

On Recognizing Map Graphs

Master Thesis of

Matthias Hanl

At the Department of Informatics and Mathematics
Chair of Theoretical Computer Science



Reviewers:	Prof. Dr. Ignaz Rutter
Advisors:	Prof. Dr. Ignaz Rutter
	Lukas Bachmann

Time Period: 28th November 2025 – 28th May 2026

Statement of Authorship

I hereby declare that this document has been composed by myself and describes my own work, unless otherwise acknowledged in the text.

Passau, May 28, 2026

Abstract

Let $G = (V, E)$ be a graph. We call G a *map graph* if we can map every vertex to a closed region on the sphere that is homeomorphic to a disk, such that those regions do not overlap and they share a boundary if and only if the associated vertices are adjacent. If we consider two regions adjacent when they share a boundary curve, we get exactly the set of planar graphs. But if we allow intersections of boundaries in single boundary points, we get a superset of the planar graphs, the map graphs. Those graphs can have arbitrarily large cliques. Consider, for example, a map drawn like the slices of a pizza with an arbitrary amount of such slice-regions meeting in the middle of the pizza.

We give a broad overview over the characterization of map graphs, as well as solved and open problems on map graphs, including current results on parametrized problems and the recognition of map graphs with bounded tree width. Further, we look into the details of a polynomial recognition algorithm for general map graphs, introduced by Mikkel Thorup in 1998. Unfortunately, in his work many details were left out and many proofs are incomplete. The basic concept of his algorithm is a recursive Local Mapping Problem (LMP), which finds an embedding for a subgraph, such that this embedding has a hole in which we can embed the rest of the vertices into. For the embedding of the rest of the vertices, which we call *interior vertices*, the only information needed is the order of the vertices around the hole, which are called the *boundary vertices*. Thus, the first steps of simplifying the instance of the problem are small cuts. Then, further operations are applied on the LMP. For example, (1) ignoring all the boundary vertices that have no adjacent neighbour among the interior vertices and (2) simplifying to multiple LMPs, such that for each LMP the interior vertices form a connected component. For both of these steps, we give (partial) LMPs with a valid solution, for which Thorups algorithm fails. At the end, we establish an overview over potential further steps.

Deutsche Zusammenfassung

Sei $G = (V, E)$ ein Graph. Wir nennen G einen *Kartengraphen*, wenn wir jeden Knoten auf eine Region auf der Sphäre abbilden können, welche homeomorph zu einer abgeschlossenen Scheibe ist, so dass sich die Regionen nicht überlappen und zwei Regionen sich genau dann eine Grenze teilen, wenn die entsprechenden Knoten adjazent sind. Wenn wir zwei derartige Regionen als adjazent betrachten, wenn die Grenze, die sie sich teilen, eine Kurve ist, bekommen wir genau die Menge der planaren Graphen. Wenn wir aber auch Adjazenzen über einzelne Punkte auf der Grenze erlauben, bekommen wir eine Übermenge der planaren Graphen, die Kartengraphen. Die Graphen können beliebig große Cliques haben, zum Beispiel wenn man sich eine Karte vorstellt, die wie eine Pizza aussieht, mit einer beliebig hohen Anzahl an Regionen, die sich in der Mitte berühren. Wir geben sowohl eine ausführliche Übersicht über die Charakterisierung von Kartengraphen, als auch über gelöste und offene Probleme für Kartengraphen. Darunter sind aktuelle Ergebnisse zu parameterisierten Problemen und die Erkennung von Kartengraphen mit beschränkter Baumweite. Anschließend betrachten wir einen Algorithmus für allgemeine Kartengraphen, welcher 1998 von Mikkel Thorup eingeführt wurde und Kartengraphen in polynomieller Zeit erkennen soll. Leider fehlen in seiner Arbeit viele Details und die Beweise sind unvollständig. Die grundlegende Idee hinter dem Algorithmus ist ein rekursives "Local Mapping Problem" (LMP), welches eine Einbettung für einen Teilgraphen findet, so dass diese Einbettung eine Region freilässt, in welche wir den restlichen Graphen als Karte einbetten können. Für die Einbettung der Knoten des restlichen Graphen, welche wir *interior vertices* nennen, benötigen wir nur die Information, in welcher Reihenfolge die Regionen um die freigelassene Region eingebettet werden können, die zu diesen Regionen entsprechenden Knoten nennen wir *boundary vertices* (Randknoten). Die ersten Schritte um eine Instanz des Problems zu vereinfachen sind kleine Schnitte, welche die Instanzen verkleinern. Danach werden weitere Operationen auf dem LMP angewandt. Zum Beispiel (1) die Randknoten zu ignorieren, welche keinen Nachbarn im Inneren haben und (2) zu mehreren LMPs zu vereinfachen, so dass für jedes LMP die inneren Knoten eine Zusammenhangskomponente formen. Für beide Schritte geben wir ein (partiell) LMP mit einer validen Lösung an, für welches Thorup's Algorithmus feststellt, dass kein Kartengraph vorliegt. Am Ende liefern wir eine Übersicht über mögliche nächste Schritte und offene Fragen.

