Discrete Mathematics and Theoretical Computer Science DMTCS vol. **16:3**, 2014, 129–146

Genus distributions of cubic series-parallel graphs

Jonathan L. Gross[1*] Michal Kotrbčík[2†] Timothy Sun[1‡]

[1]*Department of Computer Science, Columbia University, USA*
[2]*Department of Computer Science, Masaryk University, Czech Republic*

received 10th May 2012, revised 13th Apr. 2014, accepted 18th June 2014.

We derive a quadratic-time algorithm for the genus distribution of any 3-regular, biconnected series-parallel graph, which we extend to any biconnected series-parallel graph of maximum degree at most 3. Since the biconnected components of every graph of treewidth 2 are series-parallel graphs, this yields, by use of bar-amalgamation, a quadratic-time algorithm for every graph of treewidth at most 2 and maximum degree at most 3.

Keywords: graph embedding, genus distribution, series-parallel graphs, bounded treewidth

1 Introduction

For $i = 0, 1, 2, \ldots$, let $g_i(G)$ be the number of topologically distinct cellular embeddings of the graph G in the orientable surface S_i of genus i. The *genus distribution* of the graph G is the sequence of numbers

$$g_i(G) : i = 0, 1, \ldots \tag{1}$$

By the interpolation principle (see Theorem 3.4.1 of [GrTu87] or Theorem 4.5.3 of [MoTh01]), the set $\{i : g_i(G) > 0\}$ is a set of consecutive integers. The smallest number in this set is the *minimum genus* of the graph G, and the largest is the *maximum genus* of G.

The main focus of this paper is the derivation of a quadratic-time algorithm for the genus distribution of any 3-regular, biconnected *series-parallel graph*. This algorithm is readily extended to a quadratic-time algorithm for the genus distribution of any graph of treewidth at most 2 and maximum degree at most 3. The simplicity with which this specialized algorithm can be implemented, or applied by hand with the aid of a spreadsheet, distinguishes it from the recently derived quadratic-time algorithm [Gr14] for the genus distribution of any class of graphs of fixed treewidth and bounded degree.

Thanks to Maria Chudnovsky for suggesting series-parallel graphs as an interesting family of low-treewidth graphs for a genus distribution calculation.

*Email: gross@cs.columbia.edu
†Email: kotrbcik@fi.muni.cz
‡Email: tim@cs.columbia.edu

1.1 Basic results on genus distribution

Five fundamental papers [GKP10, Gr11a, PKG10, KPG10, PKG12] of the first author and his co-authors Khan and Poshni have established methods for calculating the genus distribution of a graph that is constructed by various kinds of amalgamation of graphs of known genus distribution. These methods involve *partitioned genus distributions* and *productions*. In order to develop an algorithm for a specific class of graphs, the starting point is to formulate a recursive specification of the graphs in that class, in which the operations used to create larger graphs from smaller graphs are varieties of amalgamation or self-amalgamation. Then methods similar to those of the five fundamental papers are used to calculate the genus distribution recursively. This paradigm was used successfully in calculating the genus distributions of 3-regular outerplanar graphs [Gr11b], of 4-regular outerplanar graphs [PKG11], of Halin graphs [Gr13], and of the $3 \times n$-mesh graphs [KPG12]. We adopt the same paradigm in this paper.

1.2 Connections of treewidth to embedding problems

Since the introduction of the concept of *treewidth* by Robertson and Seymour, bounding the treewidth has been widely used to obtain polynomial-time algorithms for problems that are otherwise NP-hard. In particular, deciding whether an arbitrarily selected graph can be embedded in a given surface is NP-complete [Th89]; however, for any class of graphs of bounded treewidth, Kawarabayashi, Mohar, and Reed [KMR09] have derived a linear-time algorithm for calculating the minimum genus.

Although outerplanar graphs have treewidth 2, and although Halin graphs and $P_3 \times P_n$ meshes have treewidth 3 (see [Bo98]), decomposition trees have not occurred in the calculation of specific genus distributions in any papers as yet. Nonetheless, low treewidth plays an implicit role in the recursive specification of the family of graphs in each of those papers. Similarly, in the present paper, low treewidth plays an implicit role, since it allows for a simple recursive construction of the graphs under consideration.

1.3 Terminology

In what follows, a *graph* is taken to be connected and devoid of self-loops, unless something else can be inferred from the immediate context. Multi-edges are to be expected. We use V_G and E_G to denote the vertex set and edge set of a graph G. A connected graph is *biconnected* if it has no cutpoints.

The *embeddings* in this paper are cellular embeddings in oriented surfaces. The terminology used here is predominantly consistent with [GrTu87] and [BWGT09]. See also [MoTh01], for a slightly different approach. We abbreviate "face-boundary walk" as *fb-walk*.

A *two-terminal series-parallel graph* is a doubly vertex-rooted graph (G, p, q), as per the following recursive definition.

- B The graph (K_2, p, q) is a two-terminal series-parallel graph, where p and q are the vertices of K_2, called the *source root* and the *target root*, respectively.

- R_1 *series operation* $(G, p, q) \odot_s (G', p', q')$ Target root q of G is merged with source root p' of G'. The amalgamated graph $G \odot_s G'$ with roots p and q', as in Figure 1(a), is a two-terminal series-parallel graph.

- R_2 *parallel operation* $(G, p, q) \odot_p (G', p', q')$ The result of merging source root p with source root p' , and also merging target root q with target root q', as in Figure 1(b), is a two-terminal series-parallel graph.

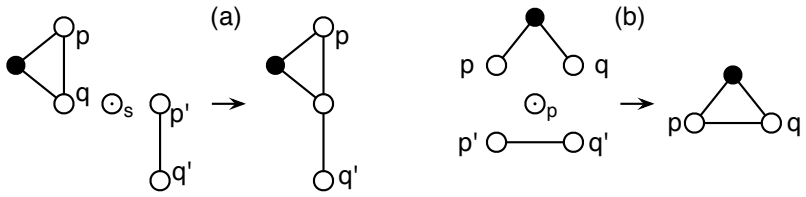

Fig. 1: Operations on series-parallel graphs.

