On Maximum Symmetric Subgraphs

Ho-Lin Chen[1], Hsueh-I. Lu[2], and Hsu-Chun Yen[1]

[1] Department of Electrical Engineering, National Taiwan University, Taipei 106, Taiwan, R.O.C. `dnbcom@ms4.hinet.net, yen@cc.ee.ntu.edu.tw`.
[2] Institute of Information Science, Academia Sinica, Taipei 115, Taiwan, R.O.C. `hil@iis.sinica.edu.tw`.

Abstract. Let G be an n-node graph. We address the problem of computing a maximum symmetric graph H from G by deleting nodes, deleting edges, and contracting edges. This NP-complete problem arises naturally from the objective of drawing G as symmetrically as possible. We show that its tractability for the special cases of G being a plane graph, an ordered tree, and an unordered tree, depends on the type of operations used to obtain H from G. Moreover, we give an $O(\log n)$-approximation algorithm for the intractable case that H is obtained from a tree G by contracting edges. As a by-product, we give an $O(\log n)$-approximation algorithm for an NP-complete edit-distance problem.

1 Introduction

As graphs are known to be one of the most important abstract models in various scientific and engineering areas, *graph drawing* has naturally emerged as a fast growing research topic in computer science. Among various aesthetics investigated in the literature, *symmetry* has received much attention recently [2,3,4,5,6,9,10,16,15,17]. As a symmetric graph can be decomposed into a number of isomorphic subgraphs, only a portion of the graph, together with the symmetry information, is sufficient to define the original graph. In this way, symmetric graphs can often be represented in a more succinct fashion than their asymmetric counterparts. In reality, however, we often have to lower our expectations by allowing imperfection while considering the so-called 'nearly symmetric' drawings of graphs. To draw a graph in a nearly symmetric fashion, a good starting point might be to draw its symmetric subgraph as large as possible first, and then add the remaining nodes and edges to the drawing. Like many of the graph drawing problems, determining whether a graph has an axial or rotational symmetry is computationally intractable [15]. The maximum symmetric subtrees (i.e., subtrees that exhibit symmetric drawings), however, can be computed in polynomial time [4]. For series-parallel graphs, algorithms that display as much symmetry as possible can be found in [10]. Aside from the above algorithmic aspects of symmetric drawings, in [6] several types of symmetries, including axial symmetries and rotational symmetries, have been characterized in a unified way using geometric automorphism groups.

To formulate the notion of *near symmetry*, in this paper we propose a quantitative measure to capture the *degree of symmetry* in drawing graphs, and then

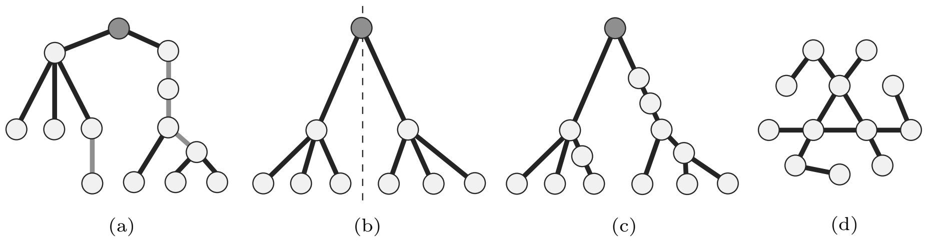

Fig. 1. (a) An asymmetric graph G rooted at the gray node. (b) A maximum axially symmetric graph H obtained from G by contracting edges. (c) A nearly symmetric drawing of G. (d) A 3-rotational symmetric graph.

investigate the complexity of computing such a measure for ordered trees, unordered trees, and plane graphs. An $O(\log n)$-approximation algorithm is given for an NP-complete case for unordered trees. Given a graph G, our symmetry measure of G is the maximum number of edges in a symmetric graph H that is obtained from G by applying a sequence of edge contractions, node deletions, or edge deletions. For example, the asymmetric graph G in Figure 1(a) can be turned into a symmetric graph H in Figure 1(b) by contracting the gray edges. One can verify that H is a maximum axially symmetric graph obtainable from G by contracting edges. Therefore, the degree of axial symmetry for G is 8. Visually, the graph in Figure 1(a) admits a nearly symmetric drawing as displayed in Figure 1(c). In light of the above, our symmetry measure can be thought of as a quantitative way of defining the notion of *near symmetry* in graph drawings. Graphs with higher degrees of symmetry have the tendency to exhibit better symmetric appearances visually. For more about nearly symmetric drawings, the reader is referred to [9].

As stated above, our main concern is to turn a graph into a symmetric graph with maximum number of edges through edge contractions, edge deletions, and node deletions. If the resulting graph is axially (respectively, rotationally) symmetric, we call the problem DAS (respectively, DRS), standing for degree of axial (respectively, rotational) symmetry. The tractability results are summarized in Tables 1 and 2. By allowing nodes to be drawn sufficiently close to each other (see Figure 1(c)), edge contractions, node deletions, and edge deletions seem to be rather natural to define near symmetry. In our setting, if a graph cannot be turned into a symmetric one, then the degree of symmetry is zero.