Contents

1	Introduction	1
2	Preliminaries	3
3	Previous Results on Map Graphs	9
3.1	Hole-Free Map Graphs	10
3.2	About k -Map Graphs	12
3.3	Parametrized Problems on Map Graphs	17
3.4	Recognition of Map Graphs with Outerplanar Witnesses	20
3.5	Recognition of Map Graphs of Bounded Treewidth	22
4	Recognizing Map Graphs	25
4.1	Local Map Problem	25
4.2	Initiate the LMP	29
4.3	Reducing the Boundary	30
4.4	Disconnected Interior Components	31
4.5	Split Pizzas and Further Steps in the Algorithm	34
5	Conclusion	39
	Bibliography	41

1. Introduction

Suppose you want to design some kind of infrastructure network such as electricity supply or public transport. Let us model the regions and their neighbourhoods as a graph by using the regions as vertices and connecting two vertices if and only if their corresponding regions share a border. If we consider two regions to be neighboured if they share a boundary curve, the result will be a planar graph. But what if it should be possible to connect the infrastructure via boundary points of the regions? The result might not be a planar graph because it is possible to construct arbitrarily large cliques, for example if many regions meet in one point like slices of a pizza, see Figure 2.1(a) for example. But it still can not be just any graph due to implied restrictions: For example, we can not get a $K_{3,3}$ by converting a map into a graph because it is not representable as a map. We call graphs that can be drawn as a map *map graphs*. They are a superset of planar graphs but do not contain all graphs. A further classification based on [CGP98] is presented in the preliminary section.

This thesis first gives a broad overview over combinatorial and algorithmic results on map graphs, then it follows the next natural question after the classification: Given a graph $G = (V, E)$, how can we see if G can be represented as a map and how do we get such a representation? Those questions are already answered for map graphs allowing at most four nations to meet at any point in [CGP98], but for map graphs in general the question is open. Chen et al. proved that the problem lies in NP. In [Tho98a], Mikkel Thorup introduced an idea to solve this problem recursively in polynomial time. Unfortunately, this idea has never been completed. This is where this thesis steps into place. Is it possible to complete Thorup's ideas? Here, we use another view on map graphs than Thorup, who directly maps the graph to nations. We look at map graphs as Chen, Grigni and Papadimitriou characterized them, namely as half-squares of planar, bipartite graphs, also called witnesses [CGP98]. The basis for the algorithm is the definition of the local mapping problem, a recursive problem to transfer our graph into such a witness graph.

Thorup's original idea is presented in two documents: one short version, published in the context of a conference [Tho98a] and a longer, "unpolished" version published on the internet, which is currently not available [Tho98b]. We prove that there are mistakes in both versions and discuss other minor inaccuracies. We also give some ideas for improvement. However, the overall completion of Thorup's idea was outside the scope of this work.

2. Preliminaries

Intuitively, the definition of a map graph is quite simple: The vertices correspond to the "nations" and an edge represents a neighbourhood of two nations, meaning that they intersect at least at one point on the border. Formally, this is written by Chen, Grigni and Papadimitriou as follows:

Definition 2.1 ([CGP02]). *Suppose G is a simple graph. A map of G is a function M taking each vertex v of G to a [region $M(v)$ homeomorphic to a closed disc] in the sphere, such that:*

- *For every pair of distinct vertices u and v , the interiors of $M(u)$ and $M(v)$ are disjoint.*
- *Two vertices u and v are adjacent in G if and only if the boundaries of $M(u)$ and $M(v)$ intersect.*

If G has a map, then it is a map graph. The regions $M(v)$ are the nations of M . The uncovered points of the sphere fall into open connected regions; the closure of each such region is a hole of M .

In literature, an often used view is that the map of a graph is also a graph with labeled faces (the nations) and unlabeled faces (the holes). Two countries are adjacent if their boundaries intersect. We call this definition for a map *gmap*. One can easily see that the concepts and proofs can be transferred between both views. We say two nations of a map *strongly touch* if they share a boundary curve. We say two nations of a map *weakly touch* if they touch, but do not strongly touch, i.e., they share a boundary point. If one allows at most k nations to meet at one point, it gives rise to k -map graphs. Planar graphs are exactly 3-map graphs [CGP98].

Chen, Grigni and Papadimitriou introduced an overview in [CGP98] of how a clique can be represented in a map graph. They proved that there are only four possible ways for a clique with n vertices, visualized in Figure 2.1:

- (a) All nations in the clique share one boundary point. We call this a *pizza*.
- (b) Some $n - 1$ nations share one boundary point, the last one touches all other nations at another boundary. We call this a *pizza with crust*.
- (c) The nations form a *hamantach*, meaning there are three points realizing all the neighbourhood relations, with at most two nations touching all three of those points.