A graph G is a *series-parallel graph* if there is a choice of terminals p and q such that (G, p, q) is a two-terminal series-parallel graph. Our definition here is consistent with that of [Bo98] and [Ep92]. A third operation, called a *jackknife operation* is allowed by [BPT09], and the resulting class of graphs that they call "series-parallel" is equivalent to that of [Du65] — see Remark 8.1 of [BPT09]. The "series-parallel graphs" in [Du65] are identified there as the graphs with no embedded "Wheatstone bridge" (which is Duffin's terminology for a K_4 topological minor). These are precisely the graphs that have no K_4-minor (e.g., see Proposition 1.7.2 of [Die06]). According to Theorem 17 of [Bo98], the graphs without a K_4-minor are exactly the graphs of treewidth at most 2. It follows that our extended genus-distribution algorithm can be applied to any of them in quadratic time, and that we do not need to further explore the distinctions between the varying definitions of "series-parallel graphs".

1.4 Outline of this paper

Section 2 derives a characterization of 3-regular, biconnected series-parallel graphs that facilitates the genus distribution algorithm for that family of graphs. Section 3 introduces the concepts of *partitioned genus distributions* and *productions*. The top-level description of an algorithm for the genus distribution of any 3-regular, biconnected series-parallel graph is given in Section 4. Sections 5 and 6 derive the productions needed to complete the calculation, as well as their application to calculating the genus distribution of an illustrative example. Section 6 also gives proof that the algorithm runs in quadratic time. Section 7 extends the algorithm to all graphs of treewidth 2 and maximum degree 3.

This paper is almost entirely self-contained, except for some details of the well-established concept of partitioned genus distributions and of the methods (as in [GKP10] and [PKG10]) for constructing productions (which are quite necessary for the algorithm). Prior experience with calculating genus distributions of graph amalgamations, especially as in [Gr11b] and [Gr13], is likely to be quite helpful.

2 Cubic Biconnected Series-Parallel Graphs

The *dipole* D_n is the graph with two vertices and an n-fold multi-edge joining them. In this section, we prove that every 3-regular, biconnected series-parallel graph can be obtained by iterated application of the following operation to the dipole D_3.

- τ Trisect an arbitrary edge e of a graph G and install a new edge in parallel to the "middle third" of edge e.

This operation, which is applicable to an non-empty graph, is called a *dmt-step* ("dmt" is an abbreviation of "double the middle third"), is illustrated by Figure 2.
We define a second operation τ^{-1}, which is applicable to 3-regular multigraphs, other than the dipole D_3. We observe that the operation τ^{-1} can be used as an inverse to the operation τ.

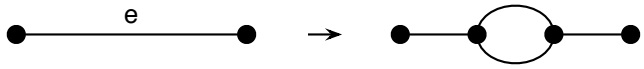

Fig. 2: A dmt-step: double the middle third.

τ^{-1} In a 3-regular graph G, let d and e be edges that share the same two endpoints, u and v, and in which no other edge shares these two endpoints. Delete edge d, and then smooth away vertices u and v.

The proof of our characterization of 3-regular, biconnected series-parallel graphs uses the following four propositions.

Proposition 2.1 (Theorem 42 of [Bo98]) *A graph G has treewidth at most 2 if and only if every biconnected component of G is a series-parallel graph.*

Proposition 2.2 *A graph G has treewidth at most 2 if and only if contains no K_4-minor.*

Proof: This follows immediately from Theorem 17 of [Bo98]. □

Proposition 2.3 *Let G be a 3-regular, biconnected series-parallel graph, and let G' be a graph with at least two vertices, obtained by applying operation τ^{-1} to edges d and e of G , with shared endpoints u and v. Then G' is 3-regular, biconnected, and series-parallel.*

Proof: Since G is 3-regular, all of its vertices are 3-valent. The operation τ^{-1} eliminates two vertices of G without changing the valences of the remaining vertices. Thus, the graph G' is 3-regular.

Let x and y be any two vertices of G'. Since G is biconnected, there is a pair of internally disjoint paths in G joining x and y, by Menger's theorem. If one of these paths contains either of the vertices u or v, then it also contains one of the edges d or e, since G is 3-regular, and the other path contains neither vertex u nor vertex v. Thus the images of the two internally disjoint path in G are two internally disjoint paths in G'. Hence, the graph G' is biconnected.

Since the graph G is series-parallel, its treewidth is at most 2, by Proposition 2.1. It follows from Proposition 2.2 that G has no K_4-minor. Accordingly, the graph G' has no K_4 minor. Therefore, by Proposition 2.2, the graph G' has treewidth at most 2. We conclude from Proposition 2.1 that the graph G' is series-parallel. □

Proposition 2.4 (Dirac's theorem, [Dir52]) *Let G be a biconnected simple graph of minimum degree 3. Then G contains a subgraph that is homeomorphic to the complete graph K_4.*

Theorem 2.5 *Let G be a biconnected 3-regular series-parallel graph. Then there exist vertices $p, q \in V_G$ such that the two-terminal series-parallel graph (G, p, q) is derivable from (D_3, p, q) by a sequence of applications of the operation τ.*

Proof: Let H be a smallest graph obtainable by iterative application of operation τ^{-1} to the graph G, that is, a graph such that no pair of vertices is joined by exactly two edges. By Proposition 2.3, the graph H is 3-regular, biconnected, and series-parallel. We observe that H cannot be simple, lest it contain, by Dirac's theorem, a homeomorphic copy of K_4, a contradiction, in view of Propositions 2.2 and 2.1. Accordingly,

there is a pair of vertices $p, q \in V_H$ with at least two edges joining them. Since there cannot be exactly two edges joining p and q, by the minimality of the graph H, and since H is 3-regular, it follows that $H \cong D_3$. Reversing the sequence of τ^{-1}-operations, we obtain a derivation of (G, p, q) from (D_3, p, q) by iterative application of the operation τ. □

We define a *dmt-string* to be a graph obtained by iterative application of dmt-steps to the graph K_2. We observe that each dmt-string has two univalent vertices and that all other vertices are trivalent.

Corollary 2.6 *Let (G, p, q) be a 3-regular, biconnected two-terminal series-parallel graph. Then (G, p, q) can be represented by a set of three dmt-strings, each with a univalent p-vertex and a univalent q-vertex, from which (G, p, q) is formed by two parallel operations.*

3 Partials and Productions

When calculating the genus distribution of a family of graphs, we commonly use a finer partition of the embeddings during the intermediate steps. For computational purposes, we need to isolate subsets of embeddings upon which the surgical operations used in the recursive construction of that family have the same effect. Whereas the genus distribution of a graph is an inventory according only (as per (1)) to the genus of the embedding surface, a *partitioned genus distribution* refines the genus distribution of a rooted graph, according to the incidence of fb-walks on the roots, which is the critical factor in the behavior of a surgical operation on an embedding. In other words, a partitioned genus distribution is a partition of all embeddings of the graph with a given genus into into several types, which allows us to keep under control the structure of the faces incident with the roots before, respectively after the amalgamation. The cells of the finer partition are called *partials*. In this context, we sometimes abbreviate genus distribution as *gd* and partitioned genus distribution as *pgd*.

