Discussiones Mathematicae
Graph Theory xx (xxxx) 1–18
https://doi.org/10.7151/dmgt.2606

MAXIMUM COMMON INDUCED SUBFORESTS AND MINIMUM COMMON INDUCED SUPERFORESTS OF A SET OF FORESTS

Dieter Rautenbach and Florian Werner

Institute of Optimization and Operations Research
Ulm University, Ulm, Germany

e-mail: dieter.rautenbach@uni-ulm.de
florian.werner@uni-ulm.de

Abstract

Graph isomorphism, subgraph isomorphism, and maximum common subgraphs are classical well-investigated objects. In the present paper, for a given set of forests, we study maximum common induced subforests and minimum common induced superforests. We show that finding a maximum subforest is NP-hard for two given subdivided stars while finding a minimum superforest is tractable for two trees but NP-hard for three trees. For a given set of k trees, we present an efficient greedy $\left(\frac{k}{2}-\frac{1}{2}+\frac{1}{k}\right)$-approximation algorithm for the minimum superforest problem. Finally, we present a polynomial time approximation scheme for the maximum subforest problem for any given set of forests.

Keywords: subgraph isomorphism, common subgraph.

2020 Mathematics Subject Classification: 05C60, 05C05.

1. Introduction

We consider finite, simple, and undirected graphs and all considered subgraphs are induced. As usual, a *forest* is a graph without cycles and a *tree* is a connected forest. In particular, we consider trees that are unrooted and unlabeled. For a graph G and a vertex u of G, let $n(G)$ denote the order of G and let $d_G(u)$ denote the degree of u in G. If two graphs G and H are isomorphic, we write $G \simeq H$. Let $\mathbb{N}$ be the set of positive integers, and let $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$. For a positive integer k, let $[k]$ denote the set of positive integers at most k, and let $[k]_0 = [k] \cup \{0\}$.

Let $\mathcal{G}$ be a set of graphs. A *subgraph* of $\mathcal{G}$ is a graph H such that, for every graph G in $\mathcal{G}$, the graph G has an induced subgraph that is isomorphic to H.

A *supergraph* of $\mathcal{G}$ is a graph H such that, for every graph G in $\mathcal{G}$, the graph H has an induced subgraph that is isomorphic to G. A subgraph that is a forest or tree is called a *subforest* or *subtree*, respectively. A supergraph that is a forest or tree is called a *superforest* or *supertree*, respectively.

In this paper we consider the following natural optimization problems.

MAXIMUM SUBFOREST

Instance: A set $\mathcal{F}$ of forests.
Task: Determine a subforest F of $\mathcal{F}$ of maximum order.

MINIMUM SUPERFOREST

Instance: A set $\mathcal{F}$ of forests.
Task: Determine a superforest F of $\mathcal{F}$ of minimum order.

Both problems are NP-hard even when restricted to instances $\mathcal{F} = \{F_1, F_2\}$, where F_1 and F_2 are unions of paths. Let I be an instance of the strongly NP-complete problem 3-PARTITION, cf. problem [SP15] in the appendix of [10] containing a list of NP-complete problems. Let I consist of $3m$ positive integers $a_1, \ldots, a_{3m}$ with $A/4 < a_i < A/2$ for each $i \in [3m]$, where $A = \frac{1}{m}(a_1 + \cdots + a_{3m})$. The task for I is to decide whether there is a partition of $[3m]$ into m sets $I_1, \ldots, I_m$ each containing exactly three elements such that $\sum_{j \in I_i} a_j = A$ for each $i \in [m]$. Let F_1 be the forest with $3m$ components that are paths of order $a_1, \ldots, a_{3m}$ and let F_2 be the forest with m components that are paths of order $A + 2$. Note that $n(F_2) = n(F_1) + 2m$.

Obviously, the following statements are equivalent.

(i) I is a yes-instance of 3-PARTITION.
(ii) F_1 is isomorphic to an induced subtree of F_2.
(iii) F_1 is a maximum subforest of $\{F_1, F_2\}$.
(iv) F_2 is a minimum superforest of $\{F_1, F_2\}$.

These equivalences imply the stated hardness of MAXIMUM SUBFOREST and MINIMUM SUPERFOREST. They also show that these problems are closely related to the very well-studied subtree/subgraph isomorphism problem [1, 7, 12, 15, 16]. Maximum common (induced and non-induced) subgraphs were first studied by Bokhari [8] in the context of array processing and are applied in areas ranging from molecular chemistry [17] to pattern matching [18]. The maximum common connected induced subgraph problem was shown to be NP-hard for 3-outerplanar labeled graphs of maximum degree and treewidth at most 4 [5, 6] and for two biconnected series-parallel graphs [14]. It can be solved efficiently [19] for a degree-bounded partial k-tree and a connected graph, whose number of spanning trees is polynomial. For the maximum common induced subgraph problem the parameterized complexity is studied in [2, 3].

Modifying the above NP-hardness comments similarly as in [11] yields the following. A *subdivided star* is a tree that arises from a star by repeatedly subdividing edges.

Proposition 1. MAXIMUM SUBFOREST *restricted to instances* $\{T_1, T_2\}$ *consisting of two subdivided stars is NP-hard.*

Note that all proofs are postponed to Section 2.

If the set $\mathcal{F}$ contains only trees and F is a minimum superforest of $\mathcal{F}$, then each copy of a tree in $\mathcal{F}$ is completely contained in a single component of F. If F would not be connected, then selecting one vertex from each component of F and identifying all selected vertices to a single vertex would yield a strictly smaller superforest of $\mathcal{F}$. This argument implies the following.

Proposition 2. *Every minimum superforest of a set of trees is a tree.*

For two trees T_1 and T_2, minimum supertree $T_\cup$ of $\{T_1, T_2\}$, and a maximum subtree $T_\cap$ of $\{T_1, T_2\}$, the following inclusion-exclusion formula concerning the orders of these trees is straightforward.

$$n(T_\cup) = n(T_1) + n(T_2) - n(T_\cap). \tag{1}$$

Furthermore, given subtrees of T_1 and T_2 isomorphic to $T_\cap$, a minimum superforest of $\{T_1, T_2\}$ can easily be constructed by extending the copy of $T_\cap$ within T_1 by adding $n(T_2) - n(T_\cap)$ new vertices and suitable edges creating a copy of T_2. Refering to Edmonds and Matula, Akutsu [4] showed that, for two given trees T_1 and T_2, some maximum subtree of $\{T_1, T_2\}$ can be determined efficiently combining a weighted bipartite matching algorithm with dynamic programming.

