Polynomial algorithms for Chromatic Index and Graph Isomorphism on partial k-trees

Hans L. Bodlaender

RUU-CS-87-17
October 1987

Polynomial algorithms for Chromatic Index and Graph Isomorphism on partial k-trees*

Hans L. Bodlaender
Dept. of Computer Science, University of Utrecht
P.O. Box 80.012, 3508 TA Utrecht, the Netherlands

Abstract

In this paper we show that the CHROMATIC INDEX and the GRAPH ISOMORPHISM problems are solvable in polynomial time when restricted to the class of graphs with treewidth $\leq k$ (k a constant) (or equivalently, the class of partial k-trees).

Keywords: Chromatic Index, Graph Isomorphism, graphs with small treewidth, partial k-trees, NP-complete problems, polynomial time algorithms.

1 Introduction

In [2,6,14] it has been shown that polynomial time algorithms and even linear time algorithms exist for a large number of NP-complete problems when these problems are restricted to the class of partial k-trees, for constant k. Presently, no polynomial time algorithms were known for CHROMATIC INDEX and GRAPH ISOMORPHISM when restricted to the partial k-trees (k constant). In this paper we show that such polynomial time algorithms exist. This solves two open problems from [10].

The CHROMATIC INDEX problem asks whether for a given graph $G = (V, E)$ and a given integer K, one can color the edges of G with at most K colors, such that for each vertex v all adjacent edges have a different color. The CHROMATIC INDEX problem is NP-complete, even for cubic graphs [8]. In [6] and [14] it has been shown that the CHROMATIC INDEX problem can be solved in polynomial and even linear time for partial k trees with maximum vertex degree d, (k, d constants). In this paper, we do not restrict the degree of the graphs. However, our algorithm has complexity $\mathcal{O}(n^{1+2^{2(k+1)}})$, which indicates that the algorithm is impractical, even for k as small as 2. Hence, it remains an open problem to find algorithms for CHROMATIC INDEX on partial k-trees with a better running time.

The GRAPH ISOMORPHISM problem is a well-studied problem that is not known to be solvable in polynomial time, but is also not known to be NP-complete. In [10], Johnson conjectures that GRAPH ISOMORPHISM is solvable in polynomial time for partial k-trees. In this paper we show that this is indeed the case. Previously, only a polynomial algorithm was known for the case that $k = 2$. (This result follows from the fact that every partial 2-tree is planar. Hence we can use a GRAPH ISOMORPHISM algorithm for planar graphs.)

*This research was done while the author was visiting the Laboratory for Computer Science of the Massachusetts Institute of Technology, with a grant from the Netherlands Organization for the Advancement of Pure Research Z.W.O.

There are various ways to characterize the class of partial k-trees. Robertson and Seymour [12] introduced the notion of treewidth. It follows easily that each graph with treewidth $\leq k$ is a partial k-tree and vice versa. In [1] an overview of several different characterizations is given. In this paper we use two different characterizations: in section 3 we use the representation as graphs with treewidth $\leq k$, and in section 4 we use the representation as partial k-trees. This facilitates the presentation of the results.

The first step of both algorithms is to find a representation as partial k-tree (or tree-decomposition). This can be done in polynomial time for fixed k. An $\mathcal{O}(n^{k+2})$ algorithm was designed by Arnborg, Corneil and Proskurowski [2]. Faster algorithms exist for $k \leq 3$ [3]. Recent results of Robertson and Seymour indicate the existence of asymptotically faster algorithms ($\mathcal{O}(n^2)$ or $\mathcal{O}(n^3)$ but with a very large constant factor). For a discussion of the latter results, see e.g. [11].

The class of partial k-trees has several important subclasses. Each of the following classes of graphs has associated with it some constant k', such that each graph in the class is a partial k'-tree: the series-parallel graphs, the outerplanar graphs, the k-outerplanar graphs, graphs with bandwidth $\leq k$, graphs with cutwidth $\leq k$, Halin-graphs, chordal graphs with maximum cliquesize k. An overview of results of this type can be found in [5]. Hence, it follows from the results in this paper that the CHROMATIC INDEX and GRAPH ISOMORPHISM problems are solvable in polynomial time for each of these classes of graphs too.

2 Definitions

In this section we give a number of basic definitions.

DEFINITION 2.1 *The* chromatic index *of a graph $G = (V, E)$ is the smallest integer K, such that there exist a function $col : E \rightarrow \{1, \ldots, K\}$, such that for each pair of edges $e_1, e_2 \in E$, that have an endpoint in common, $col(e_1) \neq col(e_2)$.*

It is a well-known fact that the chromatic index of a graph either equals its maximum vertex degree or its maximum vertex degree +1 (Vizing's theorem). We will not use this fact in this paper.

DEFINITION 2.2 *Let $G = (V, E)$ be a graph. A tree-decomposition of G is a pair $(\{X_i \mid i \in I\}, T = (I, F))$, with $\{X_i \mid i \in I\}$ a family of subsets of V, and T a tree, with the following properties:*

- $\bigcup_{i \in I} X_i = V$
- *For every edge $e = (v, w) \in E$, there is a subset X_i, $i \in I$, with $v \in X_i$ and $w \in X_i$.*
- *For all i, j, $k \in I$, if j lies on the path in T from i to k, then $X_i \cap X_k \subseteq X_j$.*

The tree-width of a tree-decomposition $(\{X_i \mid i \in I\}, T)$ is $\max_{i \in I} |X_i| - 1$. The tree-width of G, denoted by tree-width(G) is the minimum tree-width of a tree-decomposition of G, taken over all possible tree-decomposition of G.

DEFINITION 2.3 *The class of the k-trees is defined recursively as follows.*

1. *A complete graph with k vertices is a k-tree.*
2. *If $G = (V, E)$ is a k-tree, and $w \notin V$, and $v_1, \ldots, v_k$ form a complete subgraph of G with k vertices, then $H = (V \cup \{v\}, E \cup \{(v_i, v) \mid 1 \leq i \leq k\})$ is a k-tree.*
3. *All k-trees can be formed with rules 1 and 2.*

DEFINITION 2.4 *A graph is a partial k-tree, if and only if it is the subgraph of a k-tree.*

It can be shown that every graph with treewidth $\leq k$ is a partial k-tree, and conversely, that every partial k-tree has treewidth $\leq k$. We leave this as an easy exercise to the reader. (For other equivalent characterizations, see e.g. [1].)