- (d) An ordinary planar clique such as the rice ball, meaning a clique where all nations share a boundary curve, not just a boundary point.

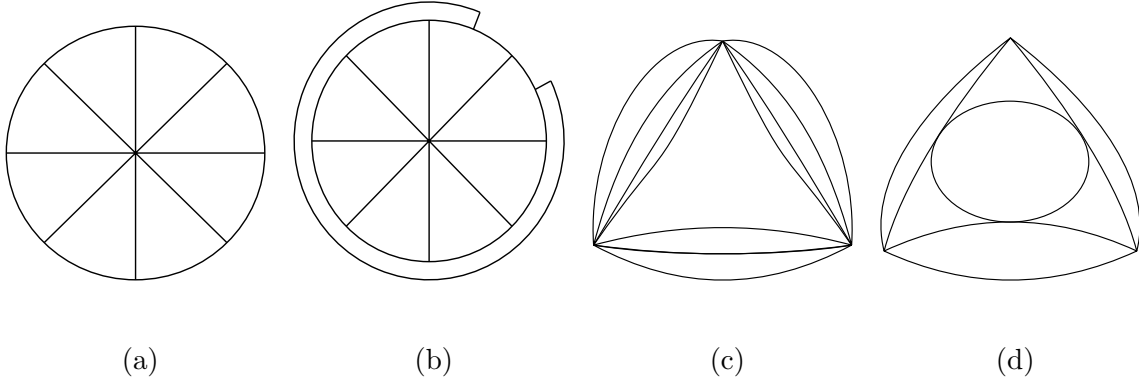


Figure 2.1: Cliques in map graphs: (a) A pizza, (b) a pizza with crust, (c) a hamantach and (d) a rice ball.

We denote a maximal clique on k vertices as MC_k . For the proof that cliques in map graphs can only be drawn in this four ways and for later presented constructions and algorithms, we use another view on map graphs than in the presented definition, which was also introduced in [CGP02].

Let H^2 denote the *square* of a graph H with H^2 having the same vertex set as H and an edge between two vertices a, b if and only if there is a 2-edge path between a and b in H .

Definition 2.2 ([CGP02]). *Suppose $H = (S; T; E)$ is a bipartite graph. Then $H^2[S]$ is the half-square of H . Having $V(H^2[S]) = S$ and with two vertices being adjacent in $H^2[S]$ if and only if they have a common neighbour in T in H . We call H a witness of G .*

Theorem 2.3. [[CGP02]] *A graph G is a map graph if and only if it is the half-square of some planar bipartite graph H .*

Proof Sketch. [[CGP02]]

" $\Rightarrow$ "

Let $G = (V, E)$ be a map graph with $|V| = n$ nations. Let V' be the set of nations, each one bijectively identified with a vertex of V . For each nation $v \in V'$, at most $n - 1$ points suffice on its boundary to realize all of its adjacencies. This leads to a finite set T of such points on the boundary of nations for the whole map. Then, we choose for each nation $v \in V'$ one point p in the interior to represent the nation and connect those interior points s to the points of T on the boundary of their nation. We denote the set of all interior points s by S . This clearly creates a planar bipartite graph $H = (S; T; E')$, where two nations touch if and only if they have a distance of 2. So $H^2[S] = G$.

" $\Leftarrow$ "

Given a planar bipartite graph $H = (S; T; E')$, take a planar embedding of H by drawing a sufficiently thin star-shaped nation around the vertices of S and its adjacent edges in E' , we get a map of $G = H^2[S]$. $\square$

Figure 2.2 shows an example of a graph $G = (V, E)$ and its planar witness graph $H = (V_H, E_H)$.

For the recognition of map graphs, it suffices to check if there exists such a witness. Of course, the four kinds of cliques can also be represented as witness graphs, see Figure 2.3.

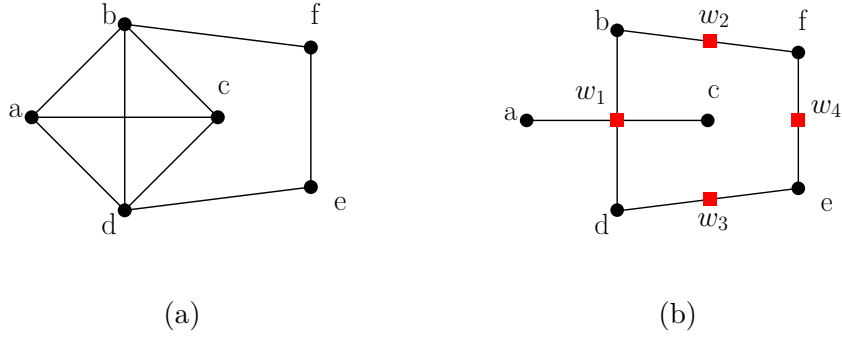


Figure 2.2: (a) A graph G and (b) its witness graph H . The black dots represent vertices of the original graph G , while we call the red squares *witness vertices*.