Our DAS problems are related to GRAPH ISOMORPHISM problems. Such problems include TREE INCLUSION [12] and EDIT DISTANCE [20], which have applications to analyzing molecular structures in biology. Given two labelled trees A and B, TREE INCLUSION is to determine whether A can be obtained from B by 'contracting' nodes, whereas EDIT DISTANCE is to determine the minimum number of 'changes,' 'contracting' (or its dual) needed to transform A into B. The main disparity between our DAS problem and those related to graph isomorphism is that the latter deal with two or more graphs from which some common substructures are extracted, whereas in the case of DAS, a single graph

Table 1. The tractability for maximum axially symmetric subgraph.

		node deletion	edge deletion	edge contraction
tree	ordered	P [Theorem 1]	P [4]	P [Theorem 1]
	unordered	P [Theorem 3]	P [4]	NPC [Theorem 2]
graph	plane	P [Theorem 4]	NPC [Theorem 5]	NPC [Theorem 5]
	general	NPC [15]	NPC [15]	NPC [15]

Table 2. The tractability for maximum rotationally symmetric subgraph with respect to unordered trees.

	node deletion	edge deletion	edge contraction
$O(1)$-degree trees	P [Theorem 8]	P [1]	Do not preserve degree bound
general trees	NPC [Theorem 9]	NPC [Theorem 9]	NPC [Theorem 9]

is considered. (Although, as we shall see later, a number of techniques commonly seen in the graph isomorphism research turn out to play a constructive role in our design and analysis.) As for trees, our DAS problem also differs from the TREE INCLUSION and EDIT DISTANCE in the following. First, our DAS problem is defined over graphs (while treating trees, both ordered and unordered, as a special case), as opposed to TREE INCLUSION and EDIT DISTANCE which are explicitly targeted for trees only. Second, instead of given two 'labelled' trees as inputs in the TREE INCLUSION and EDIT DISTANCE cases, in our symmetric drawing setting the input consists of a single graph (or tree). In addition, our graphs are not labelled. Finally, in addition to *contractions*, we also consider node deletions and edge deletions. An interesting by-product of our work is an $O(\log n)$-approximation algorithm for an NP-complete problem related to EDIT DISTANCE problems.

The rest of the paper is organized as follows. Section 2 defines the degree-of-symmetry problems. Section 3 studies the tractability of the problems for axial symmetry, and gives two approximation algorithms, one for a symmetry problem, and the other for a problem related to edit distance. Section 4 studies the tractability of the problems for rotational symmetry with respect to unordered trees. Section 5 concludes the paper with some future research directions.

2 Preliminaries

Let G be a graph. Let $|G|$ denote the number of nodes in G. A *drawing* of G on the plane is a mapping D from the nodes of G to $\mathfrak{R}^2$, where $\mathfrak{R}$ is the set of real numbers. That is, each node v is placed at point $D(v)$ on the plane, and each edge (u, v) is displayed as a line segment connecting $D(u)$ and $D(v)$. A graph G has an *axial symmetry* if there exists a drawing D under which the image of G is symmetric with respect to a straight line on the plane. G has a *k-rotational symmetry* if there exists a drawing D such that D is unchanged if

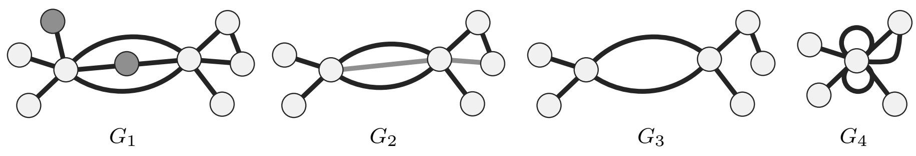

Fig. 2. Four graphs for illustrating the operations of deleting nodes, deleting edges, and contracting edges.

the plane is rotated at some point by $360/k$ degrees. For example, the drawing shown in Figure 1(b) has an axial symmetry; and the drawing in Figure 1(d) has a 3-rotational symmetry. Axial symmetry and rotational symmetry are two special kinds of *geometric automorphisms* defined by Hong, Eades, and Lee [8]. The reader is referred to [6,8,9,15] for more about symmetry in graph drawing.

For a connected G, the following basic graph operations are used throughout the paper.

– *Node deletion.* Only nodes of degree no more than two can be deleted from G. For any degree-one node w of G, deleting w is the operation of removing w and its incident edge from G. If the degree of w is two, where u and v are its neighbor in G, then deleting w is the operation of deleting w, (u, w), and (v, w) from G, and then adding a copy of (u, v) to G.
– *Edge deletion.* The operation can only be applied to an edge (u, v) whose removal does not disconnect G, or one of u and v is of degree one. Deleting (u, v) is the operation of removing edge (u, v) (or exactly one copy of the parallel edges incident to u and v). In case a degree-one node is involved, the node is removed as well.
– *Edge contraction.* For any edge (u, v) of G, contracting (u, v) is the operation of deleting the edge incident to u and v, and then merging u and v into a single node. Note that if G has m parallel edges incident to u and v, then all but one of those m parallel edges become self-loops incident to the merged node in the resulting graph.

