# Correlation Clustering and Two-edge-connected Augmentation for Planar Graphs*

**Philip N. Klein[†‡1], Claire Mathieu[‡2], and Hang Zhou[‡3]**

1 **Brown University, United States**
klein@brown.edu
2 **CNRS, École Normale Supérieure, France**
cmathieu@di.ens.fr
3 **École Normale Supérieure, France**
hangzhou@di.ens.fr

**Abstract**

In *correlation clustering*, the input is a graph with edge-weights, where every edge is labelled either $+$ or $-$ according to similarity of its endpoints. The goal is to produce a partition of the vertices that disagrees with the edge labels as little as possible.

In *two-edge-connected augmentation*, the input is a graph with edge-weights and a subset $R$ of edges of the graph. The goal is to produce a minimum weight subset $S$ of edges of the graph, such that for every edge in $R$, its endpoints are two-edge-connected in $R \cup S$.

For *planar graphs*, we prove that correlation clustering reduces to two-edge-connected augmentation, and that both problems have a polynomial-time approximation scheme.

**1998 ACM Subject Classification** G.2.2 Graph Theory, F.2.2 Nonnumerical Algorithms and Problems

**Keywords and phrases** correlation clustering, two-edge-connected augmentation, polynomial-time approximation scheme, planar graphs

**Digital Object Identifier** 10.4230/LIPIcs.STACS.2015.554

## 1 Introduction

### 1.1 Correlation Clustering

*Correlation clustering* takes as input a graph whose edges are labelled either $\langle + \rangle$ or $\langle - \rangle$. A $\langle + \rangle$ edge represents evidence that its endpoints belong in the same cluster, and a $\langle - \rangle$ edge represents evidence that its endpoints belong in different clusters. Each edge has a non-negative *weight* reflecting the strength of the evidence. The goal is to find a clustering minimizing the total weight of edges inconsistent with that evidence. This formulation, previously from computational biology [10], was introduced by Bansal, Blum, and Chawla [8]. They suggested as an application the clustering of documents into topics.

In this paper, we study the case when the graph is planar. The motivation comes from *image segmentation*. The goal is to partition the image into regions representing different image components. An image is represented by a grid of pixels. For each pair of neighboring pixels, comparing the pixels' values yields an assessment of how likely the pixels are to belong

* The full version of the paper is available on the authors' websites.
† Research funded by NSF Grants CCF-0964037 and CCF-1409520.
‡ Some of this research was done during a semester program at ICERM, the NSF-supported Institute for Computational and Experimental Research in Mathematics at Brown University.

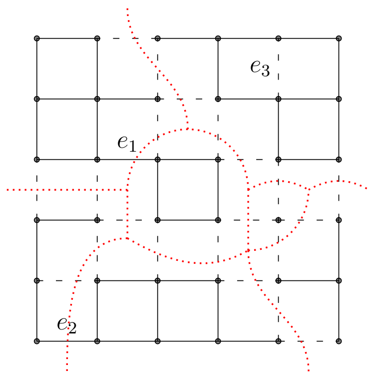


**Figure 1** In this unweighted grid, every solid (resp. dashed) edge represents a pair of similar (resp. dissimilar) pixels. Dotted lines indicate an optimal partition with inconsistent edges $e_1$, $e_2$, $e_3$.

to the same region. There can be spurious assessments. So global optimization is needed to find a good segmentation. See Figure 1. When an image is large, it is common for a visual task to first coalesce coherent uniform neighborhoods of pixels into superpixels, using preprocessing based on local properties such as brightness, color, and texture, see [3, 30]. We then extract a local similarity measure on pairs of adjacent superpixels, and the goal is to find a good segmentation of the superpixel graph under that measure. To achieve this, researchers used correlation clustering as the formulation [4, 5, 6, 26, 36]. They gave experimental results based on techniques such as integer linear programming or linear programming relaxation.

Note that the superpixel graph is planar. However, correlation clustering is NP-hard for planar graphs [7]. Prior to this work, the best result with theoretical guarantee was a constant-factor approximation for minor-excluded graphs by Demaine, Emanuel, Fiat, and Immorlica [17]. In this paper, we give a *polynomial-time approximation scheme* (PTAS).

▶ **Theorem 1.** *For any $\epsilon > 0$, there is a polynomial-time $(1 + \epsilon)$-approximation algorithm for correlation clustering in weighted planar graphs.*

### Related work

Why do we restrict ourselves to planar graphs? Because the general (weighted) problem is APX-hard [8]. Charikar, Guruswami, and Wirth [16] and independently Demaine, Emanuel, Fiat, and Immorlica [17] gave logarithmic-factor approximation algorithms. There have been improved approximation algorithms when the graph is complete [1, 8, 16]; or when, for each edge, the agreement weight and disagreement weight of that edge sum to one [1, 8]; or when, in addition, the weights satisfy the triangle inequality [22]. When the number of clusters is limited to a constant, Giotis and Guruswami [23] gave a PTAS. The problem was also studied in a planted model [31] and from the viewpoint of fixed-parameter tractability [15].

We discussed the problem of minimizing weight of disagreement; maximizing weight of agreement is equivalent at optimality but easier to approximate [8, 16, 35]. Researchers have also considered other objective functions [2].

## 1.2 Two-edge-connected Augmentation

In the field of telecommunications, an important task is to ensure that the network is resilient against single-link failures [34]. The *two-edge-connected augmentation* problem takes as input a graph $G$ with non-negative edge-weights and a subset $R$ of edges of the graph. The goal is to find a minimum-weight subset $S$ of edges of the graph such that for every edge $uv \in R$, $u$ and $v$ are two-edge-connected in the subgraph $R \cup S$. We give a PTAS for this problem when the graph is planar.