For different problems, different representations may be easier to use. We will use in section 3 the tree-representations, and in section 4 the representation as partial k-trees.

3 Chromatic Index

In this section we show that for each $k \geq 1$ there exists a polynomial time algorithm to determine the chromatic index of a given graph $G = (V, E)$ with treewidth $\leq k$.

The first step of the algorithm is to find a tree-decomposition of G with treewidth $\leq k$. Then an arbitrary node $r \in I$ is chosen, and T is considered as a rooted tree with root r. We will use notions as: children, descendants, leaves, internal nodes, in their usual meaning.

By using transformations of the tree as illustrated in figure 1 one can obtain a tree-decomposition of G, with the same treewidth as the old tree-decomposition, that has the following characteristics:

1. $|I| = \mathcal{O}(n)$.
2. Each internal node i has exactly 2 children, say j and k. Furthermore, $X_i = X_j$ or $X_i = X_k$.
3. For each edge $e = (v, w) \in E$, there is at least one leaf-node $i \in I$, with $v \in X_i$ and $w \in X_i$.

Further, note that the total transformation can be done in polynomial time (and even linear time). In the remainder of this section we suppose that we have a tree-decomposition of G with T rooted at r of the form as described above.

We continue with a large number of definitions.

DEFINITION 3.1 *For each edge e we chose arbitrarily one of the leaf-nodes i, such that X_i contains both endpoints of e. For each e this chosen node i is denoted as $rep(e)$. For all $i \in I$ let $E_i = \{e \in E \mid rep(e) = i\}$.*

Note that for *internal* nodes $i \in I$ one has $E_i = \emptyset$.

DEFINITION 3.2 *The set of colors, that can be used to color the edges is denoted by $C = \{1, \cdots, K\}$.*

DEFINITION 3.3 *The subtree of T, formed by i and all its descendants is denoted by $T(i)$. We denote $E(T(i)) = \bigcup_{j \in T(i)} E_j$.*

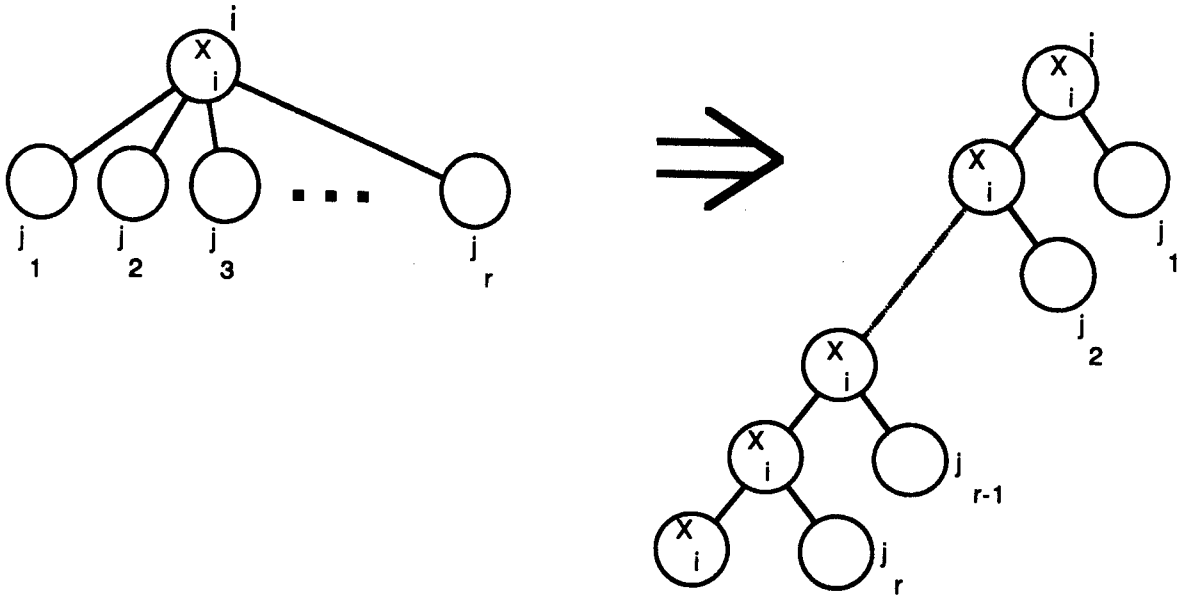

Figure 1: Transformation of T

Note that if $E(T(i)) \cap E(T(j)) \neq \emptyset$, then either i is a descendant of j or j is a descendant of i, or $i = j$.

DEFINITION 3.4 *1. A mapping $f : E(T(i)) \to C$ is called a* partial coloring *of G,* rooted *at i.*

2. A mapping $f : E \to C$ is called a total coloring.

3. A (partial or total) coloring $F \to C$ $(F \subseteq E)$ is correct *iff for all vertices $v \in V$ no two adjacent edges are colored with the same color.*

4. A partial coloring is called feasible, *iff it can be extended to a correct total coloring.*

DEFINITION 3.5 *Let $f : E(T(i)) \to C$ be a partial coloring rooted at i. The* active set *of f is the pair $(i,\ a : X_i \to \mathcal{P}(C))$, with for all $v \in X_i$, $a(v)$ denoting the set of colors of edges in $E(T(i))$, adjacent to v. We denote the active set of f by $act(f)$.*

We will be a little imprecise, and denote the active set of f, $act(f) = (i, a)$ only by a, if the root of f, i is known. We also say that i is the root of a.

Lemma 3.1 *Let f, g be correct partial colorings, with the same active set, and rooted at the same node $i \in I$. Then f is feasible, if and only if g is feasible.*

Proof. Consider an extension $\bar{f}$ of f, that is correct. Now one can extend g in the same way to a correct total coloring: use the coloring $\bar{g}$ with $\bar{g}(e) = g(e)$, if $e \in E(T(i))$, and $\bar{g}(e) = \bar{f}(e)$, if $e \notin E(T(i))$. One easily checks that $\bar{g}$ also is correct. The reverse implication can be obtained with the same argument, with switched roles of f and g. **Q.E.D.**

(The same technique was used by Saxe [13] for recognition of graphs with small bandwidth.)