Examples are shown in Figure 2: G_2 is obtained from G_1 by deleting the gray nodes; G_3 is obtained from G_2 by deleting the gray edges; and G_4 is obtained from G_2 by contracting the gray edges. Given graphs G and H, we write $G \overset{\text{ec}}{\to} H$, $G \overset{\text{nd}}{\to} H$, and $G \overset{\text{ed}}{\to} H$ to signify that H can be obtained from G through a sequence of edge contractions, node deletions, and edge deletions, respectively. Clearly, a node deletion can be viewed as an edge contraction. Hence, $G \overset{\text{nd}}{\to} H$ implies $G \overset{\text{ec}}{\to} H$. One can easily see that if H consists of a single node and no edges, then $G \overset{\text{ec}}{\to} H$ and $G \overset{\text{ed}}{\to} H$ hold for any nonempty G. However, $G \overset{\text{nd}}{\to} H$ does not necessarily hold, since parallel edges cannot be removed through node deletions.

Given a graph G, the *degree of axial* (respectively, *rotational*) *symmetry* of G is defined to be the size of a maximum axially (respectively, rotationally) symmetric graph H that can be derived from G through basic graph operations. In this paper, we investigate the tractability of determining the degree of axial symmetry (DAS) and the degree of rotational symmetry (DRS). Let DAS_{ec}, DAS_{nd}, and DAS_{ed} denote the DAS problem with respect to edge contraction, node deletion, and edge deletion, respectively. Let DRS_{ec}, DRS_{nd}, and DRS_{ed} be defined similarly. For any graphs G_1 and G_2, let $\text{dist}_{\text{ec}}(G_1, G_2)$, $\text{dist}_{\text{ed}}(G_1, G_2)$, and $\text{dist}_{\text{nd}}(G_1, G_2)$ denote the minimum number of edge contractions, edge deletions, and node deletions required to transform G_1 and G_2 into two identical graphs, respectively.

Fact 1 (see [15]) *For each operation* $\text{op} \in \{\text{ec}, \text{ed}, \text{nd}\}$*,* DAS_{op} *and* DRS_{op} *for graphs are NP-complete.*

3 Axial Symmetry

3.1 Ordered Trees

We begin by considering the DAS problems for ordered and unordered trees. Unless stated otherwise, trees are assumed to be rooted for the rest of the paper. For any ordered forest F, let reflection(F) be the ordered forest obtained from F by (a) reversing the order of the trees in F, and (b) reversing the order of the children of v for each node v in F. For any ordered tree T, let $T(i)$ be the subtree of T rooted at v_i, where $v_1, v_2, \ldots, v_n$ is the postordering of T.

Fact 2 (see [20,13]) *For any ordered forests* F_1 *and* F_2*,* $\text{dist}_{\text{ec}}(F_1, F_2)$ *can be computed in polynomial time.*

Theorem 1. DAS_{ec} *and* DAS_{nd} *for ordered trees can be solved in polynomial time.*

Proof. Clearly, one of the following two cases holds for any axially symmetric tree T.

- *Case 1*: The root of T has $2k + 1$ children for some integer k. Let v_i be the $(k + 1)$-st child of the root of T. Let F_1 be the subforest of T induced by the nodes v_j with $j < \ell$, where v_ℓ is the leftmost leaf node of $T(i)$. Let F_2 be the subforest of T induced by the nodes v_j with $i < j < n$. Clearly, F_1 is identical to reflection(F_2).
- *Case 2*: The root of T has $2k$ children for some integer k. If $k \geq 1$, then let v_i be the k-th child of the root of T. Let F_1 be the subforest of T induced by the nodes v_j with $1 \leq j \leq i$. Let F_2 be the subforest of T induced by the nodes v_j with $i < j < n$. Clearly, F_1 is identical to reflection(F_2).

We prove the statement for DAS_{ec} by giving a dynamic-programming algorithm. Let T be the given ordered tree, where $v_1, v_2, \ldots, v_{|T|}$ is the postordering of T. Clearly, $v_{|T|}$ is the root of T. By the above observations for any axially symmetric ordered tree, one can verify that the algorithm shown in Figure 3 correctly computes the minimum number of edge contractions required to turn T into an axially symmetric ordered tree. An example that illustrates number-of-contraction for v_7 is shown in Figure 4. The correctness of

```
function number-of-contraction($T$) {
  for $i = 1$ to $|T| - 1$ do {
    let $A_i$ be the set of ancestors of $v_i$ in $T_i$;
    let $F_1$ be the subforest of $T$ induced by $\{v_1, v_2, \ldots, v_i\}$;
    let $F_2$ be the subforest of $T$ induced by $\{v_{i+1}, v_{i+2}, \ldots, v_{|T|}\} - A_i$;
    let $F_3$ be obtained by removing $T(i)$ from $F_1$;
    let $c_i = \min\{\mathrm{dist}_{\mathrm{ec}}(F_1, \mathrm{reflection}(F_2)),$
                 $\mathrm{dist}_{\mathrm{ec}}(F_3, \mathrm{reflection}(F_2)) + \text{number-of-contraction}(T(i))\}$;
  }
  return $\min_{1 \le i \le |T|-1} |A_i| + c_i$;
}
```

Fig. 3. An algorithm for computing the minimum number of edge contractions required to turn T into an axially symmetric graph.