Together our observations imply the following.

Proposition 3. MINIMUM SUPERFOREST *restricted to instances* $\{T_1, T_2\}$ *consisting of two trees can be solved in polynomial time.*

In [4] Akutsu also showed that it is NP-hard to determine a maximum subtree of three given trees. Reflecting this result, we show the following, which does not follows from Akutsu's result.

Theorem 4. MINIMUM SUPERFOREST *restricted to instances* $\{T_1, T_2, T_3\}$ *consisting of three trees is NP-hard.*

For instances of bounded maximum degree, the problem can be solved efficiently.

Theorem 5. *For every* $\Delta \in \mathbb{N}$*, there is some* $p \in \mathbb{N}$ *with the following property. For a given set* $\mathcal{T} = \{T_1, T_2, T_3\}$ *consisting of three trees of order at most* n *and maximum degree at most* Δ*, one can determine in time* $O(n^p)$ *a minimum superforest* T *of* $\mathcal{T}$*.*

By Proposition 3, some minimum supertree, say $s(T,T')$, of two given trees T and T' can be determined efficiently. Repeated applications of this lead to the following natural simple greedy algorithm.

Input: A set $\{T_1,\dots,T_k\}$ of trees.
Output: A supertree T of $\{T_1,\dots,T_k\}$.
begin
  for $i=1$ **to** k **do**
    $S_i \leftarrow T_i$;
    for $j=2$ **to** k **do**
      $S_i \leftarrow s(S_i,T_{i+j-1})$, where indices are identified modulo k;
    end
  end
  $\ell \leftarrow \operatorname{argmin}\{n(S_i) : i \in [k]\}$;
  return S_ℓ;
end

Algorithm 1: Greedy Supertree.

Theorem 6. Greedy Supertree *is a* $\left(\frac{k}{2}-\frac{1}{2}+\frac{1}{k}\right)$*-approximation algorithm with polynomial running time for* MINIMUM SUPERFOREST *restricted to instances* $\{T_1,\dots,T_k\}$ *consisting of* k *trees.*

For $k=3$, Theorem 6 provides the approximation factor $4/3$. In Section 2 we show that our analysis of `Greedy Supertree` is essentially best possible and that this factor cannot be improved. The appearance of the factor $4/3$ in this context is actually not surprising. A natural simple dynamic programming algorithm that determines a minimum supertree of two given trees uses a maximum bipartite matching algorithm as a subroutine. Extending this dynamic programming approach from two to three trees would require to replace this subroutine with a 3-dimensional matching algorithm. Now, $4/3+\epsilon$ is the best known approximation factor for 3-dimensional matching [9] with no improvement during the past decade. More generally, the approximation factor in Theorem 6 reflects that the best known [13] approximation factor for the k-set packing problem is $k/2+\epsilon$. Altogether, a natural challenging problem in this context is to improve the approximation factor of $4/3$ for MINIMUM SUPERFOREST for sets $\{T_1,T_2,T_3\}$ of three given trees.

In contrast to that MAXIMUM SUBFOREST allows a polynomial time approximation.

Theorem 7. *For every* $\epsilon>0$*, there is some* $p\in\mathbb{N}$ *with the following property. For a given set* $\mathcal{F}=\{F_1,\dots,F_k\}$ *consisting of* k *forests of order at most* n*, one can determine in time* $O(kn^p)$ *a subforest* F *of* $\mathcal{F}$ *with* $n(F)\geq(1-\epsilon)n(F_{\rm opt})$*, where* $F_{\rm opt}$ *is some maximum subforest of* $\mathcal{F}$*.*

2. PROOFS

For convenience, we restate the results from Section 1 whose proofs are given here.

Proposition 1. MAXIMUM SUBFOREST *restricted to instances* $\{T_1, T_2\}$ *consisting of two subdivided stars is NP-hard.*

Proof. Let I be an instance of 3-PARTITION that consists of $3m$ positive integers $a_1, \ldots, a_{3m}$ with $A/4 < a_i < A/2$ for each $i \in [3m]$, where $A = \frac{1}{m}(a_1 + \cdots + a_{3m})$. Let F_1 be the forest with $3m$ components that are paths of order $a_1, \ldots, a_{3m}$ and let F_2 be the forest with m components that are paths of order $A + 2$. Let T_1 arise from F_1 by adding one new vertex r_1 as well as $3m$ new edges between r_1 and one endvertex in each component of F_1. Similarly, let T_2 arise from F_2 by adding one new vertex r_2 as well as m new edges between r_2 and one endvertex in each component of F_2. Note that T_1 and T_2 are subdivided stars.

In order to complete the proof, we show that I is a yes-instance of 3-PARTITION if and only if a maximum subforest of $\{T_1, T_2\}$ has order $n(T_1) - 1 = n(F_1) = a_1 + \cdots + a_{3m}$. Clearly, we may assume that $m \geq 8$. Note that, since T_2 contains no vertex of degree $d_{T_1}(r_1) = 3m$, T_1 is not a subtree of T_2, and, hence, a maximum subforest of $\{T_1, T_2\}$ has order at most $n(T_1) - 1$.

If I is a yes-instance of 3-PARTITION, then removing from T_1 only the vertex r_1 and removing from T_2 the vertex r_2 as well as two further vertices from each component of F_2 corresponding to a feasible solution for I yields two forests that are both isomorphic to F_1. Conversely, suppose now that F is an induced subforest of T_1 of order $n(T_1) - 1$ that is isomorphic to an induced subforest F' of T_2. Note that F arises from T_1 by removing a single vertex. Suppose, for a contradiction, that r_1 belongs to F. This implies $d_F(r_1) \geq d_{T_1}(r_1) - 1 = 3m - 1 > m$. Since m is the maximum degree of T_2, this is impossible, which implies $F = T_1 - r_1 = F_1$. Suppose, for a contradiction, that r_2 belongs to F'. Since F is the union of paths, this implies that there are $m - 2$ neighbors $u_1, \ldots, u_{m-2}$ of r_2 in T_2 that do not belong to F'. Let $P_1, \ldots, P_{m-2}$ be the components of F_2 such that P_i contains u_i for $i \in [m - 2]$. Since each $P_i - u_i$ is a path of order $A + 1$ and each a_i is strictly less than $A/2$, for each P_i, there are at least three vertices that do not belong to F'. Since $m \geq 8$, this implies the contradiction $n(F') \leq n(T_2) - 3(m - 2) < n(T_2) - 2m - 1 = n(T_1) - 1 = n(F_1) = n(F)$. Hence, r_2 does not belong to F', that is, F' is an induced subforest of F_2. Again, since each component of F_2 is a path of order $A + 2$ and each a_i is strictly less than $A/2$, for each component of F_2, there are at least two vertices that do not belong to F'. Since $n(F') = n(F) = n(T_2) - 2m - 1 = n(F_2) - 2m$, it follows that each component of F_2 contains exactly two vertices that do not belong to F'. These two vertices from each component of F_2 indicate a feasible solution for I, which