Calculating the genus distributions of 3-regular series-parallel graphs involves amalgamating subgraphs at pairs of terminals, such that the sum of the degrees of an amalgamated pair of vertices is at most 3. Thus, the possible degrees of the terminals prior to the final operation are 1 and 2. When both terminals are univalent, the genus distribution of (G, p, q) is partitioned into the following partials:

$$\begin{aligned} uu_i^{\bullet}(G, p, q) &= \text{the number of embeddings } G \to S_i \text{ such that} \\ &\quad \text{terminals } p \text{ and } q \text{ do not occur on the same fb-walk;} \\ uu_i'(G, p, q) &= \text{the number of embeddings } G \to S_i \text{ such that} \\ &\quad \text{terminals } p \text{ and } q \text{ occur on the same fb-walk.} \end{aligned}$$

The letter u in the name of the partial is a mnemonic for *univalent*. For every $i = 0, 1, 2, \ldots$, the set of all embeddings of (G, p, q) with genus i gives a partitioned genus distribution given by the formula

$$g_i(G, p, q) \;=\; uu_i^{\bullet}(G, p, q) + uu_i'(G, p, q)$$

When terminal p is univalent and terminal q is bivalent, the letters d or s in the name of the partial mean, respectively, that q occurs on two *different* fb-walks or that q occurs twice on the *same* fb-walk.

There are four partials:

$$\begin{aligned}
ud_i^\bullet(G,p,q) &= \text{the number of embeddings } G \to S_i \text{ such that} \\
&\quad \text{terminal } p \text{ occurs on neither fb-walk incident at } q; \\
ud_i'(G,p,q) &= \text{the number of embeddings } G \to S_i \text{ such that} \\
&\quad \text{terminal } p \text{ occurs on one fb-walk incident at } q; \\
us_i^\bullet(G,p,q) &= \text{the number of embeddings } G \to S_i \text{ such that} \\
&\quad \text{terminal } p \text{ does not occur on the fb-walk incident at } q; \\
us_i'(G,p,q) &= \text{the number of embeddings } G \to S_i \text{ such that} \\
&\quad \text{terminal } p \text{ occurs on the fb-walk incident at } q.
\end{aligned}$$

As before, for every $i = 0, 1, 2, \ldots$, the set of all embeddings of (G, p, q) with genus i has a partitioned genus distribution according to the formula

$$g_i(G,p,q) \;=\; ud_i^\bullet(G,p,q) + ud_i'(G,p,q) + us_i^\bullet(G,p,q) + us_i'(G,p,q)$$

Similarly, when p is bivalent and q is univalent, there are four partials:

$$\begin{aligned}
du_i^\bullet(G,p,q) &= \text{the number of embeddings } G \to S_i \text{ such that} \\
&\quad \text{terminal } q \text{ occurs on neither fb-walk incident at } p; \\
du_i'(G,p,q) &= \text{the number of embeddings } G \to S_i \text{ such that} \\
&\quad \text{terminal } q \text{ occurs on one fb-walk incident at } p; \\
su_i^\bullet(G,p,q) &= \text{the number of embeddings } G \to S_i \text{ such that} \\
&\quad \text{terminal } q \text{ does not occur on the fb-walk incident at } p; \\
su_i'(G,p,q) &= \text{the number of embeddings } G \to S_i \text{ such that} \\
&\quad \text{terminal } q \text{ occurs on the fb-walk incident at } p.
\end{aligned}$$

Suppose that $p^1, p^2, \ldots, p^s$ is a set of partials for a genus distribution. A *production* for a given surgical operation that transforms a graph embedding $X \to S_i$ (or a tuple of graph embeddings) into a set of graph embeddings of the graph Y is an algebraic rule of this form:

$$p_i^j(X) \longrightarrow c_1 p^1_{f_1^j(i)}(Y) + \cdots + c_t p^s_{f_s^j(i)}(Y) \tag{2}$$

The left side is called the *antecedent*, and the right side is called the *consequent*. The meaning is that the operation transforms a single embedding of graph X of type p^j on the orientable surface S_i of genus i into a set of embeddings of the graph Y, of which c_k are of type p^k on the surface $S_{f_k^j(i)}$, for each i, j, and k. A drawing is usually used as an aid in deriving the production and in proving its correctness. The names of the graphs and their roots can be suppressed when there is in context no ambiguity. Thus, we may write

$$p_i^j \longrightarrow c_1 p^1_{f_1^j(i)} + \cdots + c_t p^s_{f_s^j(i)} \tag{3}$$

In general, when there are n partials, a surgical operation on two graphs is represented by n^2 productions for the partials. It is clear to someone familiar with the use of partials and productions that it is possible to represent the parallel operation and the series operation by respective lists of productions, one for each ordered pair of partials. This would also lead to an algorithm that requires quadratic-time. Since our present objective is an algorithm that can be described concisely and calculated by hand for small graphs, we intend here to construct shorter lists of productions.

4 An Algorithm

Our Algorithm 4.1 for calculating the genus distribution of any cubic, biconnected, two-terminal series-parallel graph G has five steps.

Algorithm 4.1 ***Genus distribution algorithm for a cubic, biconnected, series-parallel graph*** G**.**

Input: A 3-regular biconnected series-parallel graph G.
Output: The genus distribution of the graph G.

1. Choose the endpoints p and q of an edge as the terminals.
2. Determine the three dmt-strings N^1, N^2, N^3 corresponding to the graph (G, p, q).
3. Calculate the pgd of each of the three dmt-strings N^1, N^2, N^3.
4. Calculate the pgd of the graph $N^1 \odot_p N^2$.
5. Calculate the gd of the graph $G = (N^1 \odot_p N^2) \odot_p N^3$.

Step (1)**.** Lemma 9 of [Ep92] shows that a biconnected series-parallel graph is two-terminal series-parallel, for any pair of terminals p and q that are joined by an edge. Therefore, we can select as terminals p and q the endpoints of any edge of G.

Step (2)**.** Determine the three dmt-strings $(N^1, p, q), (N^2, p, q), (N^3, p, q)$ for the graph G, by splitting both of the vertices p and q of the dipole D_3 into three vertices, each an endpoint of one of the edges incident on the split vertex.