DEFINITION 3.6 *1. An active set is* good, *if it is the active set of a correct partial coloring.*

2. An active set is feasible, *if it is the active set of feasible partial coloring.*

We now remark that a polynomial algorithm for the case that $|C|$ is bounded by a constant, i.e. for graphs with constant degree, can be obtained by recursively computing all good active sets (see e.g. [6]). However, if the degree of the graphs can be arbitrary large (and hence $|C|$ can be arbitrary large), then the number of active sets can be exponentially in n. In order to overcome this problem, we introduce an extra equivalence relation.

DEFINITION 3.7 *Two active sets (i,a), (i,b) (rooted at the same node i) are* isomorphic *(denoted by $(i,a) \equiv (i,b)$, or, in short, $a \equiv b$) if and only if $\exists$ an isomorphism $\psi : C \rightarrow C$, such that $\forall\, c \in C, v \in X_i : c \in a(v) \Leftrightarrow \psi(c) \in b(v)$.*

In other words, $a \equiv b$, if b can be obtained from a by "renaming" all colors.

Lemma 3.2 *If $a \equiv b$, then a is good, iff b is good; and a is feasible, iff b is feasible.*

Proof. Suppose $a \equiv b$; a, b are both rooted at i. Suppose a is good. Let f be a partial mapping with active set a. Then g, defined by $g(e) = \psi(f(e))$ is a good partial mapping with active set b. The other cases are similar. **Q.E.D.**

DEFINITION 3.8 *Let (i,a) be an active set. For all $c \in C$, we denote $S(c,a) = \{v \in X_i \mid c \in a(v)\}$, i.e., $S(c,a)$ is the set of vertices in X_i, adjacent to color c in a partial coloring corresponding to (i,a).*

We can characterize the equivalence classes of good active sets by the following lemma.

Lemma 3.3 *Let (i,a) and (i,b) be active sets. $a \equiv b$, if and only if $\forall\, S \subseteq X_i : |\{c \in C \mid S = S(c,a)\}| = |\{c \in C \mid S = S(c,b)\}|$.*

Proof. $\Rightarrow$ Suppose $a \equiv b$. Let ψ be the isomorphism $C \rightarrow C$, with $\forall\, c \in C, v \in X_i : c \in a(v) \Leftrightarrow \psi(c) \in b(v)$. Let $S \subseteq X_i$. Now $|\{c \in C \mid S = S(c,a)\}| = |\{c \in C \mid \forall\, v \in X_i : c \in a(v) \Leftrightarrow v \in S)\}| = |\{c \in C \mid \forall\, v \in X_i : \psi(c) \in a(v) \Leftrightarrow v \in S)\}| = |\{c \in C \mid \forall\, v \in X_i : c \in b(v) \Leftrightarrow v \in S)\}| = |\{c \in C \mid S = S(c,b)\}|$.
$\Leftarrow$ Suppose that $\forall\, S \subseteq X_i$: $|\{c \in C \mid S = S(c,a)\}| = |\{c \in C \mid S = S(c,b)\}|$. Hence, there exists an isomorphism $\psi : C \rightarrow C$, with $\forall\, c \in C : S(c,a) = S(\psi(c),b)$. Now, for all $c \in C, v \in X_i : c \in a(v) \Leftrightarrow v \in S(c,a) = S(\psi(c),b) \Leftrightarrow \psi(c) \in b(v)$. Hence $a \equiv b$. **Q.E.D.**

It follows that one can characterize equivalence-classes of active sets rooted at i, by mappings $nb : \mathcal{P}(X_i) \rightarrow \{0,1,\ldots,|C|\}$.

DEFINITION 3.9 *Let (i,a) be an active set. The* active count *of (i,a) is the pair $(i, nb_a : \mathcal{P}(X_i) \rightarrow \{0,1,\cdots,|C|\})$, with $\forall\, S \subseteq X_i : nb_a(S) = |\{c \in C \mid S = S(c,a)\}|$.*

Lemma 3.4 *Let (i,a), (i,b) be active sets rooted at i. Then:*

1. $\sum_{S \subseteq X_i} nb_a(S) = |C|$.

2. $a \equiv b \Leftrightarrow nb_a = nb_b$.

We say that the active count (i, nb) is *rooted at* i. Note that for each $i \in I$, the total number of active counts rooted at i, is bounded by $(|C|+1)^{2^{k+1}}$.

DEFINITION 3.10 *An active count* (i, nb) *is* good, *if there exists a good active set* (i, a), *with* $nb = nb_a$.

If (i, nb) is the active count of the active set of a partial coloring f, we also say that (i, nb) is the active count of f. In order to be able to compute all active counts rooted at internal nodes i, we introduce the notion of active pair-counts.

DEFINITION 3.11 *Let* i *be an internal node with children* j, k *and suppose* $X_i = X_k$. *The* active pair-count *of a partial coloring* $f : E(T(i)) \rightarrow C$ *is the pair* $(i, pc_f : \mathcal{P}(X_i) * \mathcal{P}(X_j) \rightarrow \{0, 1, \cdots, |C|\})$, *with* $\forall\ S \subseteq X_i,\ T \subseteq X_j$: $pc_f(S,T) = |\{c \in C \mid S = S(c, act(f')) \wedge T = S(c, act(f''))\}|$, *where* f' *is obtained by restricting* f *to* $E(T(k))$ *and* f'' *is obtained by restricting* f *to* $E(T(j))$.

In other words pc_f maps each pair (S,T) to the number of colors c, with $\forall\ v \in X_i$: $(\exists e \in E(T(k))$, adjacent to v, with $f(e) = c) \Leftrightarrow v \in S$ and $\forall\ v \in X_j : (\exists e \in E(T(j))$, adjacent to v, with $f(e) = c) \Leftrightarrow v \in T$. Note that conversely we have for each color $c \in C$ a unique pair (S,T) such that $S = S(c, act(f'))$ and $T = S(c, act(f''))$, f', f'' as above.