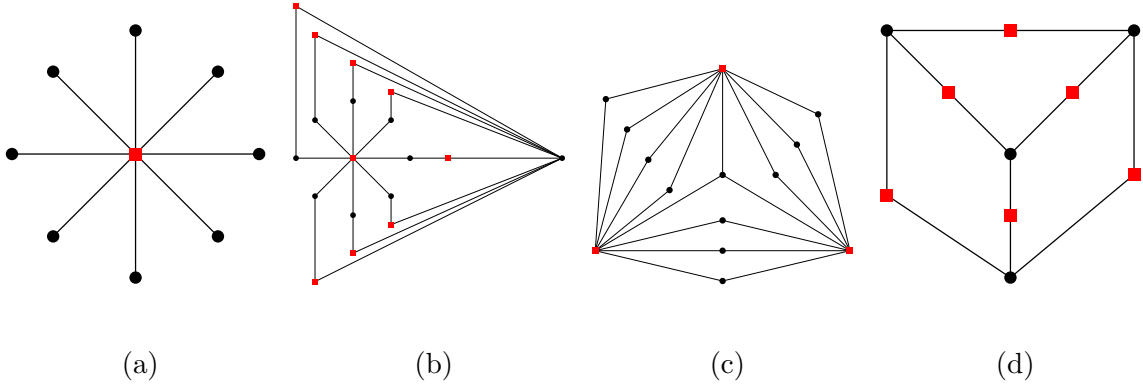


Figure 2.3: Cliques in witness perspective: (a) A pizza, (b) a pizza with crust, (c) a hamantach and (d) a rice ball.

Given a witness $H = (S; T; E')$ for a graph G , we call a witness point $w \in T$ *redundant* if all pairs of its neighbours already share a common witness. We call a witness without redundant witness points *compact*.

Now, with this preliminaries it is possible to present the idea of the proof of the following:

Theorem 2.4 ([CGP02]). *A map graph clique must be one of the above four types.*

Proof Sketch. The idea is to look at the map of one clique $C = K_n$ and make a case distinction over the maximal degree d of all witness points and the clique size n . If $n = d$, we get a pizza, if $n = d + 1$, we get a pizza with crust. So, for the rest of the proof sketch, assume $n \geq d + 2$. For $d \leq 3$, we get a planar clique, but if $d = 3$, we get $n = 5$ and K_5 is not planar. So, we have $d = 2$ and $n = 4$, the rice ball. Now, assume $d \geq 4$. Then, having at least 6 vertices and two witness points w_1, w_2 with one of them having maximal degree, we always get the graph in Figure 2.4(a) as a subgraph, since for both w_1 and w_2 , there have to be at least two points only touching the other witness point ($n \geq d + 2$). Now, we need witness points $w \neq w_i$ for $i \in \{1, 2\}$ that connect the pairs $(a, c), (a, f), (d, c), (d, f)$ since (a, b, c, d, e, f) is a clique. Assume we connect those independently with one witness vertex per pair. Then, the witness would not be planar as illustrated in Figure 2.4(b). Since the paths of length 2 of the pairs (a, f) and (d, c) (with exactly one witness vertex on each path) always cross, we need to merge those two witness points. This leads to the situation illustrated in Figure 2.4(c). But now, the paths of length 2 of the pairs (a, c) and (d, f) are redundant and we can delete them and the witness vertices on them. This leads to a hamantach as illustrated in Figure 2.4(d). $\square$

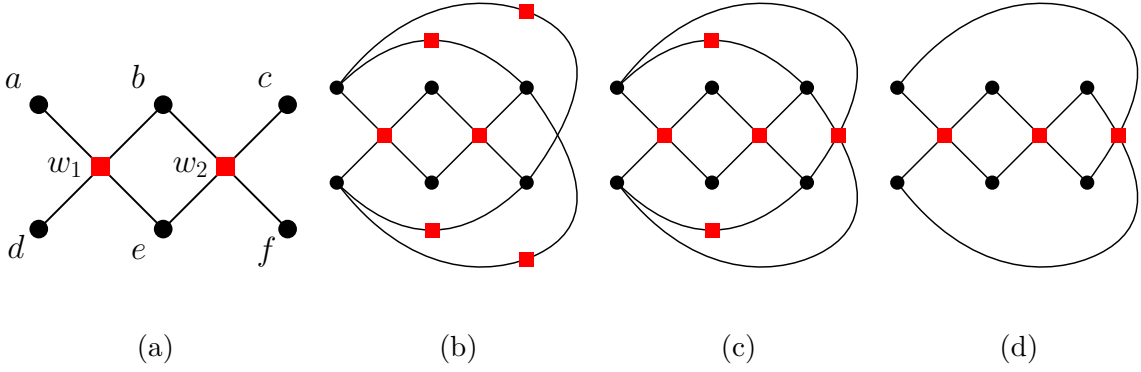


Figure 2.4: (a) The witness of a subgraph S of G , (b) a nonplanar witness W if S is a clique, (c) a planar witness W' and (d) W' with deleted redundant edges and witness points.