implies that I is a yes-instance of 3-PARTITION. ∎

Theorem 4. MINIMUM SUPERFOREST *restricted to instances* $\{T_1, T_2, T_3\}$ *consisting of three trees is NP-hard.*

Proof. We show this result by a polynomial-time reduction of the well-known NP-complete problem 3-DIMENSIONAL MATCHING (3DM), cf. [SP1] in [10], to MINIMUM SUPERFOREST. Let I be an instance of 3DM consisting of three disjoint sets $X = \{x_1, \ldots, x_q\}$, $Y = \{y_1, \ldots, y_q\}$, and $Z = \{z_1, \ldots, z_q\}$ as well as a set $M \subseteq X \times Y \times Z$ of triples. As 3DM remains NP-complete under this restriction [10], we assume that every element of $X \cup Y \cup Z$ occurs in some triple but no element of $X \cup Y \cup Z$ occurs in more than three triples. For each $i \in [q]$, let $T_0(x_i)$ be the tree that arises from the disjoint union of an isolated vertex $r(x_i)$ and three paths P_1, P_2, and P_3, each of order $2q$, by adding an edge between $r(x_i)$ and an endvertex of each P_ℓ. The three vertices in $T_0(x_i)$ at distance j from $r(x_i)$ are associated with y_j and the three vertices in $T_0(x_i)$ at distance $q + k$ from $r(x_i)$ are associated with z_k.

If x_i is contained in three triples from M, then the tree $T(x_i)$ arises from $T_0(x_i)$ by associating each triple (x_i, y_j, z_k) from M containing x_i with a different path P_ℓ and attaching one new endvertex to each of the two vertices in that P_ℓ at distances j and $q + k$ from $r(x_i)$, that is, the two vertices associated with y_j and z_k, respectively. See Figure 1 for an illustration.

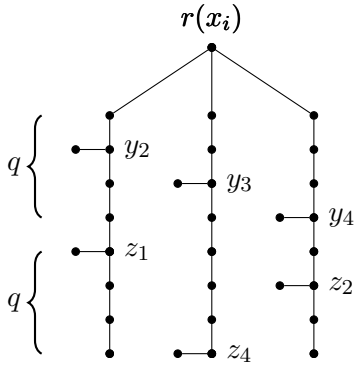

Figure 1. The tree $T(x_i)$ if $q = 4$ and x_i is contained in the three triples (x_i, y_2, z_1), (x_i, y_3, z_4), and (x_i, y_4, z_2).

If x_i is contained in less than three triples, then proceed as before for the one or two triples containing x_i and attach a new endvertex to each of the $2q$ vertices of those P_ℓ that are not associated with some triple containing x_i. See Figure 2 for an illustration.

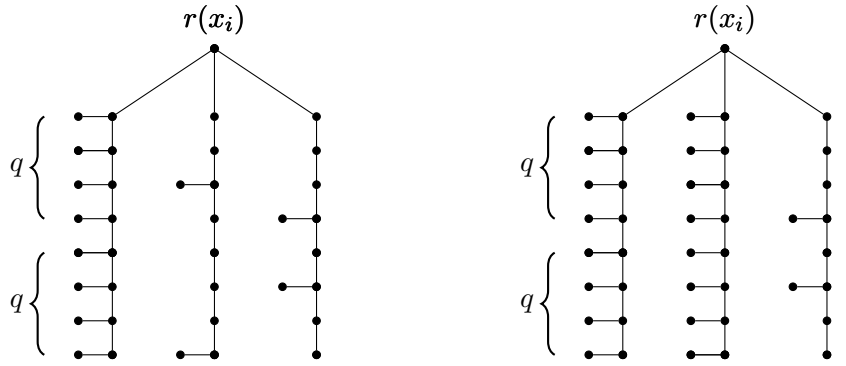

Figure 2. The left shows $T(x_i)$ if $q = 4$ and x_i is contained in exactly the two triples (x_i, y_3, z_4) and (x_i, y_4, z_2). The right shows $T(x_i)$ if $q = 4$ and x_i is contained in only one triple (x_i, y_4, z_2).

For each $j \in [q]$, let $T(y_j)$ be the tree that arises from the disjoint union of an isolated vertex $r(y_j)$ and three paths P_1, P_2, and P_3, each of order $2q$, by

- adding an edge between $r(y_j)$ and an endvertex of each P_ℓ,
- attaching a new endvertex to each of the $2q$ vertices of two of the P_ℓ, and
- attaching one new endvertex to the vertex at distance j from $r(y_j)$ on the third P_ℓ, which we call *the relevant branch for* y_j in what follows.

Let $T(z_k)$ be defined similarly. In particular, $T(z_k)$ has an endvertex attached to a vertex at distance $q + k$ from $r(z_k)$; see Figure 3 for an illustration.

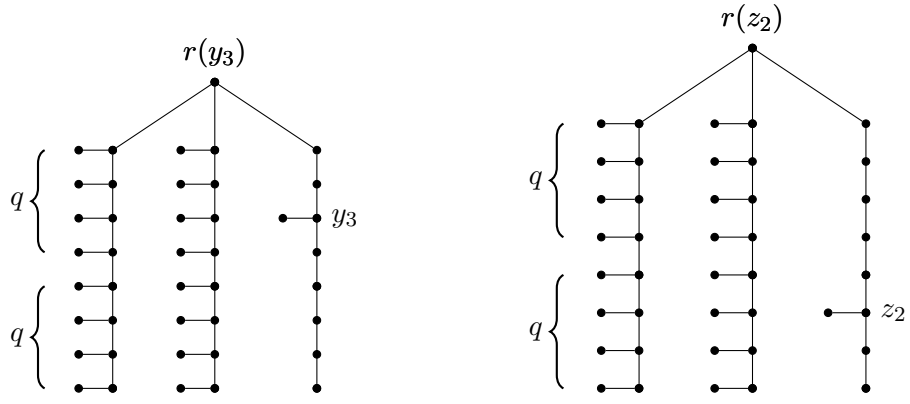

Figure 3. $T(y_3)$ on the left and $T(z_2)$ on the right.