DEFINITION 3.12 *An active pair-count* (i, pc) *is* good, *iff there exists a correct partial coloring* $f : E(T(i)) \rightarrow C$, *with* $pc_f = pc$.

Lemma 3.5 *Let* i, j, k *be as above. For each active pair-count* (i, pc_f) *of a partial coloring* f, *rooted at* i:

$$\sum_{S \subseteq X_i} \sum_{T \subseteq X_j} pc_f(S,T) = |C|.$$

It follows that the number of active pair-counts rooted at i, is bounded by $\mathcal{O}(|C|^{2^{2(k+1)}})$.

We now will show how to compute all good active counts and good active pair-counts. After these have been computed, it is easy to decide on the chromatic index of G. First we consider leaf-nodes $i \in I$. We use the following lemma:

Lemma 3.6 *Let* $i \in I$ *be a leaf. Let* $C' \subseteq C$, *with* $|C'| \geq |E_i|$. *For every partial coloring* $f : E(T(i)) \rightarrow C$, *with active set* (i, a), *there exists a partial coloring* $g : E(T(i)) \rightarrow C$, *with active set* (i, b), *such that* $a \equiv b$ *and* $\forall\ e \in E_i : g(e) \in C'$.

Proof. For every f, there exists a coloring g rooted at i, with for all $e \in E_i : g(e) \in C'$, such that there exists an isomorphism $\psi : C \rightarrow C$, with $g(e) = \psi(f(e))$. Let (i, b) be the active set of g. It follows that $a \equiv b$. **Q.E.D.**

Corollary 3.7 *Let* $i \in I$ *be a leaf. The set of all good active counts rooted at* i *can be computed in* $\mathcal{O}(1)$ *time.*

Proof. Lemma 3.6 shows that we can use the following procedure: Fix a set of $min(|C|, |E_i|) \leq \frac{1}{2}k(k+1)$ colors $C' \subseteq C$. Enumerate all partial colorings $E_i \rightarrow C'$. Remove all partial colorings that are not correct. For each remaining partial coloring, compute the corresponding active count. Some active counts may appear more than once. Remove all second and later occurrences of the same active count. The remaining table consists of all good active counts rooted at i.

As there are at most $|E_i|^{|C'|} = \mathcal{O}(1)$ different partial colorings $E_i \rightarrow C^i$, it follows that the described procedure takes $\mathcal{O}(1)$ time. **Q.E.D.**

Next we show how to compute all good active pair-counts rooted at some *internal* node i, given tables consisting of all good active counts rooted at the children of i.

Lemma 3.8 *Let i be an internal node. Let j, k be the children of i, and suppose $X_i = X_k$. Let (i, pc) be an active pair-count rooted at i. Then (i, pc) is good, if and only if there exist an active count (j, nb_1), and an active count (k, nb_2), such that*

1. *(j, nb_1) is good.*
2. *(k, nb_2) is good.*
3. *for all $S \subseteq X_i$: $nb_1(S) = \sum_{T \subseteq X_j} pc(S, T)$.*
4. *for all $T \subseteq X_j$: $nb_2(T) = \sum_{S \subseteq X_i} pc(S, T)$.*
5. *$\forall\, S \subseteq X_i,\ T \subseteq X_j$: if $S \cap T \neq \emptyset$, then $pc(S, T) = 0$.*

Proof. $\Rightarrow$ Suppose (i, pc) is a good active count, rooted at i, and let f be a corresponding correct partial coloring $E(T(i)) \rightarrow C$. Let (j, nb_1), (k, nb_2) be the active counts, defined by:

$$\text{for all } S \subseteq X_i\text{: } nb_1(S) = \sum_{T \subseteq X_j} pc(S, T),$$

$$\text{for all } T \subseteq X_j\text{: } nb_2(T) = \sum_{S \subseteq X_i} pc(S, T).$$

It easily follows that (j, nb_1) is the active count of f, restricted to $E(T(k))$, and (k, nb_2) is the active count of g, restricted to $E(T(j))$, hence (j, nb_1) and (k, nb_2) are good. Further, suppose that there are $S \subseteq X_i$, $T \subseteq X_j$, with $S \cap T \neq \emptyset$, and $pc(S, T) \neq 0$. It follows that there are $pc(S, T)$ (hence at least 1) different colors c, that are adjacent to all vertices in $S \cap T$ with an edge in $E(T(k))$ and another edge in $E(T(j))$. This contradicts the fact that (i, pc) is good.

$\Leftarrow$ Suppose we have an active pair-count (i, pc), and active counts (j, nb_1), (k, nb_2), such that conditions (1)-(5) hold. Assign to each pair (S, T), $(S \subseteq X_i,\ T \subseteq X_j)$ $pc(S, T)$ different colors, such that each color is assigned to exactly one pair (S, T). Denote the set of colors, assigned to (S, T) by $c(S, T)$. Let $c_1(S) = \bigcup_{T \subseteq X_j} c(S, T)$, $c_2(T) = \bigcup_{S \subseteq X_i} c(S, T)$.

Now note that there exists a partial coloring $f : E(T(k) \rightarrow C$, that is correct, has associated active count (j, nb_1), and for all $c \in C$, $v \in X_i$: there is an edge in $E(T(k))$, adjacent to v, that is colored with c, if and only if $\exists S \subset X_i : v \in S \wedge c \in c_1(S)$. (Use that for all $S \subseteq X_i$: $|c_1(S)| = nb_1(S)$. Find a partial coloring f', corresponding to nb_1, and then use an isomorphism on C.)

Similar, one can find a partial coloring $g : E(T(k) \rightarrow C$, that is correct, has associated active count (k, nb_2), and for all $c \in C$, $v \in X_j$: there is an edge in $E(T(j))$, adjacent to v, that is colored with c, if and only if $\exists T \subset X_j : v \in T \wedge c \in c_2(T)$.