▶ **Theorem 2.** *For any $\epsilon > 0$, there is a polynomial-time $(1 + \epsilon)$-approximation algorithm for two-edge-connected augmentation in weighted planar graphs.*

### Related work

A closely related problem is *two-edge-connected spanning subgraph*, for which a constant-factor approximation algorithm was known [25]. When the graph is planar, Berger and Grigni [11] gave a PTAS. One might think that this would lead to a PTAS for our problem, but it is not the case because the weight of a two-edge-connected augmentation can be much smaller than the minimum weight of a two-edge-connected spanning subgraph. For the *Steiner-type generalization* of the two-edge-connected subgraph problem, there was a constant-factor approximation algorithms [28]. When the graph is planar, Borradaile and Klein [13] gave a PTAS.[1]

There is a variety of other related work, see [29] for a survey. Some studied the special case when the weights are all one [20, 21, 32], or when, in addition, the graph is complete [19]. There was a 2-approximation algorithm for the related problem of augmenting a connected subgraph to achieve two-edge-connectivity among a pre-specified set of terminal vertices [33]. Edge-connectivity augmentation problems were subsumed by the work of Jain [24] on survivable network design.

## 2 Techniques and Notations

Our techniques for proving Theorem 1 and Theorem 2 include *planar duality*, *prize-collecting clustering*, *brick decomposition*, *sphere-cut decomposition*, and *dynamic programming*.

Throughout the paper, we allow graphs to have parallel edges. For a graph $G$, we note $V[G]$ as its vertex set and $E[G]$ as its edge set. For a subset $H \subseteq E[G]$, we identify $H$ with the subgraph induced by edges from $H$. The weight of $H$ is defined by $\sum_{e \in H} \text{weight}\,(e)$. The *boundary* $\partial(H)$ is the set of vertices $u$ that are incident to some edge of $H$ and to some edge of $E[G] \setminus H$. Similarly, for a subset $U \subseteq V[G]$, its *boundary* $\partial(U)$ is the set of edges $uv$ such that $u \in U$ and $v \in V[G] \setminus U$. A *plane graph* is a planar graph together with a planar embedding. We use the phrases *plane graph* and *planar graph* interchangeably. We use $OPT(G, R)$ to denote the weight of the optimal two-edge-connected augmentation for $(G, R)$. The parameters $G$ and $R$ are omitted when they are clear from the context.

## 3 Theorem 2 Implies Theorem 1

We address correlation clustering and two-edge-connected augmentation in one paper because of the reduction in Theorem 3, which shows that Theorem 2 implies Theorem 1.

▶ **Theorem 3.** *There is an approximation-preserving reduction from correlation clustering in a weighted planar graph to two-edge-connected augmentation in a weighted planar graph.*

[1] In their problem, a solution is allowed to include multiple copies of edges of the input graph.

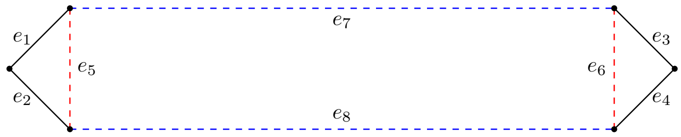


**Figure 2** In the example, $R = \{e_1, e_2, e_3, e_4\}$. The optimal two-edge-connected augmentation consists of edges $e_5$ and $e_6$. However, any Steiner tree connecting the edges of $R$ must include one of the edges $e_7$ and $e_8$, whose weight may be much higher than weight $(\{e_5, e_6\})$.

▶ Remark. In practice, we may use an approximation algorithm for two-edge-connected augmentation that is different from the algorithm in Theorem 2, and then from the reduction (Theorem 3), we obtain an algorithm for planar correlation clustering with the same approximation factor.

▶ **Lemma 4** (Bridge-Deletion Lemma). *Let $G$ be a plane graph. Let $R$ be a subset of $E[G]$. Let $S$ be a minimal two-edge-connected augmentation for $(G, R)$. Then every connected component in the subgraph $R \cup S$ is two-edge-connected.*

**Proof of Theorem 3.** Given a correlation-clustering instance $G_0$ with $\langle - \rangle$ edges, construct an instance $(G, R)$ of two-edge-connected augmentation as follows: To obtain the graph $G$, start with the planar dual of the $G_0$, and add duplicates of the duals of the $\langle - \rangle$ edges. The weights are preserved. Define $R$ to be the original (non-duplicate) duals of the $\langle - \rangle$ edges. Let $S$ be a minimal two-edge-connected augmentation. By the Bridge-Deletion Lemma, every connected component in $R \cup S$ is two-edge-connected. Define the clusters of $G_0$ to be the connected components when edges dual to $R \cup S$ are removed. ◀

## 4 Reduction to Instance with a Connected Skeleton

Without loss of generality, we assume for the rest of the paper that the edges of $R$ have weight zero.

To prove Theorem 2, we focus on a related version (Theorem 5), where we are given in addition a connected subgraph $T$ that contains every edge of $R$. We defer the proof of Theorem 5 to later sections.

▶ **Theorem 5** (Augmentation Theorem). *Let $G$ be a plane graph with edge-weights. Let $R$ be a subset of $E[G]$. Let $T$ be a connected subgraph of $G$ that contains every edge of $R$. For every $\epsilon > 0$, there is a polynomial-time algorithm* AUGMENT-CONNECTED$(G, R, T, \epsilon)$ *that computes a two-edge-connected augmentation $S$ for $(G, R)$ such that* weight $(S) \leq (1 + \epsilon)OPT(G, R) + \epsilon^2 \cdot$ weight $(T)$.

In the rest of this section, we prove Theorem 2 using the Augmentation Theorem. One might consider connecting all edges of $R$ with a Steiner tree $T$, and then applying the Augmentation Theorem. However, $OPT$ could be much smaller than the minimum weight of a Steiner tree when the solution is not connected (see Figure 2). In that case, the upper bound given by the Augmentation Theorem would not imply an approximation scheme.