Step (3)**.** It simplifies this calculation if we define a small modification of the parallel operation $\odot_p$. When we combine two dmt-strings with a parallel operation, we obtain two 2-valent vertices. Our modified operation $\overline{\odot}_p$ attaches a spike at each of these 2-valent vertices, as illustrated in Figure 3, so that we once again have a dmt-string.

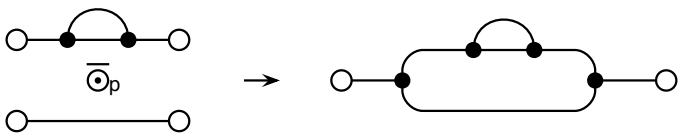

Fig. 3: The modified parallel operation $\overline{\odot}_p$.

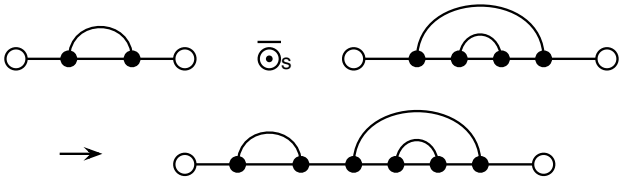

Fig. 4: The modified parallel operation $\overline{\odot}_s$.

Similarly, our modified operation $\overline{\odot}_s$ merges the second terminal of the first graph with the first terminal of the second graph, and then smooths away the merged vertex, as illustrated in Figure 4.

The method for calculating the pgd of a dmt-string is given in Section 5.

Step (4)**.** This step concerns the composition of an amalgamation of two double-rooted graphs with univalent roots at one pair of univalent roots, followed by a self-amalgamation at the other two roots. See Section 6 for a description of this computation.

Step (5)**.** This step is the composition of an amalgamation of two double-rooted graphs with univalent roots at one pair of univalent roots, followed by a self-amalgamation at the other two roots. See Section 6 for a description of this computation.

5 PGD of DMT-Strings

Four parallel productions and four series productions will be sufficient to calculate the values of the partials of any dmt-string. The following two sets of four productions each are sufficient to calculate all the partials of a dmt-string. The parallel productions are derived with the aid of Figures 5, 6, and 7. The series productions are self-evident. The genus of each resultant embedding is calculated from its Euler characteristic.

$$uu_i^\bullet \,\overline{\odot}_p\, uu_j^\bullet \longrightarrow 4uu_{i+j+1}^\bullet \quad \text{(Figure 5)} \tag{4}$$
$$uu_i^\bullet \,\overline{\odot}_p\, uu_j' \longrightarrow 4uu_{i+j+1}' \quad \text{(Figure 6)} \tag{5}$$
$$uu_i' \,\overline{\odot}_p\, uu_j^\bullet \longrightarrow 4uu_{i+j+1}' \quad \text{(mirror of Figure 6)} \tag{6}$$
$$uu_i' \,\overline{\odot}_p\, uu_j' \longrightarrow 2uu_{i+j}^\bullet + 2uu_{i+j}' \quad \text{(Figure 7)} \tag{7}$$

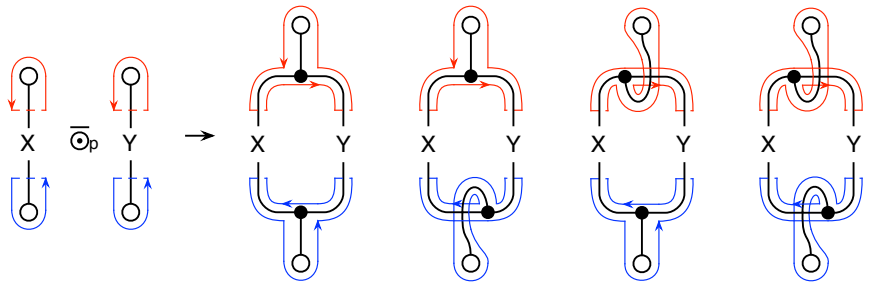

Fig. 5: Production: $uu_i^\bullet \,\overline{\odot}_p\, uu_j^\bullet \longrightarrow 4uu_{i+j+1}^\bullet$.

$$uu_i^\bullet \,\overline{\odot}_s\, uu_j^\bullet \longrightarrow uu_{i+j}^\bullet \tag{8}$$
$$uu_i^\bullet \,\overline{\odot}_s\, uu_j' \longrightarrow uu_{i+j}^\bullet \tag{9}$$
$$uu_i' \,\overline{\odot}_s\, uu_j^\bullet \longrightarrow uu_{i+j}^\bullet \tag{10}$$
$$uu_i' \,\overline{\odot}_s\, uu_j' \longrightarrow uu_{i+j}' \tag{11}$$

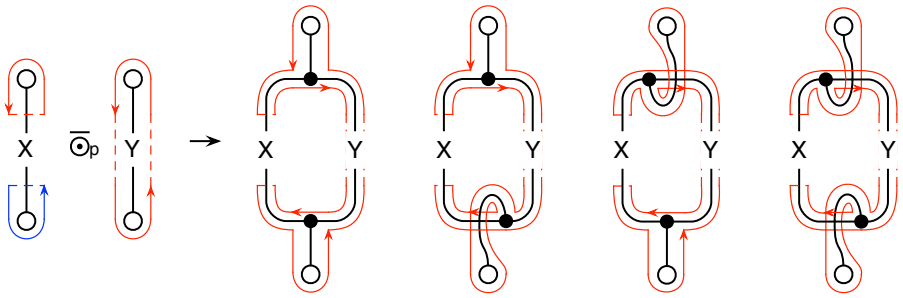

Fig. 6: Production: $uu_i^{\bullet} \,\overline{\odot}_p\, uu'_j \longrightarrow 4uu'_{i+j+1}$.

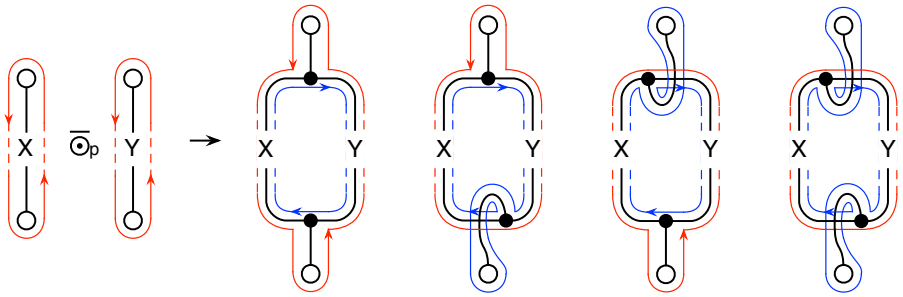

Fig. 7: Production: $uu'_i \,\overline{\odot}_p\, uu'_j \longrightarrow 2uu'_{i+j} + 2uu^{\bullet}_{i+j}$.