Now, let T_x arise from the disjoint union of an isolated vertex r_x and the trees $T(x_1), \ldots, T(x_q)$ by adding q new edges between r_x and $r(x_1), \ldots, r(x_q)$. Let T_y and T_z be defined similarly. Note that the trees T_x, T_y, and T_z are rooted in the vertices r_x, r_y, and r_z of degree q, respectively. The order of T_y and T_z is $n = 10q^2 + 2q + 1$ while the order of T_x depends on the instance I. In order to complete the proof, we show that I is a yes-instance of 3DM if and only if a minimum superforest for $\{T_x, T_y, T_z\}$ has order at most $10q^2 + 3q + 1$.

Suppose that I is a yes-instance of 3DM. Let $M^* \subseteq M$ be such that every element of $X \cup Y \cup Z$ belongs to exactly one triple from M^*. Let the tree T arise from T_y as follows. For each $j \in [q]$, consider the unique triple, say (x_i, y_j, z_k), from M^* that contains y_j, and attach a new endvertex to the vertex at distance $q + k$ from $r(y_j)$ associated with z_k that belongs to the relevant branch for y_j. Clearly, the order of T is $n(T_y) + q = 10q^2 + 3q + 1$ and it is easy to verify that T contains three induced subtrees isomorphic to T_x, T_y, and T_z, respectively.

Conversely, suppose that a minimum superforest T for $\{T_x, T_y, T_z\}$ has order at most $10q^2 + 3q + 1$, which equals $n(T_y) + q$. By renaming vertices, we may assume that T arises from T_y by adding at most q vertices and suitable edges. The structure of T_y and T_z implies that T arises from T_y by attaching one new endvertex to some vertex of each of the q relevant branches within T_y; these q additional vertices are attached to vertices associated with the distinct elements of Z. Since T_x is an induced subgraph of T and r_y is the only vertex of T of degree q, for a copy of T_x within T, the root vertex r_x of T_x is mapped to r_y and the q children of r_x in T_x are mapped in a bijective way to the q children of r_y within T. This bijective mapping indicates how to choose, for each $i \in [q]$, a triple from M containing x_i, for which the set M^* of all q selected triples is such that every element of $X \cup Y \cup Z$ is contained in exactly one triple from M^*. This completes the proof. ∎

Theorem 5. *For every $\Delta \in \mathbb{N}$, there is some $p \in \mathbb{N}$ with the following property. For a given set $\mathcal{T} = \{T_1, T_2, T_3\}$ consisting of three trees of order at most n and maximum degree at most Δ, one can determine in time $O(n^p)$ a minimum superforest T of $\mathcal{T}$.*

Proof. Let Δ be a fixed positive integer. Let $\mathcal{T} = \{T_1, T_2, T_3\}$ be the set of the three given trees of order at most n and maximum degree at most Δ. For notational simplicity, assume that the trees T_i have disjoint sets of vertices. We explain how to determine in time polynomially bounded in n supertrees T of $\mathcal{T}$ that

- either contain disjoint copies of two of the three trees (type 1),
- or contain copies of all three trees that pairwise intersect (type 2)

and are of minimum order subject to this condition. Returning the smallest such supertree yields a minimum supertree of $\mathcal{T}$. Note that the polynomial bound on the running time will depend on the fixed Δ.

Firstly, consider supertrees of type 1 that contain disjoint copies of T_2 and T_3; the other two pairs can be treated symmetrically. Let $\{T_4, \ldots, T_q\}$ be the set of all trees that arise from disjoint copies of T_2 and T_3 and a path P of order between 2 and n by identifying some vertex u in T_2 with one endvertex of P and some vertex v of T_3 with the other endvertex of P. Since there are at most n choices for the length of P, for the vertex u, and for the vertex v, we have $q = O(n^3)$. By Proposition 3, the $O(n^3)$ minimum supertrees of $\{T_1, T_4\}, \{T_1, T_5\}, \ldots, \{T_1, T_q\}$ can be determined in polynomial time, and a smallest of all these trees is a supertree of $\mathcal{T}$ containing disjoint copies of T_2 and T_3 that is of minimum order subject to this condition.

Secondly, consider a supertree T of type 2. Let T_i' be an induced copy of T_i within T such that T_1', T_2', and T_3' pairwise intersect. By the Helly property of subtrees of a tree, some vertex, say r, belongs to T_1', T_2', and T_3'. For all possible at most $O(n^3)$ choices for vertices r_1 in T_1, r_2 in T_2, and r_3 in T_3 corresponding to r, we proceed as follows for every $i \in [3]$.

- Root T_i in r_i.
- For $u_i \in V(T_i)$, let $T_i(u_i)$ be the subtree of T_i rooted in u_i that is induced by u_i and all its descendants within T_i.
- Let $f(\{u_i\}) = n(T_i(u_i))$.
- Let $f(\{u_1, u_2, u_3\})$ be the minimum order of a supertree T of the set $\{T_1(u_1), T_2(u_2), T_3(u_3)\}$ rooted in some vertex s such that T contains a copy of T_i in which s corresponds to u_i for every $i \in [3]$.
- Let $f(\{u_1, u_2\})$ be the minimum order of a supertree T of $\{T_1(u_1), T_2(u_2)\}$ rooted in some vertex s such that T contains a copy of T_i in which s corresponds to u_i for every $i \in [2]$. Define $f(\{u_1, u_3\})$ and $f(\{u_2, u_3\})$ symmetrically. By Proposition 3, $f(\{u_1, u_2\})$, $f(\{u_1, u_3\})$, and $f(\{u_2, u_3\})$ can be determined efficiently.

Note that $f(\{r_1, r_2, r_3\})$ is the minimum order of a supertree T of $\{T_1, T_2, T_3\}$ rooted in some vertex r that contains a copy of T_i in which r corresponds to r_i for every $i \in [3]$. Since we consider all $O(n^3)$ choices for the r_i, the smallest such tree is a minimum supertree of type 2.

