Local Optimization Algorithms for Maximum Planar Subgraph

Gruia Călinescu, Sumedha Uniyal

Local Optimization Algorithms for Maximum Planar Subgraph

Gruia Călinescu @ ORCID
Department of Computer Science, Illinois Institute of Technology, USA.

Sumedha Uniyal @ ORCID
Department of Computer Science, Aalto University, Finland

Abstract

Consider the NP-hard problem of, given a simple graph G, to find a planar subgraph of G with the maximum number of edges. This is called the Maximum Planar Subgraph problem and the best known approximation is 4/9 and is obtained by sophisticated Graphic Matroid Parity algorithms. Here we show that applying a local optimization phase to the output of this known algorithm improves this approximation ratio by a small $\epsilon = 1/747 > 0$. This is the first improvement in approximation ratio in more than a quarter century. The analysis relies on a more refined extremal bound on the Lovász cactus number in planar graphs, compared to the earlier (tight) bound of [5, 8].

A second local optimization algorithm achieves a tight ratio of 5/12 for Maximum Planar Subgraph without using Graphic Matroid Parity. We also show that applying a greedy algorithm before this second optimization algorithm improves its ratio to at least $91/216 < 4/9$. The motivation for not using Graphic Matroid Parity is that it requires sophisticated algorithms that are not considered practical by previous work. Also, we could not find any implementation [1]. The best previously published [7] approximation ratio without Graphic Matroid Parity is $13/33 < 5/12$.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis

Keywords and phrases planar graph, maximum subgraph, approximation algorithm, matroid parity, local optimization

Digital Object Identifier 10.4230/LIPIcs.ESA.2024.3

Funding Gruia Călinescu's work performed in part while visiting Northwestern University. Sumedha's work performed in part at Aalto University, Finland, and in part at Northwestern University, United States of America. She was supported by the Academy of Finland (grant agreement number 314284).
Sumedha Uniyal: Sumedha Uniyal passed away in February 2020. Her husband, Prajna Uniyal, email uniyal.prajna@gmail.com, has approved this submission.

Acknowledgements Gruia thanks Michael Pelsmajer for many comments that improved the presentation. Gruia thanks the anonymous SODA 2024 reviewer B who suggested using a positive linear combination (instead of case analysis) when computing the ϵ-improvement of Algorithm MTLK4, which lead to a slightly simpler proof with a bigger ϵ. Gruia also thanks Konstantin Makarychev for hints on finding the best linear combination.

[1] After this paper was accepted by ESA 2024, We found out from Matthias Stallmann about https://github.com/Arkhist/Graphic-Matroid-Parity

1 Introduction

Maximum Planar Subgraph (MPS) is the following problem: given a graph G, find a planar subgraph of G with the maximum number of edges. MPS is known to be NP-complete [24] and APX-hard [5], meaning that there exists a small $\epsilon > 0$ such that a polynomial-time algorithm with approximation ratio of at least $1 - \epsilon$ for MPS implies that $P = NP$. In this paper all graphs are undirected, nonempty, finite, simple graphs unless otherwise noted.

Besides theoretical appeal, MPS has applications in circuit layout, facility layout, and graph drawing (see [17, 31, 27, 23, 14, 12, 21] and the references contained there). Poranen [27] uses simulated annealing and interestingly reports that using the output of better approximation algorithms as the start of simulated annealing obtains better practical results. Chimani, Klein, and Wiedera [13] also report benefits of using better approximation algorithms for MPS.

We assume that the reader has basic knowledge of planar graphs and approximation algorithms. For a quarter century, the best known approximation algorithm for MPS has a tight approximation ratio[2]of 4/9 [5] and uses polynomial-time algorithms for Graphic Matroid Parity[3] to construct a maximum triangular cactus (a graph all of whose blocks[4]are triangles[5]), followed by adding edges to connect the components of this cactus without creating any new cycles. Graphic Matroid Parity has polynomial-time algorithms [26, 25, 9, 18, 19] and allows us to find a triangular cactus with maximum number of triangles.

In this paper we show that applying a local optimization phase after finding a maximum triangular cactus achieves an approximation ratio of $(4/9) + \epsilon$, for a small $\epsilon > 0$. A second local optimization algorithm achieves a tight ratio of 5/12 for MPS without using Graphic Matroid Parity. We also show that applying a "greedy" algorithm before this second local optimization algorithm improves its ratio to at least 91/216. This variant has an upper limit of $3/7 < 4/9$ on its approximation ratio. The motivation for not using Graphic Matroid Parity is that it requires sophisticated algorithms that are not considered practical by Chimani and Wiedera [14][6]. The best previously published [7] approximation ratio without Graphic Matroid Parity is $13/33 < 5/12$.

Besides these theoretical guarantees, both of our local optimizations "make sense" (since each local improvement step increases the size of the output by one) and can be used in heuristics, or as a starting point of simulated annealing. Other ideas that make sense are further discussed in Conclusions (Section 6).

1.1 Previous Theoretical Work

A planar graph on n vertices has at most $3n - 6$ edges. As we can assume that the input graph is connected (or else we can run an algorithm separately on the connected components of the input), simply outputting a spanning tree achieves a tight approximation ratio of 1/3. Dyer, Foulds, and Frieze [16] proved that the Maximal Planar Subgraph, which simply outputs an inclusion-maximal planar subgraph, has performance ratio 1/3. Cimikowski [15] proved that the heuristics of Chiba, Nishioka, and Shirakawa [10] and of Cai, Han, and

[2] *tight* in this paper means that there are matching examples for the approximation ratio of the algorithm
[3] See Section 2 for the exact definition
[4] We use the standars definition of *block*, explicitly stated in Section 2
[5] A simple graph is a *cactus* if any two distinct simple cycles have at most one vertex in common. A triangular cactus as defined by us is a cactus.
[6] Also, we could not find any implementation.

Tarjan [2] have performance ratios not exceeding 1/3.

Călinescu, Fernandes, Finkler, and Karloff [5] were the first to improve this. Their first algorithm (GT, from Greedy-Triangles) (greedily) adds triangles, as long as the graph stays a triangular cactus. Then the algorithm greedily adds edges connecting the components of the cactus, without creating any new cycles. This "connecting" phase appears at the end of all the approximation algorithms and we will not mention it from now on. The GT algorithm has a tight approximation ratio of 7/18. Their second algorithm (MT, from Maximum-Triangular Cactus) employs exact algorithms for Graphic Matroid Parity (i.e., [25]) to construct a triangular cactus with a maximum number of triangles. This algorithm achieves a tight approximation of 4/9, and the analysis is long and complicated. Another claim of the 4/9 ratio was sent in a personal communication in 1996 or 1997 by Danny Raz, and is also proven in Chalermsook, Schmid, and Uniyal [8][7]; these proofs are also long and/or complicated.

Poranen [28] proposes two new approximation algorithms but their ratio is below 4/9 [11]. Chalermsook and Schmid [7] obtain a 13/33 ratio without using Graphic Matroid Parity. Note that $13/33 > 7/18$. We have examples showing that this algorithm has ratio at most $11/27 < 5/12$. See Figure 1 for a graphical representation of approximation ratios.

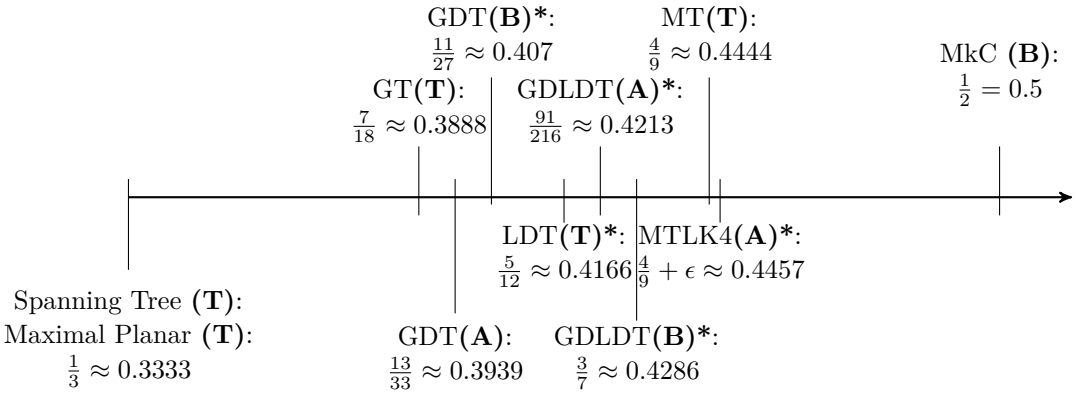


Figure 1 Ratios for various approaches. **(T)** says that the result is tight with matching example. **(A)** says that an algorithm is proven to achieve that ratio. **(B)** states an upper bound for the approximation ratio of an algorithm. New results marked by *.

1.2 Techniques discussion

A graph A is a kt-structure if every block in every non-trivial connected component of A is either a K_4 or a triangle. A diamond is the graph resulting from deleting any single edge from a K_4. A graph A is a dt-structure if every block in every non-trivial connected component of A is either a diamond or a triangle. A graph A is a 4-structure if every block in A has at most 4 vertices. All 4-structures are planar graphs. See Figure 2 for illustrations.

[7] [8] also has the sentence "Therefore, combining this with our bound implies that local search arguments are sufficient to get us to a $\frac{4}{9}+\epsilon$ approximation for MPS", but no proof for this sentence; their previous arguments give a $\frac{4}{9}-\epsilon$ approximation.

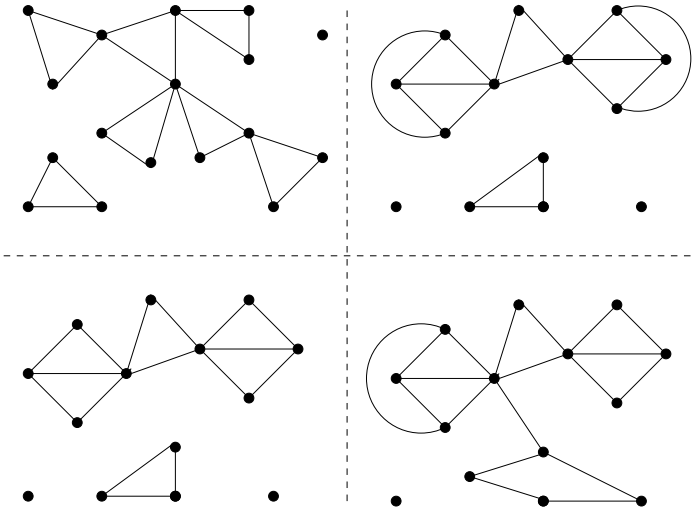


Figure 2 In the upper left, a triangular cactus. In the upper right, a kt-structure. In the lower left, a dt-structure. In the lower right, a 4-structure.

4-Structures can have more edges than triangular cacti, but finding a 4-structure with maximum number of edges is known to be NP-hard [4] (easy reduction from 3D-Matching [20]). It is known [3] that even if we fix k and allow blocks of size up to k in our output, and could find the k-structure with maximum number of edges and planar blocks (algorithm MkC), we would still get an approximation ratio of smaller than 1/2.

Local optimization is a powerful technique for unweighted maximization problems. For example, Lee, Sviridenko, and Vondrák [22] prove that a basic local optimization algorithm achieves a $(1-\epsilon)$-approximation for Matroid Parity, a generalization of Graphic Matroid Parity that is NP-hard [29]. Using local optimization for MPS was suggested by [8].

Our first algorithm, denoted by MTLK4 (Maximum-Triangular-Local-K4) from now on, starts with a maximum triangular cactus and simply replaces, as long as we still have a kt-structure, at most two triangles by a K_4. Previous approximation algorithms for MPS have not been able to exploit K_4's in getting a better approximation ratio, and in fact our analysis giving that the approximation ratio of MTLK4 is at least $(4/9)+\epsilon$ is complicated and not tight[8]. The analysis relies on Theorem 5, which gives a more refined extremal bound on the Lovász cactus number in planar graphs, compared to the earlier (tight) bound of [5, 8].

The algorithm of Chalermsook and Schmid [7] (which we call GDT, from Greedy-Diamonds-Triangles) consists of greedily adding diamonds followed by greedily adding triangles (while staying a dt-structure), and has approximation ratio between 13/33 and 11/27. Our 5/12 approximation ratio of our second local optimization algorithm, denoted by LDT (Local-Diamonds-Triangles) from now on, is based on the fact that LDT does "break" diamonds to add more diamonds, assuming a triangle is left from our broken diamond. While this 5/12-approximation is tight, it can be improved by greedily adding diamonds (while staying a dt-structure)[9], followed by LDT. We call this algorithm GDLDT. We prove that GDLDT has an approximation ratio between 91/216 and 3/7.

While our algorithms are fairly simple, their analyses are long and complicated. The

[8] We believe that another page of arguments will make our ϵ a very little bit bigger.
[9] This is exactly the start of the GDT algorithm of [7].

reader may want to start with the (7/18)-approximation of [5] (also presented in [4]) for a simple non-trivial algorithm with a somewhat simple analysis.

One can also use local optimization to approximate Graphic Matroid Parity, as described in [22]. As pointed out in [8], this leads to a $4/9-\epsilon$ approximation for MPS, as outlined below. The local optimization algorithm used by [22], adapted to our terminology, replaces sets of k triangles by sets of $k+1$ triangles, as long as it keeps a triangular cactus. We have (so far unpublished, and involving a different set of authors) evidence that swapping two triangles for one gives a ratio of at least 11/27. Theorem 1.1 of [8] also mentions a 2-swap algorithm, but it swaps only triangles of a plane graph and is not analyzed as an MPS algorithm. Let n be the number of vertices in the input graph, and M be the number of triangles in the input graph. The analysis of [22] requires $k = 5^{\lceil 1/(2\epsilon)\rceil}$ to achieve a $(1-\epsilon)$-approximation for Matroid Parity. To achieve a 5/12 approximation for MPS, one would need an $\epsilon = 1/4$ above: if the optimum is triangulated and the maximum triangular cactus has $n/3$ triangles (which we know that it can happen, from [5]), then we need a triangular cactus with almost $n/4$ triangles in the algorithm's cactus to get a ratio of 5/12, which means we must approximate Graphic Matroid Parity with a factor of 3/4. Directly using the analysis of [22] gives an $O(M^{25})$-time algorithm. We do not fully understand [1] and maybe it provides a better analysis of the swapping heuristics and achieves a $O(n^{4+3/\epsilon})$ running time. This would be $O(n^{16})$ time to get a ratio of 5/12 for MPS. By comparison, our LDT algorithm has an implementation that runs in $O(n^5)$ time.

2 Preliminaries

Generally speaking we follow West [32] for terminology and notation. Given a graph $G = (V, E)$ and $V' \subseteq V$, we denote by $G[V']$, the *induced* subgraph of G with vertex set given by V' (we keep in this subgraph with vertex set V' all the edges of G with both endpoints in V').

A *triangle* in a graph is a C_3, a cycle of length three. A graph is *planar* if it can be drawn in the 2-dimensional plane so that no two edges meet in a point other than a common end. A triangle of a plane graph is a *facial triangle* if the triangle is the boundary of a face of the plane graph. We call a face of a plane graph a *triangular face* if its boundary is a triangle.

A *cut-vertex* (respectively, *bridge*) is a vertex (resp., edge) whose deletion increases the number of components. A graph is said to be a *biconnected graph*, if it has at least three vertices and it does not contain any cut-vertices. A *block* is a maximal connected subgraph without a cut-vertex. Thus, every block is either a maximal biconnected subgraph, or a bridge.

Graphic Matroid Parity is the following problem: given a multigraph $G' = (V', E')$ and a partition of the edge set E' into pairs of distinct edges $\{f, f'\}$, find a (simple) forest $F \subseteq E'$ with the maximum number of edges, such that $f \in F$ if and only if $f' \in F$, for all $f \in E'$.

Graphic Matroid Parity algorithms can be used to construct a maximum triangular cactus in a given graph [25]. Indeed, given a graph $G = (V, E)$, one constructs G' by having $V' = V$, and for each triangle T of G, let e, e' be any pair of distinct edges in T. Add two new edges f and f' to E', f with the same endpoints as e, and f' with the same endpoints as e'. Pair f with f' in the partition of E'. The straightforward lemma 2.13 of [5] states that a forest F of G' as above with $2p$ edges exists if and only if a triangular cactus with p triangles exists in G.

3 Local Optimization after Max Triangular Cactus

Here is our first approximation algorithm, Algorithm **MTLK4**: Start with a maximum triangular cactus. As long as possible, if there exists a K_4 in the input graph and at most two edge-disjoint triangles in our kt-structure such that removing the (one or two) triangles and adding the K_4 results in a kt-structure, do this replacement. See figures 3 and 4 for illustrations. Each such replacement increases the cyclomatic number[10] of the kt-structure, and thus this is a polynomial-time algorithm. Precisely, there are $O(n^4)$ K_4's to consider and for each we can look at $O(n^2)$ ways to choose two existing triangles for replacement (these two triangles come from a graph with $O(n)$ edges). So we can achieve each replacement in $O(n^7)$ time, for a total running time, excluding the Matroid Parity algorithm, of $O(n^8)$.