5.1 Examples

Our first example is the dmt-string $\hat{D}_2$ of Figure 8, which is of fundamental use in our further calculations. Here we use pgd (which stands for *partitioned genus distribution*) as a function.

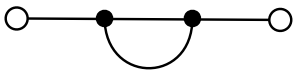

Fig. 8: The dmt-string $\hat{D}_2$.

We observe that $\hat{D}_2$ is representable as $K_2 \,\overline{\odot}_p\, K_2$. Therefore,

$$\begin{aligned} pgd(\hat{D}_2) &= pgd(K_2 \,\overline{\odot}_p\, K_2) \\ &= uu'_0 \,\overline{\odot}_p\, uu'_0 \qquad (12) \\ pgd(\hat{D}_2) &= 2uu_0^{\bullet} + 2uu'_0 \qquad \text{by Prod. (7)} \qquad (13) \end{aligned}$$

We now apply Eq. (7) and the productions above to calculate the partials of the three dmt-strings of the graph of Figure 9.

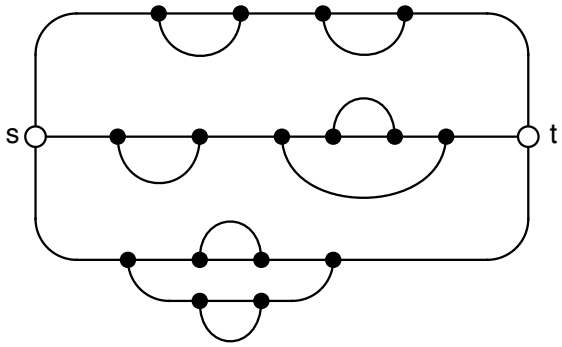

Fig. 9: A cubic series-parallel graph.

The top dmt-string N^1 is representable as $\hat{D}_2 \,\overline{\odot}_s\, \hat{D}_2$. Therefore,

$$\begin{aligned}
pgd(N^1) &= pgd(\hat{D}_2 \,\overline{\odot}_s\, \hat{D}_2) \\
&= (2uu_0^\bullet + 2uu_0')\,\overline{\odot}_s\,(2uu_0^\bullet + 2uu_0') \\
&= 2uu_0^\bullet \,\overline{\odot}_s\, 2uu_0^\bullet + 2uu_0^\bullet \,\overline{\odot}_s\, 2uu_0' \\
&\qquad + 2uu_0' \,\overline{\odot}_s\, 2uu_0^\bullet + 2uu_0' \,\overline{\odot}_s\, 2uu_0' \\
&= 4uu_0^\bullet + 4uu_0^\bullet + \ 4uu_0^\bullet + 4uu_0' \\
&\qquad \text{by Prods. (8), (9), (10), (11)} \\
pgd(N^1) &= 12uu_0^\bullet + 4uu_0' \qquad (14)
\end{aligned}$$

The dmt-string N^2 is representable as $\hat{D}_2 \,\overline{\odot}_s\, (\hat{D}_2 \,\overline{\odot}_p\, K_2)$. Therefore,

$$\begin{aligned}
pgd(N^2) &= pgd(\hat{D}_2 \,\overline{\odot}_s\, (\hat{D}_2 \,\overline{\odot}_p\, K_2)) \\
&= (2uu_0^\bullet + 2uu_0')\,\overline{\odot}_s\,((2uu_0^\bullet + 2uu_0')\,\overline{\odot}_p\, uu_0') \\
&= (2uu_0^\bullet + 2uu_0')\,\overline{\odot}_s\,(8uu_1' + 4uu_0^\bullet + 4uu_0') \\
&\qquad \text{by Prods. (5), (7)} \\
&= 16uu_1^\bullet + 8uu_0^\bullet + 8uu_0^\bullet + 16uu_1' + 8uu_0^\bullet + 8uu_0' \\
&\qquad \text{by Prods. (4), (5), (6), (7)} \\
pgd(N^2) &= 24uu_0^\bullet + 8uu_0' + 16uu_1^\bullet + 16uu_1' \qquad (15)
\end{aligned}$$

The dmt-string N^3 is representable as $\hat{D}_2 \,\overline{\odot}_p\, \hat{D}_2$. Therefore,

$$\begin{aligned}
pgd(N^3) &= pgd(\hat{D}_2 \,\overline{\odot}_p\, \hat{D}_2) \\
&= (2uu_0^\bullet + 2uu_0')\,\overline{\odot}_p\,(2uu_0^\bullet + 2uu_0') \\
&= 16uu_1^\bullet + 16uu_1' + 16uu_1' + 8uu_0^\bullet + 8uu_0' \\
&\qquad \text{by Prods. (4), (5), (6), (7)} \\
pgd(N^3) &= 8uu_0^\bullet + 8uu_0' + 16uu_1^\bullet + 32uu_1' \qquad (16)
\end{aligned}$$

6 Amalgamating DMT-Strings

A parallel operation on two dmt-strings yields a series-parallel graph whose source and target roots are both 2-valent. Figure 10 illustrates the productions used to transform the partials of the two antecedent dmt-strings into the partials of the consequent series-parallel graph. We define two partials for the case where p and q are both bivalent and a single walk is twice incident at each:

$$\begin{aligned} ss_i^{\bullet}(G,p,q) &= \text{the number of embeddings } G \to S_i \text{ such that} \\ &\quad \text{the fb-walks twice incident at } p \text{ and } q \text{ are different;} \\ ss_i'(G,p,q) &= \text{the number of embeddings } G \to S_i \text{ such that} \\ &\quad \text{the same fb-walk is twice incident at } p \text{ and } q. \end{aligned}$$

We also define

$$dd_i''(G,p,q) = \text{the number of embeddings } G \to S_i \text{ such that} \\ \text{the same two fb-walks are twice incident at } p \text{ and } q.$$

