-partition problem, and the $(10,20)$-partition with six components in Fig. 3(b) is a solution for the maximum $(10,20)$-partition problem.

The (p,l,u)-partition problem for general p is NP-complete and the minimum and maximum (l,u)-partition problems for general l and u are NP-hard even for series-parallel graphs. Moreover, the (p,l,u)-partition

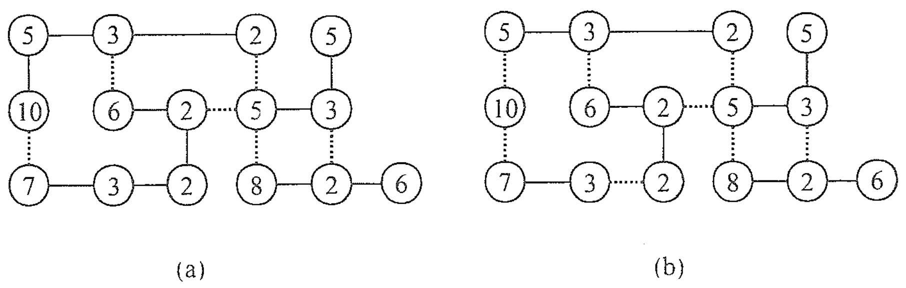

Figure 3: (a) Solution for the minimum $(10, 20)$-partition problem, and (b) solution for the maximum $(10, 20)$-partition problem.

problem is strongly NP-complete and the minimum and maximum (l, u)-partition problems are strongly NP-hard for general graphs. One can observe that, for any $c > 0$, there is no polynomial-time c-approximation algorithm for the minimum (l, u)-partition problem or the maximum (l, u)-partition problem on series-parallel graphs unless P = NP.

In this chapter we obtain pseudo-polynomial-time algorithms to solve the three partition problems for series-parallel graphs and partial k-trees, as follows.

Theorem 5 *Both the minimum and maximum (l, u)-partition problems can be solved in time $O(u^4 n)$ and the (p, l, u)-partition problem can be solved in time $O(p^2 u^4 n)$ for any series-parallel graph.*

Theorem 6 *Both the minimum and maximum (l, u)-partition problems can be solved in time $O\left(u^{2(k+1)} n\right)$ and the (p, l, u)-partition problem can be solved in time $O\left(p^2 u^{2(k+1)} n\right)$ for any partial k-tree, where k is a constant.*

5 Distance-edge-colorings

Let $G = (V, E)$ be a graph with vertex set V and edge set E, and let $|V| = n$. For two vertices u and v, we denote by $\mathrm{dist}(u, v)$ the *distance between u and v* in G, that is, the number of edges in a shortest path between u and v in G. For two edges $e = (u, v)$ and $e' = (u', v')$, the *distance between e and e'* in G is defined as follows:

$$\mathrm{dist}(e, e') = \min\{\mathrm{dist}(u, u'), \mathrm{dist}(u, v'), \mathrm{dist}(v, u'), \mathrm{dist}(v, v')\}.$$

For a given bounded nonnegative integer ℓ, we wish to color all edges of G so that any two edges e and e' with $\mathrm{dist}(e, e') \leq \ell$ have different colors. Such a coloring is called an *ℓ-edge-coloring* of G. Thus a 0-edge-coloring is merely an ordinary edge-coloring, and a 1-edge-coloring is a "strong edge-coloring." The *ℓ-chromatic index* $\chi'_\ell(G)$ of G is the minimum number of colors required for an ℓ-edge-coloring of G. The *ℓ-edge-coloring problem* is to compute the ℓ-chromatic index $\chi'_\ell(G)$ of a given graph G. For example, the coloring of a graph in Fig. 4 is a 1-edge-coloring with six colors $c_1, c_2, \cdots, c_6$, and is of course a 0-edge-coloring, but is not a 2-edge-coloring. One can easily observe that $\chi'_1(G) = 6$ for the graph G in Fig. 4.

Since the edge-coloring problem is NP-hard, the ℓ-edge-coloring problem is NP-hard in general. However, the ordinary edge-coloring problem and the 1-edge-colring problem can be solved in linear time and polynomial time, respectively, for partial k-trees.

In this chapter we give two polynomial-time algorithms for the ℓ-edge-coloring problem. The first one is the following exact algorithm.