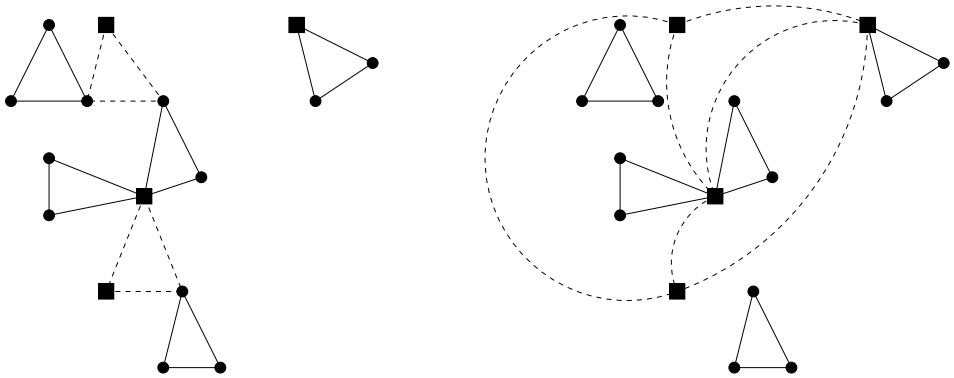


Figure 3 On the left side, the current kt-structure, with two connected components. There is a K_4 in the input graph, with its four vertices represented by small filled squares. Two triangles from the current kt-structure, dashed, can be removed to disconnect these four vertices, after which the K_4 (with edges represented by dashed cycle arcs) can be be added resulting in another kt-structure (on the right side, with three connected components) with a higher cyclomatic number.

[10] The cyclomatic number (also called circuit rank, cycle rank, or nullity) of an undirected graph is the minimum number of edges that must be removed from the graph to break all its cycles, making it into a tree or forest.

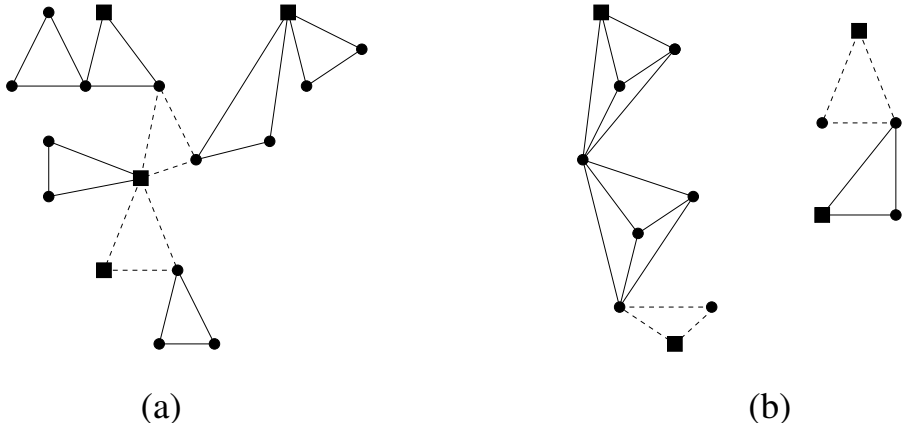


Figure 4 Both (a) and (b) show kt-structures where, for each of them, two triangles (dashed) can be removed to allow a K_4, whose vertices are represented by small filled squares, to be added resulting in a kt-structure.

We believe that an $O(n^5)$-running time version of the algorithm exists, based on the following idea. One needs to read the proof of the approximation ratio first, as we modify the algorithm. We only check for each K_4 if there is no strong bond, and its four vertices are not all in the same component of the current kt-structure. This check can be done in $O(1)$ after computing connected components by Depth First Search (and this has to be done only once after each local improvement). A K_4 that does not pass this check is not used for the local improvement, even though in MTLK4 it could possibly lead to improvement (see Figure 4 (a)). However, we cannot get a $O(n^4)$ running time by showing that a K_4 that was found not useful once will not become useful in the future. We have a counterexample.

The rest of the section will be spent on the approximation ratio analysis.

First, some intuition. Let $\beta(\bar{G})$ be the number of triangles in a maximum triangular cactus in graph $\bar{G}$. It is known from [5] (see also Inequality (2) below) that, if $\tilde{H}$ is a planar graph on n vertices with $|E(\tilde{H})| = 3n-6-t$, then $\beta(\tilde{H}) \geq (1/3)(n-t-2)$. With G the input graph and H an optimum solution (a maximum planar subgraph of G), we obtain that the output of the MT algorithm (which, recall, computes the triangular cactus with maximum number of triangles, and then connects the components of this cactus) has $n-1+\beta(G)$ edges (since every triangle gains one edge over a spanning tree), and we obtain a ratio of $4/9$ from $\beta(G) \geq \beta(H)$. With θ such that H has $3n-6-\theta$ edges, from simple algebraic manipulation, we would have an $\epsilon > 0$ improvement over $4/9$ if $\theta \geq \epsilon' n$ for some $\epsilon' > 0$. But we cannot count on this happening.

So now we assume that θ is close to 0. As we use K_4's, the second question is what happens if H (the optimum solution as above) does not have any K_4's. The bound $\beta(H) \geq (1/3)(n-\theta-2)$ is not tight anymore, and in fact with $\theta = 0$ we are able to prove that $\beta(H) \geq (3/7)n-1$ when H has no K_4's. This is good as $(n+(3/7)n-2)/(3n-6) \geq 10/21 > 4/9$.

So we assume that θ is close to 0 and H has K_4's. To quantify the benefits of these K_4's, we introduce some notation (see Figure 5 for an illustration):

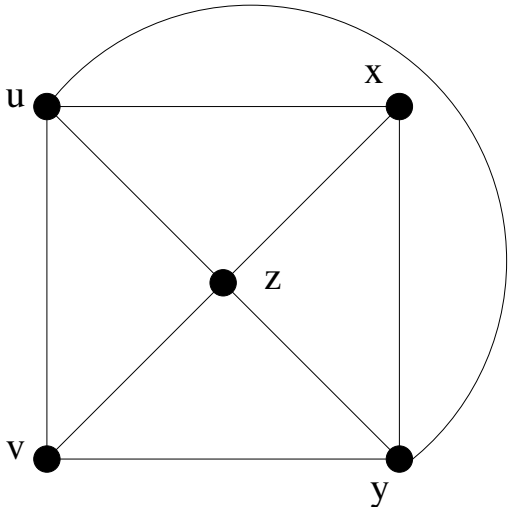


Figure 5 In this triangulated plane graph, x and v are the cubic (degree three) vertices.

▶ **Definition 1.** *Call a vertex v of a triangulated plane graph $\tilde{H}$* cubic *if it has degree three (so we have a K_4 formed by v and its neighbors). Let $c(\tilde{H})$ be the number of cubic vertices in $\tilde{H}$.*

Theorem 5 below obtains that $\beta(\tilde{H}) \geq (3/7)n - (1/7)c(\tilde{H}) - 1$, with $n = |V(\tilde{H})|$. With $\tilde{H}$ a triangulated supergraph of H on the same vertex set, we have that $\tilde{H}$ contains exactly θ edges not in H. Then we can obtain that $\beta(H) \geq (3/7)n - (1/7)c(\tilde{H}) - 1 - \theta$. Now, if $c(\tilde{H})$ is also small in addition to θ being close to 0, we have again a ratio bigger than 4/9. And when $c(\tilde{H})$ is "large" compared to n, and θ is close to 0, then there are enough K_4's in H for the local optimization phase of MTLK4 to use, as shown after Theorem 5. We are done with providing some intuition. As small graphs are easy to handle, we assume that $n = |V(G)| > 4$ in this section. On our way to Theorem 5 we need a lot of notation.

Double-partitions. Let $\bar{G} = (V, E)$ be a graph on n vertices. Let $\mathcal{P} = \{V_1, \ldots, V_k\}$ be a partition of the vertices of $\bar{G}$ into *vertex classes*, and $\mathcal{Q} = \{E_1, \ldots, E_q\}$ be a partition of the edges of $\bar{G}$ into *edge colors*. We say that an edge color and a vertex class are incident if at least one of the edges of the edge color is incident to at least one of the vertices of the vertex class. For $1 \leq i \leq q$, let u_i denote the number of vertex classes V_j of $\mathcal{P}$ incident to edge color E_i of $\mathcal{Q}$. We say that "double-partition" $(\mathcal{P}, \mathcal{Q})$ *covers* a triangle if the triangle has at least two vertices in the same vertex class of $\mathcal{P}$ (when we also say that the triangle is *covered* by this vertex class) or all three edges in the same edge color of $\mathcal{Q}$ (when we also say that the triangle is *covered* by this edge color). We call the double-partition $(\mathcal{P}, \mathcal{Q})$ *valid for* $\bar{G}$ if every triangle of $\bar{G}$ is covered. Set[11]:

$$\Phi(\mathcal{P}, \mathcal{Q}) = n - k + \sum_{i=1}^{m} \lfloor \frac{u_i - 1}{2} \rfloor. \tag{1}$$

Since $k \leq n$ and $u_i \geq 1$ for all i, we have that $\Phi(\mathcal{P}, \mathcal{Q}) \geq 0$, and note that there is always a valid double-partition $(\mathcal{P}, \mathcal{Q})$ for $\bar{G}$ (e.g., $k = 1$ and $q = 1$ and $V_1 = V(\bar{G})$ and $E_1 = E(\bar{G})$).

[11] The formula given here, which differs from that in [25] by having floors, is correct. See also [30] or Theorem 11.3.2 of [25].

Call a vertex class a *trivial vertex class* if it has just one vertex, call this vertex a *singleton*, and an edge color a *trivial edge color* if it has just one edge. The *contribution* of an edge color E_i of $\mathcal{Q}$ to $\Phi(\mathcal{P},\mathcal{Q})$ is the quantity $\lfloor \frac{u_i-1}{2} \rfloor$. The *contribution* of $\mathcal{P}$ to Φ is defined to be $n-k$.

According to Lovász and Plummer [25], we have:

▶ **Proposition 2** (Theorem 11.3.6 in [25]). *The number of triangles in a maximum triangular cactus in a graph $\bar{G}$ is equal to the minimum of $\Phi(\mathcal{P},\mathcal{Q})$ taken over all valid double-partitions $(\mathcal{P},\mathcal{Q})$ for $\bar{G}$.*

Let $\tilde{H}$ be a planar graph with $n \geq 3$ vertices. Embed $\tilde{H}$ in the plane without crossing edges, obtaining a plane graph. Let t be the number of edges *missing* for this embedding to be triangulated (meaning that adding t edges to $\tilde{H}$ would result in a triangulated planar graph). A triangulated plane graph has $3n-6$ edges, if $n \geq 3$. So $t = (3n-6) - |E(\tilde{H})|$; t does not depend on the embedding. The 4/9-approximation of [5] is based on:

▶ **Proposition 3** (Theorem 2.3 of [5], also equivalent to Corollary 1.2 of [8]). *Let $\tilde{H}$ be a connected planar graph with $n \geq 3$ vertices. Let t be the number of missing edges, defined as above. Then*

$$\Phi(\mathcal{P},\mathcal{Q}) \geq \frac{1}{3}(n-2-t),$$

for all valid double-partitions $(\mathcal{P},\mathcal{Q})$ for $\tilde{H}$.

This bound is known to be asymptotically tight when $t=0$, see for example [5] where their 4/9-approximation is proven tight. Recall that $\beta(\tilde{H})$ is the number of triangles in a maximum triangular cactus in $\tilde{H}$. From Propositions 2 and 3 we obtain:

$$\beta(\tilde{H}) \geq \frac{1}{3}(n-2-t), \tag{2}$$

If a double-partition $(\mathcal{P},\mathcal{Q})$ is valid for plane graph $\tilde{H}$, then it must cover all the facial triangles of $\tilde{H}$.

▶ **Definition 4.** *Call a double-partition $(\mathcal{P},\mathcal{Q})$ of a plane graph $\tilde{H}$* p-valid *if it covers the facial triangles of $\tilde{H}$.*

Note that a valid double-partition $(\mathcal{P},\mathcal{Q})$ of a graph $\bar{G}$ is p-valid for any plane embedding of $\bar{G}$. The main technical result of this section is Theorem 5 below, which gives a lower bound for $\Phi(\mathcal{P},\mathcal{Q})$ for all p-valid double partitions $(\mathcal{P},\mathcal{Q})$ of a triangulated plane graph $\tilde{H}$ in terms of n and $c(\tilde{H})$. The accounting part of the lower bound is proven assuming that the double partition $(\mathcal{P},\mathcal{Q})$ satisfies a number of conditions. Ideally, we would want that $\mathcal{P}$ would have only one vertex class P_1 with more than one vertex (we are not able to ensure this condition); this condition would make the accounting much simpler.

We will enumerate the actual conditions later. To get that $(\mathcal{P},\mathcal{Q})$ satisfy these actual conditions, we *modify* $(\mathcal{P},\mathcal{Q})$, which really means obtaining from a p-valid $(\mathcal{P},\mathcal{Q})$ another double partition $(\mathcal{P}',\mathcal{Q}')$ such that $\Phi(\mathcal{P},\mathcal{Q}) \geq \Phi(\mathcal{P}',\mathcal{Q}')$ and such that $(\mathcal{P}',\mathcal{Q}')$ is also p-valid for $\tilde{H}$. It is fine to make such modifications since, once we prove a lower bound for $(\mathcal{P}',\mathcal{Q}')$, then it holds for $(\mathcal{P},\mathcal{Q})$ as well.

Assuming that $\tilde{H}$ is a triangulated plane graph, one can easily verify based on the definition of $\Phi(\mathcal{P},\mathcal{Q})$ (Equation (1)) that we can do the following **modifications**:

1. Any edge of $\tilde{H}$ such that both facial triangles that contain this edge are covered by vertex classes is put into an edge color by itself (we can do this since this trivial edge color contributes 0 to Φ, and the contribution of any other edge color does not increase). In particular, any edge that is incident to two vertices in the same vertex class is put into an edge color by itself.
2. If we have two distinct edge colors both incident to the same two (or more) distinct vertex classes, these two edge colors can be merged into one edge color.
3. If an edge color Q_i can be partitioned into two edge colors Q' and Q'' such that all the facial triangles covered by Q_i are covered by Q' or Q'', and Q' and Q'' are incident to at most one common neighbor among the vertex classes, split Q_i into Q' and Q''.

Recall Definition 1. One main technical result is:

▶ **Theorem 5.** *Let $\tilde{H}$ be a triangulated plane graph with $n \geq 3$ vertices, and let $(\mathcal{P}, \mathcal{Q})$ be a double-partition p-valid for $\tilde{H}$. Then:*

$$\Phi(\mathcal{P}, \mathcal{Q}) \geq \frac{3}{7}n - \frac{1}{7}c(\tilde{H}) - 1. \tag{3}$$

This very long proof appears later. We give some shorter intuition now. Equation (3) is tight (excluding a small additive term), as shown by the following two constructions. Both of them provide useful examples for the intuition behind the proof. Below, we use c to denote the number of cubic vertices of the plane graph being discussed. Consider a triangulation W on r vertices containing no K_4. From Euler's formula we get that W has $2r-4$ faces. In the first construction, insert one cubic vertex in every face. We have $n = r + (2r-4)$ and $c = 2r-4$. We take $\mathcal{P}$ to have only one non-trivial vertex class, W, and $\mathcal{Q}$ to have only trivial edge colors; note that the double partition $(\mathcal{P}, \mathcal{Q})$ is p-valid for the constructed plane graph. Thus $\Phi(\mathcal{P}, \mathcal{Q}) = r - 1 = (3/7)n - (1/7)c - (1/7)$.

In the second construction, insert three new vertices into every face of W as in Figure 6. We have $n = r + 3(2r-4) = 7r - 12$ and $c = 0$. We take $\mathcal{P}$ to have only one non-trivial vertex class, W, and $\mathcal{Q}$ to have $2r-4$ non-trivial edge colors, each with the edges embedded strictly in each face of W. Each of these non-trivial edge colors is incident to exactly 4 vertex classes: the three singletons (singletons are vertices) inside the face of W, and the non-trivial class comprised of W. Thus $\Phi(\mathcal{P}, \mathcal{Q}) = r - 1 + 2r - 4 = (3/7)n + (1/7)$. Also note that the double partition $(\mathcal{P}, \mathcal{Q})$ is p-valid for the constructed plane graph.