It is also possible to get an bound for the vertices and edges of a witness H for a graph $G = (V, E)$. A quadratic bound is quite obvious, but it is also possible to prove the following:

Lemma 2.5 ([CGP02]). *If G is a map graph with n vertices, then it has a witness H with $\mathcal{O}(n)$ vertices and edges.*

Proof. Consider a compact drawing of $H = (S; T; E')$. Now, every witness vertex w has two neighbored vertices a, b , which do not have another path of length two in between them. Delete w and all adjacent edges, and connect a, b via a direct edge. Then, we have $|S| = n$ vertices and $|T| = m$ edges in a simple planar graphs. By Euler's formula we get $m \leq 3n - 6$. So at most $4n - 6$ vertices for H . Since H has no 3-cycles because it is bipartite, by again applying one of Euler's formulas on H , we get $|E'| \leq 2(4n - 6) - 4 = 8n - 16$. $\square$

Also, we have for k -map graphs:

Lemma 2.6 ([CGP02]). *For $k \geq 3$, k -map graphs are those with witnesses H such that every witness point has at most degree k . Moreover, a k -map graph with n vertices has $\mathcal{O}(kn)$ edges.*

Proof. The first part follows directly from Theorem 2.3. For the second part, in the proof of Lemma 2.5, we proved that a witness $H = (V; T; E')$ of a map graph $G = (V, E)$ has less than $8n$ edges. So, there is a vertex $v \in S$ of degree at most 7. Since G is a k -map graph, v has in G a maximal degree of $7(k - 1)$. For an induction, assume a graph with n vertices has $c \cdot kn$ edges for a constant c and let us look on a graph G' with $n + 1$ vertices. Then, we can just delete a vertex v in G' of maximal degree $7(k - 1)$. This results in a number of edges of G' of $c \cdot kn + 7k - 7 = c \cdot k(n + 1) + 6k - 7 \in \mathcal{O}(k(n + 1))$. For $k = 3$, which are planar graphs, this holds, since a planar graph has at most $3n - 6$ edges (Euler's formula) and we can construct a witness by putting one witness vertex between each pair of adjacent vertices of the planar graph. $\square$

This bound has been improved to $kn - 2k$ edges in [Che01] by looking at "well-formed" maps, which are efficient in the amount of edges.

Since the main difference between map graphs and planar graphs is the size of cliques, there will be a special focus on them. An important concept for cliques is a *contractible curve*. If we view our graph as a map, a contractible curve is a curve that intersects the border

of all nations of a clique and intersects no other nations. If we have a planar bipartite graph $H = (S; T; E)$, a contractible curve is a set of vertices $t \in T$ touching only vertices of one clique and in total touching all vertices of this clique, which can be connected with a non-crossing edge. In both cases, if there exists a contractible curve turning a clique into a pizza by contracting the curve, we call this clique a *safe pizza*.

Another important concept in this paper is a data structure called *PQ-tree*. This is a tree whose nodes are called *P*- and *Q*-nodes. They allow two kinds of operations on their nodes: the children of *P* (Permutation-) nodes can be ordered arbitrarily and the children of *Q* (Queue-) nodes have a fixed order, which can only be reversed. For a node v in this tree, we denote by $L(v)$ the leaves whose lowest common ancestor (*lca*) is v .

In such a *PQ*-tree, we often have to *cluster* some leaves to restrict the possible orderings, meaning that we adjust the *PQ*-tree such that those leaves are neighboured in all possible orderings as described in [BL76].

Lemma 2.7 ([Tho98a]). *If S is clustered in σ , the ordering within S is independent of the rest of the ordering. More precisely, let π be a σ -ordering and π_S be any $(\sigma|_S)$ -ordering. Consider the two orderings π' and π'' obtained from π by replacing the cluster $\pi|_S$ either with π_S , or with π_S reversed. Then, either π' or π'' is a σ -ordering.*

Proof. Let $v = lca(S)$ and let S be clustered in σ . The other parts of the *PQ*-tree σ are not relevant for the ordering of S . Since v is the lowest common ancestor of the vertices in S , there are at least two children c_1, c_2 of v with leafs that are part of S . If v is a *Q*-node, there can not be a child of v with leafs in- and outside of S , since then, we can just mirror the order in this child and the leafs of S are not consecutive anymore, which contradicts S being clustered. So, the children of v having S as leafs contain S and nothing else, which means that those children already have to be clustered beyond v and the order of the children's children does not matter. If v is a *P*-node, again, there can not be a child of P having vertices of S and vertices not of S , since else, we can just swap this child with another one containing (parts of) S and get a contradiction to S being clustered. Also, there can not be a child of v with leafs not contained in S , since else, we can just put this child in between the (at least) two children containing S and get a contradiction to S being clustered. $\square$

3. Previous Results on Map Graphs

The basis of map graph theory was essentially established by Chen, Grigni and Papadimitrou with the definition of map graphs, analyzing their cliques and introducing the witness perspective. By looking at the latter, one can also quickly prove that the recognition problem for map graphs is in NP: given a witness and the algorithm in [HT74], it is possible to check if the witness can be embedded in the plane in linear time. Another important result is that the property of being a map graph is not monotone, meaning that we can not characterize map graphs by forbidden subgraphs or minors. This can be seen in the following example. Consider the graph illustrated in Figure 3.1.