Fortunately, there is an algorithmic tool, called *prize-collecting clustering*, due to Bateni, Hajiaghayi, and Marx [9], that addresses exactly this kind of obstacle. They used it in addressing the Steiner forest problem. They started with a 2-approximate solution, and used prize-collecting clustering to decompose the instance into subinstances. We use the same approach for two-edge-connected augmentation, see Algorithm 1.

**Algorithm 1** REDUCE-TO-CONNECTED

**Input:** a weighted planar graph $G$ and a subset $R$ of edges, $\epsilon > 0$
**Output:** connected subgraphs $T_1, \ldots, T_k$

```
1: Y ← two-edge-connected augmentation with weight at most 2 · OPT
2: (U_1, ··· , U_ℓ) ← two-edge-connected components of R ∪ Y
3: Contract each component U_i to build a new graph Ĝ
4: For every v ∈ Ĝ, let φ_v be ε^{-1} times the weight of the component corresponding to v
5: Do prize-collecting clustering on Ĝ and φ, obtaining a forest F
6: Return the connected components T_1, ..., T_k of the subgraph F ∪ R ∪ Y of G
```

**Algorithm 2** AUGMENT

**Input:** a planar graph $G$, a subset of edges $R$, and $\epsilon > 0$
**Output:** two-edge-connected augmentation $S$ for $(G, R)$

```
(T_1, ..., T_k) ← REDUCE-TO-CONNECTED(G, R, ε/7)        ▷ Theorem 7
for i ← 1 to k do
    S_i ← AUGMENT-CONNECTED(G, R ∩ T_i, T_i, ε/7)       ▷ Theorem 5
return (⋃_i S_i) \ R
```

Line 1 computes a 2-approximate solution using Jain's algorithm [24] (which solves a much more general problem).

▶ **Lemma 6** (corollary from [24]). *There is an algorithm that computes in polynomial time a two-edge-connected augmentation $Y$ for $(G, R)$ such that* weight $(Y) \leq 2 \cdot OPT$ *and that every connected component in $R \cup Y$ is two-edge-connected.*

Line 5 uses prize-collecting clustering, which receives a graph with vertex-potentials $\phi_v$ and returns a forest $F$ of edges of weight at most $2 \sum_v \phi_v$. Since the sum of vertex-potentials is at most $2\epsilon^{-1} \cdot OPT$, the weight of $F$ is at most $4\epsilon^{-1} \cdot OPT$. Using essentially the same arguments as in [9], we obtain the following.

▶ **Theorem 7** (variant of Theorem 1.3 in [9]). *Let $G$ be a plane graph with edge-weights. Let $R$ be a subset of $E[G]$. For fixed $\epsilon$, Algorithm 1 computes in polynomial time a set of connected subgraphs $T_1, \ldots, T_k$ with the following properties:*

- $\bigcup_i T_i$ *contains every edge of $R$.*
- $\sum_i \text{weight}\,(T_i) \leq (4/\epsilon + 2)OPT(G, R)$.
- $\sum_i OPT(G_i, R_i) \leq (1 + \epsilon)OPT(G, R \cap T_i)$

**Proof of Theorem 2.** The top-level algorithm of Theorem 2 is given in Algorithm 2. By the Augmentation Theorem (Theorem 5) and Property 1 of Theorem 7, the output is a two-edge-connected augmentation for $(G, R)$.

For each $i$, the weight of $S_i$ is at most $(1 + \epsilon/7)OPT(G, R \cap T_i) + (\epsilon/7)^2 \cdot \text{weight}\,(T_i)$. Summing over $i$ and combining Properties 2 and 3 of Theorem 7, we infer that the weight of the output solution is at most $(1 + \epsilon)OPT(G, R)$. ◀

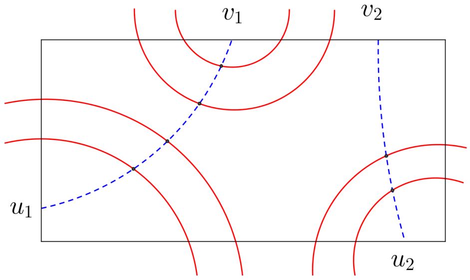


**Figure 3** The rectangle is a brick. The solid curves are parts of a near-optimal solution. The dashed curves illustrate the $u_1$-to-$v_1$ path and the $u_2$-to-$v_2$ path inside the brick.

# 5 Techniques for Proving the Augmentation Theorem

## 5.1 New Use of Brick Decomposition

For non-local problems in weighted planar graphs in which the weight of the optimal solution can be much smaller than the weight of the graph, the *brick decomposition* technique of [14] has proved to be quite versatile: a planar embedded subgraph $M$ (called the *mortar graph*) is selected, and the *bricks* are the subgraphs of $G$ embedded in the faces of $M$ (see Section 6.1). The key is the following properties of $M$.

**Property 1:** $M$ has weight $O(OPT)$;

**Property 2:** There exists a near-optimal solution that crosses the boundary of each brick only a constant number of times.

Both properties are achievable for problems such as *Steiner tree* [14], *Steiner forest* [9], *TSP* [12], and *two-edge-connected survivability* [13] for the variant in which the solution is allowed to include multiple copies of edges of the input graph.

The main obstacle in applying this approach to *two-edge-connected augmentation* is that Property 2 seems unachievable using the known brick-decomposition construction. We therefore use the mortar graph in a new way. We take additional care in the construction of the mortar graph because of the edges of $R$. As a consequence, instead of Property 2, we can show that, after a transformation[2] of the instance, we have:

**Property 2′ (Structure Theorem):** There exists a near-optimal solution such that, for any brick and any two vertices $u, v$ on the boundary of the brick, there exists a $u$-to-$v$ Jordan curve inside the brick that **intersects the near-optimal solution at only a constant number of points.**[3] See Figure 3.

Property 2′ is proved by *reducing nesting* and *adding boundary cycles*. See Section 6.