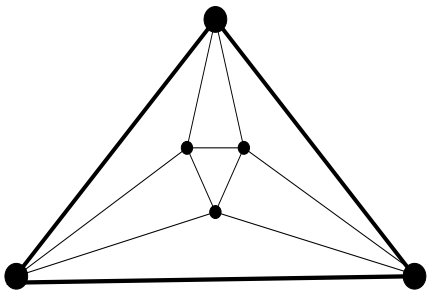


Figure 6 A face of W is shown in solid edges.

Intuition. To show some of the proof ideas, we make the simplifying assumption that $\mathcal{P}$ has only one non-trivial part, which we call R (we could not find a way to enforce this

assumption and ended up with a much longer proof). We also make another two simplifying assumptions, that $r := |R| > 2$ and that $\tilde{H}[R]$ is a connected graph (Removing these two assumptions seems possible with some effort). First, apply Modification 1 such that any edge of $\tilde{H}[R]$ is put into an edge color by itself.

For a face F of $\tilde{H}[R]$, call S_F the set of singletons embedded strictly inside F. We discuss now the case $|S_F| > 1$. Due to $\tilde{H}$ being triangulated, $\tilde{H}[S_F]$ is connected. Walking around $s \in S_F$ in clockwise manner, we get a circular list L of vertices that are not all in the same vertex class. When we switch in L from one class to another, we have one facial triangle T_1 of $\tilde{H}$ that is not covered by a vertex class. Later in the walk, when again we switch in L from one vertex class to another, we have another facial triangle T_2 of $\tilde{H}$ that is not covered by a vertex class. If there are two distinct edge colors Q_1 and Q_2 of $\mathcal{Q}$ covering T_1 and T_2, then there are two distinct vertex classes incident to both Q_1 and Q_2 and we can apply Modification 2. Continue this traversal of L and get that all the facial triangles incident to s that are not covered by R are covered by one single edge color. $\tilde{H}[S_F]$ is connected, and we can do a depth-first search traversal and as we meet the vertices of S_F, we do the walk-around described above for each of them to obtain that all the facial triangles of $\tilde{H}$ embedded in F that are not covered by R are covered by one single edge color which we call Q_F. By construction, Q_F meets all the singletons of S_F and the vertex class R.

If edge color Q_F as above also covers facial triangles outside F, then we can apply Modification 3, where Q' consist of the edges of Q_F embedded strictly inside F and Q'' consists of the other edges of Q_F; note that only R among the vertex classes can be incident to both Q' and Q''. After doing this for all the faces of $\tilde{H}[R]$, we get that any non-trivial edge color is the Q_F for some face F with $|S_F| > 1$ of $\tilde{H}[R]$, and all the edges of Q_F are embedded strictly inside F. Note that when $|S_F| = 0$ or $|S_F| = 1$, all the triangular faces of $\tilde{H}$ embedded in F are covered by R. And all the triangular faces of $\tilde{H}$ are embedded in some face F of $\tilde{H}[R]$. We now have:

$$\Phi(\mathcal{P}, \mathcal{Q}) = (r-1) + \sum_{F \text{ face of } \tilde{H}[R]} \lfloor |S_F|/2 \rfloor = (r-1) + \sum_{i \geq 0} f_i \lfloor \frac{i}{2} \rfloor, \tag{4}$$

where f_i is the number of faces of $\tilde{H}[R]$ each with i singletons embedded inside. Let c_1 be the number of triangular faces of $\tilde{H}[R]$ with exactly one singleton embedded inside. As any such singleton is a cubic vertex,

$$c_1 \leq c. \tag{5}$$

Let d_1 be the number of non-triangular faces of $\tilde{H}[R]$ with one singleton inside, so that

$$f_1 = c_1 + d_1. \tag{6}$$

We would have $2r - 4$ triangular faces of $\tilde{H}[R]$ if it were triangulated. A non-triangulated face can be replaced by two or more triangulated faces by adding “fake" edges, and thus from which we deduce:

$$2r \geq c_1 + 2d_1 + f_3 + f_5, \tag{7}$$

and therefore

$$4r \geq 2c_1 + 4d_1 + 2f_3 + 2f_5 \geq 2c_1 + 3d_1 + 2f_3 + f_5. \tag{8}$$

Using the equation above, Equation (6), and Equation (5) we obtain:

$$\frac{4}{7}r \geq \frac{2}{7}c_1 + \frac{3}{7}d_1 + \frac{2}{7}f_3 + \frac{1}{7}f_5 = \frac{3}{7}f_1 + \frac{2}{7}f_3 + \frac{1}{7}f_5 - \frac{1}{7}c_1 \geq \frac{3}{7}f_1 + \frac{2}{7}f_3 + \frac{1}{7}f_5 - \frac{1}{7}c.$$

For $i \in \{0, 2, 4, 6, 7, 8, 9, \ldots\}$, we have $\lfloor i/2 \rfloor \geq \frac{3}{7}i$. Using this, the equation above, Equation (4), and $n = |V(\tilde{H})| = r + \sum_{i \geq 0} i \cdot f_i$, we obtain:

$$\Phi(\mathcal{P}, \mathcal{Q}) = (r-1) + \sum_{i \geq 0} f_i \lfloor \frac{i}{2} \rfloor \geq \frac{3}{7}(r + \sum_{i \geq 0} i \cdot f_i) - \frac{1}{7}c - 1 = \frac{3}{7}n - \frac{1}{7}c - 1,$$

or Equation (3). We are done providing intuition for the proof of Theorem 5.

Proof. (of Theorem 5) This long proof has two stages: first stage modifies (if needed) the double-partition $(\mathcal{P}, \mathcal{Q})$ so that it satisfies five conditions, and the second stage does the accounting using the five conditions. A number of definitions are needed before the first stage starts.

We use c to denote $c(\tilde{H})$ and Φ to denote $\Phi(\mathcal{P}, \mathcal{Q})$. Let $P_1, P_2, \ldots, P_h$ be the non-trivial vertex classes of $\mathcal{P}$, and let $p_i = |P_i|$.

Define $B_i := \tilde{H}[P_i]$, and let B_i's connected components be $B_i^1, B_i^2, \ldots, B_i^{j_i}$. Note that if a vertex class covers a facial triangle, then there exists an i with B_i containing an edge e of the triangle. If this happens we say that e *v-covers* the facial triangle.

Let B_i^j be some component of some B_i and let F be an inner face of B_i^j that contains at least one vertex of $\tilde{H}$. Let $L_i^j(F)$ be the border of F ($L_i^j(F)$ is a subgraph of B_i^j). Consider all the vertices v and edges e embedded in F but not embedded strictly inside any face F' of some $B_{i'}^{j'}$ such that F' is included in F. These vertices and edges plus $L_i^j(F)$ comprise a plane subgraph of $\tilde{H}$ that we call an *internal soup*. $L_i^j(F)$ is called the *main border* of the internal soup. See Figure 7 for an example. Call F (as a region of the plane) the *territory* of the internal soup. Define a *relevant facial triangle* of an internal soup to be a facial triangle whose three vertices belong to more than one vertex class (equivalently, these three vertices are not in the same B_i).

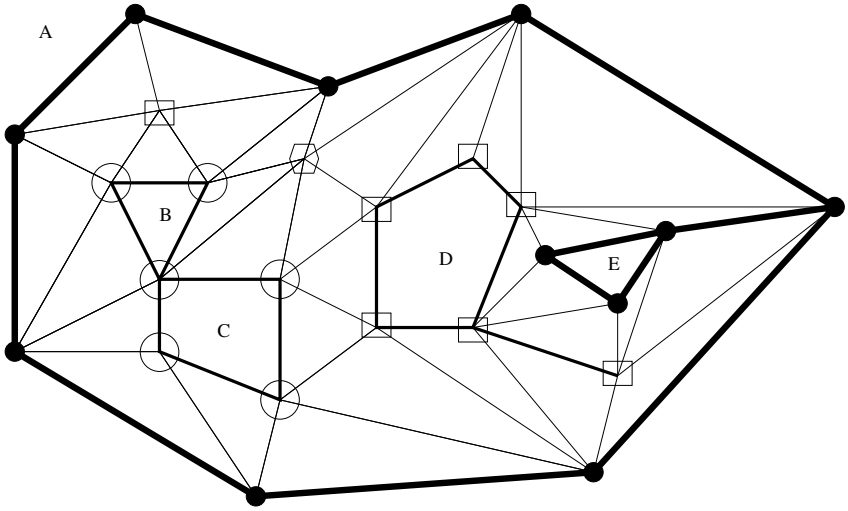


Figure 7 An example of an internal soup. The edges and vertices strictly inside the faces A, B, C, D, and E are not part of the soup. There are four vertex classes here, depicted by circles, squares, filled circles, and hexagons. Here, the vertex class depicted by squares is not connected. The main border of this soup is represented by very thick segments. Thick segments represent other edges with both endpoints in the same vertex class. Thin segments represent edges of the soup with endpoints in different vertex classes. All the depicted triangles except possibly B and E are facial triangles of $\tilde{H}$. The territory of this soup is the plane without the interior of the face A. There are 33 relevant facial triangles of this soup — they all have at least one thin edge. If B or E are facial triangles of $\tilde{H}$, they are not relevant facial triangles of the soup.

Let u, v, w be the three vertices on the border of the outer face of $\tilde{H}$. If u, v, w are not contained in the same B_i, for some i, we also have an *external soup* defined as above using as F (in the definition) the region of the plane that is the complement of the outer face of $\tilde{H}$. The external soup does not have a main border. The territory of the external soup is defined to be the region of the plane that is the complement of the outer face of $\tilde{H}$. See Figure 8 for an example.

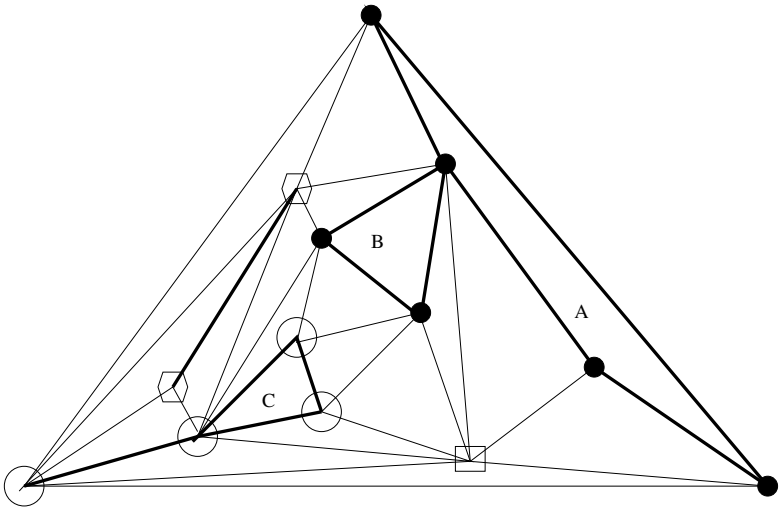


Figure 8 An example of an external soup. Here F is the region of the plane that is the complement of the outer face of $\tilde{H}$ (unlike internal soups, and despite its name, here F is not a face of some B_i^j). The edges and vertices strictly inside the faces A, B, and C, are not part of the soup. There are four vertex classes here, depicted by circles, squares, filled circles, and hexagons. Thick segments represent edges with both endpoints in the same vertex class. Thin segments represent edges of the soup with endpoints in different vertex classes. All the depicted triangles except possibly B and C are facial triangles of $\tilde{H}$. The territory of this soup is F. There are 18 relevant facial triangles of this soup — they all have at least one thin edge and include the facial triangle of the outer face of $\tilde{H}$. If B or C are facial triangles of $\tilde{H}$, they are not relevant facial triangles of the soup.

Note that every facial triangle of $\tilde{H}$ is either a facial triangle of some soup, or has all its three edges in some B_i (and vertex class P_i v-covers the facial triangle).

We say that a component B_i^j *participates* in a soup D if B_i^j does not contain the main border of the soup D and at least one vertex of B_i^j is a vertex of the soup. Recall that a singleton is a vertex that belongs to a size 1 vertex class. Singletons that are part of the soup also *participate* in the soup. For example, in Figure 7, we have three components and a singleton that participate in the depicted soup (this is since the vertex class represented by squares has two components that participate in the soup).

Later, the proof requires that the double-partition $(\mathcal{Q}, \mathcal{P})$ satisfies the following **five conditions** (goals):

1. Any edge of $\tilde{H}$ such that both facial triangles that contain this edge are covered by vertex classes is in an edge color by itself. In particular, any edge that is incident to two vertices in the same vertex class is in an edge color by itself;
2. At most one edge color covers facial triangles in a soup;
3. Each B_i is connected (recall that $B_i = \tilde{H}[P_i]$, where P_i is a vertex class of $\mathcal{P}$);
4. An edge color covers facial triangles in at most one soup;
5. There is no i with $|P_i| = 2$.

We start the **first stage** now, working towards these goals. All the relevant facial triangles in a soup must be covered by double-partition $(\mathcal{P}, \mathcal{Q})$, so any relevant facial triangle is covered by either one vertex class of $\mathcal{P}$ or by an edge color of $\mathcal{Q}$. Apply Modification 1 to $(\mathcal{P}, \mathcal{Q})$, so that any edge of $\tilde{H}$ such that both facial triangles that contain this edge are covered by vertex classes is put into an edge color by itself, achieving our *first goal*. In particular, any edge of $\tilde{H}$ connecting two vertices in the same vertex class is put into an edge color by itself.

Now every relevant facial triangle of every soup is covered either by a vertex class or by an edge color but not both.

Next we modify, if needed, $(\mathcal{P}, \mathcal{Q})$, so that at most one edge color is used for covering all the relevant facial triangles in a soup that are not covered by a vertex class. If this is not the case, we use Modification 2, as described below. That we can do this relies on topological arguments. Let $L_i^j(D)$ be the outer border of some component B_i^j that participates in a soup D. As we walk clockwise on $L_i^j(D)$, we encounter relevant facial triangles of the soup, some with one edge in $L_i^j(D)$, and others with only one vertex in $L_i^j(D)$. These relevant facial triangles form the *moat* of B_i^j in D. See Figure 9 for an illustration. Similarly, we define the *moat* of a singleton w to consist of all the relevant facial triangles that contain w. A moat of a component B_i^j (or a singleton) that participates in the external soup can include the outer face of $\tilde{H}$.

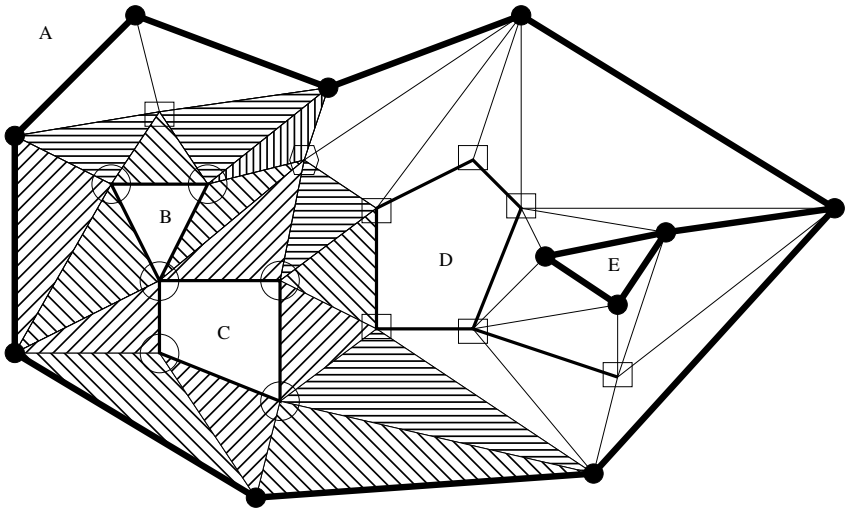


Figure 9 An example of a moat, inside the soup represented in Figure 7. The facial triangles (and corresponding triangular faces) with patterns illustrate the moat of the connected component of the vertex class whose vertices are marked by empty circles. The pattern is diagonal if the facial triangle is covered by a vertex class, and horizontal or vertical if the facial triangle must be covered by an edge color.

If on the outer border of a moat in an internal soup D we only have vertices from the main border of D, one can easily check that all the soup D consists of this moat, and all the relevant facial triangles in the soup D are covered by vertex classes. For an external soup D, if on the outer border of a moat we only have vertices from some a single $L_{i'}^{j'}(D)$ or a single singleton, one can easily check that all the soup D consists of this moat, and all the relevant facial triangles in the soup D are covered by vertex classes. If not, as we walk on the outer border of the moat, when we switch from some $L_{i'}^{j'}(D)$ to another (or to a singleton or to the main border, or from a singleton to another or to some $L_{i'}^{j'}(D)$ or to the main border, or from the main border to a singleton or to some $L_{i'}^{j'}(D)$), we have a relevant facial triangle T_1 that has its three vertices in three different vertex classes and therefore must be covered by an edge color. And later in the walk we have another facial triangle T_2 that must be covered by an edge color (when we switch the vertex class outside the moat, as in Figure 9). If there are two distinct edge colors Q_1 and Q_2 of $\mathcal{Q}$ covering T_1 and T_2, then there are two distinct vertex classes incident to both Q_1 and Q_2 and we can apply Modification 2. Continue around this moat and other moats in the soup D, until we get that all the relevant facial triangles in

the soup D that are not covered by vertex classes are covered by the same edge color. This continuation works since $\tilde{H}$ is triangulated and thus all the moats of a soup are "connected" as they overlap. More precisely, we treat two moats as being adjacent if they share a relevant facial triangle, and then one can easily check that the space of moats of a soup is connected, allowing us to obtain that every relevant facial triangle in the soup that is covered by an edge color is covered by the same edge color.