In order to complete the proof, we explain how to determine the values $f(\{u_1, u_2, u_3\})$ by dynamic programming in polynomial time. If u_1 is an endvertex of T_1, then $f(\{u_1, u_2, u_3\}) = f(\{u_2, u_3\})$; similarly, if u_2 or u_3 are endvertices. Hence, we may assume that u_1, u_2, and u_3 are not endvertices. Let U_i be the set of children of u_i in T_i. The definitions imply that $f(\{u_1, u_2, u_3\})$ is the minimum

of

$$f(\mathcal{P}) = 1 + \sum_{j=1}^{k} f(e_j)$$

over all partitions $\mathcal{P} = \{e_1, \ldots, e_k\}$ of $U_1 \cup U_2 \cup U_3$ into sets e_j with $|e_j \cap U_i| \leq 1$ for every $i \in [3]$ and $j \in [k]$. Since $|U_1 \cup U_2 \cup U_3| \leq 3\Delta$, there are finitely many such partitions; since $k \leq 3\Delta$, a trivial upper bound on the number of these partitions is $(3\Delta)^{3\Delta}$. It follows that the values $f(\cdot)$ (together with suitable realizers) can be determined efficiently by dynamic programming, which completes the proof. ■

By an inductive argument also considering type 1 and type 2 supertrees and using the Helly property, Theorem 5 easily generalizes to MINIMUM SUPERTREE for given sets of k trees with the polynomial bounding the running time depending on k.

Theorem 6. Greedy Supertree *is a* $\left(\frac{k}{2} - \frac{1}{2} + \frac{1}{k}\right)$*-approximation algorithm with polynomial running time for* MINIMUM SUPERFOREST *restricted to instances* $\{T_1, \ldots, T_k\}$ *consisting of* k *trees.*

Proof. Let $\mathcal{T} = \{T_1, \ldots, T_k\}$ be the given set of k trees. By Proposition 3, for two given trees T and T', some minimum supertree $s(T, T')$ of $\{T, T'\}$ can be found efficiently. The trees S_i determined by `Greedy Supertree` are of the form

$$S_i = s(\cdots s(s(s(T_i, T_{i+1}), T_{i+2}), T_{i+3}), \ldots, T_{i+k-1}) \text{ for } i \in [k],$$

where indices are identified modulo k. We show that returning the smallest of the S_i yields a $\left(\frac{k}{2} - \frac{1}{2} + \frac{1}{k}\right)$-approximation algorithm for MINIMUM SUPERFOREST on $\mathcal{T}$. Therefore, let T be a minimum superforest of $\mathcal{T}$. Let $n = n(T)$. For $i \in [k]$, let $n_i = n(T_i)$ and let $V_i \subseteq V(T)$ be such that $T[V_i] \simeq T_i$. For $ij \in \binom{[k]}{2}$, let $n_{ij} = |V_i \cap V_j|$. Clearly,

$$\begin{aligned} n_i + n_j - n_{ij} &\leq n \text{ for every } ij \in \tbinom{[k]}{2}, \text{ and} \\ \sum_{i \in [k]} n_i - \sum_{ij \in \binom{[k]}{2}} n_{ij} &\leq n. \end{aligned}$$

Adding these $\binom{k}{2} + 1$ inequalities yields

$$k \sum_{i \in [k]} n_i - 2 \sum_{ij \in \binom{[k]}{2}} n_{ij} \leq \left(\binom{k}{2} + 1\right) n, \tag{2}$$

and, hence,

$$\frac{1}{k}\sum_{j\in[k]}\left(\sum_{i\in[k]} n_i - \sum_{i\in[k]\setminus\{j\}} n_{ij}\right) = \sum_{i\in[k]} n_i - \frac{2}{k}\sum_{ij\in\binom{[k]}{2}} n_{ij}$$

$$\overset{(2)}{\leq} \frac{1}{k}\left(\binom{k}{2}+1\right)n = \left(\frac{k}{2}-\frac{1}{2}+\frac{1}{k}\right)n. \tag{3}$$

Since $n_{i(i+1)}$ is the order of some possibly not largest subtree of $\{T_i, T_{i+1}\}$, we have $n(s(T_i, T_{i+1})) \overset{(1)}{\leq} n_i + n_{i+1} - n_{i(i+1)}$. Since $n_{i(i+2)}$ is the order of some subtree of $\{T_i, T_{i+2}\}$ and the tree $s(T_i, T_{i+1})$ contains a copy of T_i, we have $n(s(s(T_i, T_{i+1}), T_{i+2})) \leq n(s(T_i, T_{i+1})) + n_{i+2} - n_{i(i+2)} \leq n_i + n_{i+1} - n_{i(i+1)} + n_{i+2} - n_{i(i+2)}$. Using that n_{ij} is the order of some subtree of $\{T_i, T_j\}$ and that each tree of the form $s(\ldots s(s(T_i, T_{i+1}), T_{i+2}), \ldots, T_{i+\ell})$ for some ℓ contains a copy of T_i, it now follows inductively that

$$n(S_j) \leq \sum_{i\in[k]} n_i - \sum_{i\in[k]\setminus\{j\}} n_{ij}. \tag{4}$$

Altogether, we obtain

$$\min\{n(S_i) : i \in [k]\} \leq \frac{1}{k}\sum_{j\in[k]} n(S_j) \overset{(4)}{\leq} \frac{1}{k}\sum_{j\in[k]}\left(\sum_{i\in[k]} n_i - \sum_{i\in[k]\setminus\{j\}} n_{ij}\right)$$

$$\overset{(3)}{\leq} \left(\frac{k}{2}-\frac{1}{2}+\frac{1}{k}\right)n,$$

which completes the proof. ∎

The analysis of `Greedy Supertree` is essentially best possible. We give an example for $k = 3$ showing that the factor $4/3$ cannot be improved. For non-negative integers $n_1, \ldots, n_p$, let the tree $T(n_1, \ldots, n_p)$ arise from a path $P : u_1 \cdots u_p$ of order p by attaching, for every $i \in [p]$, exactly n_i new endvertices to u_i. For positive integers a, b, and c with $a > b > c \geq 1$, consider the three trees

$$T_1 = T(0, b, a, a, c, 0),$$
$$T_2 = T(0, b, 0, a, 0, 0, 0, a, 0), \text{ and}$$
$$T_3 = T(0, b, 0, 0, 0, 0, 0, a, 0, c, a, 0)$$

illustrated in Figure 4.