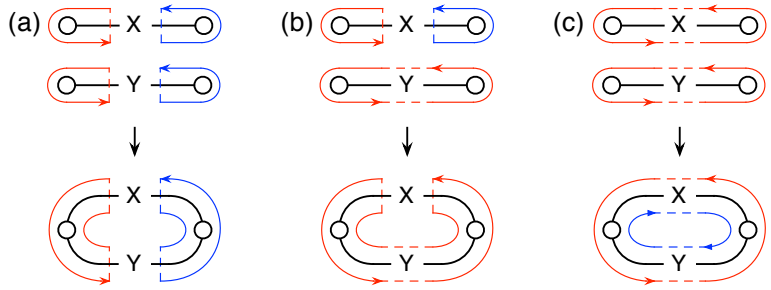

Fig. 10: Parallel operations on two dmt-strings.

$$uu_i^{\bullet} \odot_p uu_j^{\bullet} \longrightarrow ss_{i+j+1}^{\bullet} \quad \text{(Figure 10(a))} \tag{17}$$
$$uu_i^{\bullet} \odot_p uu_j' \longrightarrow ss_{i+j+1}' \quad \text{(Figure 10(b))} \tag{18}$$
$$uu_i' \odot_p uu_j^{\bullet} \longrightarrow ss_{i+j+1}' \quad \text{(use Figure 10(b))} \tag{19}$$
$$uu_i' \odot_p uu_j' \longrightarrow dd_{i+j}'' \quad \text{(Figure 10(c))} \tag{20}$$

Once again, the genus of each resultant embedding is calculated from its Euler characteristic. Continuing with our example of the graph from Figure 5.5, these productions enable us, in turn, to calculate $pgd(N^1 \overline{\odot}_p N^2)$.

$$\begin{aligned} pgd(N^1 \odot_p N^2) &= (12uu_0^\bullet + 4uu_0') \\ &\quad \odot_p (24uu_0^\bullet + 8uu_0' + 16uu_1^\bullet + 16uu_1') \\ &= 12uu_0^\bullet \odot_p (24uu_0^\bullet + 8uu_0' + 16uu_1^\bullet + 16uu_1') \\ &\quad + 4uu_0' \odot_p (24uu_0^\bullet + 8uu_0' + 16uu_1^\bullet + 16uu_1') \\ &= 288ss_1^\bullet + 96ss_1' + 192ss_2^\bullet + 192ss_2' \\ &\quad + 96ss_1' + 32dd_0'' + 64ss_2' + 64dd_1'' \\ pgd(N^1 \odot_p N^2) &= 32dd_0'' + 64dd_1'' + 288ss_1^\bullet + 192ss_2^\bullet \\ &\quad + 192ss_1' + 256ss_2' \end{aligned} \tag{21}$$

The following six productions will enable us to complete our calculation of the genus distribution of the graph of Figure 9, starting from the pgd (21) for $N^1 \overline{\odot}_p N^2$ and the pgd (16) for N^3. We observe that the result of applying these productions is a genus distribution, rather than a partitioned genus distribution. Accordingly, the consequents are of the form g_i rather than subscripted partials, thereby indicating only that the resulting embedding surface is S_i.

$$dd_i'' \odot_p uu_j^\bullet \longrightarrow 4g_{i+j+1} \quad \text{(Figure 11)} \tag{22}$$
$$dd_i'' \odot_p uu_j' \longrightarrow 2g_{i+j} + 2g_{i+j+1} \quad \text{(Figure 12)} \tag{23}$$
$$ss_i^\bullet \odot_p uu_j^\bullet \longrightarrow 4g_{i+j+1} \quad \text{(Figure 13)} \tag{24}$$
$$ss_i^\bullet \odot_p uu_j' \longrightarrow 4g_{i+j+1} \quad \text{(Figure 14)} \tag{25}$$
$$ss_i' \odot_p uu_j^\bullet \longrightarrow 4g_{i+j+1} \quad \text{(Figure 15)} \tag{26}$$
$$ss_i' \odot_p uu_j' \longrightarrow 4g_{i+j} \quad \text{(Figure 16)} \tag{27}$$

Figures 11, 12, 13, 14, 15, and 16 illustrate the six productions (22), (23), (24), (25), (26), and (27), respectively, that collectively provide an algebraic representation of the parallel operation.

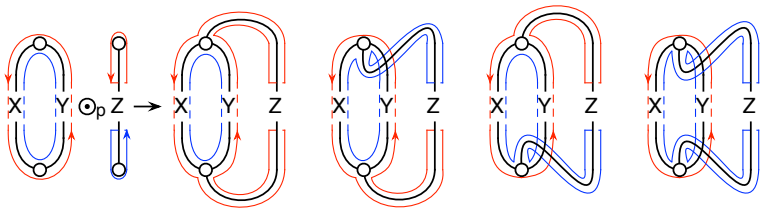

Fig. 11: $dd_i'' \odot_p uu_j^{\bullet} \longrightarrow 4g_{i+j+1}$.

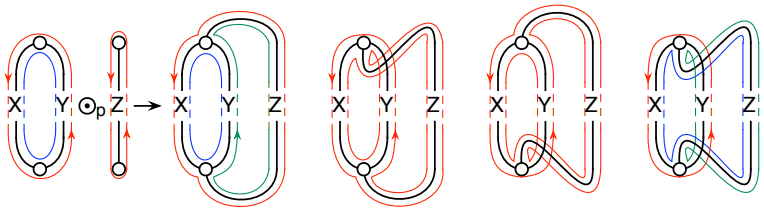

Fig. 12: $dd_i'' \odot_p uu_j' \longrightarrow 2g_{i+j} + 2g_{i+j+1}$.

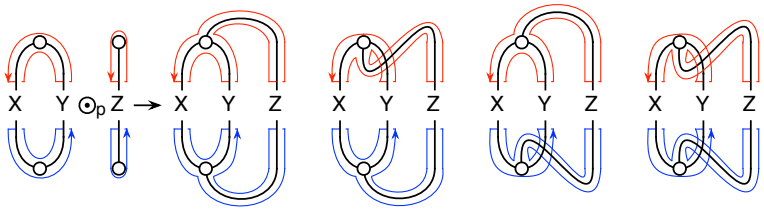

Fig. 13: $ss_i^{\bullet} \odot_p uu_j^{\bullet} \longrightarrow 4g_{i+j+1}$.

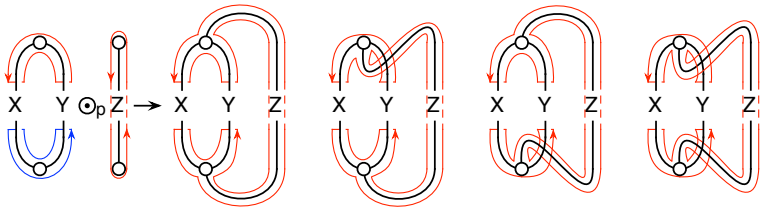

Fig. 14: $ss_i^{\bullet} \odot_p uu_j' \longrightarrow 4g_{i+j+1}$.