The changes done here do not invalidate the first goal, and do not change any of the soups. Thus from now on $(\mathcal{P}, \mathcal{Q})$ satisfies that at most one edge color is used for covering all the relevant facial triangles in a soup not covered by vertex classes, achieving our *second goal*.

Our *third goal* is to modify the double-partition $(\mathcal{P}, \mathcal{Q})$ to obtain that each B_i is connected. We also want that each non-trivial edge color covers relevant facial triangles in only one soup (*fourth goal*). Recall that P_i's are the parts of $\mathcal{P}$ and that $B_i := \tilde{H}[P_i]$, and B_i's connected components are $B_i^1, B_i^2, \ldots, B_i^{j_i}$.

The soups of $\tilde{H}$ can be arranged to be the vertices of a forest $\mathcal{F}$, by containment of their territories: if the territory of soup D is inside the territory of soup D', then D is a descendant of D'. If an external soup exists, $\mathcal{F}$ is a tree with the external soup as the root. Leaves of $\mathcal{F}$ are the soups whose territory does not contain the territory of any other soup. We traverse $\mathcal{F}$ in postorder, keeping the **invariant** that, when processing soup D, all the soups D' descendants of D satisfy the following property: All the B_i^j embedded in the territory of D', except possibly for the B_i^j that contains the main border of D', are in fact a B_i (in other words, B_i is connected), the edge color that covers relevant facial triangles of D', if any, has all of its edges in D', and the first two goals are also achieved. We now work on D to obtain the same properties and maintain the invariant during this postorder proposal.

As above, $L_i^j(D)$ denotes the border of some B_i^j as it participates in a soup D (by the definition of "participate", $L_i^j(D)$ is not the main border of D). We have (from the second goal) that at most one edge color covers relevant facial triangles in the soup D. If B_i has some components other than B_i^j, we modify $\mathcal{P}$ by splitting the vertex class P_i into two vertex classes, one consisting of the vertices of $V(B_i^j)$ and the other consisting of $P_i \setminus V(B_i^j)$. All the facial triangles are still covered by the new double-partition, since a facial triangle T covered before the modification by the vertex class P_i must be v-covered before the modification by an edge of B_i and this edge would be in either B_i^j or in what is left from B_i after removing $V(B_i^j)$ from B_i, thus v-covering after the modification the facial triangle T. We claim that Φ does not go up: the contribution of $\mathcal{P}$ decreases by 1, while the total contribution of edge colors increases by at most 1, since trivial edge colors contribute 0 to Φ, and only the (single) non-trivial edge color, which we call Q_l, that contains edges in the soup D can have its u_l incremented (recall that u_l is the number of vertex classes incident to Q_l). Indeed, other non-trivial edge colors that are incident to the new vertex class that consists of $V(B_i^j)$ can only cover relevant facial triangles of soups with territory included in interior faces of B_i^j, and these edge colors are not incident to any $B_i^{j'}$, for $j' \neq j$, from the invariant. After we apply this step to all the B_i^j participating in the soup D, except for the main border of the soup D, we do have that all such B_i^j are in fact a B_i.

If there exists an edge color Q covering relevant facial triangles of D and Q also covers facial triangles elsewhere, apply Modification 3 and split Q into Q' and $\bar{Q}$, where Q' consist of the edges of Q inside the soup D. Note that the edges of $\bar{Q}$ are embedded outside the territory of D, from the invariant. All the facial triangles covered by Q are now covered by either Q' or $\bar{Q}$; this is since the facial triangles that have one edge in the main border of D are not covered by an edge color (from the first goal). Also, there is at most one vertex class, (specifically, the P_i that contains the vertices of the main border of D) that is incident to

both Q' and $\bar{Q}$, since the edges of $\bar{Q}$ are embedded outside the territory of D and all the vertex classes $P_{i'}$ that have vertices inside the territory of D, except for the P_i that contains the vertices of the main border of D, form a connected $B_{i'}$ (from the invariant) and hence $P_{i'}$ cannot have vertices outside the territory of D.

Note that none of the changes invalidate the first two goals, or modify the soups. By now, we achieved our first four goals: any edge of $\tilde{H}$ such that both facial triangles that contain this edge are covered by vertex classes is in an edge color by itself (in particular any edge of $\tilde{H}$ connecting two vertices in the same vertex class is in an edge color by itself), at most one edge color covers facial triangles in a soup, double-partition $(\mathcal{P}, \mathcal{Q})$ has each B_i connected, and each non-trivial edge color covers facial triangles in only one soup. Moreover, the facial triangles in a soup that are not covered by vertex classes, if they exist, are all covered by one single edge color.

We further modify the double-partition $(\mathcal{P}, \mathcal{Q})$, to have that $|P_i| \neq 2$, for all i. Let us look at one P_i with $|P_i| = 2$. The graph B_i is connected by now, so let us look at the soup D where B_i participates (B_i cannot contain the main border of a soup), and at the moat around B_i. In a first case, the outer border of this moat is the main border of the soup D. Then all facial triangles of the soup D are covered by vertex classes. We split the vertex class P_i into two singleton vertex classes, and put all the edges of D into a new edge color thus covering the two facial triangles that B_i used to cover. Then we apply Modification 1. The contribution of $\mathcal{P}$ to Φ decreased by 1, while the contribution of this new edge color is exactly 1 (it is incident to exactly three vertex classes). Note that this modification does not destroy any of our first four goals. In a second case, the moat around B_i has two or more vertex classes on its outer border. Then there is one edge color that covers facial triangles in the soup D. In particular, this edge color is incident to P_i as it covers at least one facial triangle with (exactly) one vertex in P_i. Once again split the vertex class P_i into two singleton vertex classes, and put all the edges incident to P_i in the edge color that covers facial triangles in the soup D, thus covering the two facial triangles that B_i used to cover. Then we apply Modification 1. Once again, the contribution of $\mathcal{P}$ to Φ decreased by 1, while in this second case the contribution of this edge color covering relevant facial triangles in D increased by at most 1, as it is incident to one more vertex class. The contribution of other edge colors does not change. Note that this modification does not destroy any of our first four goals and does not modify the soups.

We have finished the first stage of the proof, and by now the double-partition $(\mathcal{Q}, \mathcal{P})$ satisfies the five conditions.

We continue with the **second stage** of the proof. Fix D to be a soup which meets g vertex classes. These g vertex classes give raise to the components or singletons that participate in the soup, and to the main border of the soup if the soup is an internal soup. Each of these vertex classes is connected in $\tilde{H}$ and also its intersection with the soup is connected.

▷ Claim 6. Exactly $2g - 4$ relevant facial triangles of the soup D are not covered by vertex classes.

Proof. Suppose a vertex class P_i has $L_i(D)$ as the border which is part of the soup D; $V(L_i(D))$ belong to P_i. Let S_D be the set of singletons participating in soup D. Let INS be the set of indices of non-singleton vertex classes that meet D (non-trivial vertex classes that participate in D, or that give the main border of D), and let $l_i = |V(L_i(D))|$ when $i \in INS$. Let b_i be the number of blocks of such an $L_i(D)$.

As $l_i \geq 2$, the edges of $L_i(D)$ v-cover exactly $l_i + b_i - 1$ relevant facial triangles of the soup D, as the walk around $L_i(D)$ has length exactly $l_i + b_i - 1$. A relevant triangle of soup D that is covered by a vertex class is v-covered exactly once during these walks around all the $L_i(D)$.

The total number of relevant facial triangles in the soup D is $2(S_D + \sum_{i \in INS} l_i) - 4 - \sum_{i \in INS}(l_i - b_i - 1)$, which is obtained as described in this paragraph. Construct a plane graph H' as follows. Start with H' being the soup D (which is a plane graph). Then all the relevant facial triangles of D are faces of H'. Take a B_i (given by the non-trivial vertex class P_i) that participates in the soup. Keep only $L_i(D)$ from B_i and triangulate every inner face of $L_i(D)$. This way, for B_i, we get exactly $l_i - b_i - 1$ triangular faces of H' that are not the faces of relevant facial triangles of D. Assume now that D has a main border and is given by vertices of B_i; let F be the face of B_i used in the definition of soup D. Triangulate all the inner faces of $L_i(D)$ except for F, and triangulate the outer face of $L_i(D)$. Again, for B_i, we get exactly $l_i - b_i - 1$ triangular faces of H' that are not the faces of relevant facial triangles of D. Now H' is a triangulated graph with $2(S_D + \sum_{i \in INS} l_i)$ vertices, and has exactly $2(S_D + \sum_{i \in INS} l_i) - 4$ triangular faces. These are all faces of relevant facial triangles of D except for the $\sum_{i \in INS}(l_i - b_i - 1)$ faces that we described above.

Thus the number of relevant facial triangles of D that are not covered by vertex classes is exactly:

$$2(S_D + \sum_{i \in INS} l_i) - 4 - \sum_{i \in INS}(l_i - b_i - 1) - \sum_{i \in INS}(l_i + b_i - 1) = 2(S_D + |INS|) - 4 = 2g - 4.$$

This concludes the proof of the claim. ■

Note that when $g = 2$, this number is 0 (which we have seen before, when we looked at moats whose outside border is included in one vertex class). When $g > 2$, there are relevant facial triangles in D that are not covered by vertex classes, and thus must be covered by an edge color Q_j (just one color, from our second condition). Condition 4 of $(\mathcal{P}, \mathcal{Q})$ implies that edge color Q_j only covers relevant facial triangles of D, and in fact we can assume edge color Q_j only contains edges of D. Thus Q_j only covers facial triangles of D; as the facial triangles of D that are not relevant (if any) are covered by a vertex class, we conclude that Q_j covers a set of facial triangles that is exactly the set of relevant facial triangles of D not covered by a vertex class. Recall that u_j is the number of vertex classes incident to edge color j. We have that $u_j = g$, based on earlier arguments when walking around moats with the space of moats of a soup being "connected". One can easily check that $2g - 4 = 2u_j - 4 \leq 4\lfloor \frac{u_j - 1}{2} \rfloor$, and therefore

$$\mathit{coverage}(j) = 2g - 4 \leq 4\lfloor \frac{u_j - 1}{2} \rfloor, \tag{9}$$

where $\mathit{coverage}(j)$ is the number of facial triangles covered by edge color Q_j. Using Condition 1 above, every facial triangle covered by an edge color is a relevant triangle in some soup and thus, using Condition 4, the equation above holds for all edge colors.

We have $2n - 4$ facial triangles that need to be covered by double-partition $(\mathcal{P}, \mathcal{Q})$ in $\tilde{H}$. Define $\lambda_i = 3p_i - 6 - E(B_i)$, the number of edges missing from B_i to be triangulated (we used here that $p_i = |V(B_i)| \geq 3$, from Condition 5). Each edge of B_i can v-cover at most two facial triangles of $\tilde{H}$. But there is an exception: if three edges of B_i form a facial triangle of $\tilde{H}$, then we must subtract 2 (as this facial triangle is covered three times when we computed above). Let $\bar{f}_i^3$ be the number of faces of B_i that are faces of $\tilde{H}$. We obtain that the vertex class $V(B_i)$ can cover at most $2(3p_i - 6 - \lambda_i - \bar{f}_i^3)$ facial triangles. Based on this discussion we have

$$2n - 4 \leq 2\sum_{i=1}^{h}(3p_i - 6 - \lambda_i - \bar{f}_i^3) + \sum_{j=1}^{q} coverage(j), \tag{10}$$

where as before h is the number of non-trivial vertex classes of $\mathcal{P}$ and q is the number of edge colors of $\mathcal{Q}$.

Let $p = \sum_{i=1}^{h} p_i$, $\bar{f} = \sum_{i=1}^{h} \bar{f}_i^3$, and $\lambda = \sum_{i=1}^{h} \lambda_i$. Let c' be the number of triangular faces of all B_i that contain exactly one singleton (of the double-partition) and no other vertex of $\tilde{H}$. This singleton is a cubic vertex of $\tilde{H}$. Let d be the number of triangular faces of all B_i that contain exactly two singletons, and no other vertex of $\tilde{H}$. Let s be the number of singletons of $\mathcal{P}$ that are embedded in a triangular face of some B_i, where all the vertices of $\tilde{H}$ contained in this face are singletons, and there are at least three of them. Let s' be the number of the other singletons of $\mathcal{P}$, not as above. Then $n = p + c' + 2d + s + s'$. From Equation (10) we deduce:

$$2(p + c' + 2d + s + s') - 4 \leq 6p - 12h - 2\lambda - 2\bar{f} + \sum_{j=1}^{q} coverage(j),$$

from which we deduce:

$$2s + 2c' + 4d + 2s' + 8h + 2\lambda + 2\bar{f} - 4 \leq 4p - 4h + \sum_{j=1}^{q} coverage(j).$$

Recall that k is the number of parts (vertex classes) of $\mathcal{P}$, h is the number of non-trivial vertex classes of $\mathcal{P}$, and p is the total number of vertices in non-trivial vertex classes. The contribution of of $\mathcal{P}$ to $\Phi(\mathcal{P}, \mathcal{Q})$ is $n - k = \sum_{i=1}^{h}(p_i - 1) = p - h$. Thus, using Equation (9),

$$\frac{1}{2}s + \frac{1}{2}c' + d + \frac{1}{2}s' + 2h + \frac{1}{2}\lambda + \frac{1}{2}\bar{f} - 1 \leq \Phi(\mathcal{P}, \mathcal{Q}). \tag{11}$$

Recall that each B_i is a connected plane graph with $|V(B_i)| = p_i \geq 3$. If it were triangulated, B_i would have exactly $2p_i - 4$ triangular faces. B_i is missing λ_i edges, and each of these can destroy at most two triangular faces. So we have at least $2p_i - 2\lambda_i - 5$ inner triangular faces of B_i. If F, one of these inner triangular faces of B_i, contains some B_j, then we call F a *green* face. For a green face F of B_i, there exists a $j \neq i$, with B_j contained in F and such that there is no $k \notin \{i, j\}$ with B_k contained in F and B_j in an inner face of B_k. We say that B_j *compels* F to become a green face and we note that B_j can only compel one face to become green. Thus there are at most h green faces. Let r_i be the number of non-green inner triangular faces of B_i that are not empty of points of $\tilde{H}$ and let $r = \sum_{i=1}^{h} r_i$. Based on the discussion above, we have:

$$r \geq 2p - 2\lambda - 5h - \bar{f} - h = 2p - 6h - 2\lambda - \bar{f}. \tag{12}$$

From the definition of s,

$$s \geq 3(r - c' - d)$$

as triangular faces of some B_i that are non-empty and non-green contain either one singleton (there are at most c' such faces), or two singletons (there are at most d such faces) or contain three or more singletons. From this and Equation (12) we obtain:

$$s + 3c' + 3d \geq 3(2p - 6h - 2\lambda - \bar{f})$$

and therefore:

$$7s + 7c' + 14d + 28h + 7\lambda + 7\bar{f} \geq 6p + 6s + 6c' + 12d - 2(c' + d) + 10h,$$

which implies, as $n = p + s + c' + 2d + s'$, that

$$\frac{1}{2}s + \frac{1}{2}c' + d + \frac{1}{2}s' + 2h + \frac{1}{2}\lambda + \frac{1}{2}\bar{f} - 1 \geq \frac{3}{7}n - \frac{1}{7}(c' + d) + \frac{5}{7}h - 1 \geq \frac{3}{7}n - \frac{1}{7}c - 1,$$

where for the last inequality we use the fact that $c \geq c' + d$, which follows from the fact that if two vertices are embedded in a triangular face, and together with the triangle form a triangulated planar graph, then one of these two vertices is cubic. This combined with Equation (11) implies Equation (3), finishing the proof of Theorem 5. ■

We continue the analysis of the approximation ratio of Algorithm MTLK4. As before, G denotes the input graph and H an optimum solution (a maximum planar subgraph of G). Let $\tilde{H}$ be a triangulated plane supergraph of H, and θ be $|E(\tilde{H})| - |E(H)|$. As above, the value of the optimum solution is $|E(H)| = 3n - 6 - \theta$. Below, c counts the number of cubic vertices in $\tilde{H}$.