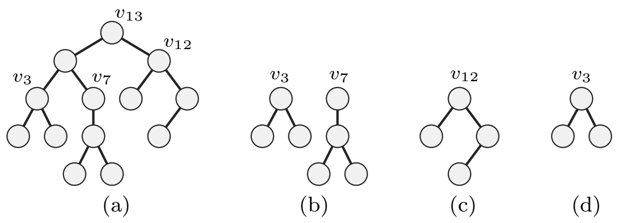

Fig. 4. (a) An ordered tree. (b) The F_1 for v_7. (c) The F_2 for v_7. (d) The F_3 for v_7.

number-of-contraction is immediate from the above two-case observation. By Fact 2, it is not difficult to see that number-of-contraction runs in polynomial time.

The statement for $\mathrm{DAS}_{\mathrm{nd}}$ can be proved similarly. □

3.2 Unordered Trees

Theorem 2. $\mathrm{DAS}_{\mathrm{ec}}$ *for unordered trees is NP-complete.*

Proof. By Fact 1, it suffices to show a reduction from the following variant of SATISFIABILITY, which remains NP-complete [12]: The input f is a set of m clauses $C_1, C_2, \ldots, C_m$ over variables $v_1, v_2, \ldots, v_n$, where each $\neg v_i$ appears in at most one clause of f. Let $\ell = (12m + 10)(m + \sum_{i=1}^{m} |C_i|) + n$, where $|C_i|$ is the number of literals in C_i. For each $1 \le k \le m$, let G_k be as shown in Figure 5(b). Clearly, the height of each G_i is $3m + 5$. The unordered tree instance T_f consists of two subtrees L and R, each begins with a long path of length ℓ. The length-ℓ path of L has subtrees $G_1, \ldots, G_m$. The length-ℓ path of R has subtrees $P_1, \ldots, P_n$, where P_i is defined as follows. If $\neg v_i$ does not appear in f, then P_i has a subtree G_j for each index j with $v_i \in C_j$. If $\neg v_i$ appears in a clause, say, C_r of f, then P_i is exactly G_r, whose leftmost leaf node has a subtree G_j for each index j with $v_i \in C_j$. An example of T_f is shown in Figure 5(a).

Now we prove that f is satisfiable if and only if $T_f \overset{\mathrm{ec}}{\to} T$ holds for some axially symmetric tree T with $|T| \ge 1 + 2|L|$. Suppose f is satisfiable by a truth

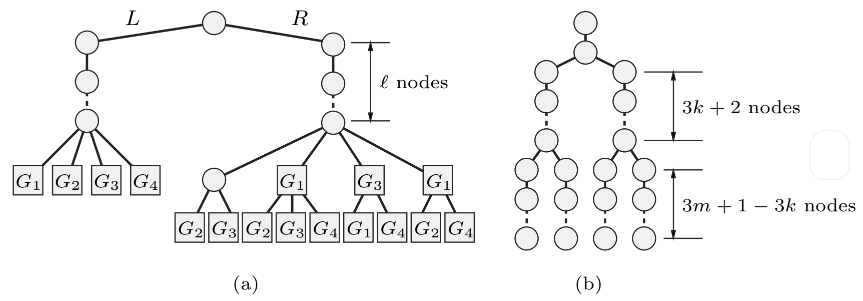

Fig. 5. (a) The T_f for $f = \{\neg v_2 \vee v_3 \vee \neg v_4,\ v_1 \vee v_2 \vee v_4,\ v_1 \vee v_2 \vee \neg v_3,\ v_2 \vee v_3 \vee v_4\}$. (b) The building block G_k.

assignment ϕ. We show how to obtain L by applying edge contractions to R. If $\phi(v_i)$ = false, then we remove all those G_j with $v_i \in C_j$ from P_i by edge contractions. If $\phi(v_i)$ = true, then we remove all but those G_j with $v_i \in C_j$ from P_i by edge contractions. Since each C_j, with $1 \leq j \leq m$, is satisfied by ϕ, it is clear that at least one copy of G_j remains in the resulting R. Thus, we can apply additional edge contractions on the resulting R to obtain L.