Now, consider the partial coloring $h : E(T(i)) \rightarrow C$, defined by $h(e) = f(e)$, if $e \in E(T(k))$, and $h(e) = g(e)$, if $e \in E(T(j))$. (Note that $E(T(k)) \cup E(T(j)) = E(T(i))$ and $E(T(k)) \cap E(T(j)) = \emptyset$.) Now for each pair $(S,T) \in \mathcal{P}(X_i) * \mathcal{P}(X_j)$, and each color $c \in C$: $(\forall\, v \in X_i : (\exists e \in E(T(k))$, adjacent to v, with $f(e) = c) \Leftrightarrow v \in S$, and $\forall\, v \in X_j : (\exists e \in E(T(j))$, adjacent to v, with $f(e) = c) \Leftrightarrow v \in T)$, if and only if $(c \in c_1(S) \cap c_2(T) = c(S,T))$. It follows that the active pair-count, associated to h, is (i, pc). We will now show that h is correct. Suppose h is not correct. Then $\exists e \in E(T(k)), e' \in E(T(j)) : h(e) = h(e')$, and $\exists v \in X_i$ is adjacent to e and e'. Then the color $h(e)$ is associated with a pair (S,T), with $v \in S \cap T$. Hence $S \cap T \neq \emptyset$ and $pc(S,T) > 0$. Contradiction. It follows that h is correct, hence (i, pc) is good. **Q.E.D.**

Corollary 3.9 *Let i be an internal node with children j and k and suppose that $X_i = X_k$. Suppose tables, consisting of all good active counts rooted at j and rooted at k are given. Then one can compute all good active pair-counts rooted at i in time $\mathcal{O}(|C|^{2^{2(k+1)}})$.*

Proof. One can check for each function $pc : \mathcal{P}(X_i) * \mathcal{P}(X_j) \rightarrow \{0, 1, \cdots, |C|\}$ with $\sum_{S \subseteq X_i} \sum_{T \subseteq X_j} pc(S,T) = |C|$, whether (i, pc) is a good active pair-count, by first directly checking condition 5 of lemma 3.8, then computing nb_1 and nb_2 as indicated by conditions 3 and 4, and then looking up in the tables whether (j, nb_1) and (k, nb_2) are good. This costs $\mathcal{O}(1)$ time per active pair-count, so in total $\mathcal{O}(|C|^{2^{2(k+1)}})$. **Q.E.D.**

Next we show how to compute all good active counts rooted at an internal node i, given a table, consisting of all good active pair-counts rooted at i.

Lemma 3.10 *Let $i \in I$ be an internal node. An active count (i, nb) is good, if and only if there exists a good active pair-count (i, pc), such that for all $S \subseteq X_i$, $nb(S) = \sum pc(S', T')$, where the sum is taken over all pairs (S', T'), with $S' \subseteq X_i$, $T' \subseteq X_j$, $S' \cup T' = S$, $S' \cap T' = \emptyset$.*

Proof. Let f be a correct partial coloring $E(T(i)) \rightarrow C$. Note that a color c is adjacent to all vertices in $S \subseteq X_i$, and to no vertex in $X_i - S$, if and only if there are subsets $S' \subseteq X_i$, $T' \subseteq X_j$, with $S' \cup T' = S$, such that c is adjacent to all vertices in S' with an edge in $E(T(k))$ (and to no other vertex in $X_i - S'$ with an edge in $E(T(k))$,) and to all vertices in T' with an edge in $E(T(j))$ (and to no other vertex in $X_i - T'$ with an edge in $E(T(j))$.) Because f is correct, we have also that $S' \cap T' = \emptyset$.

Let (i, nb) be the active count, corresponding to f (i.e. $nb = nb_f$), and (i, pc) the active pair-count, corresponding to f (i.e. $pc = pc_f$). It follows that for all $S \subseteq X_i$, $nb(S) = \sum pc(S', T')$, the sum taken over all pairs (S', T'), with $S' \subseteq X_i$, $T' \subseteq X_j$, $S' \cup T' = S$, $S' \cap T' = \emptyset$. Hence the result follows. **Q.E.D.**

Corollary 3.11 *Let i be an internal node. Suppose a table, consisting of all good active pair-counts rooted at i, is given. Then one can compute a table, consisting of all good active counts rooted at i, in time $\mathcal{O}(|C|^{2^{2(k+1)}})$.*

Proof. Use the following procedure: For each good active pair-count, compute the corresponding active count, as indicated by lemma 3.10. If an active count appears more than once, remove all multiple copies from the table. Clearly this procedure uses at most $\mathcal{O}(|C|^{2^{2(k+1)}})$ time. **Q.E.D.**

Finally, we note that one easily determines the answer to the CHROMATIC INDEX problem, given a table of all good active counts, rooted at root r.

Lemma 3.12 *There exists a correct total coloring, if and only if there exists a good active count rooted at r.*

Proof. If there exists a good active count rooted at r, then there exists a good active set rooted at r, hence there exists a correct partial coloring $f : E(T(r)) \rightarrow C$. But f is also a total coloring, because $E(T(r)) = E$. **Q.E.D.**

Theorem 3.13 *For each $k \geq 1$, there exists an algorithm that solves the* CHROMATIC INDEX *problem in $\mathcal{O}(n^{1+2^{2(k+1)}})$ time.*

Proof. We can use the following algorithm: first find the desired tree-decomposition of G, and then recursively calculate all good active counts and good active pair-counts, as indicated in corollary 3.7, 3.9 and 3.11. Finally check in $\mathcal{O}(1)$ time, whether the table, consisting of all good active counts, rooted at r is not empty. This algorithm uses $\mathcal{O}(|I| \cdot |C|^{2^{2(k+1)}}) = \mathcal{O}(n^{1+2^{2(k+1)}})$ time. **Q.E.D.**

4 Graph Isomorphism

In this section we show that for each $k \geq 1$, there exists a polynomial algorithm that decides whether two partial k-trees are isomorphic or not.

We let $G = (V_G, E_G)$ and $H = (V_H, E_H)$ be partial k-trees, with $|V_G| = |V_H| = n$, for which it must be determined whether G and H are isomorphic.