From Theorem 5 and Proposition 2 we immediately obtain

$$\beta(G) \geq \beta(H) \geq \beta(\tilde{H}) - \theta \geq \frac{3}{7}n - \frac{1}{7}c - \theta - 1 \tag{13}$$

as one edge of $E(\tilde{H}) \setminus E(H)$ can be part of only one triangle of the maximum triangular cactus of $\tilde{H}$.

Let A be the kt-structure produced by MTLK4. Say that there is a *strong bond* between two vertices if there is a path between them in A with all edges in K_4's of A. Call a cubic vertex v of $\tilde{H}$ *blocked* if $K_4(v)$, the K_4 formed by v and its three neighbors in $\tilde{H}$, is not added to the kt-structure. A cubic vertex v is blocked because of one of the following:

- one of the edges of $K_4(v)$ is not in G (also not in H). We call v *absent*.
- all the edges of $K_4(v)$ are in G (and in H - since H is a maximum planar subgraph of G) and at least two of the four vertices of $K_4(v)$ are connected by a strong bond. We call v *neutralized*.
- all the edges of $K_4(v)$ are in G (and in H) and there are no strong bonds between any of the vertices of $K_4(v)$, and all four of the vertices of $K_4(v)$ are in the same connected component of A. We call v *subdued*.

Indeed, if all the edges of $K_4(v)$ are in G (and in H) and there is no strong bond between any two vertices of $K_4(v)$ and only three of its vertices are in the same component of A, then we could remove at most two triangles from this component of A to disconnect these three vertices, and add $K_4(v)$ while maintaining a kt-structure. Also, assuming there is no strong bond between any two vertices of $K_4(v)$: if two of the vertices of $K_4(v)$ are in one component of A and the other two in another component of A, then we can remove one triangle from each of these two components and add $K_4(v)$ while maintaining a kt-structure, and if the vertices of $K_4(v)$ are in three or four components of A, it is easy to remove just one triangle from A and add $K_4(v)$ while maintaining a kt-structure.

We use c_a to denote the number of absent cubics, c_s to denote the number of subdued cubics, and let c_n be the number of neutralized cubics. Let $A'_1, \ldots, A'_{q'}$ be the non-trivial sub-kt-structures obtained from A by removing all the triangles that are not part of K_4's. If we have j'_i K_4's in A'_i, then A'_i has $3j'_i + 1$ vertices, each two of them connected by a strong bond. All the strong bonds are obtained this way. Let γ be the number of K_4's in A. Then $\gamma = \sum_{i=1}^{q'} j'_i$.

Let B'_i be $\tilde{H}[V(A'_i)]$. Then B'_i has at most $3(1+3j'_i)-6 = 9j'_i - 3$ edges. For a cubic v to be neutralized, one of the four edges of $K_4(v)$ must be an edge of some B'_i. Recall than $n > 4$ and that $\tilde{H}$ is a triangulated plane graph. Then cubic vertices cannot be adjacent in $\tilde{H}$ and one can easily check that one edge e of $\tilde{H}$ can participate in at most two $K_4(u)$. Precisely, if one of the endpoints of e is a cubic, then this endpoint is the only possible u above, and if not, then the only possible u above are the two vertices that each forms a facial triangle with e.

Thus the total number of neutralized cubics is at most

$$c_n \leq 2\sum_{i=1}^{q'}(9j'_i - 3) \leq 18\gamma. \tag{14}$$

Also by the reasoning above,

$$c_a \leq 2\theta. \tag{15}$$

Let α be the number of triangles in A and recall that γ is the number of K_4's in A. Let A have non-trivial components $A_1, \ldots, A_q$, each with α_i triangles and γ_i K_4's. Let $B_i = \tilde{H}[V(A_i)]$. Thus $|V(B_i)| = |V(A_i)| = 1 + 2\alpha_i + 3\gamma_i$. Now we count how many subdued cubics $c_s(i)$ are in $V(A_i)$. Remove all these cubics from B_i and we get a planar graph D_i with $1 + 2\alpha_i + 3\gamma_i - c_s(i)$ vertices and where each subdued cubic vertex of A_i is the only vertex of $\tilde{H}$ embedded in a triangular face of D_i (recall that $n > 4$ and cubic vertices cannot be adjacent in $\tilde{H}$). We know that D_i can have at most $2(1 + 2\alpha_i + 3\gamma_i - c_s(i)) - 4$ triangular faces, and thus:

$$c_s(i) \leq 2(1 + 2\alpha_i + 3\gamma_i - c_s(i)) - 4$$

from which we deduce:

$$c_s \leq \frac{4}{3}\alpha + 2\gamma - \frac{2}{3}q$$

(recall that here q is the number of non-trivial components of A). As $\alpha \leq \beta(G)$ (recall that $\beta(G)$ is the number of triangles in a maximum triangular cactus of the graph G), adding the equation above to Equations (14) and (15) we obtain:

$$c \leq 20\gamma + \frac{4}{3}\beta(G) + 2\theta,$$

which we rewrite as

$$\gamma \geq -\frac{1}{15}\beta(G) - \frac{1}{10}\theta + \frac{1}{20}c. \tag{16}$$

Now we have all the ingredients to obtain our $((4/9)+\epsilon)$-approximation. As before, G denotes the input graph and H an optimum solution (a maximum planar subgraph of G). Let $\tilde{H}$ be a triangulated plane supergraph of H, and θ be $|E(\tilde{H})| - |E(H)|$. As above, the value of the optimum solution is $|E(H)| = 3n - 6 - \theta$. From Inequality (3) for H, we obtain:

$$\beta(G) \geq \beta(H) \geq \frac{1}{3}(n - 2 - \theta).$$

Multiply the inequality above by 141/166, Inequality (13) by 21/166, and Inequality (16) by 60/166 (we solved by hand a small linear program to obtain these numbers) and add them up to obtain:

$$\beta(G) + \gamma \geq \beta(G) + \frac{60}{166}\gamma \geq \frac{28}{83}n - \frac{37}{83}\theta - 1.$$

As every K_4 introduced by the algorithm gains at least one more edge compared to the one or two triangles it replaces (in other words, it increases the cyclomatic number by at leasy one), our output has at least $n - 1 + \beta(G) + \gamma$ edges. Based on the inequality above, one can check that

$$n - 1 + \beta(G) + \gamma \geq \frac{111}{83}n - \frac{37}{83}\theta - 2 \geq \frac{37}{83}(3n - \theta - 6).$$

In conclusion, we have:

▶ **Theorem 7.** *Algorithm MTLK4 is a* (37/83)*-approximation for MPS.*

The improvement $\epsilon = 1/747$.

4 Local Optimization without Max Triangular Cactus

Recall that a diamond subgraph is a graph that is isomorphic to the graph resulting from deleting any single edge from a K_4. We call the two vertices of degree three in a diamond as the bases of the diamond, and the two vertices of degree two as the tips of the diamond. We call the edge between the two bases of a diamond the base-edge of the diamond. A diamond of a plane graph is a *facial diamond* if the two triangles of the diamond are facial triangles.

In this section, we present **LDT**, a local optimization algorithm (inspired by the one of [6]) using only diamonds and triangles that is a (5/12)-approximation algorithm for MPS on general graphs. There is a tight example for this approximation ratio. The input is a graph G, which we assume to be connected as one can run any approximation algorithm on separate connected components if needed.

4.0.0.1 Algorithm LDT:

We maintain a spanning subgraph A (thus $V(A) = V(G)$) that is a dt-structure in G, starting with $E(A) = \emptyset$. The algorithm has two phases.

Phase I: The goal of this phase is to increase the cyclomatic number of A. For each connected component A' of A, the algorithm keeps a weighted tree $T_{A'}$ whose vertex set is $V(A')$ and edge set is as follows (see Figure 10 for an illustration). For every diamond of A, the edge connecting the bases is included in $T_{A'}$ with weight 2. For each of the two tips of every diamond, include in $T_{A'}$ an edge between this tip and one of the bases, with weight 1. For every triangle of A', include in $T_{A'}$ two of its edges, each with weight 1. One can easily check that $T_{A'}$ is a spanning tree of A'. Let $F(A)$ be the spanning forest on $V(G)$ obtained by taking all the edges of $E(T_{A'})$ for all the components A' of A.

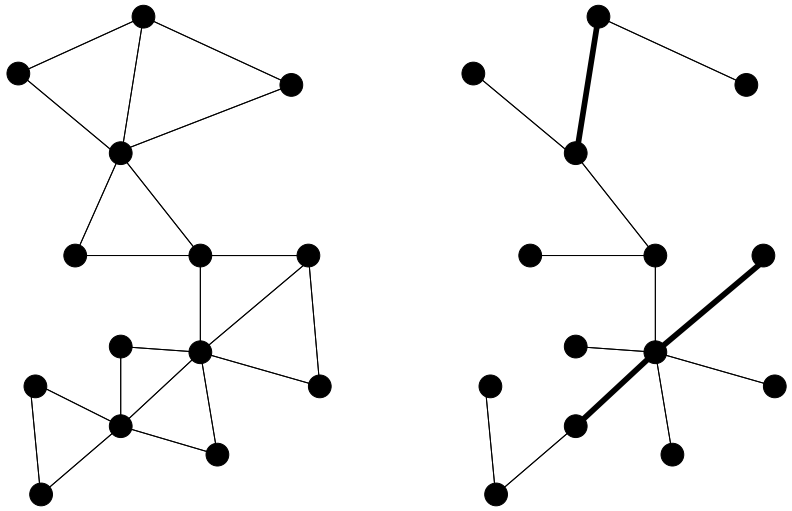


Figure 10 On the right, the tree $T_{A'}$ obtained from a connected component A' (depicted on the left) of A. Edges of weight 2 are thick

One local optimization step is obtained as follows: if there exist a triangle of G with its three vertices in three different components of A, then add this triangle to A, and resume. If no such triangle exists, go through all the diamonds D of G. Go through all the edges $e \in E(F(A))$ of weight 1. Let $F_e(A)$ be the spanning forest of G whose edge set is $E(F(A)) \setminus \{e\}$. If the four vertices of D are in four components of $F_e(A)$, and e was in $E(F(A))$ as an edge of a triangle, then remove from A this triangle and add D. If the four vertices of D are in four components of $F_e(A)$, and e was in $E(F(A))$ as an edge that connects a tip of a diamond D' to one of the bases of D', then remove from A the two edges connecting this tip from the bases (leaving a triangle from the diamond D') and add D.

Phase II: As long as possible, greedily add edges connecting various components in graph $(V(A), E(A))$ to get the output L (L is connected since the input G is connected).

See Figure 11 for an example of a local optimization step. One can easily check that applying the local optimization step of Phase I keeps A a dt-structure and its cyclomatic number indeed increases.

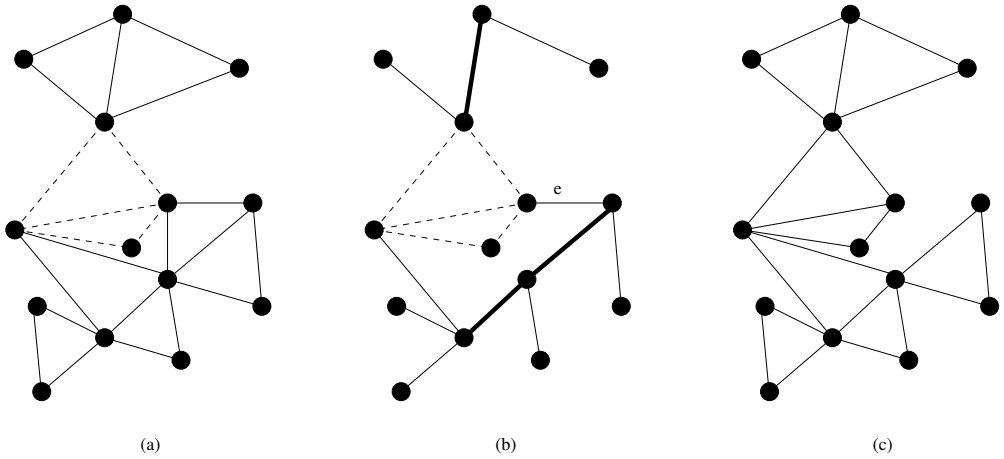


Figure 11 (a) shows in solid line segments the current dt-structure A. A diamond D, represented by the dashed line segments, is considered. (b) A forest $F(A)$ is represented by thick solid line segments having weight 2 and thin solid line segments having weight 1. We can see that by removing from $F(A)$ the weight-1 edge e, the four vertices of the diamond D are in four different components. (c) The resulting dt-structure after the local optimization step.

Running time We discuss the running time of one local improvement, or finding out that no such improvement exists. There are $O(n)$ edges in the current dt-structure A. Using Depth First Search, computing the connected components of the current dt-structure can be done in $O(n)$. There are $O(n^3)$ triangles in the input graph. Using the information given by the connected components of the current dt-structure, it takes constant time to check whether a triangle can be used for improving the dt-structure (this involves finding, for each vertex v of the triangle, the representative of the connected component containing v, and then checking if we have three distinct representatives). For the local optimization using diamonds, we use a more elaborate method to save a factor of n in the running time.

Let $\hat{F}(A)$ be the spanning forest on $V(G)$ obtained by taking all the edges of $E(T_{A'})$ that have weight 2, for all the components A' of A. We construct $\hat{F}(A)$ and, using Depth First Search, compute its connected components in time $O(n)$. Then we go through all the $O(n^4)$ diamonds and for each such diamond D we check if its four vertices belong to at least three connected components of A. If no, D cannot be used. If yes, we check if the two nodes that are in the same connected component of A are also in the same connected component of $\hat{F}(A)$; if yes, then D cannot be used. If no, in time $O(n)$ we can find one edge $e \in F(A) \setminus \hat{F}(A)$ such that the four vertices of D are in four different components of $F_e(A)$. In time $O(n^4)$ we can easily update A to include D.

There are $O(n)$ local improvement steps, for a total running time of $O(n^5)$.

Approximation Ratio Analysis First we show that the approximation ratio does not exceed 5/12 by more than an $o(1)$ term. Consider a large triangular grid H as in Figure 12, which occupies a regular hexagon, and such that H has an odd number of vertices, $2p + 1$, with a large p. Let S be some triangular cactus, with p triangles, and with $V(S) = V(H)$. We make S edge-disjoint from H, and also we avoid having the endpoints of an edge of S at distance less than 2 in H. One can easily check that H has $4p - o(p)$ faces. In "every second" face of H add a "new" point as in the figure, so that points are not added in two faces of H that share an edge in H. Connect each of these new points to the three vertices of H lying on the border of the face of H where the new point is added. The input graph G consists of the union of H and S and these new points with their incident new edges. Thus

G has $4p - o(p)$ vertices. The union of H and the new points induces an almost triangulated planar graph with $12p - o(p)$ edges. The triangular cactus S can be the solution produced by the Phase I of the algorithm, as we show in the next paragraph. Then our output has no diamonds and p triangles, for a total of at most $5p$ edges (the output has cyclomatic number of p and less than $4p$ vertices). By choosing p as large as we wish, we get a ratio as close to $5/12$ as we want.

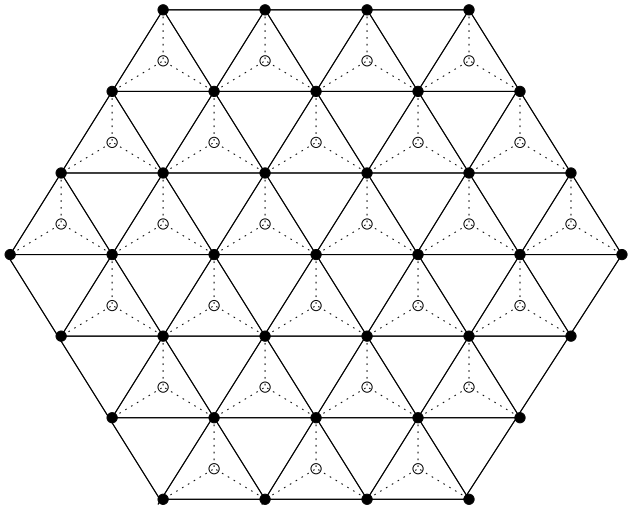


Figure 12 The planar graph H is given by filled circles representing the vertices and solid lines representing the edges. The new vertices are represented by empty circles, and the edges adjacent to these new vertices are represented by dashed lines.

It is easy to check that no triangle can be added to S. Now we check that no diamond can be swapped for a triangle in S according to Phase I of the algorithm. From our construction, any diamond of G has at least three vertices in $V(S)$, and for any edge e from $E(F(E(S)))$, the graph $F_e(E(S))$ will have one component with at least two of these three vertices.