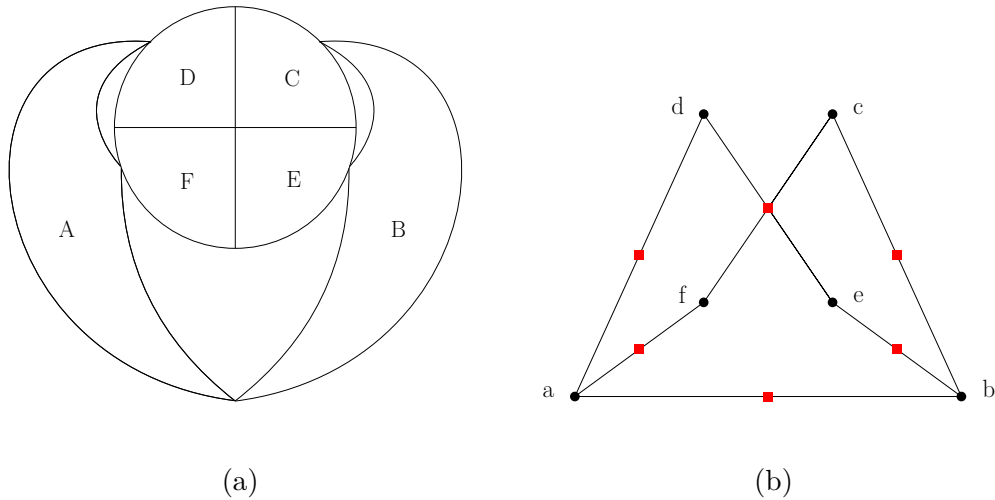


Figure 3.1: (a) A graph G in map perspective and (b) in witness perspective.

If we delete the edge $u = (c, e)$, we get a witness that is not planar, see Figure 3.2. Because a graph is a map graph if and only if it has a planar witness, $G - u$ is not a map graph.

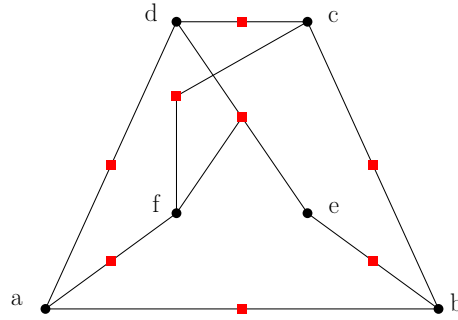


Figure 3.2: Graph $G - u$.

3.1 Hole-Free Map Graphs

A *hole-free* map M of a graph G is a map where every point of the sphere is covered by some nation of M [CGP02]. If G has such a map, we call G a *hole-free map graph*. An example is illustrated in Figure 3.3.

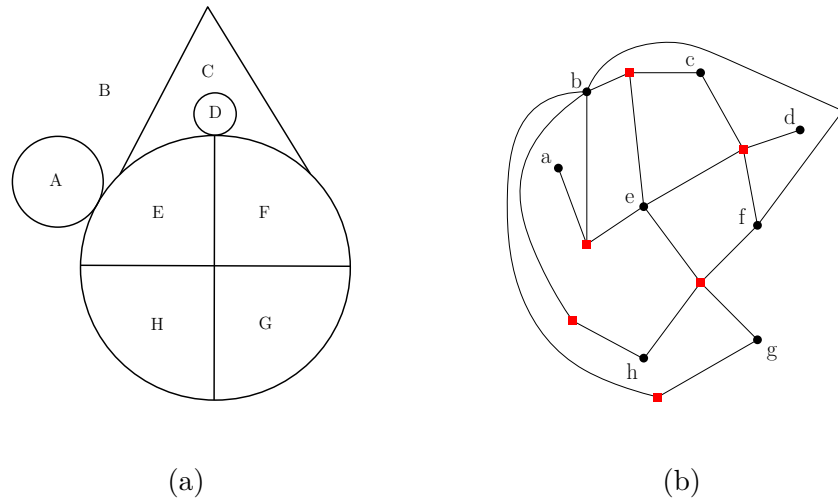


Figure 3.3: (a) A hole-free map graph in map perspective and (b) in witness perspective.

Hole-free map graphs are biconnected and one can conclude directly from Theorem 2.3 that a map graph is hole-free if and only if the faces of its witness have each exactly 4 or 6 vertices on its boundary [CGP02].