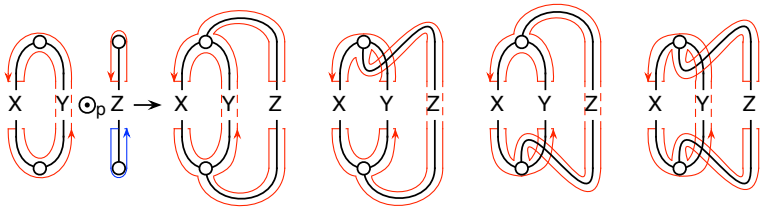

Fig. 15: $ss_i' \odot_p uu_j \longrightarrow 4g_{i+j+1}$.

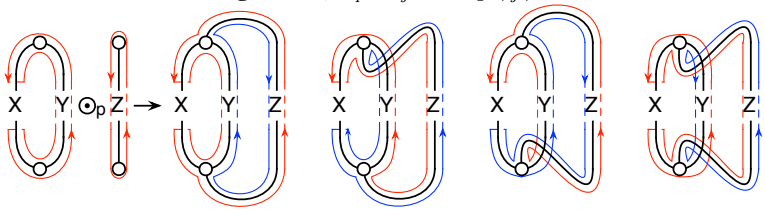

Fig. 16: $ss_i' \odot_p uu_j' \longrightarrow 4g_{i+j}$.

$$\begin{aligned} pgd(N^1 \odot_p N^2) &= 32dd''_0 + 64dd''_1 + 288ss^\bullet_1 + 192ss^\bullet_2 + 192ss'_1 + 256ss'_2 \\ pgd(N^3) &= 8uu^\bullet_0 + 8uu'_0 + 16uu^\bullet_1 + 32uu'_1 \end{aligned}$$

$$\begin{aligned} 32dd''_0 \odot_p 8uu^\bullet_0 &= 1024g_1 \\ 32dd''_0 \odot_p 8uu'_0 &= 512g_0 + 512g_1 \\ 32dd''_0 \odot_p 16uu^\bullet_1 &= 2048g_2 \\ 32dd''_0 \odot_p 32uu'_1 &= 2048g_1 + 2048g_2 \\ 64dd''_1 \odot_p 8uu^\bullet_0 &= 2048g_2 \\ 64dd''_1 \odot_p 8uu'_0 &= 1024g_1 + 1024g_2 \\ 64dd''_1 \odot_p 16uu^\bullet_1 &= 4096g_3 \\ 64dd''_1 \odot_p 32uu'_1 &= 4096g_2 + 4096g_3 \\ 288ss^\bullet_1 \odot_p 8uu^\bullet_0 &= 9216g_2 \\ 288ss^\bullet_1 \odot_p 8uu'_0 &= 9216g_2 \\ 288ss^\bullet_1 \odot_p 16uu^\bullet_1 &= 18432g_3 \\ 288ss^\bullet_1 \odot_p 32uu'_1 &= 36864g_3 \\ 192ss^\bullet_2 \odot_p 8uu^\bullet_0 &= 6144g_3 \\ 192ss^\bullet_2 \odot_p 8uu'_0 &= 6144g_3 \\ 192ss^\bullet_2 \odot_p 16uu^\bullet_1 &= 12288g_4 \\ 192ss^\bullet_2 \odot_p 32uu'_1 &= 24576g_4 \\ 192ss'_1 \odot_p 8uu^\bullet_0 &= 6144g_2 \\ 192ss'_1 \odot_p 8uu'_0 &= 6144g_1 \\ 192ss'_1 \odot_p 16uu^\bullet_1 &= 12288g_3 \\ 192ss'_1 \odot_p 32uu'_1 &= 24576g_2 \\ 256ss'_2 \odot_p 8uu^\bullet_0 &= 8192g_3 \\ 256ss'_2 \odot_p 8uu'_0 &= 8192g_2 \\ 256ss'_2 \odot_p 16uu^\bullet_1 &= 16384g_4 \\ 256ss'_2 \odot_p 32uu'_1 &= 32768g_3 \end{aligned}$$

By summation of the right-hand sides of the equations above, we obtain the genus distribution

genus	0	1	2	3	4
#embeddings	512	10752	68608	129024	53248

This calculation has been confirmed by a computer program based on the Heffter-Edmonds algorithm.

Theorem 6.1 *The time required by Algorithm 4.1 is at most quadratic in the number of vertices of the graph G supplied as input.*

Proof: Choosing an edge and its endpoints s and t, as required by Step (1), takes constant time. The time needed for Step (2), which is achieved by partitioning the edge-set of the given graph G into the edge-sets of the three dmt-strings, is linear in the number of vertices of G, using depth-first search.

When two subgraphs are amalgamated during Step (3), the contribution to the partials of the merged graph corresponding to a pair of nonzero-valued subscripted partials, one in the pgd of the first amalgamand and the other in the pgd of the second amalgamand, is calculated by the application of one of the Productions (4), ..., (11). Since the number of nonzero-valued subscripted partials for any graph is linear in the number of vertices, the time to calculate the pgd of the merged graph is proportional to the product of the numbers of vertices in the two amalgamands. Suppose that the numbers of vertices of the fragments of an dmt-string are $x_1, x_2, \ldots, x_p$. Since the vertices in two different fragments will be merged into a combined fragment only once during the reassembly of the dmt-string, the number of applications of productions during the entire reassembly is at most

$$\sum_{i \neq j} x_i x_j$$

However,

$$\sum_{i \neq j} x_i x_j \ < \ (x_1 + x_2 + \cdots + x_p)^2$$

from which we infer that the total time is at most quadratic in the number of vertices of the dmt-string.

To see that Step (4) can be done in quadratic time, first observe that the number of nonzero partials in $pgd(N^1)$ and $pdg(N^2)$ is linear in the maximum genus of N^1 and N^2, since partials can be nonzero only for genera with genus less than or equal to maximum genus and for every genus in this range there is only a constant number of different types of partials. Furthermore, it is well known that the maximum genus of any graph G is bounded from above by $\beta(G)$, the cycle rank of G, which in turn is linear in the number of vertices for any graph with maximum degree 3. Accordingly, there is only a linear number of nonzero partials for both N^1 and N^2, yielding at most a quadratic number of combinations needed to calculate $pdg(N^1 \odot_p N^2)$. Finally, each combination of two partials can be computed in constant time using appropriate choices from Productions (4), ..., (7), which implies that Step (4) can indeed be done in quadratic time in the number of vertices.