We continue with the proof that the approximation ratio of LDT is at least $5/12$. Recall that G, the input graph, is connected and let OPT be an optimal MPS solution of G. Define $n := |V(G)| = |V(OPT)|$. We fix a planar embedding of OPT which will be used through out the section. Add to OPT a set Θ of edges to obtain a triangulated simple plane graph H. Let $\theta := |\Theta|$. A triangulated plane graph with n vertices has $3n - 6$ edges, if $n \geq 3$. So $\theta = (3n - 6) - |E(OPT)|$; θ does not depend on the embedding of OPT or H.

Let A be the dt-structure at the end of Phase I of the LDT algorithm. Let A have c non-trivial connected components $A_1, A_2, \ldots, A_c$. Let t_i, d_i be the number of triangles and diamonds in A_i respectively. This implies that the number of vertices n_i in the component A_i is exactly $3d_i + 2t_i + 1$ and the number of edges in A_i is exactly $5d_i + 3t_i$.

Let $d := \sum_{i=1}^{c} d_i$ and $t := \sum_{i=1}^{c} t_i$. We can also see that the output L has $(n-1)+2d+t$ edges, as every diamond adds two edges and every triangle adds one edge to a spanning tree (recall that the input graph G is connected and thus has a spanning tree).

First, we notice that if $c = 0$ (there are no triangles in G), then OPT has no triangle and it is a simple exercise that OPT has at most $2n - 2$ edges (this follows from Euler's formula and the fact that all the faces of an embedding of OPT have at least four edges on their boundary). As Phase II produces a graph with at least $n - 1$ edges, we have a $(1/2)$-approximation. Thus, from now on, we assume $c \geq 1$.

Second, we notice that, unless n is large compared to d and t, we get the desired approximation ratio. Precisely,

$$\frac{(n-1)+2d+t}{3n-6-\theta} \geq \frac{(n-1)+2d+t}{3n-6} \geq \frac{5}{12}$$

if $n \leq 8d+4t+6$. So from now on, we assume that:

$$n > 8d+4t+6. \tag{17}$$

Let $B_i := H[V(A_i)]$ be the subgraph of H induced by the vertices of A_i. Then we have $n_i = |V(B_i)|$. Let $m_i := |E(B_i)|$.

In T_{A_i}, defined in the description of Phase I of the LDT algorithm (also illustrated in Figure 10), remove the edges of weight 1 and we are left with several connected non-trivial components $J_i^1, J_i^2, \ldots J_i^{k_i}$, where $k_i > 0$ if $d_i > 0$ (if $d_i = 0$ there are no weight-2 edges in T_{A_i} and there are no non-trivial components above). The graphs $H[V(J_i^j)]$ are all planar and each can have at most $3|V(J_i^j)| - 5 = 3(|V(J_i^j)| - 1) - 2$ edges. Note that J_i^j uses $(|V(J_i^j)| - 1)$ weight-2 edges in T_{A_i}. We have d_i weight-2 edges in T_{A_i}. From this we obtain that the total number α_i of all edges of all the k_i graphs $H[V(J_i^j)]$ cannot exceed $3d_i - 2$:

$$\alpha_i \leq 3d_i - 2 \tag{18}$$

No border of a triangular face f of H can be added to A in **Phase I**, and based on this we classify all the triangular faces of H into one of the following three categories:

Category 1: at least one edge bordering f is in Θ (and the triangle given by the border of f does not exists in OPT; recall that Θ is the set of edges we have added to OPT to obtain the triangulated plane graph H).

Category 2: no edge bordering f is in Θ and exactly two vertices on the border of f are in the same $V(B_i)$.

Category 3: no edge bordering f is in Θ and all three vertices on the border of f are in the same $V(B_i)$.

There are $2n-4$ triangular faces of H and at most 2θ can fall in the first category above, as one edge of Θ only borders two triangular faces of H. Therefore at least $2n-4-2\theta$ triangular faces fall in either the second or the third category.

If an edge of B_i borders two triangular faces in the second category above, we call it a *strong* edge of B_i, and let S_i be the set of strong edges of B_i; also let $s_i = |S_i|$. The other edges of B_i are called *weak* edges. A weak edge can border at most one triangular face in the second category, and can border triangular faces in the third category. Think of this weak edge as contributing 1 to the Category 2 triangular faces and 1/3 to the Category 3 triangular faces. With these reasonings, we obtain:

$$\sum_{i=1}^{c}(2s_i + (4/3)(m_i - s_i)) \geq 2n - 4 - 2\theta,$$

which we rewrite as

$$\sum_{i=1}^{c}((2/3)s_i + (4/3)m_i) \geq 2n - 4 - 2\theta. \tag{19}$$

Define Γ_i to be the set of (strong) edges in S_i that are not in any of the k_i graphs $H[V(J_i^j)]$ mentioned above, and let $\Gamma = \cup_i \Gamma_i$. (This set Γ will give us technical issue complicating the proof.) Let $\gamma_i := |\Gamma_i|$, and note that $s_i \leq \alpha_i + \gamma_i$. The graph B_i does not have to be

triangulated; let $\lambda_i := 3n_i - 6 - m_i$ (λ_i is the number of edges missing from B_i to make a triangulated planar graph, noting that $n_i \geq 3$).

Then from $s_i \leq \alpha_i + \gamma_i$ and Inequality (19) we obtain

$$\sum_{i=1}^{c}((2/3)(\alpha_i + \gamma_i) + (4/3)(3n_i - 6 - \lambda_i)) \geq 2n - 4 - 2\theta,$$

which we rewrite as

$$\sum_{i=1}^{c}(\frac{2}{3}\alpha_i + 4n_i - 8) + \sum_{i=1}^{c}(\frac{2}{3}\gamma_i - \frac{4}{3}\lambda_i) \geq 2n - 4 - 2\theta. \tag{20}$$

▷ Claim 8.

$$\sum_{i=1}^{c}\gamma_i \leq 5c + 2\sum_{i=1}^{c}\lambda_i. \tag{21}$$

We believe that a stronger inequality holds, maybe without a 2 in front of the second summation; however this is enough for our purposes. We are unable to prove this without a long charging argument.

Proof. We first add some notation. The graphs B_i are considered plane graphs here. We add and embed a set of edges Λ_i to B_i to obtain a triangulated plane (simple) graph, and thus we will have $|\Lambda_i| = \lambda_i$. We do this carefully as follows. Let B_i^j, $j = 1, 2, \ldots j_i$ be the connected components of B_i, and let $b_i^j = |V(B_i^j)|$. Out of these connected components, let $\bar{j}_i$ be the number of the trivial components, with only one vertex each (also called *singletons*), and let $\hat{j}_i$ be the number of components with two vertices only; we call such a component a *doubleton*. If $b_i^j > 2$, let $B_i^j(l)$, for $l = 1, 2, \ldots l_i^j$, be the blocks of B_i^j, and let $b_i^j(l) = |V(B_i^j(l))|$. Note that $b_i^j(l) \geq 2$. Let $\bar{l}_i^j$ be the number of trivial blocks of B_i^j, meaning with two vertices (these trivial blocks are the *bridges* of the connected component B_i^j).

To construct Λ_i, we start by looking at non-trivial blocks $B_i^j(l)$, and add (and embed) a set of "diagonal" edges Δ_i in each face of each $B_i^j(l)$ until $B_i^j(l)$ becomes triangulated. See Figure 13 for an illustration. It is easy to check that this keeps the graph with vertex set $V(B_i)$ and edge set $E(B_i) \cup \Delta_i$ simple and plane; moreover each of this graph's non-trivial blocks is a triangulated plane graph.

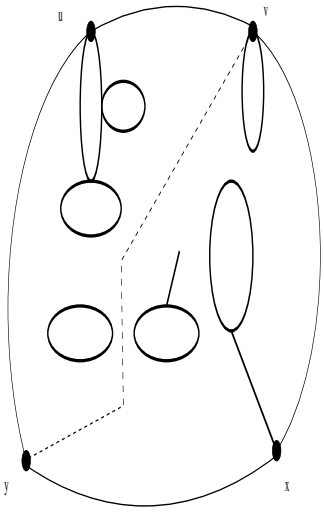


Figure 13 An illustration of adding edges in Δ_i. Here we have the bounded face with corners u, v, x, y of some $B_i^j(l)$. Other blocks or components of B_i that are embedded in this face are depicted by thick ovals. We can add and embed the diagonal edge vy, depicted by the dashed poly-line, while maintaining a plane embedding.

Next, as long as we have a trivial block (a bridge yx) sharing an endpoint x with a trivial block xz, we add the edge yz while maintaining planarity. This results in some triangulated blocks. Let Ω_i be the set of such edges added while doing this.

Next, if we have a trivial block (a bridge yx) sharing an endpoint x with a non-trivial block Q, and y is embedded in a face xuv of Q, we add the edges uy and vy resulting in a bigger planar block that contains y and the vertices of Q; see Figure 14 for an illustration. This bigger block is triangulated. Let Π_i be the edges added in this phase. The resulting graph is still planar.

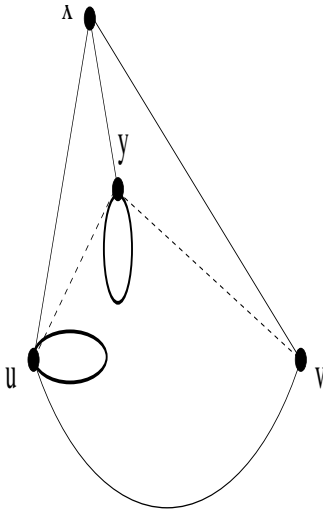
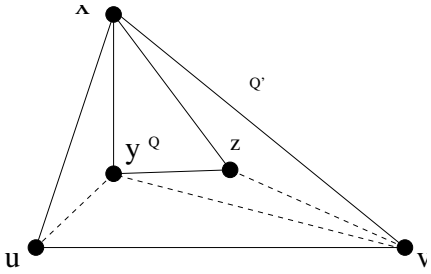
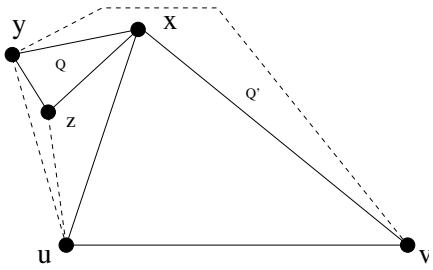


Figure 14 An illustration of adding edges in Π_i to merge the bridge xy into the block Q that has the vertices x, u, v such that y is embedded in a triangular face of Q. Here the starting edges are thin continuous segments or arc segments, and the new edges added to Π_i are dashed segments. We can do this even if some other blocks contain x or y or u or v. Two such block are depicted as the thick ovals. While in this figure y is embedded in an inner face of Q, the same idea can be used when y is embedded in the outer face of Q.

Next, we consider two non-trivial blocks that share one vertex x (an articulation point). All blocks except doubletons are triangulated by now. One block Q with outer face triangle xyz is embedded in a face xuv of the other block Q'. We add three edges to make $Q \cup Q'$ triangulated, as in Figure 15. We repeat this process, until all the blocks in the same component of B_i are merged together into one plane block; this is a triangulated plane graph except for doubletons. Let Υ_i be the set of such "reinforcing" edges added while doing this. (We could but do think of the edges of $Omega_i \cup \Pi_i$ as "reinforcing" a connected component since their goal in this proof is to eliminate the bridges).



An illustration of adding edges in Υ_i to merge a block Q with outer face xyz into the block Q', where y and z are embedded in an inner triangular face of Q' that has as corners the vertices x, u, v. Here the starting edges are thin continuous segments, and the new edges added to Υ_i are dashed segments.



An illustration of adding edges in Υ_i to merge a block Q with outer face xyz into the block Q', where y and z are embedded in the outer triangular face of Q' that has as corners the vertices x, u, v. Here the starting edges are thin continuous segments, and the new edges added to Υ_i are dashed segments.

Figure 15 Merging two non-trivial blocks

Thus, the graph with vertex set $V(B_i)$ and edge set $E(B_i) \cup \Delta_i \cup \Omega_i \cup \Pi_i \cup \Upsilon_i$ is planar; by now each of its connected components with at least three vertices is a triangulated plane graph. We have that the number of bridges of B_i that are not doubletons is at most

$$2|\Omega_i \cup \Pi_i| \tag{22}$$

as any such bridge disappears when we add $\Omega_i \cup \Pi_i$ and one edge of $\Omega_i \cup \Pi_i$ results in the disappearance of at most two such bridges. Precisely, an edge of Ω_i makes two bridges disappear (meaning that the edge that is a bridge does not form a bridge any more), two edges of Π_i make one bridge disappear.

Now we proceed to connect the graph G', where $V(G') = V(B_i)$ and initially $E(G') = E(B_i) \cup \Delta_i \cup \Omega_i \cup \Pi_i \cup \Upsilon_i$. First let us assume that G' has a component Q with at least three vertices. This component will "absorb" other components as follows. All the other components are embedded in faces of Q. Pick a component Q' so that Q' is not embedded in an interior face of any other component of G' other than possibly Q. If Q' is a singleton x and is embedded in a faces uvw of Q, add three edges xu, xv, and xw to G'; adjust Q to include vertex x and note that Q remains a triangulated plane graph. We are adding three edges per singleton. If Q' is a doubleton xy and is embedded in a (triangular) face of Q with corners u, v, w, add the five edges ux, vx, vy, wx, and wy to G'; adjust Q to include vertices x and y, and note that Q remains a triangulated plane graph. We are adding five edges per doubleton.

If Q' has at least three vertices, it is a triangulated plane graph. We have three cases: Q' is embedded in an inner face of Q, or Q is embedded in an inner face of Q', or each is embedded in the outer face of the other. The first two cases are symmetric and we only describe the first one below. Let x, y, z (clockwise order) be corners of the outer face of Q'. Here Q' is embedded in an inner face F' of Q; let u, v, w (clockwise order) be the corners of F'. Add to G' the six edges: ux, uy, vy, vz. wz, and wx. See the left side Figure 16 for an illustration. Adjust Q to included Q' and note that it is triangulated plane graph. In the third case, Q' is embedded in the outer face F' of Q. let u, v, w (clockwise order) be the corners of F'. Add to G' the six edges: ux, uy, wy, wz. vz, and vx. See the right side of Figure 16 for an illustration. Adjust Q to included Q' and note that it is triangulated plane graph.

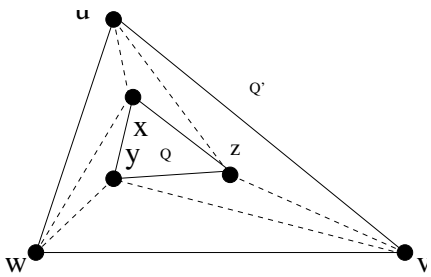
Keep doing this until G' is connected and triangulated; let Ψ_i be the set of edges added in this stage. Recall that j_i is the number of connected components of B_i, $\bar{j}_i$ is the number of singletons and $\hat{j}_i$ is the number of doubletons respectively. There are $(j_i - \bar{j}_i - \hat{j}_i)$ connected components of B_i that are not singletons nor doubletons. Thus:

$$|\Psi_i| = 3\bar{j}_i + 5\hat{j}_i + 6(j_i - \bar{j}_i - \hat{j}_i - 1) \tag{23}$$

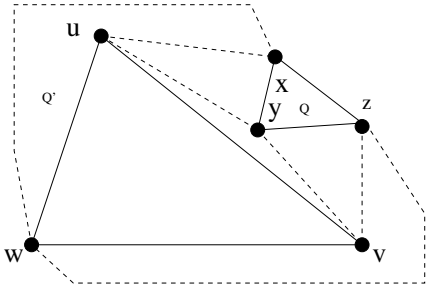
Second, assume that G' does not have components with at least three vertices. Then G' has $\bar{j}_i + 2\hat{j}_i$ vertices and $\hat{j}_i$ edges. Note that $|V(G')| = |V(B_i)| \geq 3$. A triangulated graph with this many vertices would have $3(\bar{j}_i + 2\hat{j}_i) - 6$ edges and therefore a set Ψ_i with exactly $3\bar{j}_i + 5\hat{j}_i - 6$ edges can be added to G' to make it a plane triangulated simple graph. Note that Equation (23) still holds.

Finally $\Lambda_i = \Psi_i \cup \Omega_i \cup \Pi_i \cup \Upsilon_i \cup \Delta_i$. Recall that $\lambda_i = |\Lambda_i|$.

Our credit scheme starts with credits equal to the RHS of Inequality (21). These credits are given by a procedure we describe later to each components of every B_i, bridges of every B_i, and edges of $\Delta_i \cup \Upsilon_i$ ("diagonals" and "reinforcing"). These components, bridges, diagonals, and reinforcing edges pay, as described later, for all the edges Γ so that every edge of Γ receives one credit, and nobody uses more credit than it was allocated.