Such graphs can be used to simplify proofs for other graphs, for example, maximal planar graphs are exactly 3-connected hole-free 3-map graphs [CGP06]. Currently, there is no algorithm to recognize general hole-free map graphs. However, in 2006, Chen, Grigni and Papadimitrou introduced an algorithm to recognize hole-free 4-map graphs in cubic time using *triangulated 1-planar* graphs.

Definition 3.1. A graph is called *1-planar* if there exists an embedding such that each edge is crossed at most once. A graph is called *triangulated 1-planar* if each face is bordered by exactly 3 edges.

An example of a triangulated 1-planar graph is illustrated in Figure 3.4. Chen et al. simplify the problem of recognizing hole-free 4-map graphs to recognizing triangulated 1-planar graphs. To work on those, we first introduce some definitions. A graph is called

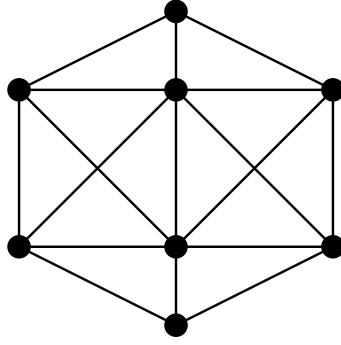
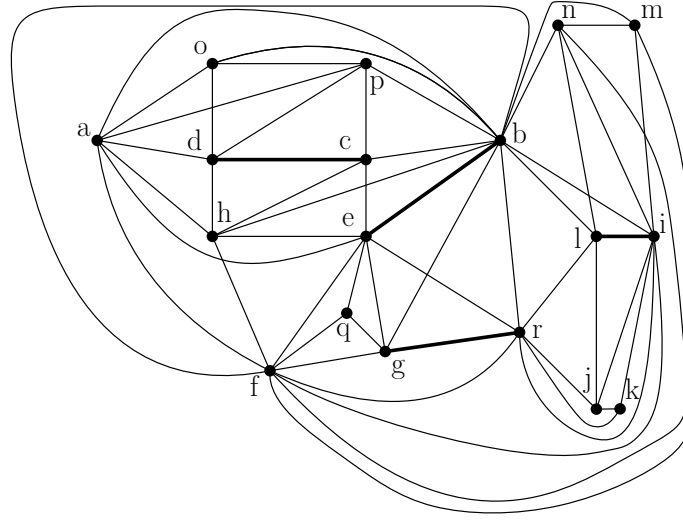


Figure 3.4: A triangulated 1-planar graph.


 Figure 3.5: A marked graph G .

marked if each edge is either marked or not. An example of a marked graph is illustrated in Figure 3.5. Marked graphs give the possibility to mark edges where we already know that the nations associated with the two vertices of the edge strongly touch, so these edges can not be crossed by another edge.

Definition 3.2 ([CGP06]). *A layout of G is a 4-gmap $\mathcal{L}$ of a marked graph $G = (V, E)$ such that*

- (1) *the degree of every vertex in $\mathcal{L}$ is at most 4, and*
- (2) *for every marked edge (u, v) in G , countries u and v strongly touch in $\mathcal{L}$.*

$\mathcal{L}$ is well-formed if there is no vertex of degree 2.

If a layout $\mathcal{L}$ of G has no hole, we call it an *atlas* of G . A layout $\mathcal{L}$ of $G[U]$ with $U \subseteq V$ is called *extensible* if and only if whenever G has an atlas A , then $\mathcal{L} = A[U]$. Also, they define edges (a, b) and (x, y) in $G = (V, E)$ as *crossable* if they are both unmarked and $\{a, b, x, y\}$ is a MC_4 in G . For an edge $e = (a, b)$, if e is unmarked, then let $\mathcal{E}[e]$ denote the set of all vertices $x \in V$ such that there is a vertex $y \in V$ with (x, y) crossable with e . Otherwise, let $\mathcal{E}[e]$ be the empty set.

The idea is to simplify the graph G via the following separators at first, which already leads to subgraphs without 6-cliques. Chen et al. search for those separators in the presented order.

- **Separating edge:** A separating edge is an edge $e = (a, b) \in E$ such that the graph $G[V \setminus (\{a, b\} \cup \mathcal{E}[e])]$ is disconnected.
- **Separating 4-cycle:** A separating 4-cycle is a chord-free 4-cycle C such that $G[V \setminus C]$ is disconnected.
- **Separating triple:** A separating triple is a 3-clique C such that $G[V \setminus (C \cup \mathcal{E}[e])]$ (with e being one of the three edges of C) is disconnected.
- **Separating quadruple:** A separating quadruple is a chord-free 4-cycle C such that $G[V \setminus (C \cup \mathcal{E}[e])]$ (with e being one of the four edges of C) is disconnected.
- **Strongly separating triangle:** A strongly separating triangle is a separating triangle, where the disconnect graph has a connected component consisting only of a single vertex.
- **Separating triangle:** A separating triangle is a 3-clique C such that the graph $G[V \setminus (C \cup \mathcal{E}[e] \cup \mathcal{E}[f])]$ (with e, f being two of the three edges of C) is disconnected.