Now we assume that there is an axially symmetric tree T such that $T_f \overset{\text{ec}}{\to} T$ and $|T| \geq 1 + 2|L|$ hold. One can easily verify that L and the length-ℓ path in R must stay intact in T. In the symmetric drawing, if block G_i in L is mapped into block G_j in R, then $i = j$. Note that the depth of each leaf node in each G_i, with $1 \leq i \leq m$, is exactly $m + 6$. Therefore, if $i \neq j$, then $G_i \overset{\text{ec}}{\to} G_j$ does not hold. It follows that each G_i in the right subtree of T comes from a P_j of R. If P_j was composed of G_i whose leftmost leaf node has some subtrees, then let $\phi(v_i)$ = false. Otherwise, let $\phi(v_i)$ = true. Hence, ϕ indeed satisfies f. □

Theorem 3. DAS$_{\text{nd}}$ *for unordered trees can be solved in polynomial time.*

Proof. Let $v_1, v_2, \ldots, v_n$ be a postordering of the input tree T. If in the outcome of the conversion between $T(i)$ and $T(j)$, v_i and v_j remain to be the roots of $T(i)$ and $T(j)$, then the minimum number of node deletions required is written as $d(T(i), T(j))$. (Note that in the resulting symmetric mapping, the two roots need not be v_i and v_j.) Let das$(T(i))$ be the number of node deletions required to turn the $T(i)$ into a symmetric one.

For each i, let V_i consist of the children of v_i in T. We first show how to compute $d(T(i), T(j))$ in polynomial time. Construct a complete bipartite graph $B = (V_i \cup V_j, E)$ as follows: For each edge (v_p, v_q) with $v_p \in V_i$ and $v_q \in V_j$, let the edge weight $w(v_p, v_q)$ be $|T(p)| + |T(q)| - \text{dist}_{\text{nd}}(T(p), T(q))$, and compute the maximum matching of graph G. Since v_i and v_j must be mapped into each other for computing $d(T(i), T(j))$, and the numbers of children of v_i and v_j do not increase under node deletions, we know $d(T(i), T(j)) = |T(i)| + |T(j)| -$ matching(B), where matching(B) is the weight of the maximum matching of B. Since maximum matching can be computed in polynomial time [14], so can $d(T(i), T(j))$.

It remains to show that each $\text{das}(T(i))$ can be computed in polynomial time. Construct a weighted graph $G_i = (V_i, E_i)$. For each pair of nodes v_p and v_q in V_i, let the weight of edge (v_p, v_q) be $|T(p)| + |T(q)| - \text{dist}_{\text{nd}}(T(p), T(q))$. We then do the following:
Step 1. The number of operations required to let $T(i)$ become symmetric without any nodes on the symmetry axis is $|T(i)| - \text{matching}(G_i)$.
Step 2. Delete one node v_j of G_i, and then compute the maximum matching of the resulting graph. The number of operations required to let $T(j)$ become symmetric with v_j on the symmetry axis is $|T(j)| -$ the sum of edge weights found $-$ $\text{das}(T(j))$.
Step 3. Repeat Step 2 for each node in G.
Step 4. The minimum number of required operations is the minimum value found in Steps 1 and 3.

Clearly, the solution of a maximum matching contains edge (v_i, v_j) of weight w if and only if $T(i)$ and $T(j)$ can be made identical by deleting w nodes. Thus, the above procedure computes $\text{das}(T(i))$ correctly, proving the theorem. □

3.3 Plane Graphs

In this subsection we study the DAS problems for planarly embedded graphs.

Theorem 4. DAS_{nd} *for plane graphs can be solved in polynomial time.*

Proof. The proof is a slight extension of a result in [15]. First we apply node deletions to all the degree-1 and degree-2 nodes of G. Let the resulting graph be H. Hence, if after finite node deletion operations on G, we get a symmetric graph G_1, and suppose P is a geometric automorphism on G_1, then P must also be a geometric automorphism on H. Using the algorithm that finds automorphism for plane graphs [11], all the geometric automorphisms of H can be obtained in polynomial time. And for each automorphism P, we then compare the numbers of degree-1 and degree-2 nodes of G on the corresponding edges of P by Theorem 1. Thus, the theorem is proved. □

Theorem 5. DAS_{ed} *and* DAS_{ec} *for plane graphs are NP-complete.*

Proof. By Fact 1, it suffices to ensure the NP-hardness of DAS_{ed} and DAS_{ec} for plane graphs. The NP-hardness of DAS_{ed} can be obtained by a reduction from the NP-complete Hamiltonian cycle problem on a plane graph G [7]. For a given n-node m-edge plane graph G, we construct a plane graph H by connecting G, an $(m+2)$-node path P_{m+2}, and an n-node cycle C_n. One can verify that there exists an axially symmetric plane graph H' whose edge number is greater than or equal to $2n+m+3$ and $H \overset{\text{ed}}{\to} H'$ if and only if G admits a Hamiltonian cycle.