An illustration of adding edges in Ψ_i to merge a connected component Q' with outer face xyz with a connected component Q, where Q' is embedded in an inner triangular face of Q that has as corners the vertices u, v, w. Here the starting edges are thin continuous segments, and the new edges added to Ψ_i are dashed segments.



An illustration of adding edges in Ψ_i to merge a connected component Q with outer face uvw with a connected component Q' with outer face xyz, when Q is embedded in the outer triangular face of Q' and vice-versa. Here the starting edges are thin continuous segments, and the new edges added to Ψ_i are dashed segments.

Figure 16 Merging two components of G'

Precisely, we give $5 + 2\lambda_i$ credits to B_i, further allocated as follows. Every edge of Λ_i has 2 credits. We keep 2 credits to every edge of Δ_i We give one credit on every bridge of B_i that is not a doubleton; these credits come from $\Omega_i \cup \Pi_i$.

We give 3 credits on every connected component of B_i, and another 1 credit one every doubleton of B_i. These credits come from the 5 allocated to B_i, and from the edges of Ψ_i. To check that we have enough credits, we look at how Ψ_i is constructed. If B_i has one component with at least three vertices, then we give this component 3 credits from the 5 we have. Whenever we add another component of B_i with at least three vertices, we give this component 3 credits, from the 6 edges we are putting in Ψ_i. Whenever we add another component of B_i with 2 vertices (a doubleton), we give this component 4 credits, from the 5 edges we are putting in Ψ_i. Whenever we add another component of B_i with 1 vertex (a singleton), we give this component 3 credits, from the 3 edges we are putting in Ψ_i. If B_i only has singletons and doubletons, we would need $3\bar{j}_i + 4\hat{j}_i$ credits, and we have two credits for every edge of Ψ_i, plus another 5 credits. In total, we have $5 + 2(3\bar{j}_i + 5\hat{j}_i - 6) = 3\bar{j}_i + 4\hat{j}_i + (3\bar{j}_i + 6\hat{j}_i - 7)$ credits, and this is enough since $3\bar{j}_i + 6\hat{j}_i \geq 9$ (recall that here $\bar{j}_i + 2\hat{j}_i = n_i \geq 3$).

We continue with the proof of Inequality (21). Consider an edge $xy \in \Gamma_i$ (for some i), and let us look at the facial diamond D of H which has xy as a base. We name u and v the tips of D. Note that u cannot be in $V(B_i)$ since no facial triangle adjacent to xy has all three vertices in $V(B_i)$, since xy is a strong edge. Also, all the edges of this diamond are in OPT and therefore in the input graph G; this follows from the fact that none of the edges of D are in Θ since, again, xy is strong. Same reasoning applies to show $v \notin V(B_i)$. And we must have the case that u and v are in the same $V(B_{i'})$, since otherwise the Phase I of the LDT algorithm will not stop yet, as argued next. Since $xy \in \Gamma_i$, x and y are not in the same $V(J_i^j)$, for some j. This means that there exists an edge e of T_{A_i} of weight 1 that can disconnect x from y in T_{A_i}. If u and v were not in the same $V(B_{i'})$, then u, v, x, y, the four vertices of D, would be in four components of $F_e(A)$ and another step of Phase I would be executed.

If the edge xy is a bridge of B_i, then we use one credit given to that bridge in the credits scheme to pay the edge; if this bridge is a doubleton, it still has three credits left. By now, the edges of Γ that are bridges in some B_i are all paid and every connected component of every B_i has at least three credits left. Also, every edge of each Δ_i has two credits left.

Assume now that xy is not a bridge of B_i and let F' be an interior face of B_i having xy as one of the edges bordering F'. We have that one of $\{u, v\}$ is embedded in F' and the other is embedded outside F', as xyv and xyu are the two facial triangles of H adjacent to edge xy. Let us assume that u is embedded in F'; then the whole component of $B_{i'}$ that contains u is embedded in F'. Moreover, there cannot be another inner face F" of some $B_{i"}$ such that the component of $B_{i'}$ that contains u is embedded in F" and some vertex w of this component has two neighbors x" and y" in $B_{i"}$ such that the there exists a triangular face of H with corners $w, x", y"$ (see Figure 17). Traversing the boundary of the face F' we meet a total of f' edges once and b' edges twice (these are bridge edges of B_i). Note that these bridge edges, if they are in Γ_i, they are paid one credit each. To triangulate F', we must use exactly $f' + 2b' - 3$ edges. As done by our procedure of constructing G', we have at most $2b'$ of these edges in $\Omega_i \cup Pi_i$ - indeed, if we use an edge in Ω_i to replace two bridges by a block that is a triangle, then we later use (three) edges from Υ_i merge this block into the larger block. And for the other bridge edges, we put exactly two edges in Π_i. This leaves us with at least $f' - 3$ edges of $\Delta_i \cup \Upsilon_i$ embedded inside F'.

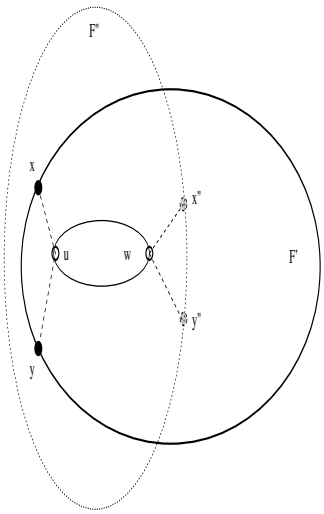


Figure 17 An illustration of the fact that F" cannot exist. The connected component of $B_{i'}$ that contains u is depicted by a an oval with thin solid borders. The dashed straight segments are edges of the input graph G.

We pay all the edge of Γ_i that appear on the boundary of F' and are not bridges by charging the component of $B_{i'}$ that contains u and the edges of $\Delta_i \cup \Upsilon_i$ embedded inside F'. Note that as argued above, $B_{i'}$ cannot be charged more than once this way, and thus it brings 3 credits. Also, the edges of $\Delta_i \cup \Upsilon_i$ embedded inside F' are not charged in any other place, so they bring $2(f' - 3)$ credits. Then we have at least f' credits to pay for all the edges of Γ_i that appear on F'. We do this and then every edge of every Γ_i is paid one credit. In conclusion, the claim holds. ■

Using Inequality (21) in Inequality (20) we obtain:

$$\sum_{i=1}^{c}\left(\frac{2}{3}\alpha_i + 4n_i - 8\right) \geq 2n - 4 - 2\theta - (10/3)c.$$

From this, the fact that $n_i = 1 + 3d_i + 2t_i$, and Inequality (18), we obtain:

$$\sum_{i=1}^{c}\left(\frac{2}{3}(3d_i - 2) + 4(1 + 3d_i + 2t_i) - 8\right) \geq 2n - 4 - 2\theta - (10/3)c,$$

which simplifies to

$$\sum_{i=1}^{c}\left(14d_i + 8t_i - \frac{16}{3}\right) \geq 2n - 4 - 2\theta - (10/3)c.$$

This in turn simplifies to:

$$14d + 8t - (16/3)c \geq 2n - 4 - 2\theta - (10/3)c,$$

and using the early assumption that $c \geq 1$, this implies

$$7d + 4t \geq n - \theta - 1. \tag{24}$$

From Inequality (17) we obtain:

$$2n \geq 11d + 8t,$$

which is equivalent to

$$5(n - 7d - 4t) + 24d + 12t \geq 3n.$$

Using Inequality (24), this implies:

$$5\theta + 24d + 12t \geq 3n - 18.$$

which is equivalent to

$$\frac{(n-1) + 2d + t}{3n - 6 - \theta} \geq \frac{5}{12}.$$

This completes the analysis of the algorithm LDT. We have:

▶ **Theorem 9.** *Algorithm LDT is an approximation algorithm with ratio of* 5/12*, and can be implemented to run in time* $O(n^5)$.

5 A Greedy Algorithm followed by LDT

In this section, we present Algorithm GDLDT (Greedy-Diamonds-Local-Diamonds-Triangles) that achieves a small improvement over Algorithm LDT from the previous section.

Algorithm GDLDT adds diamonds greedily as long as we can maintain a dt-structure[12] before we start Phase I of the LDT algorithm.

[12] As mentioned earlier, this is exactly the start of the GDT algorithm of [7].

We end up with $k \geq 0$ non-trivial connected diamond-structures[13] A_i each with d_i diamonds. A special case is $k = 0$; it will be discussed later. Recall that H is a triangulated plane graph that has all the edges of OPT and another set Δ of δ edges. We have $1 + 3d_i$ vertices in A_i or in $B_i := H[V(A_i)]$. B_i would have at most $9d_i - 3$ edges. We define $d := \sum_{i=1}^{k} d_i$. Also let $BB := \cup_{i \in \{1,2,\ldots,k\}} E(B_i)$.

H has $3n - 6$ facial diamonds - one for every edge. Call a pair of facial diamonds *interlocking* if the base of one is an edge of another. Then one facial diamond participates in four pairs of interlocking facial diamonds.

No facial diamond of H can be added when **Greedy Diamonds** stops, and based on this we classify all the facial diamonds of H into one of the following three categories:

Category I: One edge of the facial diamond is in Θ (recall that Θ is a set of edges that are in H but not OPT, and that $\theta = |\Theta|$).

Category II: Not of Category I, and one edge of the facial diamond is in BB.

Category III: Not of Categories I and II, and the two tips of the facial diamond are in the same $V(B_i)$, for some $i \in \{1, 2, \ldots, k\}$.

An edge e of Θ participates in at most 5 facial diamonds: one diamond D where the edge e is the base, and at most 4 facial diamonds where the base is one of the at most four edges other than e in the facial diamond that has e as a base. Thus there are at most 5Θ facial diamonds of Category I.

If $k = 0$ (the special case where no diamonds were found), then we only have facial diamonds of Category I. Then we have

$$5\theta \geq 3n - 6. \tag{25}$$

From Inequality (24) (here $d = 0$), we obtain:

$$4t \geq n - \theta - 1$$

and therefore

$$\frac{n - 1 + 2d + t}{3n - 6 - \theta} \geq \frac{(5/4)n - (5/4) - (1/4)\theta}{3n - 6 - \theta} \geq \frac{11}{24} > \frac{91}{216},$$

where the middle inequality above follows by simple algebra from Inequality (25).

From now on, we assume that $k > 0$. Similarly, an edge of BB participates in at most 5 facial diamonds. Thus there are at most $5|BB|$ facial diamonds of Category II.

Let us now consider the set of facial diamonds of Category III. If a facial diamond of Category III belongs to interlocking pair of facial diamonds with another facial diamond of Category III, then we put both these facial diamonds in a set Q. Let R be the set of facial diamonds of Category III that were not put in Q.

We construct a bipartite multigraph M with R as the vertices of one part, and $BB \cup \Theta$ as the vertices of the other part, and edge set defined as follows.

Let D be a facial diamond of R with base vertices x and y and tips u and v. Note that there is an i such that u and v are in $V(B_i)$ while x and y are not in $V(B_i)$. Consider the four distinct facial diamonds of H with bases xu, yu, xv, and yv. As we assumed D is not in Q, for each D' of these four facial diamonds, there must be an edge e of $\Theta \cup BB$ that makes D' ineligible for being added by the Greedy algorithm (D' is of Category I or II since if D' were to be of Category III, then D' and D would interlock). Put an edge in M between D

[13] A diamond structure is a graph whose blocks are all diamonds

and e and label this edge by D'. Do this for all the four diamonds interlocking with D. We may end up putting two edges between D and some edge e of $\Theta \cup BB$, labeled by two distinct diamonds that interlock with D and both contain e. See Figure 18 for an illustration. Note that e is not the base of the facial diamond D' when an edge of M incident to e (seen as a vertex of M) is labeled by D'.

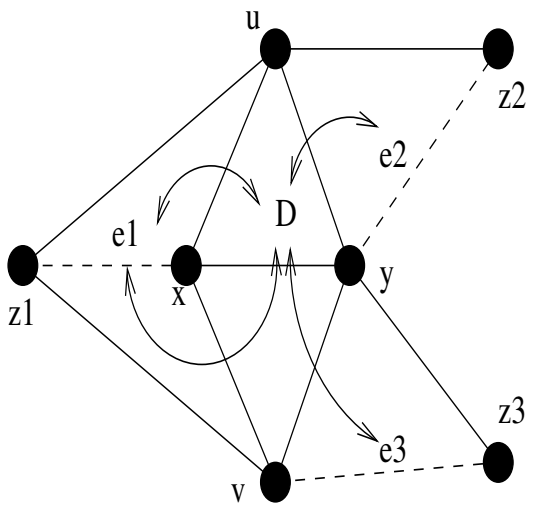


Figure 18 Straight segments, dashed or not, are edges of H. We have a diamond D that is in the set R, with base xy and tips u and v. There are four facial diamonds that interlock with D, and in this figure the facial diamonds with bases xu, xv, yu, and yv have tip set $\{z_1, y\}$, $\{z_1, y\}$, $\{z_2, x\}$, and $\{z_3, x\}$ respectively. Each of these four facial diamonds has one edge in $\Theta \cup BB$, which in this figure are the three dashed segments e_1, e_2, and e_3. Then in M we put two edges with endpoints e_1 and D, one edge with endpoints e_2 and D, and one edge with endpoints e_3 and D. These edges of M are represented by arcs with arrows on both ends.

Thus in M, every vertex that is a facial diamond of R has degree exactly four. An edge e of $\Theta \cup BB$, seen as a vertex of M, cannot gain more than one edge of M labeled by diamond D' (here D' must contain e), since e is not the base of D' and if two facial diamonds of R, say, D_1 and D_2 will get an edge to e in M labeled D', then D_1 and D_2 interlock (see Figure 19) and thus they are not in R. Thus a vertex of M that is an edge e of $\Theta \cup BB$ has degree at most four in M. By counting the degrees of the vertices of M, we conclude that:

$$|R| \leq |BB| + \theta. \tag{26}$$

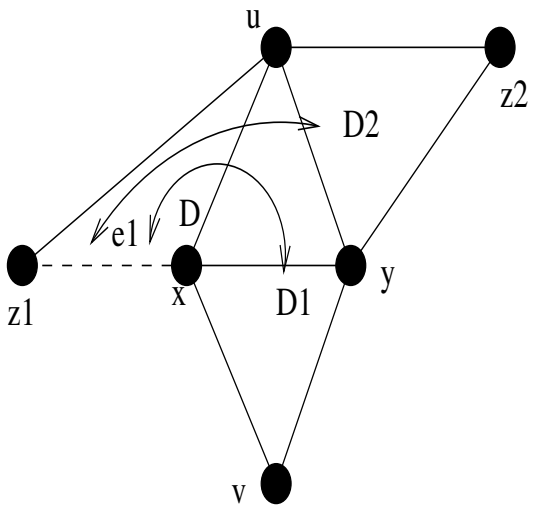


Figure 19 Here D is the facial diamond with bases x and y and tips u and v. Assume that D is a member of R. As we construct the graph M, we put an edge between D and e, and we label this edge D'. Here e is the edge between w and x, and D' is the facial diamond of H with bases x and u and tips y and w. The only other facial diamond that could have an edge of M labeled D' incident to e is D_2, but then D_2 and D_1 interlock and none of them would be a member of R.

Let $AA := E(H[\cup_{i\in\{1,2,\ldots,k\}}V(A_i)])$, where we recall that k is the number of non-trivial components A_i. Also recall that $BB = \cup_{i\in\{1,2,\ldots,k\}}E(B_i)$ and $B_i = H[V(A_i)]$; thus $BB \subseteq AA$. We will construct a bipartite graph M' with the left part being the edges of $AA \setminus BB$ and the right part being the diamonds of Q.