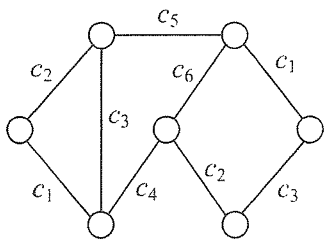

Figure 4: A 1-edge-coloring of a partial 3-tree G with six colors.

Theorem 7 *Let G be a partial k-tree, let ℓ be a bounded nonnegative integer, and let α be a positive integer. Then it can be examined in time $O\left(n(\alpha+1)^{2^{2(k+1)(\ell+1)+1}}\right)$ whether G has an ℓ-edge-coloring with α colors.*

The number α is not assumed to be a fixed constant, but can be assumed to be smaller than the number m of edges in G. Therefore, using a binary search technique, one can compute the ℓ-chromatic index $\chi'_\ell(G)$ of G by applying Theorem 7 for at most $\log_2 m$ values of α, $1 \le \alpha < m$. We thus have the following corollary.

Corollary 1 *The ℓ-chromatic index $\chi'_\ell(G)$ of a partial k-tree G can be computed in polynomial time.*

By combining our algorithm for partial k-trees with a general method for obtaining approximation algorithms for NP-complete problems on planar graphs [1], one can find an ℓ-edge-coloring of a given planar graph G with at most $2\chi'_\ell(G)$ colors in polynomial time. We thus have the following approximation algorithm.

Theorem 8 *There is a polynomial-time 2-approximation algorithm for the ℓ-edge-coloring problem on planar graphs.*

6 Multicolorings

Let each vertex v of a graph G have a positive integer weight $\omega(v)$. Let C be a set of colors, and let 2^C be the power set of C. Then a *multicoloring* Γ *of* G is a mapping from V to 2^C which assigns each vertex $u \in V$ a set $\Gamma(u)$ of $\omega(u)$ colors in C in a way that $\Gamma(v) \cap \Gamma(w) = \emptyset$ for any pair of adjacent vertices $v, w \in V$. Thus the ordinary vertex-coloring is merely a multicoloring for the special case where $\omega(v) = 1$ for every vertex v. The *multichromatic number* $\chi_\omega(G)$ of G is the minimum number of colors required for a multicoloring of G. The *multicoloring problem* is to compute the multichromatic number $\chi_\omega(G)$ of a given graph G. Consider a graph G in Fig. 5(a) where $\omega(v)$ is attached to each vertex v. Since v_1, v_2 and v_5 are adjacent with each other, we have $\chi_\omega(G) \ge \omega(v_1) + \omega(v_2) + \omega(v_5) = 5$. Since G has a multicoloring with five colors $c_1, c_2, \cdots, c_5$ as illustrated in Fig. 5(b), we have $\chi_\omega(G) \le 5$. Thus $\chi_\omega(G) = 5$.

Since the vertex-coloring problem is NP-hard, the multicoloring problem is of course NP-hard. However, it is known that the multicoloring problem can be solved efficiently for some restricted classes of graphs, such as trees, triangulated graphs, perfect graphs, and series-parallel graphs.

In this chapter, we give an algorithm to solve the multicoloring problem for partial k-trees, as follows.

Theorem 9 *The multichromatic number $\chi_\omega(G)$ of a partial k-tree $G = (V, E)$ can be computed in time $O\left(nW^{2^{2k+3}} \log_2 W\right)$, where n is the number of vertices of G and $W = \max_{v \in V} \omega(v)$.*

It is easy to modify the algorithm so that it actually finds a multicoloring of G with $\chi_\omega(G)$ colors. It should be noted that an ordinary representation of a multicoloring of G requires space of size at least $\sum_{v\in V} \omega(v) = O(nW)$.