Consider the collection of k-element vertex sets which are separators of G. We number these k-vertex separators and denote them $C_1, C_2, \ldots$. For each C_i, consider the set of connected components of the graph, obtained by removing C_i form G. We denote the sets of vertices in these connected components by $C_i^1, C_i^2, \ldots$.

Similarly, we number the k-element vertex separators of H, and denote them by $D_1, D_2, \ldots$. The sets of vertices in the connected components of the graph obtained by removing D_i from H, are denoted by $D_i^1, D_i^2, \ldots$.

For graphs $G' = (V, E)$, and $W \subseteq V$, we denote the subgraph of G', induced by W by $G'[W]$.

The subgraph of G, induced by all vertices in C_i and C_i^j, together with a complete set of vertices in C_i, is denoted by $G(C_i, C_i^j)$, or in short $G(i, j)$. (I.e. (v, w) is an edge in $G(i, j)$, if and only if $v, w \in C_i \cup C_i^j$ and ($(v, w) \in E$ or $v, w \in C_i$).)

Similarly, $H(D_i, D_i^j)$, or in short $H(i, j)$, denotes the subgraph of H, induced by all vertices in $D_i \cup D_i^j$, together with a complete set of vertices in D_i.

The algorithm to recognize partial k-trees of [2] relies heavily on two lemmas. We use the first of these lemmas, and a small modification of the second one.

Lemma 4.1 (Arnborg et. al. [2]) *Suppose $n \geq k+2$. G is a partial k-tree, if and only if there exists a k-vertex separator C_r, such that all subgraphs $G(r,j)$ are partial k-trees.*

Lemma 4.2 *A graph $G(i,j)$ with at least $k+2$ vertices is a partial k-tree, if and only if there exists a vertex $v \in C_i^j$, such that for each connected component A of the graph, obtained by removing v from C_i^j, there is a k-vertex separator $C_m \subseteq C_i \cup \{v\}$, such that*

1. *No vertex in A is adjacent to the (unique) vertex in $C_i \cup \{v\} - C_m$.*
2. *$G(C_m, A)$ is a partial k-tree.*

The proof is similar to the proof in [2]. Also one can show, that each component A, appearing in the lemma can be written as C_m^l, (with the corresponding value of m).

DEFINITION 4.1 *Let f be a bijection $C_i \rightarrow D_{i'}$. We say that the pair (C_i, C_i^j) is f-isomorphic to $(D_{i'}, D_{i'}^{j'})$, denoted by $(C_i, C_i^j) \equiv^f (D_{i'}, D_{i'}^{j'})$, if and only if there exists a function $\psi : C_i \cup C_i^j \rightarrow D_{i'} \cup D_{i'}^{j'}$, such that*

1. $\forall\, v \in C_i : \psi(v) = f(v)$.
2. $\forall\, v, w \in C_i \cup C_i^j : (v,w) \in E_G \Leftrightarrow (\psi(v), \psi(w)) \in E_H$.

The first step in our algorithm is to run the recognition algorithm of Arnborg, Corneil and Proskurowski [2] on G, with some extra bookkeeping. In this way one can not only determine that G is a partial k-tree, but one can also obtain the information, indicated in lemma 4.1 and lemma 4.2. In other words, we obtain

1. A k-vertex separator C_r, such that all subgraphs $G(r,j)$ are partial k-trees.
2. For all $G(i,j)$, arising in this collection of information, one has a vertex $v \in C_i^j$ and we write each component A of $G[C_i^j - \{v\}]$ as some C_m^l, with
   (a) No vertex in $A = C_m^l$ is adjacent to the (unique) vertex in $C_i \cup \{v\} - C_m$.
   (b) $G(m,l)$ is a partial k-tree, and either $G(m,l)$ has $\leq k+1$ vertices, or we have, recursively, similar information for $G(m,l)$.

This collection of information will be called a "representation of G as partial k-tree".

From now on we will consider only $G(i,j)$, and C_i^j, appearing in the representation of G as partial k-tree. Note that there are $\mathcal{O}(n)$ such $G(i,j)$'s and C_i^j's.

The next step of the algorithm is to find all D_i and D_i^j, as defined before. This can be done in $\mathcal{O}(n^{k+1})$ time.

The following two lemma's give the essential steps of our algorithm.

Lemma 4.3 *Suppose G has a k-vertex separator C_r, such that all graphs $G(r,j)$ are partial k-trees. Suppose the number of C_r^j is m.*

Then G is isomorphic to H, if and only if H has a k-vertex separator D_s, such that there exists a bijection $f : C_r \rightarrow D_s$, such that

1. $\forall\, v, w \in C_r : (v,w) \in E_G \Leftrightarrow (f(v), f(w)) \in E_H$.

2. *Let $D_s^1, D_s^2, \ldots, D_s^{m'}$ be the connected components of the graph, obtained by removing D_s from H, i.e. of $H[V_H - D_s]$. Then $m = m'$, and there exists a bijection $\phi : \{1, \ldots, m\} \rightarrow \{1, \ldots, m'\}$, such that for all l, $1 \leq l \leq m$, $(C_r, C_r^l) \equiv^f (D_s, D_s^{\phi(l)})$.*

Proof. ($\Leftarrow$) Suppose G is isomorphic to H. Let ψ be an isomorphism of G to H. Let $D_s = \psi(C_r)$. It follows that D_s is a k-vertex separator of H. Let f be ψ, restricted to C_r. f is a bijection $C_r \rightarrow D_s$. Each C_r^j is mapped to a unique D_s^l by ψ, say $D_s^{\phi(j)}$. Now ϕ is a bijection $\{1, \ldots, m\} \rightarrow \{1, \ldots, m\}$, and for all $l, 1 \leq l \leq m$, $(C_r, C_r^l) \equiv^f (D_s, D_s^{\phi(l)})$.
($\Rightarrow$) Define $\psi : V_G \rightarrow V_H$ as follows:

1. If $v \in C_r$, then $\psi(v) = f(v)$.
2. If $v \in C_r^l$, for some l, $1 \leq l \leq m$, then there exists a function $\psi^l : C_r \cup C_r^l \rightarrow D_r \cup D_r^{\phi(l)}$, such that
   (a) $\forall\, v \in C_r : f(v) = \psi^l(v)$.
   (b) $\forall\, v, w \in C_r \cup C_r^l : (v, w) \in E_G \Leftrightarrow (\psi^l(v), \psi^l(w)) \in E_H$.