Consider now a facial diamond D_1 from Q. Let D_1 have base vertices x and y and tips u and v. Note that there is an i such that u and v are in $V(B_i)$ while x and y are not in $V(B_i)$. Since $D_1 \in Q$, there is a diamond D_2 of Category III that interlocks with D_1; wlog we assume that D_2 has base uy and tips x and z. See Figure 19 for an illustration, ignoring the vertex w and the diamond D'. Then there exists $j \neq i$ with vertices x and z in $V(B_j)$. Note then that ux and xv are edges of $AA \setminus BB$. We put in M' edges from D_1 to the edges ux and xv. Note that neither ux nor xv is the base edges of D_1. Thus every vertex in the right part of the bipartite graph M' has degree exactly 2. An edge of $AA \setminus BB$ appears as a non-base edge in four diamonds of H and thus, seen as a vertex in the left part of M', can have degree at most 4. We conclude:

$$|Q| \leq 2|AA \setminus BB| = 2|AA| - 2|BB|.$$

As every facial diamond of H belongs to one of the three categories, using the equation above and Equation (26), we conclude:

$$3n-6 \leq 5\theta+5|BB|+|R|+|Q| \leq 6\theta+6|BB|+2(|AA|-|BB|) = 6\theta+4|BB|+2|AA|. \quad (27)$$

Now, since B_i is a simple planar graph with $1+3d_i$ vertices, we have $|E(B_i)| \leq 9d_i - 3$. Note also that $|BB| = \sum_{i=1}^{k}|E(B_i)| \leq 9d-3k$. The set AA is the edge set of a planar graph with $\sum_{i=1}^{k}(1+3d_i) = k+3d$ vertices and therefore $|AA| \leq 3(k+3d)-6 = 9d+3k-6$. Using these inequalities and Inequality (27) we obtain:

$$3n - 6 \leq 6\theta + 4(9d-3k) + 2(9d+3k-6) = 6\theta + 54d - 12 - 6k.$$

This implies

$$d \geq \frac{1}{18}n - \frac{1}{9}\theta.$$

Adding this to Inequality (24), we obtain:

$$8d + 4t \geq \frac{19}{18}n - \frac{10}{9}\theta - 1,$$

which leads to

$$2d + t \geq \frac{19}{72}n - \frac{5}{18}\theta - \frac{1}{4}.$$

Using $\theta \geq 0$, one can verify that this implies:

$$\frac{n - 1 + 2d + t}{3n - 6 - \theta} \geq \frac{91}{216},$$

which is still far from 3/7 but is a little bit better than 5/12. In conclusion we have the first statement of:

▶ **Theorem 10.** *Algorithm GDLDT (greedily adding diamonds followed by our LDT algorithm) is an approximation algorithm with ratio of at least* 91/216*, and at most* 3/7*, and can be implemented to run in time* $O(n^5)$*.*

Regarding the running time, there are $O(n^4)$ diamonds and it is straightforward to use connected components to check whether a diamond can be greedily added in time $O(n^4)$. There are at most $O(n)$ such additions. Algorithm LDT can be implemented to run in time $O(n^5)$ 9. The second statement of the theorem is proved ahead.

5.1 Upper bound on the approximation ratio

We do not have a matching upper bound for the algorithm discussed in this section. However the ratio will not exceed $3/7 < 4/9$, as shown in the following example.

Consider a dt-structure D with d diamonds and no triangles; moreover the d diamonds are connected so that the edges between the bases of each diamond form a tree T. So T has $d + 1$ vertices; there are another $2d$ tips. Arrange the $2d$ tips as our graph H from the 5/12 example and, just like there, add a new vertex in every second face of H, making it adjacent to the three vertices of H bordering that face. Construct a triangulated planar graph J on the $d + 1$ vertices of $V(T)$. Then add a new vertex in every face of J, connecting it to the three vertices of J bordering the face. The dt-structure D together with H and J and the new vertices form the input graph G. There are $2d - o(d)$ new vertices in the faces of H and $2d - 2$ new vertices in the faces of J; so G has $7d - o(d)$ vertices. Graph H combined with its new vertices is planar and has $4d - o(d)$ vertices and $12d - o(d)$ edges, and J combined with its new vertices is planar and has $3d - 1$ vertices and $9d - 9$ edges, and thus a planar subgraph of G with $21d - o(d)$ edges exists.

Just like in the 5/12 series of examples, one can check that D cannot be improved by the local optimization step. Then our output will have at most $7d + 2d$ edges (two extra edges for each diamond). By choosing d as large as we wish, we get a ratio as close to $9/21 = 3/7$ as we want.

A variation of this example gives an upper bound of $11/27 < 5/12$ for the algorithm of [7] which greedily adds diamonds followed by greedily adding triangles: put a new vertex in all the faces of H, not just half of them. We end up with $6d - o(d)$ new vertices, for a total of $9d - o(d)$ vertices and a planar subgraph with $27d - o(d)$ edges. The output will have at most $9d + 2d$ edges as no triangle can be added to the d diamonds of D.

6 Conclusions and Discussion

We improved the approximation ratio for Maximum Planar Subgraph from 4/9 to $4/9 + \epsilon$, for a very small $\epsilon > 0$, by analyzing the application of a natural local optimization step after applying the previously known, Matroid-Parity based (4/9)-approximation algorithm. Our analysis, while involved, is not tight and there may be room for a more significant improvement here.

In particular, we do not believe that Equation (13) is tight. We conjecture that the tight (excluding a small additive term) bound that generalizes Theorem 5 is:

$$\Phi(\mathcal{P}, \mathcal{Q}) \geq \frac{3}{7}n - \frac{1}{7}c - \frac{10}{21}t \tag{28}$$

Here we are looking at double-partitions $(\mathcal{P}, \mathcal{Q})$ that cover all the facial triangles of a plane graph H with n vertices and t is the number of edges missing from H to be a triangulation. As before, c is the number of cubic vertices in H, but without H being triangulated, we provide a new definition of cubic, as follows. Say vertex v is adjacent to u, x, y forming a K_4 with v embedded strictly inside the triangle uxy. We call v cubic if none of the three triangles vux, vxy, vyu have a vertex z strictly inside that forms a K_4 with its neighbors. See Figure 20 for an example. This bound would be tight, as described next.

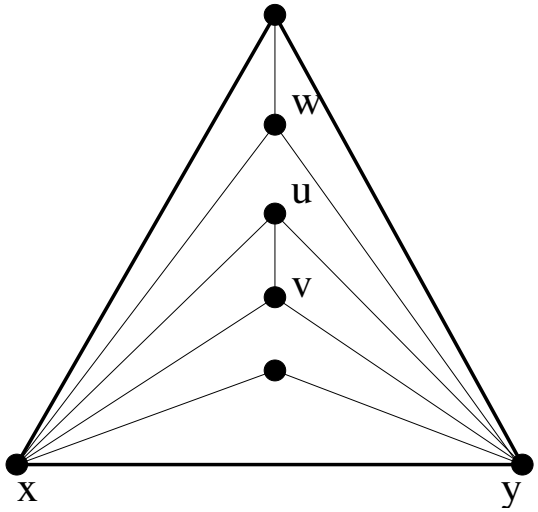


Figure 20 Here v is a cubic while w is not.

Consider a triangulation W on r vertices containing no K_4. Add four vertices in each face as in Figure 21. We have $n = r + 4(2r - 4) = 9r - 16$ and $t = 3(2r - 4) = 6r - 12$. There are no K_4's and therefore no cubic vertices in this graph. Choose $\mathcal{P}$ to have one non-trivial vertex class: W, and $\mathcal{Q}$ has no non-trivial edge color, so that $\Phi(\mathcal{P}, \mathcal{Q}) = r - 1$. One can verify that all the triangles are covered by double-partition $(\mathcal{P}, \mathcal{Q})$. Then, as r increases, Equation (28) becomes tight.

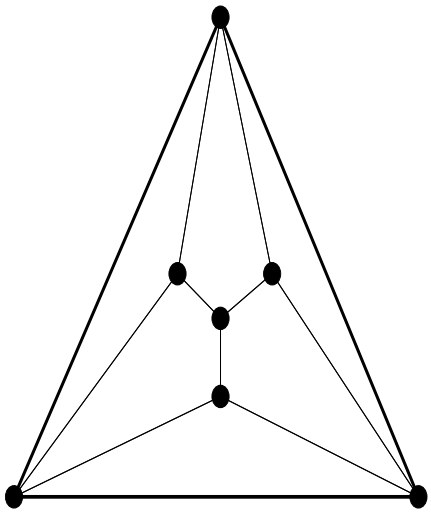


Figure 21 Faces used to show tightness of Equation (28).

Equation (28) would allow us to increase the ϵ to maybe $1/270$ with the current techniques.

We also presented a local optimization algorithm, LDT, that does not use Matroid Parity and has a tight approximation bound of $5/12$. Applying a greedy algorithm before starting this second local optimization gives a small improvement, to at least $91/216 = (5/12) + (1/216)$. Consider the following variant of our LDT algorithm: the algorithm keeps a dt-structure, and a local optimization step would either add a triangle, replace a triangle by a diamond, or replace a diamond by a diamond and a triangle. This variant has approximation ratio of at least $5/12$, as all the local steps LDT makes are done by the variant. Moreover, the series of examples showing that LDT has approximation ratio at most $5/12$ holds for this variant. And if we augment the variant to add diamonds before adding triangles, then we get a ratio between $91/216$ and $3/7$ as we did for GDLDT.

As in the previous work [5], we use the approach of using a few basic graphs for blocks in our output. This guarantees planarity and is at the basis of the analysis of all approximation algorithms for MPS with ratio bigger than $1/3$. As long as one uses this approach, this paper suggests that the application of local optimization, even if it comes before or after other algorithms, is beneficial, and more powerful than Greedy algorithms (albeit slower).

If we are to use blocks of fixed size, it is not hard to see and known since [3] that one cannot achieve an approximation better than $1/2$. But maybe the "spruces" of [6] can give us $1/2$ or better if used as the blocks of output.

One approach that does not use Matroid Parity, and we were not able to analyze beyond the $91/216$ bound presented earlier outputs a 4-structure by:

1. Use the approximation algorithm of [22] to get a $((2/3) - \epsilon)$-approximation for the maximum K_4-structure. Contract the selected $K_4's$.
2. Use again the approximation algorithm of [22] to get a $((2/3) - \epsilon)$-approximation for the maximum diamond-structure.
3. Use our LDT algorithm. Add to it three other local "new" optimizations steps:
   - If one can replace two triangles by a diamond and still have a 4-structure, do so;
   - If one can replace three triangles by a K_4 and still have a 4-structure, do so;
   - If one can replace one triangle and one diamond by a K_4 and still have a 4-structure, do so;

4. Contract the blocks and use once again the approximation algorithm of [22] to get a $(2/3) - \epsilon$ approximation for the maximum C_4-structure.
5. Expand everything contracted and connect the blocks with bridges to output a connected graph.

Maybe this or a variant of this can also beat the 4/9 ratio. The three new local optimization steps mentioned above do not increase the cyclomatic number but may be useful as they increase the number of connected components in our 4-structure.

An alternative would be to devise an approximation algorithm for Weighted Matroid 3-Parity, when the weights are 3 (from K_4's), 2 (from diamonds), and 1 (from triangles and C_4's). With such small integer weights, again local optimization seems the way to go.

References

1. Markus Blaeser, Gorav Jindal, and Anurag Pandey. A deterministic PTAS for the commutative rank of matrix spaces. *Theory of Computing*, 14:1–21, 01 2018. doi:10.4086/toc.2018.v014a003.
2. Jiazhen Cai, Xiaofeng Han, and Robert E Tarjan. An $O(m \log n)$-time algorithm for the maximal planar subgraph problem. *SIAM Journal on Computing*, 22(6):1142–1162, 1993.
3. Gruia Călinescu and Cristina G. Fernandes. On the k-structure ratio in planar and outerplanar graphs. *Discrete Mathematics & Theoretical Computer Science*, 10(3):135–148, 2008. URL: http://www.dmtcs.org/dmtcs-ojs/index.php/dmtcs/article/view/961/2387.
4. Gruia Călinescu and Cristina G. Fernandes. Maximum planar subgraph. In Teofilo F. Gonzalez, editor, *Handbook of Approximation Algorithms and Metaheuristics, Second Edition, Volume 2: Contemporary and Emerging Applications*. Chapman and Hall/CRC, 2018.
5. Gruia Călinescu, Cristina G. Fernandes, Ulrich Finkler, and Howard J. Karloff. Better approximation algorithm for finding planar subgraphs. *Journal of Algorithms*, 27(2):269–302, 1998.
6. Gruia Călinescu, Cristina G. Fernandes, Hemanshu Kaul, and Alexander Zelikovsky. Maximum series-parallel subgraph. *Algorithmica*, 63(1-2):137–157, 2012.
7. Parinya Chalermsook and Andreas Schmid. Finding triangles for maximum planar subgraphs. In Sheung-Hung Poon, Md. Saidur Rahman, and Hsu-Chun Yen, editors, *WALCOM: Algorithms and Computation*, pages 373–384, 2017.
8. Parinya Chalermsook, Andreas Schmid, and Sumedha Uniyal. A tight extremal bound on the Lovász cactus number in planar graphs. In *36th International Symposium on Theoretical Aspects of Computer Science*, 2019.
9. Ho Yee Cheung, Lap Chi Lau, and Kai Man Leung. Algebraic algorithms for linear matroid parity problems. *ACM Transactions on Algorithms (TALG)*, 10(3):1–26, 2014.
10. T Chiba, I Nishioka, and I Shirakawa. An algorithm of maximal planarization of graphs. In *Proc. IEEE Symp. on Circuits and Systems*, pages 649–652, 1979.
11. Markus Chimani, Ivo Hedtke, and Tilo Wiedera. Limits of greedy approximation algorithms for the maximum planar subgraph problem. In *Combinatorial Algorithms: 27th International Workshop, IWOCA 2016, Helsinki, Finland, August 17-19, 2016, Proceedings 27*, pages 334–346. Springer, 2016.
12. Markus Chimani, Ivo Hedtke, and Tilo Wiedera. Exact algorithms for the maximum planar subgraph problem: New models and experiments. *Journal of Experimental Algorithmics (JEA)*, 24:1–21, 2019.
13. Markus Chimani, Karsten Klein, and Tilo Wiedera. A note on the practicality of maximal planar subgraph algorithms. In Yifan Hu and Martin Nöllenburg, editors, *Graph Drawing and Network Visualization*, pages 357–364, Cham, 2016. Springer International Publishing.
14. Markus Chimani and Tilo Wiedera. Cycles to the rescue! novel constraints to compute maximum planar subgraphs fast. In Yossi Azar, Hannah Bast, and Grzegorz Herman, editors, *26th Annual European Symposium on Algorithms, ESA 2018, August 20-22, 2018, Helsinki, Finland*, volume 112 of *LIPIcs*, pages 19:1–19:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.ESA.2018.19.
15. Robert Cimikowski. An analysis of heuristics for graph planarization. *Journal of Information and Optimization Sciences*, 18(1):49–73, 1997.
16. M.E. Dyer, L.R. Foulds, and A.M. Frieze. Analysis of heuristics for finding a maximum weight planar subgraph. *European Journal of Operations Research*, 20:102–114, 1985.
17. Leslie R Foulds. *Graph theory applications*. Springer Science & Business Media, 2012.
18. Harold N. Gabow and Matthias F. Stallmann. An augmenting path algorithm for linear matroid parity. *Combinatorica*, 6:123–150, 1986.
19. H.N. Gabow and M. Stallmann. Efficient algorithms for graphic matroid intersection and parity. In *12th Colloq. on Automata, Language and Programming*, pages 210–220, 1985.

1210 **20** M. R. Garey and D. S. Johnson. *Computers and Intractability.* W.H. Freeman and Co., NY,
1211 1979.
1212 **21** M. Junger and P. Mutzel. Maximum planar subgraphs and nice embeddings: Practical layout
1213 tools. *Algorithmica*, 16:33–59, 1996.
1214 **22** Jon Lee, Maxim Sviridenko, and Jan Vondrák. Matroid matching: The power of local
1215 search. *SIAM Journal on Computing*, 42(1):357–379, 2013. `arXiv:https://doi.org/10.`
1216 `1137/11083232X`, `doi:10.1137/11083232X`.
1217 **23** Annegret Liebers. Planarizing graphs—a survey and annotated bibliography. In *Graph*
1218 *Algorithms And Applications 2*, pages 257–330. World Scientific, 2004.
1219 **24** P.C. Liu and R.C. Geldmacher. On the deletion of nonplanar edges of a graph. In *10th*
1220 *Southeastern Conference on Combinatorics, Graph Theory, and Computing*, pages 727–738,
1221 1977.
1222 **25** L. Lovász and M.D. Plummer. *Matching theory.* North-Holland, Amsterdam–New York, 1986.
1223 **26** James B Orlin. A fast, simpler algorithm for the matroid parity problem. In *International*
1224 *Conference on Integer Programming and Combinatorial Optimization*, pages 240–258. Springer,
1225 2008.
1226 **27** Timo Poranen. A simulated annealing algorithm for the maximum planar subgraph problem.
1227 *International Journal of Computer Mathematics*, 81(5):555–568, 2004.
1228 **28** Timo Poranen. Two new approximation algorithms for the maximum planar subgraph problem.
1229 *Acta Cybernetica*, 18(3):503–527, 2008. URL: `https://cyber.bibl.u-szeged.hu/index.php/`
1230 `actcybern/article/view/3735`.
1231 **29** A. Schrijver. *Combinatorial Optimization.* Springer, 2003.
1232 **30** Z. Szigeti. On the graphic matroid parity problem. *J. Combin. Theory Ser. B*, 88:247–260,
1233 2003.
1234 **31** Roberto Tamassia. Graph drawing. In *Handbook of computational geometry*, pages 937–971.
1235 Elsevier, 2000.
1236 **32** Douglas B. West. *Introduction to Graph Theory.* Prentice Hall, 2 edition, September 2000.