## 5.2 Outline of Algorithm Augment-Connected

We use ideas from [14] which we now summarize:

1. Build a mortar graph of $G$ based on the connected skeleton $T$.
2. Do Breadth-First Search (BFS) on the dual of the mortar graph, and select a mod-$k$ residue $j$ such that edges whose levels are congruent to $j$ have total weight at most $1/k$ times the weight of the mortar graph.

[2] The transformation is to add artificial copies of the brick boundaries. See Figure 4 in Section 6.

[3] The constant depends on $\epsilon$.

3. Commit to including these edges in the ultimate solution; this decomposes the graph into subinstances each consisting of at most $k$ levels of bricks.
4. A planar graph consisting of only $k$ BFS levels has branchwidth $2k$, i.e., can be recursively decomposed into clusters of edges such that each cluster is bounded by at most $2k$ vertices.

However, here we must diverge. Note that the branch decomposition obtained above has a special form: it is a *sphere-cut decomposition*, which means that each cluster of edges is precisely the set of edges enclosed by a Jordan curve $J$ that intersects no edges (and intersects a constant number of vertices) of the mortar graph. This is where Property $2'$ comes in: each segment of $J$ traversing a brick can be replaced with a curve that intersects a constant number of points of the near-optimal solution. This yields a new Jordan curve $J'$ that passes through a constant number of points of the near-optimal solution. Such structure enables us to design a dynamic program (DP), given in Section 7.

For each cluster of the sphere-cut decomposition, the DP enumerates all possibilities of the intersection points of the unknown near-optimal solution with the partially unknown Jordan curve $J'$. The DP also enumerates all possiblities of the connected structure of the part of the solution inside $J'$. See Section 7.2. Note that there may be some edges of the graph that are in the parent cluster but not in the child clusters (Figure 8), so the DP must do a bit of extra work to go from tables for the children to the table for the parent. See Section 7.3.

# 6 Structure Theorem

The Structure Theorem (Theorem 11) is the key to the polynomial-time performance of the dynamic program (Section 7). Before stating the theorem, we recall the definition and properties of *brick decomposition* from [14] in Section 6.1, and we illustrate the transformation of *doubling brick boundaries* in Section 6.2.

## 6.1 Mortar Graph and Brick Decomposition

▶ **Definition 8** (Mortar Graph and Bricks, slight adaptation from [14])**.** Let $G$ be a plane graph with edge-weights. Let $R$ be a subset of $E[G]$. Let $M$ be a subgraph of $G$. For each face $F$ of $M$, we define a *brick* $B$ as the planar subgraph of $G$ embedded inside the face, including the boundary edges of $F$. We denote the *interior* of $B$ as the brick without the boundary edges of $F$. We call $M$ a *mortar graph* of $G$ if the boundary of every brick $B$, in counter-clockwise order, is the concatenation of four paths $\text{North}_B$, $\text{South}_B$, $\text{East}_B$, $\text{West}_B$ (the subscript $B$ is omitted when it is clear from the context), such that:

1. No edge of $R$ is in the interior of $B$, or on $\text{South}_B$, $\text{East}_B$, or $\text{West}_B$.
2. $\text{South}_B$ is a shortest path in $B$, and every proper subpath of $\text{North}_B$ is an almost shortest path in $B$, i.e., its weight is at most $(1+\epsilon)$ times the weight of the shortest path between its endpoints in $B$;
3. There exists an integer $k = O(1/\epsilon^4)$ and vertices $s_0, s_1, \ldots, s_k$ ordered from west to east along $\text{South}_B$ such that, for any vertex $x$ on the segment $[s_i, s_{i+1})$ of $\text{South}_B$, the weight of the segment between $x$ and $s_i$ along $\text{South}_B$ is less than $\epsilon$ times the weight of the shortest path between $x$ and $\text{North}_B$ in $B$.

▶ **Lemma 9** (Brick-Decomposition Lemma, slight adaptation from [14])**.** *Let $G$ be a planar graph with edge-weights. Let $R$ be a subset of $E[G]$. Let $T$ be a connected subgraph of $G$ that contains every edge of $R$. There is a polynomial-time algorithm that computes a mortar graph $M$ of $G$ such that:*

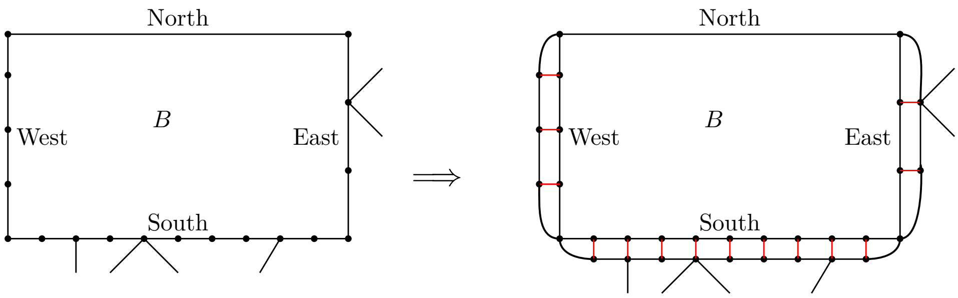


**Figure 4** Doubling the South, East, and West boundaries of the brick $B$. The new edges between vertices and their copies have weight 0.

1. $\text{weight}(M) = O(\text{weight}(T)/\epsilon)$;
2. $\sum_{brick\ B} \text{weight}(\text{East}_B \cup \text{West}_B) = O(\epsilon^2 \cdot \text{weight}(T))$.

## 6.2 Doubling Brick Boundaries

The proof of the Structure Theorem applies to a modified version of the graph in which artificial copies of the South, East, and West brick boundaries are added (Figure 4), and zero-weight edges are added between corresponding vertices. We call this *doubling* these boundaries. Note that no edges of $R$ are duplicated (according to Property 1 of Definition 8). Let $H$ be the resulting graph.