It is easy to verify that

- $s(T_1, T_2) \simeq T(0, b, a, a, c, 0, 0, a, 0)$ does not contain $T(0, a, 0, 0, a, 0)$,

- $s(T_1,T_3) \in \{T(0,b,0,0,0,0,b,a,a,c,a,0), T(0,b,0,0,0,0,c,a,a,b,a,0)\}$ and does not contain $T(0,a,0,0,0,a,0)$, and
- $s(T_2,T_3) \simeq T(0,b,0,a,0,0,0,a,0,c,a,0)$ does not contain $T(0,a,a,0)$.

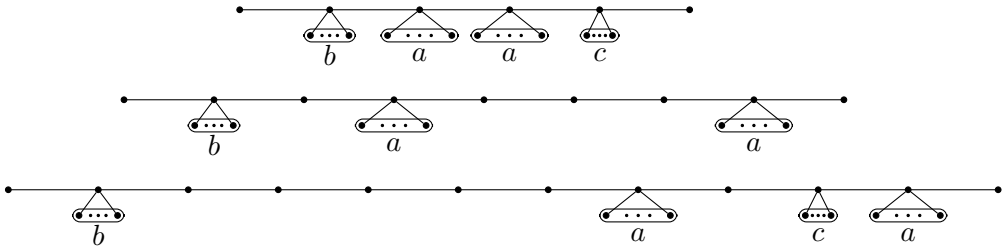

Figure 4. Three trees T_1, T_2, and T_3.

It follows that all three trees

$$s(s(T_1,T_2),T_3),\ s(s(T_1,T_3),T_2), \text{ and } s(s(T_2,T_3),T_1)$$

have order at least $4a$, while the tree

$$T(0,b,0,0,b,b,a,a,c,c,a,0)$$

of order $3a+3b+2c+12$ shown in Figure 5 contains T_1, T_2, and T_3.

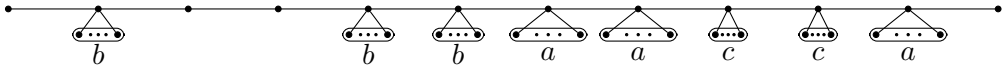

Figure 5. A supertree for $\{T_1,T_2,T_3\}$.

Choosing a large shows that the factor $4/3$ cannot be improved. Note that every supertree of $\{T_1,T_2\}$ that contains $T(0,a,0,0,a,0)$ and can therefore accomodate T_3 in a more efficient way has at least b vertices more than $s(T_1,T_2)$. Choosing a, b, and c such that $4\epsilon a \geq b > \epsilon n(s(T_1,T_2)) = \epsilon(3a+b+c+9)$ shows that the factor $4/3$ can only be improved marginally if the subroutine for $s(\cdot,\cdot)$ is allowed to return slightly suboptimal trees.

For the proof of Theorem 7, we need an auxiliary statement.

Let $\Delta \in \mathbb{N}$. Let $\mathcal{T}_\Delta = \{T_1,\ldots,T_q\}$ be the set of all trees of order at most Δ. Let $\mathcal{F}_\Delta$ be the set of all forests whose components belong to $\mathcal{T}_\Delta$. For a forest F, let $t(F) = (t_1,\ldots,t_q) \in [n(F)]_0^q$ be such that t_i is the number of components of F that are isomorphic to T_i for every $i \in [q]$ and let

$$\hat{t}(F) = \big\{t(F') : F' \text{ is an induced subforest of } F\big\}.$$

For every $t = (t_1, \ldots, t_q)$ from $\hat{t}(F)$, an induced subforest F' of F with $t(F') = t$ is a *realizer of t within F*. Note that $t(F)$ counts only small components of F but that F may have large components. Note furthermore, that $\hat{t}(F) \subseteq [n(F)]_0^q$.

For two sets $A, B \in \mathbb{N}_0^q$, let $A \oplus B = \{a + b : a \in A \text{ and } b \in B\}$.

Lemma 8. *For every $\Delta \in \mathbb{N}$, there is some $p \in \mathbb{N}$ with the following property. For every forest F of order at most n, $\hat{t}(F)$ as well as realizers within F can be determined in time $O(n^p)$.*

Proof. If F has components $F_1, \ldots, F_k$, then $\hat{t}(F) = \bigoplus_{i=1}^{k} \hat{t}(F_i)$. Since $\hat{t}(F_i) \subseteq [n(F_i)]_0^q$ and $\oplus$ is associative, in order to show the desired statement, we may assume that F is a tree. Root of F is some vertex r. Let $\mathcal{R}$ be the set of all pairs (S, s) such that $S \in \mathcal{T}_\Delta$ and $s \in V(S)$, that is, $\mathcal{R}$ captures all possible ways of selecting root vertices for the trees in $\mathcal{T}_\Delta$. Let u be some vertex of F. Let F_u be the subtree of F rooted in u that contains u and all its descendants.

For every $(S, s) \in \mathcal{R}$, let $\hat{t}_{(S,s)}(F_u)$ be the set of all $(t_1, \ldots, t_q) \in [n(F_u)]_0^q$ such that

- F_u has an induced subforest K that consists of t_i disjoint copies of T_i for every $i \in [q]$,
- the vertex u is contained in some component L of K that is isomorphic to S, and
- some isomorphism π between S and L maps s to u.
  See Figure 6 for an illustration.

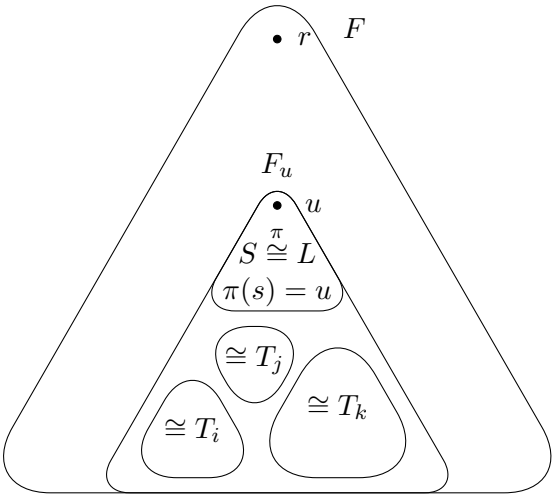

Figure 6. The structure of induced subgraphs K of F_u contributing to $\hat{t}_{(S,s)}(F_u)$.