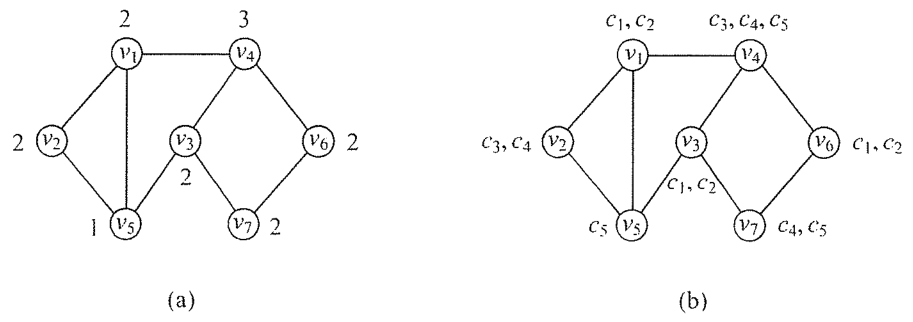

Figure 5: (a) A partial 3-tree G, and (b) a multicoloring Γ of G with five colors.

7 Conclusions

In this thesis, we presented efficient algorithms for several graph partitioning problems and two graph coloring problems. We showed that all these problems can be solved for partial k-trees G by a dynamic programming approach based on a tree-decomposition of G.

References

[1] B. S. Baker, Approximation algorithms for NP-complete problems on planar graphs, *J. ACM*, Vol. 41, pp. 153–180, 1994.

[2] H. L. Bodlaender, A linear-time algorithm for finding tree-decompositions of small treewidth, *SIAM J. Computing*, Vol. 25, pp. 1305–1317, 1996.

[3] J. A. Telle and A. Proskurowski, Algorithms for vertex partitioning problems on partial k-trees, *SIAM J. Discrete Math.*, Vol. 10, pp. 529–550, 1997.

論文審査結果の要旨

コンピュータネットワーク等に現れる様々な問題は，グラフの分割や彩色として定式化することができるが，それらの多くは一般のグラフに対して効率よく解けそうにない．そこで著者は，部分 k-木と呼ばれる非常に広いグラフのクラスに対し，需要点と供給点がある場合の分割問題や，点に整数重みが付いている場合の均一分割問題，距離辺彩色問題および多重彩色問題を効率よく解くアルゴリズムを与えた．本論文はこれらの成果をとりまとめたものであり，全編 7 章からなる．

第 1 章は序論である．

第 2 章では準備として，グラフの基本的な概念の定義を与えるとともに，グラフアルゴリズムや近似アルゴリズムの一般的な性質を明らかにしている．

第 3 章では，電力網などを需要点と供給点のあるグラフでモデル化し，全ての需要点に電力を供給することが可能かどうか判定する判定問題と，供給されている需要点の需要量の合計が最大になるような電力の流し方を求める最大化問題を扱っている．著者は，木に対して判定問題を解く線形時間アルゴリズムを与えるとともに，最大化問題を擬多項式時間で解くアルゴリズムおよび完全近似スキームを与えている．さらに，直並列グラフや部分 k-木に対して最大化問題を擬多項式時間で解くアルゴリズムを与えている．これらの問題は電力系統の配電融通問題などに応用があり，本章のアルゴリズムは極めて重要な成果である．

第 4 章では，グラフの均一分割問題，即ち与えられたグラフの辺を除去して連結成分に分割し，各連結成分に含まれる点の重みの合計が均一になるようにする問題が，木，直並列グラフおよび部分 k-木に対して擬多項式時間で解けることを示している．均一分割問題は選挙区割問題などに応用があり，有用な結果である．

第 5 章では，最少色数を用いた距離辺彩色，即ち指定された距離以内の任意の 2 辺に異なる色を割り当てる彩色が，部分 k-木に対して多項式時間で求まることを示している．さらに，平面グラフに対して，最少色数の 2 倍以内の色数を用いた距離辺彩色を多項式時間で求める近似アルゴリズムを与えている．本章の結果は，従来の辺彩色および強辺彩色の結果を包含するものであり，高く評価できる．

第 6 章では，最少色数を用いた多重彩色，即ち隣り合う 2 点の色は異なるように各点に指定された個数の色を割り当てる彩色が，部分 k-木に対して擬多項式時間で求まることを示している．多重彩色問題は並列タスクスケジューリング問題に応用でき，極めて重要な結果である．

第 7 章は結論である．

以上要するに，本論文は，様々な分割問題や彩色問題を，部分 k-木という非常に広いグラフのクラスに対して効率よく解くアルゴリズムを与えたものであり，計算機科学，特にアルゴリズム理論の発展に寄与するところが少なくない．

よって，本論文は博士（情報科学）の学位論文として合格と認める．