▶ **Lemma 10** (Boundary-Doubling Lemma). *A two-edge-connected augmentation for $(G, R)$ can be transformed into a two-edge-connected augmentation for $(H, R)$ in linear time without increasing the weight, and vice versa.*

As a consequence, it suffices to find a near-optimal solution for $(H, R)$.

## 6.3 Theorem Statement

▶ **Theorem 11** (Structure Theorem). *Let $G$ be a plane graph with edge-weights. Let $R$ be a subset of $E[G]$. Let $M$ be the mortar graph of $G$. Let $H$ be the graph obtained from $G$ by doubling the South, East, and West boundaries of every brick.*

*For any two-edge-connected augmentation $S_0$ for $(H, R)$, there exists a two-edge-connected augmentation $S$ for $(H, R)$ such that:*

- $\text{weight}(S) \le (1+\epsilon)\text{weight}(S_0) + 3\sum_{brick\ B} \text{weight}(\text{East}_B \cup \text{West}_B)$;
- *For any brick and any two vertices $u, v$ on the boundary of the brick, there exists a $u$-to-$v$ Jordan curve inside the brick that has $O(1/\epsilon^4)$ crossings with $S$, all occurring at vertices.*

## 6.4 Proof Sketch

The proof of the Structure Theorem consists in modifying the initial solution so that any pair of vertices on the boundary of a brick can be connected by a curve that has few crossings with the modified solution. Figure 5 shows the kind of curve we use. It starts at a given vertex $u$ on the brick boundary, traverses nested paths to reach the South boundary, then bypasses South-to-North paths using cycles formed by the duplicated edges of the South boundary, and finally again traverses nested paths to reach the given vertex $v$ on the brick boundary. In order to have a small number of crossings, we must ensure that the number of

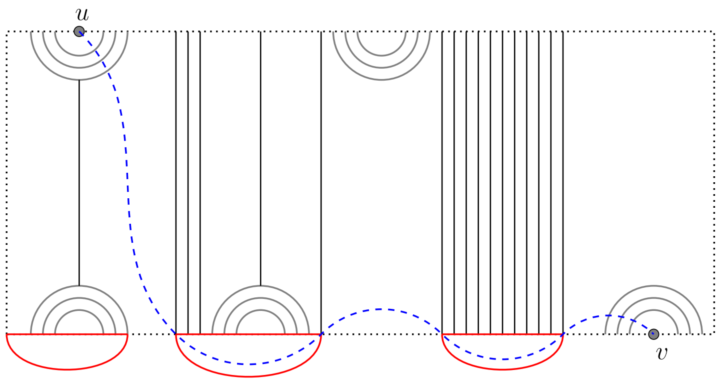


**Figure 5** The dashed path from $u$ to $v$ has few crossings with the modified solution (solid).

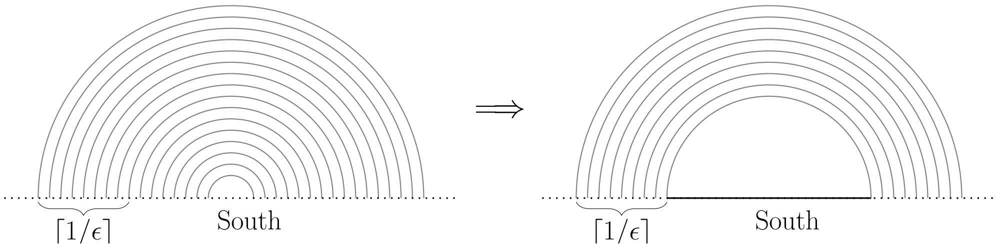


**Figure 6** Reducing nesting of the solution: if there are more than $\lceil 1/\epsilon \rceil$ nested paths (left figure), add a piece of the South boundary (the bold segment in the right figure) and empty the cycle thus created. The same operation is applied to nested paths connected to the North boundary, with the caveat that edges of $R$ need not be added to the solution (since the solution is supposed to be an augmentation of $R$).

nested paths is small and that only a small number of South cycles are used to bypass the South-to-North paths. This is illustrated in Figures 6 and 7.

The construction of the solution $S$ works on each brick in turn, modifying the initial solution $S_0$ inside that brick: adding the East and West cycles (i.e., the East and West boundaries together with their duplicates), reducing nesting as in Figure 6, and adding South cycles (i.e., parts of South together with their duplicates) as in Figure 7.

## 7 Dynamic Programming

In this section, we design a dynamic program (Theorem 12) to solve the two-edge-connected augmentation problem for $(H, R)$ in the special case where the dual of the mortar graph has bounded diameter. From the Structure Theorem, in order to get a near-optimal solution, we may restrict attention to solutions that satisfy the property defined there. A dynamic program computes the best among all such solutions.

▶ **Theorem 12** (Dynamic-Programming Theorem)**.** *Let $R, M, H$ be defined as in the Structure Theorem (Theorem 11). Assume, in addition, that the dual graph of $M$ has diameter $O(1/\epsilon^3)$. There is an algorithm that computes in polynomial time a two-edge-connected augmentation $S$ for $(H, R)$ such that* $\text{weight}(S) \le (1+\epsilon)OPT(H,R) + 3\sum_{brick\ B} \text{weight}(\text{East}_B \cup \text{West}_B)$.

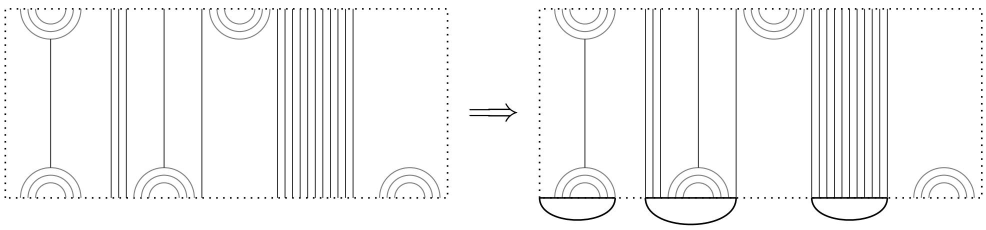

**Figure 7** Adding South cycles (the bold cycles in the right figure) into the solution. The number of South cycles needed is $O(1/\epsilon^4)$ due to Property 3 in the definition of bricks.