Now we prove the NP-hardness of DAS_{ec}. Let G^* be the dual graph of the input plane graph G with the node (and its incident edges) associated with the external face in G removed. Clearly, G admits a symmetric drawing if and only if G^* admits a symmetric drawing. Also, an edge contraction on G corresponds to an edge deletion on G^*. Therefore the NP-hardness of DAS_{ec} for G follows from that of DAS_{ed} for G^*. □

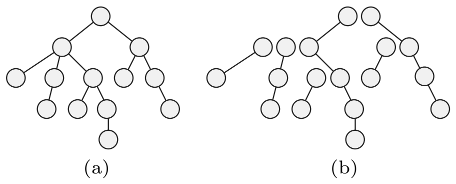

Fig. 6. (a) An unordered tree T. (b) A path decomposition of T.

3.4 Approximation Algorithms

As shown in Theorem 2, DAS$_{\text{ec}}$ for unordered trees is NP-complete. In the following, we give an approximation algorithm for DAS$_{\text{ec}}$. We shall see later that the technique of our approximation algorithm for DAS$_{\text{ec}}$ has an application to the problem of finding the maximum isomorphic subtrees (under contraction) between two unordered unlabeled trees. A related result in the literature is that the edit-distance problem for unordered labeled trees is MAX SNP-hard [19].

Let decomp(T, r) be the following procedure of decomposing a tree T with root r into k paths, where k is the number of leaves. The leaves are labeled from 1 to k such that a leaf with a smaller depth is assigned a smaller label. The k paths are constructed as follows. An edge $e = (v, w)$, where v is closer to r than w, belongs to path $P_i, 1 \leq i \leq k$, if i is the largest label among all the leaves of the subtree rooted at w. An example is shown in Figure 6. Let e be the base of the natural logarithm.

Theorem 6. *For any n-edge rooted tree T, there is a polynomial-time algorithm that outputs a symmetric tree T' such that $T \overset{\text{ec}}{\to} T'$ and T' contains at least $\frac{n}{e \ln n}$ edges.*

Proof. Let r be the root of T. We first perform decomp(T, r), and let $R = \{P_1, \ldots, P_k\}$, where k is the number of leaves of T, be the resulting set of paths. We then decompose R into $R_1, \ldots, R_{\lceil \ln n \rceil}$ such that any path in R_i has length between e^{i-1} and e^i, for each $1 \leq i \leq \lceil \ln n \rceil$. Then, find the set $R_j, 1 \leq j \leq \lceil \ln n \rceil$, with the maximum number of edges among all R_i. Now $\forall i, i \neq j$, contract all the edges in R_i and all the edges $e = (u, v)$ in R_j with the distance from v to the leaf of the path (in R) containing e greater than or equal to e^{j-1}. The resulting graph is a tree with paths of equal length $\lceil e^{j-1} \rceil$ directly attached to the root r, and hence, the graph is symmetric. For those paths in R_j, at least $1/e$ of the total number of edges of R_j is left after the mentioned contraction operations. (Recall that the length of each path in R_j has lower and upper bound of $\lceil e^{j-1} \rceil$ and $\lceil e^j \rceil$, respectively.) In view of the above, the total number of edges left in the resulting symmetric tree is at least $\frac{n}{e \ln n}$. $\square$

By Theorem 6, we are able to find an approximation algorithm of guaranteed performance for computing, given two unordered and unlabeled trees T_1 and T_2, the maximum tree T such that both $T_1 \overset{\text{ec}}{\to} T$ and $T_2 \overset{\text{ec}}{\to} T$.

Theorem 7. *There is a polynomial-time algorithm which, given two unordered unlabeled trees T_1 and T_2, outputs a tree T having at least $\frac{4 \cdot \text{opt}}{e^2 \ln n_1 \ln n_2}$ edges such* *that* $T_1 \overset{\text{ec}}{\to} T$ *and* $T_2 \overset{\text{ec}}{\to} T$*, where* n_1 *and* n_2 *are the numbers of edges in* T_1 *and* T_2*, respectively, and* opt *is the number of edges in an optimal tree.*

Proof. For any given trees T_1 and T_2, we run decomp(T_1, r_1) and decomp(T_2, r_2). Then we separate the edges in T_1 and T_2 into $\frac{\ln n_1}{2}$ and $\frac{\ln n_2}{2}$ subsets, respectively, using the same strategy stated in the proof of Theorem 6. The length of a path in the i-th set is between e^{2i-2} and e^{2i}. For the i-th subset of T_1 and the j-th subset of T_2 (assuming that $i \geq j$), since each path in the j-th subset of T_2 has length at most e^{2j}, the maximum number of edges that can be matched is at most $e^{2j} \cdot \mu(i,j)$, where $\mu(i,j)$ is the minimum of (a) the number of paths in the i-th subset of T_1, and (b) the number of paths in the j-th subset of T_2. However, according to the contraction method used in Theorem 6, it is easy to see that the number of edges matched is at least $\mu(i,j) \cdot e^{2j-2}$ (i.e., $\frac{1}{e^2}$ of the maximum number of edges matched). Our contraction procedure is similar to that of Theorem 6. In the partitions of T_1 and T_2 mentioned above, we mark those edges belonging to the optimal solution opt. Since there are only $\frac{\ln n_1}{2}$ subsets of T_1 and $\frac{\ln n_2}{2}$ subsets of T_2, there exists a pair (i,j) such that the i-th subset of T_1 and the j-th subset of T_2 have more than $\frac{4 \cdot \text{opt}}{\ln n_1 \ln n_2}$ edges matched in opt. Our algorithm generates a tree of at least $\frac{1}{e^2} \cdot \frac{4 \cdot \text{opt}}{\ln n_1 \ln n_2}$ edges. The subtree achieving the above approximation ratio can be obtained by considering all possible combinations of i and j. □