Note that $\hat{t}_{(S,s)}(F_u)$ is empty, if F_u does not contain a suitable copy of S.

Similarly, let $\hat{t}_{\emptyset}(F_u)$ be the set of all $(t_1,\ldots,t_q) \in [n(F_u)]_0^q$ such that

- F_u has an induced subforest K that consists of t_i disjoint copies of T_i for every $i \in [q]$ and
- the vertex u does not belong to K.

Clearly,

$$\hat{t}(F_u) = \hat{t}_{\emptyset}(F_u) \cup \bigcup_{(S,s)\in\mathcal{R}} \hat{t}_{(S,s)}(F_u).$$

Let u have the children $v_1,\ldots,v_d$ in F.

We have

$$\hat{t}_{\emptyset}(F_u) = \hat{t}(F_{v_1}) \oplus \hat{t}(F_{v_2}) \oplus \cdots \oplus \hat{t}(F_{v_d}).$$

Now, let $(S,s) \in \mathcal{R}$. Let $s_1,\ldots,s_{d'}$ be the neighbors of s in S and let S_i be the component of $S-s$ containing s_i. Since S has order at most Δ, we have $d' < \Delta$.

Furthermore, we have

$$\hat{t}_{(S,s)}(F_u) = \bigcup_{f:[d'] \xrightarrow{injective} [d]} \left(\bigoplus_{i\in[d']} \hat{t}_{(S_i,s_i)}(F_{v_{f(i)}}) \ \oplus \bigoplus_{i\in[d]\setminus f([d'])} \hat{t}_{\emptyset}(F_{v_i}) \right),$$

where the $O(d^{\Delta})$ injective functions f capture the different ways of associating the neighbors of s in S with the children of u in F. See Figure 7 for an illustration.

Using these formulas, a simple dynamic programming approach allows to determine $\hat{t}(F)$ as well as suitable realizers within F in time $O(n^p)$. ∎

Theorem 7. *For every $\epsilon > 0$, there is some $p \in \mathbb{N}$ with the following property. For a given set $\mathcal{F} = \{F_1,\ldots,F_k\}$ consisting of k forests of order at most n, one can determine in time $O(kn^p)$ a subforest F of $\mathcal{F}$ with $n(F) \geq (1-\epsilon)n(F_{\rm opt})$, where $F_{\rm opt}$ is some maximum subforest of $\mathcal{F}$.*

Proof. Let $\epsilon > 0$ be fixed. Let $\mathcal{F} = \{F_1,\ldots,F_k\}$ be a given set of k forests of order at most n. For $i \in [k]$, let $n_i = n(F_i)$, and let $n_1 = \min\{n_1,\ldots,n_k\}$. Let $F_{\rm opt}$ be some maximum subforest of $\mathcal{F}$. Since each forest F_i in $\mathcal{F}$ has an independent set of order at least $n_i/2 \geq n_1/2$, we have $n(F_{\rm opt}) \geq n_1/2$.

Let $\Delta = \lceil \frac{2}{\epsilon} \rceil$. Let $\mathcal{F}_{\Delta}$ be as above, that is, $\mathcal{F}_{\Delta}$ is the set of all forests whose components all have order at most Δ. Rooting each component of F_1 in some vertex and iteratively removing vertices u of maximum depth for which u has at least Δ descendants, yields a set X of at most $n_1/\Delta \leq 2n(F_{\rm opt})/\Delta$ vertices of F_1 such that $F_1' = F_1 - X$ belongs to $\mathcal{F}_{\Delta}$. Let $F_{\rm opt}'$ be a maximum subforest

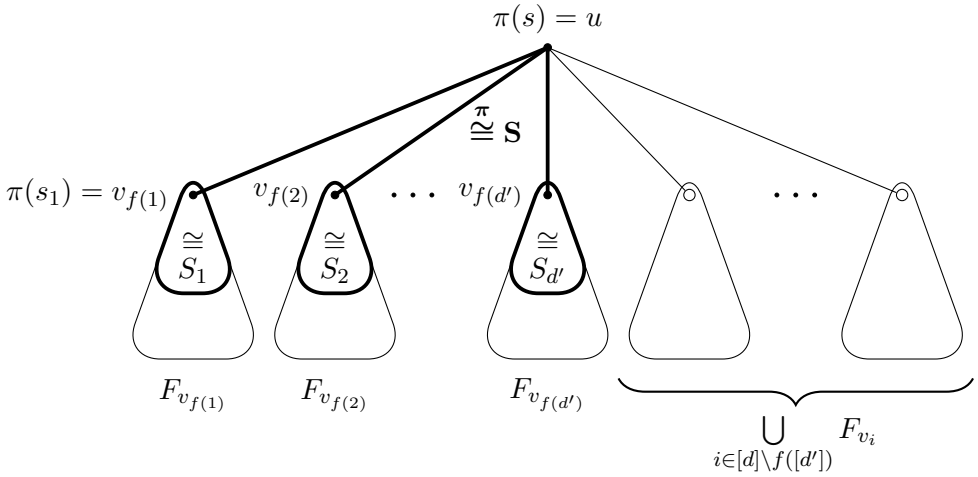

Figure 7. Embedding S (as well as the rest of K) into F_u mapping the root s of S to u and the children s_i of s in S to children $v_{f(i)}$ of u as selected by f.

of $(\mathcal{F} \setminus \{F_1\}) \cup \{F_1'\}$. Clearly, F'_{opt} is a subforest of $\mathcal{F}$ that belongs to $\mathcal{F}_\Delta$ and satisfies

$$n(F'_{\text{opt}}) \geq n(F_{\text{opt}}) - |X| \geq \left(1 - \frac{2}{\Delta}\right) n(F_{\text{opt}}) \geq (1 - \epsilon) n(F_{\text{opt}}).$$

Therefore, in order to complete the proof, it suffices to show that a subforest of $\mathcal{F}$ that belongs to $\mathcal{F}_\Delta$ and has maximum possible order subject to this condition, can be found efficiently.