## 7.1 Sphere-Cut Decomposition

Our DP is based on a special kind of branch-decomposition of plane graphs, called a *sphere-cut decomposition* (see [18]): A *noose* of a plane graph is a Jordan curve that intersects only vertices of the graph and not edges. A *sphere-cut decomposition of width $w$* is a family of non-crossing nooses each intersecting at most $w$ vertices; the nooses form a binary tree by the enclosure relation, each leaf noose encloses exactly one edge, and each edge is enclosed by a leaf noose. For each noose in the sphere-cut decomposition, we refer to the set of edges enclosed as a *cluster*.

▶ **Lemma 13** (trivial adaptation from [27]). *Let $G$ be a plane graph whose dual graph has diameter $k$. Then $G$ has a sphere-cut decomposition of width $2k$, and it can be computed in linear time.*

## 7.2 Specification of DP Table

In this section, we define the index of the DP table and the value at an index.

By Lemma 13, $M$ has a sphere-cut decomposition $\mathcal{SC}$ of width $O(1/\epsilon^3)$. The first index of the DP table is a cluster $E$ of $\mathcal{SC}$.

Let $S_0$ be the optimal two-edge-connected augmentation for $(H, R)$, and let $S$ be the solution obtained in the Structure Theorem (Theorem 11). By the Bridge-Deletion Lemma (Lemma 4), we can modify $S$ so that every connected component in $R \cup S$ is two-edge-connected, without increasing the weight of $S$. For every cluster $E$ of $\mathcal{SC}$, let $J_E$ be the noose enclosing $E$ and of minimum number of crossings with $R \cup S$ (all occurring at vertices), breaking ties by choosing the minimally enclosing one.[4] It is easy to show that the family of nooses $\{J_E\}_{E \in \mathcal{SC}}$ is non-crossing.

▶ **Lemma 14.** *For every cluster $E$ of $\mathcal{SC}$, $J_E$ intersects $O(1/\epsilon^7)$ vertices of $R \cup S$.*

**Proof.** Since $\mathcal{SC}$ has width $O(1/\epsilon^3)$, there is a noose enclosing $E$ that has $O(1/\epsilon^3)$ intersections with $M$. From one intersection to the next, it goes across a single brick, and by the Structure Theorem (Theorem 11), the part inside this brick can be chosen so as to have $O(1/\epsilon^4)$ intersections with $S$. This results in a noose enclosing $E$ that has $O(1/\epsilon^7)$ intersections with $R \cup S$. ◀

Let $Q^* \subseteq V[H]$ denote the (unknown) set of $O(1/\epsilon^7)$ intersection vertices of $J_E$ with $S \cup R$. The second index of the DP table is a subset $Q \subseteq V[H]$ of size $O(1/\epsilon^7)$.

[4] Since the noose is a geometric object, it is not uniquely defined, but a discrete formulation can be given using the *face-vertex incidence graph*.

Next, we encode the connectivity structure of the part of $R \cup S$ inside $J_E$. Let $R_E$ (resp. $\Gamma^*$) denote the set of edges of $R$ (resp. $S$) that are inside $J_E$. Define a forest $F_0^*$ from $R_E \cup \Gamma^*$ by contracting every two-edge-connected component into a node. A node of $F_0^*$ is called *internal* if its corresponding two-edge-connected component in $R_E \cup \Gamma^*$ does not contain any node from $Q^*$, i.e., the component is strictly inside $J_E$. We then define a forest $F^*$ from $F_0^*$ by splicing internal nodes of degree 2 and removing internal nodes that are singletons. By the construction, $F^*$ has at most $|Q^*|$ non-internal nodes, and it does not contain internal nodes of degree 0, 1, or 2. So $F^*$ has at most $2|Q^*|-2$ nodes. The third index of the DP table is a forest $F$ of at most $2|Q|-2$ nodes. Moreover, there is a map $\psi^*$ giving the natural many-to-one map from $Q^*$ to nodes of $F^*$. The fourth index of the DP table is a map $\psi$ from $Q$ to $V[F]$. To summarize:

▶ **Definition 15** (DP index)**.** An *index of the DP table*, also called a *DP index*, contains the following:
- $E$: a cluster of the sphere-cut decomposition $\mathcal{SC}$
- $Q$: a subset of $V[H]$ of size $O(1/\epsilon^7)$
- $F$: a forest of size at most $2|Q|-2$
- $\psi$: a map from $Q$ to $V[F]$, such that every node of degree 0, 1, or 2 in the forest $F$ belongs to the image of $\psi$.

In addition, the triple $(Q, F, \psi)$ as defined above is called a *partial DP index.*[5]

A set of edges $\Gamma$ is *consistent* with a DP index $(E, Q, F, \psi)$ if applying the previous construction to $R_E \cup \Gamma$ leads to the connectivity structure described by $(F, \psi)$. For every DP index $(E, Q, F, \psi)$, define its *value* $DP(E, Q, F, \psi)$ as the minimum weight among a collection of $\Gamma$'s, such that:
- *Correctness:* Every $\Gamma$ in this collection is consistent with $(E, Q, F, \psi)$;
- *Optimality:* If $(Q, F, \psi) = (Q^*, F^*, \psi^*)$, then $\Gamma^*$ is in this collection.

In order to prove the Dynamic-Programming Theorem (Theorem 12), we only need to find a polynomial-time algorithm to fill in the DP table and to output the value $\mathrm{DP}(M, \emptyset, \emptyset, \emptyset^{\emptyset})$. [6]

## 7.3 Hole Region between Parent and Children

Let $E$ be a cluster of $\mathcal{SC}$ and let $E_1$ and $E_2$ be its child clusters. Let $Q^*, Q_1^*, Q_2^* \subseteq V[H]$ be the sets of intersections of $R \cup S$ with $J_E, J_{E_1}, J_{E_2}$. The *hole region* is the area inside $J_E$ but outside $J_{E_1}$ and $J_{E_2}$ in the plane.[7] See Figure 8. We remark that the hole region cannot contain edges from $R$.