4 Rotational Symmetry

Theorem 8. *For any constant* $k \geq 3$*,* k-DRS$_{\text{nd}}$ *for unordered trees with* $O(1)$ *degree can be solved in polynomial time.*

Proof. Let T be a given tree, whose maximum number of children of each node is at most δ. For brevity, we may assume that each non-leaf node of T has exactly δ children. This assumption can be removed without too much effort. Let $v_1, v_2, \ldots, v_n$ be the postordering of T. Let $v_{c(i,j)}$ be the j-th child of v_i. Let $d(i_1, i_2, \ldots, i_k)$ be the minimum number of node deletions required to transform $T(i_1), T(i_2), \ldots, T(i_k)$ into k identical trees. Given a set $S = \{s_1, s_2, \ldots, s_k\}$, a mapping $\sigma : S \to S$ is called a *permutation* of S if $\{\sigma(s_1), \sigma(s_2), \ldots, \sigma(s_k)\} = S$. It is not difficult to see that $d(i_1, i_2, \ldots, i_k)$ can be computed by the following recursive procedure: If $\{v_{i_1}, \ldots, v_{i_k}\}$ contains a leaf of T, then let $d(i_1, i_2, \ldots, i_k)$ be the sum of $|T(i_j)|$ over all leaves v_{i_j} of T. Otherwise, $d(i_1, \ldots, i_k)$ is the minimum of (a) $\min_{1 \leq p \leq k, 1 \leq q \leq \delta} d(i_1, \ldots, i_{p-1}, c(i_p, q), i_{p+1}, \ldots, i_k) + |T(i_p)| - |T(c(i_p, q))|$ and (b) $d_2 = \min_{\sigma_1, \ldots, \sigma_k \in \Sigma} \sum_{1 \leq t \leq \delta} d(c(i_1, \sigma_1(t)), \ldots, c(i_k, \sigma_k(t)))$, where Σ consists of all permutations of $\{1, \ldots, \delta\}$. Since $k = O(1)$ and $\delta = O(1)$, we know that $d(i_1, i_2, \ldots, i_k)$ can be computed in polynomial time. If the number of children of the root is no more than k, then the degree of k-rotational symmetry of T is zero. Otherwise, the degree of k-rotational symmetry of T is exactly $n-1$ minus the minimum of $\sum_{i=1}^{\lfloor \frac{\delta}{k} \rfloor} d(r_{i,1}, r_{i,2}, \ldots, r_{i,k}) + \sum \left\{ |T(j)| : v_j \text{ is a child of the root}, j \notin \left\{ r_{i,j} \mid 1 \leq i \leq \left\lfloor \frac{\delta}{k} \right\rfloor, 1 \leq j \leq k \right\} \right\}$, where the minimum is taken over all choices of distinct children $v_{r_{i,j}}$ $(1 \leq i \leq \left\lfloor \frac{\delta}{k} \right\rfloor, 1 \leq j \leq k)$ of the root. □

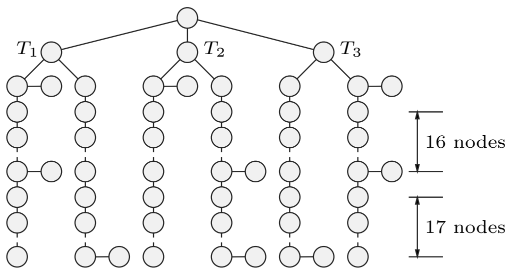

Fig. 7. The tree T_R constructed for the problem instance R of TRIPARTITE MATCHING, where $R = \{\{b_{1,1}, b_{2,1}, b_{3,2}\}, \{b_{1,1}, b_{2,2}, b_{3,2}\}, \{b_{1,2}, b_{2,2}, b_{3,1}\}\}$.