The argument that Step (5) uses only quadratic time is similar to that for Step (4) — the number of nonzero partials is again linear in the number of vertices; and each computation, this time involving one of the Productions (22), ..., (27), can be done in constant time. □

7 Extending to all Graphs of Treewidth ≤ 2 and Maximum Degree ≤ 3

The *bar-amalgamation* of two disjoint rooted graphs (G, u) and (H, v) is the result of running a new edge (the "bar") between u and v.

Proposition 7.1 (Theorem 5 of [GrFu87]) *The genus distribution of the bar-amalgamation of graphs (G, u) and (H, v) is the constant multiple of the convolution of the genus distributions of G and H. The constant factor is the product of the degree of u in G and the degree of v in H.*

To extend Algorithm 4.1 to any graph G of treewidth at most 2 and maximum degree 3, we infer from Proposition 2.1 that each biconnected component of G either is isomorphic to K_2 or is a homeomorphic copy of a 3-regular biconnected graph. Moreover, each of the latter kind of biconnected components meets only the K_2-type components. Thus, the graph G is an iterated bar-amalgamation of biconnected series-parallel graphs. Accordingly, to calculate its genus distribution, we calculate the genus distributions of its biconnected components, take corresponding convolutions, and multiply by scalars corresponding to degrees of vertices at the ends of the bars.

8 Conclusions

Starting with a quadratic-time algorithm for calculating the genus distribution of all cubic biconnected series-parallel graphs, we have constructed a quadratic-time algorithm for the genus distribution of any graph of treewidth at most 2 and maximum degree at most 3. The advantage of this case-specific algorithm over the more general algorithm of [Gr14] is the ease with the case-specific algorithm can be used to obtain numerical results for specific graphs.

Acknowledgements

The second author was partly supported by Nadácia Tatra Banky grant 11sds071, SAIA NSP, grant APVV-0223-10, the grant APVV-ESF-EC-0009-10 within the EUROCORES Programme EUROGIGA (project GReGAS) of the European Science Foundation, and Ministry of Education, Youth, and Sport project No. CZ.1.07/2.3.00/30.0009 – Employment of Newly Graduated Doctors of Science for Scientific Excellence. The research was done while the second author was a Ph.D. student at the Department of Computer Science, Comenius University, Bratislava, Slovakia, and was visiting the Department of Computer Science at Columbia University. He would like to thank his host Prof. J. L. Gross and the department for the hospitality.

References

[BWGT09] L. W. Beineke, R. J. Wilson, J. L. Gross, and T. W. Tucker, Editors, *Topics in Topological Graph Theory*, Cambridge Univ. Press, 2009.

[Bo98] H. L. Bodlaender, A partial k-arboretum of graphs with bounded treewidth, *Theoretical Comp. Sci.* **209** (1998), 1–45.

[BPT09] R. B. Borie, R. G. Parker, and C. A. Tovey, Solving problems on recursively constructed graphs, *ACM Compu. Surveys* **41** (2009), 1–51.

[Die06] R. Diestel, *Graph Theory*, Third Edition, Springer, 2006.

[Dir52] G. A. Dirac, A property of 4-chromatic graphs and some remarks on critical graphs, *J. London Math. Soc.* **27** (1952), 85–92.

[Du65] R. J. Duffin, Topology of series-parallel networks, *J. Math. Analysis and Applications* **10** (1965), 303–318.

[Ep92] D. Eppstein, Parallel recognition of series-parallel graphs, *Information and Computation* **98** (1992), 41–55.

[Gr10] J. L. Gross, Genus distribution of graphs under surgery: adding edges and splitting vertices, *New York J. Mathematics* **16** (2010), 161–178.

[Gr11a] J. L. Gross, Genus distribution of graph amalgamations: Self-pasting at root-vertices, *Australasian J. Combin.* **49** (2011), 19–38.

[Gr11b] J. L. Gross, Genus distributions of cubic outerplanar graphs, *J. of Graph Algorithms and Applications* **15** (2011), 295–316.

[Gr13] J. L. Gross, Embeddings of cubic Halin graphs: genus distributions, *Ars Math. Contemporanea* **6** (2013), 37–56.

[Gr14] J. L. Gross, Embeddings of graphs of fixed treewidth and bounded degree, *Ars Math. Contemporanea* **7** (2014), 379–403.

[GrFu87] J. L. Gross and M. Furst, Hierarchy for imbedding-distribution invariants of a graph, *J. Graph Theory* **11** (1987), 205–220.

[GKP10] J. L. Gross, I. F. Khan, and M. I. Poshni, Genus distribution of graph amalgamations: Pasting at root-vertices, *Ars Combin.* **94** (2010), 33–53.

[GrTu87] J. L. Gross and T. W. Tucker, *Topological Graph Theory*, Dover, 2001; (original edn. Wiley, 1987).

[KMR09] K. Kawarabayashi, B. Mohar, and B. Reed, A simpler linear time algorithm for embedding graphs into an arbitrary surface and the genus of graphs of bounded tree-width, *Proc. 49th Ann. Symp. on Foundations of Computer Science (FOCS'08)* IEEE (2008), 771–780.

[KPG10] I. F. Khan, M. I. Poshni, and J. L. Gross, Genus distribution of graph amalgamations at roots of higher degree, *Ars Math. Contemporanea* **3** (2010), 121–138.

[KPG12] I. F. Khan, M. I. Poshni, and J. L. Gross, Genus distribution of $P_3 \times P_n$, *Discrete Math.* **312** (2012), 2863–2871.

[MoTh01] B. Mohar and C. Thomassen, *Graphs on Surfaces*, Johns Hopkins University Press, 2001.

[PKG10] M. I. Poshni, I. F. Khan, and J. L. Gross, Genus distribution of edge-amalgamations, *Ars Math. Contemporanea* **3** (2010), 69–86.

[PKG11] M. I. Poshni, I. F. Khan, and J. L. Gross, Genus distribution of 4-regular outerplanar graphs, *Electronic J. Combin.* **18** (2011) #P212, 25pp.

[PKG12] M. I. Poshni, I. F. Khan, and J. L. Gross, Genus distribution of graphs under self-edge-amalgamations, *Ars Math. Contemporanea* **5** (2012), 127–148.

[Th89] C. Thomassen, The graph genus problem is NP-complete, *J. Algorithms* **10** (1989), 568–576.