Let $\hat{\Gamma}^*$ denote the set of edges of $S$ in the hole region. Let $\hat{Q}^*$ denote the set of intersections of $S$ with the boundary of the hole region. We have $\hat{Q}^* \subseteq Q^* \cup Q_1^* \cup Q_2^*$, thus $|\hat{Q}^*| = O(1/\epsilon^7)$. From $\hat{\Gamma}^*$ and $\hat{Q}^*$, we encode the connectivity structure of the part of $S$ in the hole region as a forest $\hat{F}^*$ of at most $2|\hat{Q}^*|-2$ nodes and a map $\hat{\psi}^* : \hat{Q}^* \to V[\hat{F}^*]$. This is similar to the encoding in Section 7.2.

We use a side table $T$ for the computation at hole regions. The table is indexed by a partial DP index $(\hat{Q}, \hat{F}, \hat{\psi})$. The *value* $T(\hat{Q}, \hat{F}, \hat{\psi})$ is defined as the minimum weight of any $\hat{\Gamma}$ that is consistent with $(\hat{Q}, \hat{F}, \hat{\psi})$ and contains no cycles.

---

[5] Note that the description of $Q, F, \psi$ is independent of $E$.

[6] The DP outputs the *value* of a solution, not the solution itself; but it is easy to enrich the DP in the standard manner so that it also outputs the solution achieving the value.

[7] Note that $J_E$, $J_{E_1}$, and $J_{E_2}$ are non-crossing.

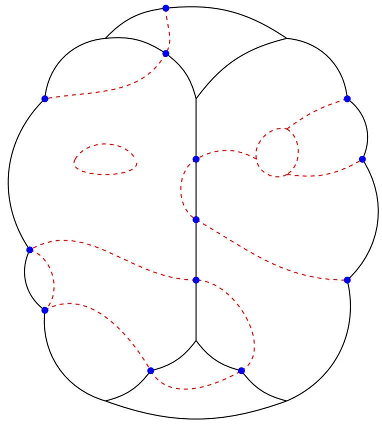

**Figure 8** $J_E$ is the outermost boundary. It encloses 4 areas that are separated by the solid curves. $J_{E_1}$ (resp. $J_{E_2}$) is the boundary of the left (resp. right) area. The *hole region* contains the top and bottom areas. The dashed paths represent $R \cup S$ inside $J_E$. The points represent vertices from $Q^* \cup Q_1^* \cup Q_2^*$.

## 7.4 Implementation of DP Table

First, the algorithm fills in the side table $T$ during the preprocessing. Notice that any $\hat{\Gamma} \subseteq E[H]$ that is consistent with $(\hat{Q}, \hat{F}, \hat{\psi})$ and contains no cycles is such that, every node $a$ in $\hat{F}$ corresponds to a vertex $u_a$ in the graph $H$, and every edge $ab$ in $\hat{F}$ corresponds to a path between $u_a$ and $u_b$ in $\hat{\Gamma}$. Therefore, to compute the value $T(\hat{Q}, \hat{F}, \hat{\psi})$, the algorithm enumerates, for every $a \in \hat{F}$, the vertex $u_a$ among $V[H]$. For every $ab \in \hat{F}$, it then computes the shortest path between $u_a$ and $u_b$ in $H$. The union of all these shortest paths defines the current $\hat{\Gamma}$. The value $T(\hat{Q}, \hat{F}, \hat{\psi})$ is the minimum weight of all $\hat{\Gamma}$'s during the enumeration. The overall running time of the preprocessing is thus polynomial.

Next, the algorithm fills in the DP table in the order of the index $E$ from bottom up in $\mathcal{SC}$. Consider a DP index $(E, Q, F, \psi)$. Let $E_1$ and $E_2$ be the child clusters of $E$. The algorithm enumerates every combination of $(E_1, Q_1, F_1, \psi_1)$, $(E_2, Q_2, F_2, \psi_2)$, and $(\hat{Q}, \hat{F}, \hat{\psi})$ that are *compatible* with $(E, Q, F, \psi)$, and the current weight is the sum of the three entries. $DP(E, Q, F, \psi)$ is assigned with the minimum weight during the enumeration.

**Acknowledgements** We would like to thank Howard J. Karloff for helping make the connection between correlation clustering and two-edge-connected augmentation in planar graphs; Nabil Mustafa for numerous discussions; and Grigory Yaroslavtsev.

## References

1. Nir Ailon, Moses Charikar, and Alantha Newman. Aggregating inconsistent information: Ranking and clustering. *Journal of the ACM*, 55(5), 2008.
2. Nir Ailon and Edo Liberty. Correlation clustering revisited: The true cost of error minimization problems. In *International Colloquium on Automata, Languages and Programming*, pages 24–36. Springer, 2009.
3. Sharon Alpert, Meirav Galun, Ronen Basri, and Achi Brandt. Image segmentation by probabilistic bottom-up aggregation and cue integration. In *Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.

4 Amir Alush and Jacob Goldberger. Ensemble segmentation using efficient integer linear programming. *Pattern Analysis and Machine Intelligence*, 34(10):1966–1977, 2012.

5 Amir Alush and Jacob Goldberger. Break and conquer: Efficient correlation clustering for image segmentation. In *Similarity-Based Pattern Recognition*, volume 7953, pages 134–147. Springer, 2013.

6 Bjoern Andres, Jörg H. Kappes, Thorsten Beier, Ullrich Kothe, and Fred A. Hamprecht. Probabilistic image segmentation with closedness constraints. In *International Conference on Computer Vision*, pages 2611–2618. IEEE, 2011.