Theorem 9. *For any constant $k \geq 3$, k-DRS$_{\text{nd}}$, k-DRS$_{\text{ed}}$ and k-DRS$_{\text{ec}}$ for unordered trees are NP-complete in general.*

Proof. By Fact 1, a reduction from TRIPARTITE MATCHING [18] suffices. Let R be a given instance for TRIPARTITE MATCHING, which is a ternary relation over $B_1 \times B_2 \times B_3$, where $B_i = \{b_{i,1}, b_{i,2}, \ldots, b_{i,n}\}$ for each $i \in \{1, 2, 3\}$. We define the tree T_R for the DRS problems as follows. Define a function $f(k)$ by letting $f(1) = 1$ and $f(k) = f(k-1) + n^4 + k - 2$ for each k with $2 \leq k \leq |R|$. Let P be a path of $f(|R|)$ nodes rooted at an endpoint of P. For each $i \in \{1, 2, 3\}$ and $j \in \{1, 2, \ldots, n\}$, let $T_{i,j}$ be a rooted binary tree obtained from P as follows: the node of depth $f(k)$ is attached by a node if and only if the k-th relation in R contains $b_{i,j}$. Let T_R be a rooted tree consisting of the subtrees T_1, T_2, and T_3, where each T_i is a rooted tree consisting of the subtrees $T_{i,j}$ with $1 \leq j \leq n$. An example is shown in Figure 7. One can verify that R admits a size-n tripartite matching if and only if the degree of 3-rotational symmetry of T_R is at least $3|R| \cdot f(|R|) + 3n + 4$. ⊓⊔

5 Conclusions

From a computational complexity viewpoint, we have investigated the problem of transforming a graph into a symmetric one using a number of graph operations including edge contractions, edge deletions, and node deletions. For a given graph, our goal is to find the maximum symmetric graph that can be derived from the original graph. An approximation algorithm with a guaranteed performance has been given to one of the NP-complete problems. Our study here can be viewed as a step towards a formal treatment of the notion of near symmetry in graph drawings. An interesting future research would be to find a sequence of operations that leads to the symmetric graph, and see how such information can be used to give a nearly symmetric display of an originally asymmetric graph (perhaps, by adjusting edge lengths and levels).

References

1. T. Akutsu, An RNC algorithm for finding a largest common subtree of two trees, *IEICE Transactions on Information Systems*, E75-D, pp. 95–101, 1992.
2. S. Bachl, Isomorphic Subgraphs, *International Symposium on Graph Drawing* (GD'99), LNCS 1731, pp. 286–296, 1999.
3. G. Di Battista, P. Eades, and R. Tamassia and I. Tollis, *Graph Drawing: Algorithms for the Visualization of Graphs*, Prentice-Hall, 1999.
4. K. Chin and H. Yen, The Symmetry Number Problem for Trees, manuscript, 1998.
5. F. de Fraysseix, A Heuristic for Graph Symmetry Detection, *International Symposium on Graph Drawing* (GD'99), LNCS 1731, pp. 276–285, 1999.
6. P. Eades and X. Lin, Spring Algorithms and Symmetry, *Computing and Combinatorics* (COCOON'97), LNCS 1276, pp. 202–211, 1997.
7. M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, New York, 1979.
8. S. Hong, P. Eades, and S. Lee, Finding Planar Geometric Automorphisms in Planar Graphs, *9th International Symposium on Algorithms and Computation* (ISAAC'98) LNCS 1533, Springer, pp. 277–286, 1998.
9. S. Hong, P. Eades, A. Quigley, and S. Lee, Drawing Algorithms for Series-Parallel Digraphs in Three Dimensions, *International Symposium on Graph Drawing* (GD'98), LNCS 1547, pp. 198–209, 1998.
10. S. Hong, P. Eades, A. Quigley, and S. Lee, Drawing Series-Parallel Digraphs Symmetrically, manuscript, 1999.
11. J. Hopcroft and R. Tarjan, A V^2 Algorithm for Determining Isomorphism of Planar Graphs, *Information Processing Letters*, 1(1):32–34, 1971.
12. P. Kilpelainen, and H. Mannila, Ordered and Unordered Tree Inclusion, *SIAM Journal on Computing* 24(2):340–356, 1995.
13. P. Klein, Computing the Edit Distance Between Unrooted Ordered Trees, *6th European Symposium on Algorithms* (ESA'98), LNCS 1461, 91–102, 1998.
14. E. Lawer, *Combinatorial Optimization: Networks and Matroids*, New York: Holt, Rinehart & Winston, 1976.
15. J. Manning, *Geometric Symmetry in Graphs*, Ph.D. Dissertation, Department of Computer Science, Purdue University, 1990.
16. J. Manning and M. Atallah. Fast Detection and Display of Symmetry in Trees, *Congressus Numerantium*, 64:159–169, 1988.
17. J. Manning, M. Atallah, K. Cudjoe, J. Lozito, and R. Pacheco. A System for Drawing Graphs with Geometric Symmetry, *International Symposium on Graph Drawing* (GD'95), LNCS 894, pp. 262–265, 1995.
18. C. Papadimitriou, *Computational Complexity*, Addison-Wesley, 1994.
19. K. Zhang, and T. Jiang, Some MAX SNP-hard Results Concerning Unordered Labeled Trees, *Information Processing Letters* 49(5):249–254, 1994.
20. K. Zhang, and D. Shasha, Simple Fast Algorithms for the Editing Distance between Trees and Related Problems, *SIAM Journal on Computing* 18(6):1245–1262, 1989.