We now claim that ψ is an isomorphism from G to H. Let $v, w \in V_G$.

Case 1: $v, w \in C_r$. Now $(v, w) \in E_G \Leftrightarrow (f(v), f(w)) \in E_H \Leftrightarrow (\psi(v), \psi(w)) \in E_H$.

Case 2: $v, w \in C_r^l$. Then $(v, w) \in E_G \Leftrightarrow (\psi^l(v), \psi^l(w)) \in E_H \Leftrightarrow (\psi(v), \psi(w)) \in E_H$.

Case 3: $v \in C_r^{l_1}, w \in C_r^{-2}, l_1 \neq l_2$. It follows that $(v, w) \notin E_G$, and $\psi(v) \in D_s^{\phi(l_1)}$, $\psi(w) \in D_s^{\phi(l_2)}$, $\phi(l_1) \neq \phi(l_2)$. Hence $(\psi(v), \psi(w)) \notin E_H$.

Case 4: $v \in C_r, w \in C_r^l$. Now $(v, w) \in E_G \Leftrightarrow (\psi^l(v), \psi^l(w)) \in E_H \Leftrightarrow (f(v), \psi^l(w)) \in E_H$ $\Leftrightarrow (\psi(v), \psi(w)) \in E_H$.

Case 5: $v \in C_r^l$, $w \in C_r$. Similar to case 4.

It follows that ψ is an isomorphism from G to H. **Q.E.D.**

Lemma 4.4 *Let $G(i, j)$ be a partial k-tree, and let $v \in C_i^j$. Let for each connected component A_p of the graph, obtained by removing v from C_i^j a k-vertex separator $C_{m_p} \subseteq C_i \cup \{v\}$ be given, such that*

1. *No vertex in A_p is adjacent to the vertex in $C_i \cup \{v\} - C_{m_p}$.*
2. *$G(C_{m_p}, A_p)$ is a partial k-tree.*

Let the number of connected components of the graph, obtained by removing v from C_i^j be m. Let $D_{i'}$ be a k-vertex separator of H, and $D_{i'}^{j'}$ a connected component of $H[V_H - D_{i'}]$. Let f be a bijection $C_i \rightarrow D_{i'}$.

Then $(C_i, C_i^j) \equiv^f (D_{i'}, D_{i'}^{j'})$, if and only if the following conditions hold:

1. *$\forall\, v, w \in C_i : (v, w) \in E_G \Leftrightarrow (f(v), f(w)) \in E_H$.*
2. *$\exists w \in D_{i'}^{j'}$, such that the number of connected components of the graph, obtained by removing w from $D_{i'}^{j'}$ is m, and for each of these components B_q, we can find a k-vertex separator $D_{m_q} \subseteq D_{i'} \cup \{w\}$, such that*

(a) no vertex in B_q is adjacent to the vertex in $D_i^j \cup \{w\} - D_{m_q}$.

(b) there exists a bijection $\phi : \{1,\ldots,m\} \to \{1,\ldots,m\}$, such that for all $l, 1 \leq l \leq m$: $(C_{m_l}, A_l) \equiv^{f'} (D_{m_{\phi(l)}}, B_{\phi(l)})$, with $f'(x) = f(x)$, for $x \in C_i$, and $f'(v) = w$.

Also, for each such B_q, there is an l, such that $B_q = D^l_{m_q}$, and $|B_q| < |D_{i'}^{j'}|$.

The proof is similar to the proof of lemma 4.3. We now proceed with the description of the algorithm.

The algorithm must now determine for all (C_i, C_i^j), appearing in the representation of G as partial k-tree and for all $(D_{i'}, D_{i'}^{j'})$, and all bijections $f : C_i \to D_{i'}$, whether $(C_i, C_i^j) \equiv^f (D_i, D_i^j)$, storing the answers to these questions in some data structure. This is done in the following manner.

We first sort all (D_i, D_i^j) to increasing size; and now deal with them one by one as they appear in the sorted order; beginning with the smallest ones.

For pairs $(D_{i'}, D_{i'}^{j'})$, with $|D_{i'}^{j'}| \leq 1$, we can determine, for each (C_i, C_i^j), and $f : C_i \to D_{i'}$, whether $(C_i, C_i^j) \equiv^f (D_i, D_i^j)$, in $\mathcal{O}(1)$ time per triple. Hence the total time for all these pairs is bounded by $\mathcal{O}(n^{k+1} \cdot 1 \cdot n) = \mathcal{O}(n^{k+2})$.

Suppose now we are dealing with some $(D_{i'}, D_{i'}^{j'})$, with $|D_{i'}^{j'} > 1$. For each (C_i, C_i^j) in the collection of information of G, and each bijection $f : C_i \to D_{i'}$ we follow the same procedure. Suppose (C_i, C_i^j) and f are given.

Let $v \in C_i^j$ be given, as indicated in lemmas 4.2 and 4.4. Let for each connected component A_p of the graph, obtained by removing v from C_i^j a k-vertex separator $C_{m_p} \subseteq C_i \cup \{v\}$ be given, such that

1. No vertex in A_p is adjacent to the vertex in $C_i \cup \{v\} - C_{m_p}$.
2. $G(C_{m_p}, A_p)$ is a partial k-tree.

Let the number of connected components of the graph, obtained by removing v from C_i^j be m.

First one tests whether $\forall\, v, w \in C_i : (v, w) \in E_G \Leftrightarrow (f(v), f(w)) \in E_H$.

If this test does not fail, then we perform the following actions for each $w \in D_{i'}^{j'}$: Calculate all connected components B_q of the graph, obtained by removing w from $D_{i'}^{j'}$. Now there are at most $k+1$ different k-vertex separators $\subseteq D_i \cup \{w\}$. For each of these, say D', one must test, whether no vertex in B_q is adjacent to the (unique) vertex in $(D_{i'} \cup \{w\}) - D'$. For each D', passing this test, one must now determine, for which (C_{m_p}, A_p) one has $(C_{m_p}, A_p) \equiv^{f'} (D', B_q)$, with $f'(x) = f(x)$ for $x \in C_i$, and $f'(v) = w$. Since $|B_q| < |D_{i'}^{j'}|$, we have computed this information earlier in the algorithm, and hence the information can be "looked up".