7 Yoram Bachrach, Pushmeet Kohli, Vladimir Kolmogorov, and Morteza Zadimoghaddam. Optimal coalition structure generation in cooperative graph games. In *Conference on Artificial Intelligence*, 2013.

8 Nikhil Bansal, Avrim Blum, and Shuchi Chawla. Correlation clustering. *Machine Learning*, 56(1-3):89–113, 2004.

9 MohammadHossein Bateni, MohammadTaghi Hajiaghayi, and Dániel Marx. Approximation schemes for Steiner forest on planar graphs and graphs of bounded treewidth. *Journal of the ACM*, 58(5):21, 2011.

10 Amir Ben-Dor, Ron Shamir, and Zohar Yakhini. Clustering gene expression patterns. *Journal of Computational Biology*, 6(3/4):281–297, 1999.

11 A. Berger and M. Grigni. Minimum weight 2-edge-connected spanning subgraphs in planar graphs. In *International Colloquium on Automata, Languages and Programming*, volume 4596, pages 90–101, 2007.

12 Glencora Borradaile, Erik D. Demaine, and Siamak Tazari. Polynomial-time approximation schemes for subset-connectivity problems in bounded-genus graphs. *Algorithmica*, 68(2):287–311, 2014.

13 Glencora Borradaile and Philip N. Klein. The two-edge connectivity survivable network problem in planar graphs. *International Colloquium on Automata, Languages and Programming*, pages 485–501, 2008.

14 Glencora Borradaile, Philip N. Klein, and Claire Mathieu. An $O(n \log n)$ approximation scheme for Steiner tree in planar graphs. *ACM Transactions on Algorithms*, 5(3):31, 2009.

15 Sebastian Böcker and Jan Baumbach. Cluster editing. In Paola Bonizzoni, Vasco Brattka, and Benedikt Löwe, editors, *The Nature of Computation. Logic, Algorithms, Applications*, volume 7921, pages 33–44. Springer, 2013.

16 Moses Charikar, Venkatesan Guruswami, and Anthony Wirth. Clustering with qualitative information. *Journal of Computer and System Sciences*, 71(3):360–383, 2005.

17 Erik D. Demaine, Dotan Emanuel, Amos Fiat, and Nicole Immorlica. Correlation clustering in general weighted graphs. *Theoretical Computer Science*, 361(2):172–187, 2006.

18 Frederic Dorn, Eelko Penninkx, Hans L. Bodlaender, and Fedor V. Fomin. Efficient exact algorithms on planar graphs: Exploiting sphere cut decompositions. *Algorithmica*, 58(3):790–810, 2010.

19 Kapali P. Eswaran and R. Endre Tarjan. Augmentation problems. *SIAM Journal on Computing*, 5(4):653–665, 1976.

20 Guy Even, Jon Feldman, Guy Kortsarz, and Zeev Nutov. A 1.8 approximation algorithm for augmenting edge-connectivity of a graph from 1 to 2. *ACM Transactions on Algorithms*, 5(2):21:1–21:17, 2009.

21 Guy Even, Guy Kortsarz, and Zeev Nutov. A 1.5-approximation algorithm for augmenting edge-connectivity of a graph from 1 to 2. *Information Processing Letters*, 111(6):296–300, 2011.

22 Aristides Gionis, Heikki Mannila, and Panayiotis Tsaparas. Clustering aggregation. *ACM Transactions on Knowledge Discovery from Data*, 1(1), 2007.

**23** Ioannis Giotis and Venkatesan Guruswami. Correlation clustering with a fixed number of clusters. In *Symposium on Discrete algorithm*, pages 1167–1176. ACM, 2006.

**24** Kamal Jain. A factor 2 approximation algorithm for the generalized Steiner network problem. *Combinatorica*, 21(1):39–60, 2001.

**25** Samir Khuller and Uzi Vishkin. Biconnectivity approximations and graph carvings. *Journal of the ACM*, 41(2):214–235, 1994.

**26** Sungwoong Kim, Sebastian Nowozin, Pushmeet Kohli, and Chang Dong Yoo. Higher-order correlation clustering for image segmentation. In *Advances in Neural Information Processing Systems*, pages 1530–1538, 2011.

**27** Philip N. Klein and Shay Mozes. Optimization algorithms for planar graphs. In preparation, manuscript at http://planarity.org.

**28** Philip N. Klein and R. Ravi. When cycles collapse: A general approximation technique for constrained two-connectivity problems. In *Integer Programming and Combinatorial Optimization*, pages 39–55, 1993.

**29** Guy Kortsarz and Zeev Nutov. Approximating minimum cost connectivity problems. *Approximation Algorithms and Metahueristics*, 2007.

**30** David R. Martin, Charless C. Fowlkes, and Jitendra Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *Pattern Analysis and Machine Intelligence*, 26(5):530–549, 2004.

**31** Claire Mathieu and Warren Schudy. Correlation clustering with noisy input. In *Symposium on Discrete Algorithms*, pages 712–728, 2010.

**32** Hiroshi Nagamochi. An approximation for finding a smallest 2-edge-connected subgraph containing a specified spanning tree. *Discrete Applied Mathematics*, 126(1):83 − 113, 2003.

**33** R Ravi. Approximation algorithms for Steiner augmentations for two-connectivity. Technical Report CS-92-21, Department of Computer Science, Brown University, 1992.

**34** Mauricio Resende and Panos Pardalos. *Handbook of optimization in telecommunications*. Springer, 2008.

**35** Chaitanya Swamy. Correlation clustering: maximizing agreements via semidefinite programming. In *Symposium on Discrete Algorithms*, pages 526–527. SIAM, 2004.

**36** Julian Yarkony, Alexander Ihler, and Charless C Fowlkes. Fast planar correlation clustering for image segmentation. In *European Conference on Computer Vision*, volume 7577, pages 568–581. Springer, 2012.