By Lemma 8, we can determine $\hat{t}(F_i)$ as well as suitable realizers within F in time $O(n^{p_1})$ for every $i \in [k]$. Since

$$\begin{aligned} &\max\{n(F) : F \in \mathcal{F}_\Delta \text{ is a subforest of } \mathcal{F}\} \\ &= \max\left\{\sum_{i=1}^{q} t_i n(T_i) : (t_1, \ldots, t_q) \in \bigcap_{i=1}^{k} \hat{t}(F_i)\right\}, \end{aligned}$$

the desired statement follows. ∎

It seems interesting to study tradeoffs between supergraphs that are required to belong to different graph classes. For a set $\mathcal{F}$ of trees, for instance, a supergraph of minimum order may be much smaller than a minimum supertree. Indeed, if $\mathcal{F} = \{T(a, 0, a), T(a, 0, 0, a), \ldots, T(a, \underbrace{0, \ldots, 0}_{k \text{ times}}, a)\}$ for positive integers a and k at least 3, then suitably identifying vertices of degree $a + 1$ yields a supergraph of

$\mathcal{F}$ of order $2+2a+1+2+\cdots+k = 2a+\binom{k+1}{2}+2$, while every supertree of $\mathcal{F}$ has order $\Omega\left(\sqrt{k}a\right)$.

As stated at the very beginning, all considered subgraphs were induced. Several of our results can be adapted by small yet careful changes to non-induced subgraphs.

References

[1] A. Abboud, A. Backurs, T.D. Hansen, V.V. Williams and O. Zamir, *Subtree isomorphism revisited*, ACM Trans. Algorithms **14** (2018) 27.
https://doi.org/10.1145/3093239

[2] F.N. Abu-Khzam, *Maximum common induced subgraph parameterized by vertex cover*, Inform. Process. Lett. **114** (2014) 99–103.
https://doi.org/10.1016/j.ipl.2013.11.007

[3] F.N. Abu-Khzam, É. Bonnet and F. Sikora, *On the complexity of various parameterizations of common induced subgraph isomorphism*, Theoret. Comput. Sci. **697** (2017) 69–78.
https://doi.org/10.1016/j.tcs.2017.07.010

[4] T. Akutsu, *An RNC algorithm for finding a largest common subtree of two trees*, IEICE Trans. Inf. Syst. **E75-D** (1992) 95–101.

[5] T. Akutsu, A.A. Melkman and T. Tamura, *Improved hardness of maximum common subgraph problems on labeled graphs of bounded treewidth and bounded degree*, Internat. J. Found. Comput. Sci. **31** (2020) 253–273.
https://doi.org/10.1142/S0129054120500069

[6] T. Akutsu and T. Tamura, *On the complexity of the maximum common subgraph problem for partial k-trees of bounded degree*, in: Algorithms and Computation ISAAC 2012, K.M. Chao, Ts.Hsu and D.T.Lee (Ed(s)), Lect. Notes in Comput. Sci. **7676** (Springer, Berlin, Heidelberg, 2012) 146–155.
https://doi.org/10.1007/978-3-642-35261-4_18

[7] H.L. Bodlaender, T. Hanaka, Y. Kobayashi, Y. Kobayashi, Y. Okamoto, Y. Otachi and T.C. van der Zanden, *Subgraph isomorphism on graph classes that exclude a substructure*, Algorithmica **82** (2020) 3566–3587.
https://doi.org/10.1007/s00453-020-00737-z

[8] S.H. Bokhari, *On the mapping problem*, IEEE Trans. Comput. **30** (1981) 207–214.
https://doi.org/10.1109/TC.1981.1675756

[9] M. Cygan, *Improved approximation for 3-dimensional matching via bounded pathwidth local search*, in: 2013 IEEE 54th Annual Symposium on Foundations of Computer Science (2013) 509–518.
https://doi.org/10.1109/FOCS.2013.61

[10] M.R. Garey and D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness (Freeman, San Francisco, 1979).
https://doi.org/10.1137/1024022

[11] M. Grohe, G. Rattan and G.J. Woeginger, *Graph similarity and approximate isomorphism*, in: 43rd International Symposium on Mathematical Foundations of Computer Science (MFCS 2018), Leibniz International Proceedings in Informatics (LIPIcs) **117** (Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018) 20. https://doi.org/10.4230/LIPIcs.MFCS.2018.20

[12] P. Heggernes, P. van 't Hof, D. Meister and Y. Villanger, *Induced subgraph isomorphism on proper interval and bipartite permutation graphs*, Theoret. Comput. Sci. **562** (2015) 252–269. https://doi.org/10.1016/j.tcs.2014.10.002

[13] C.A.J. Hurkens and A. Schrijver, *On the size of systems of sets every t of which have an SDR, with an application to the worst-case ratio of heuristics for packing problems*, SIAM J. Discrete Math. **2** (1989) 68–72. https://doi.org/10.1137/0402008

[14] N. Kriege, F. Kurpicz and P. Mutzel, *On maximum common subgraph problems in series-parallel graphs*, European J. Combin. **68** (2018) 79–95. https://doi.org/10.1016/j.ejc.2017.07.012

[15] D.W. Matula, *Subtree isomorphism in* $O(n^{5/2})$, Ann. Discrete Math. **2** (1978) 91–106. https://doi.org/10.1016/S0167-5060(08)70324-8

[16] D. Marx and M. Pilipczuk, *Everything you always wanted to know about the parameterized complexity of subgraph isomorphism* (*but were afraid to ask*), in: 31st International Symposium on Theoretical Aspects of Computer Science (STACS 2014), LIPIcs. Leibniz Int. Proc. Inform. **25** (Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2014) 542–553. https://doi.org/10.4230/LIPIcs.STACS.2014.542

[17] J.W. Raymond and P. Willett, *Maximum common subgraph isomorphism algorithms for the matching of chemical structures*, J. Comput. Aided Mol. Des. **16** (2002) 521–533. https://doi.org/10.1023/A:1021271615909

[18] K. Shearer, H. Bunke and S. Venkatesh, *Video indexing and similarity retrieval by largest common subgraph detection using decision trees*, Pattern Recognition **34** (2001) 1075–1091. https://doi.org/10.1016/S0031-3203(00)00048-0

[19] A. Yamaguchi, K.F Aoki and H. Mamitsuka, *Finding the maximum common subgraph of a partial k-tree and a graph with a polynomially bounded number of spanning trees*, Inform. Process. Lett. **92** (2004) 57–63. https://doi.org/10.1016/j.ipl.2004.06.019

Received 12 August 2024
Revised 1 September 2025
Accepted 1 September 2025
Available online 27 October 2025