The problem of finding the bijection, as indicated in lemma 4.4 can now be translated to the problem of finding a perfect matching in a bipartite graph. The nodes in this graph represent the A_p's and the B_q's, there is an edge between the nodes representing A_p and B_q, if and only if there is a $D' \subseteq D_{i'} \cup \{w\}$, with $(C_{m_p}, A_p) \equiv^{f'} (D', B_q)$. It follows from lemma 4.4, that if we have such a perfect matching, then we can conclude that $C_i, C_i^j) \equiv^f (D_i, D_i^j)$. As finding a perfect matching in a bipartite graph with n vertices can be done in $\mathcal{O}(n^{2.5})$ time (see e.g. [9]), it follows that this last step takes $\mathcal{O}(|D_{i'}^{j'}|^{2.5})$ time.

Hence, the total time needed to deal with one triple $(C_i, C_i^j), (D_{i'}, D_{i'}^{j'}), f$ is bounded by $\mathcal{O}(|D_{i'}^{j'}|^{3.5})$.

It follows that the total time over all such triples is bounded by $\mathcal{O}(n^{k+4.5})$.

Finally, testing whether G and H are isomorphic is done with help of all computed information, and lemma 4.3. One must test for all k-vertex separators D_s, whether it fulfills the conditions of lemma 4.3. (Note that C_r is given.) This can be done with a similar perfect-matching procedure as before. The total time of this step is bounded by $\mathcal{O}(n^{k+3.5})$.

This completes the description of our algorithm. The following theorem follows.

Theorem 4.5 *There exists an algorithm that decides whether two given partial k-trees $G = (V_G, E_G)$ and $H = (V_H, E_H)$ are isomorphic in time $\mathcal{O}(n^{4.5})$.*

5 Final remarks

In this paper we proved that the CHROMATIC INDEX and the GRAPH ISOMORPHISM problems are solvable in polynomial time for graphs with a constant upper bound on the treewidth. In [5] it is shown, that each of the following classes of graphs has a constant upper bound on the treewidth of graphs in the class: almost trees with parameter k, graphs with bandwidth k, k-outerplanar graphs, (k constant). Hence, for these classes of graphs, the CHROMATIC INDEX and the GRAPH ISOMORPHISM problem are also solvable in polynomial time.

We did not show how to *obtain* edge-colorings of G with the desired number of colors, or how to *construct* a graph isomorphism between two partial k-trees, if one exists. However, this can be done with some small modifications of the algorithms (using some extra bookkeeping), also in polynomial time. We omit the details.

Although our algorithm for CHROMATIC INDEX on partial k-trees is polynomial, the time it uses is double-exponential in k, and hence the algorithm is far from practical, even for the smallest values of k. For the case that the treewidth is at most 2 (or, equivalently: G is a series-parallel graph), an $\mathcal{O}(n \cdot |E|) = \mathcal{O}(n^3)$ algorithm was found by Syslo [15]. Hence, it is an interesting open problem to find more *practical* polynomial algorithms for the CHROMATIC INDEX problem on graphs with a treewidth $\leq k$, for constants $k > 2$.

References

[1] S. Arnborg. Efficient algorithms for combinatorial problems on graphs with bounded decomposability – A survey. *BIT*, 25:2–23, 1985.

[2] S. Arnborg, D. Corneil, and A. Proskurowski. Complexity of finding embeddings in a k-tree. *SIAM J. Alg. Disc. Meth.*, 8:277–284, 1987.

[3] S. Arnborg and A. Proskurowski. Characterization and recognition of partial 3-trees. *Siam J. Alg. Disc. Meth.*, 7:305–314, 1986.

[4] S. Arnborg and A. Proskurowski. *Linear time algorithms for NP-hard problems on graphs embedded in k-trees.* TRITA-NA-8404, Dept. of Num. Anal. and Comp. Sci., Royal Institute of Technology, Stockholm, Sweden, 1984.

[5] H. L. Bodlaender. *Classes of Graphs with Bounded Treewidth.* Tech. Rep. RUU-CS-86-22, Dept. of Comp. Science, University of Utrecht, Utrecht, 1986.

[6] H. L. Bodlaender. *Dynamic programming algorithms on graphs with bounded treewidth.* Tech. Rep. MIT/LCS/TR-394, Lab. for Comp. Science, M.I.T., 1987.

[7] S. Fiorini and R. J. Wilson. *Edge-colorings of Graphs.* Pitman, London, 1977.

[8] I. Holyer. The NP-completeness of edge-coloring. *SIAM J. Comput*, 10:718–720, 1981.

[9] J. Hopcroft and R. M. Karp. An $n^{5/2}$ algorithm for maximum matching in bipartite graphs. *SIAM J. Comput*, 4:225-231, 1975.

[10] D. S. Johnson. The NP-completeness column: an ongoing guide. *J. of Algorithms*, 6:434–451, 1985.

[11] D. S. Johnson. The NP-completeness column: an ongoing guide. *J. of Algorithms*, 8:285–303, 1987.

[12] N. Robertson and P. Seymour. Graph minors. II. Algorithmic aspects of tree-width. *J. of Algorithms*, 7:309–322, 1986.

[13] J. Saxe. Dynamic programming algorithms for recognizing small-bandwidth graphs in polynomial time. *SIAM J. Alg. Disc. Meth.*, 1:363–369, 1980.

[14] P. Scheffler and D. Seese. A combinatorial and logical approach to linear-time computability. 1986. Extended abstract.

[15] M. Sysło. NP-complete problems on some tree-structured graphs: a review. In M. Nagl and J. Perl, editors, *Proc. WG'83 International Workshop on Graph Theoretic Concepts in Computer Science*, pages 342–353, Univ. Verlag Rudolf Trauner, Linz, West Germany, 1983.