For example in the marked graph illustrated in Figure 3.5:

- The edge $e = (a, b)$ is a separating edge with $\mathcal{E}[e] = \{o, p, h, e\}$, separating d and c from the rest of the graph.
- The chord free cycle $C = \{a, b, c, d\}$ is a separating 4-cycle, since it separates o and p from the rest of the graph.
- The clique $C = \{e, f, g\}$ is a separating triple already without deleting $\mathcal{E}[e]$ for some edge e , since it separates q from the rest of the graph.
- The cycle $C = \{b, i, r, j\}$ is a separating quadruple with $\mathcal{E}[(b, i)] = \{m, n\}$, separating l from the rest of the graph.
- The clique $\{b, i, r\}$ is a (strongly) separating triangle with $\mathcal{E}[(b, i)] = \{m, n\}$ and $\mathcal{E}[(r, i)] = \{j, k\}$, separating l from the rest of the graph.

After having applied all possible separators in this order, Chen et al. search for maximal 5-cliques (MC_5) and remove them. A 5-clique C may be removed by finding so-called correct centers of C and removing those. A *correct center* is a list of four vertices (since we are in a 4-map graph) (a, b, c, d) in C , such that C has a well-formed extensible layout in which a, b, c, d meet in a 4-node in this order [CGP06]. For example, in Figure 3.5 (e, b, r, g) form a correct center in this order, meeting in the face between them in a crossing. Chen et al. remark that in a MC_5 , such a correct center is always a *correct pizza*, which is a list (a, b, c, d) , such that if G has a well-formed atlas, it has one in which a, b, c, d meet in a 4-node in this order [CGP06]. After removing such a correct center, it might happen that we can, again, apply separators, which needs to be done before removing the next MC_5 . If no correct center can be found, there has to be a separator to decompose G to continue. Then, with the 5-cliques being removed, all 4-cliques are firstly being scanned for rice balls, then for other non-pizza cliques and at last for pizzas, in every step simplifying the problem to cases where the structure does not exist anymore. Afterwards, G has to be planar, since no 4-clique is left. Else it gets rejected [CGP06].

3.2 About k -Map Graphs

For general k -map graphs, there is no known recognition algorithm. But regarding 4-map graphs, a result in [CGP98] states that they can be recognized in polynomial time. Franz J. Brandenburg improves this bound to cubic time by characterizing them as *kite-augmented 1-planar graphs* [Bra19b]. A *kite* is a drawing of a K_4 , namely a 4-cycle with two crossing edges (either both inside or both outside the cycle) as illustrated in Figure 3.6(a).

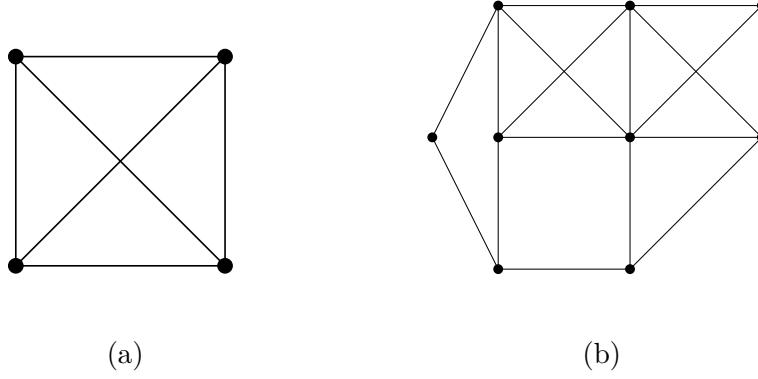


Figure 3.6: (a) A kite K and (b) a kite-augmented 1-planar graph G .

Definition 3.3 ([Bra19b]). A 1-planar embedding $\mathcal{E}(G)$ of a graph G is kite-augmented if, for every pair of edges (a, c) and (b, d) that cross in $\mathcal{E}(G)$, G contains the edges (a, b) , (b, c) , (c, d) and (d, a) . A 1-planar graph is kite-augmented if it admits a kite-augmented 1-planar embedding.

Figure 3.6(b) presents an example for a kite-augmented 1-planar graph G . Brandenburg presents the following theorem.

Theorem 3.4 ([Bra19b]). For a graph $G = (V, E)$ the following are equivalent:

- (1) G is kite-augmented 1-planar.
- (2) G is a 4-map graph.
- (3) $G = H^2[V]$ for a planar bipartite graph $H = (V, W, F)$ with vertices of degree 2 and 4 in W .

Proof. The equivalence (2) $\Leftrightarrow$ (3) follows directly from Theorem 2.3. We can not have vertices in W of degree higher than 4, since otherwise we would get cliques with more than 4 vertices resulting in map graphs, which are not 4-map graphs. We do not need vertices in W of degree 1, since they can just be left away, and we do not need vertices in W of degree 3, since they can be exchanged with 3 vertices of degree 2 [CGP02].

We achieve (3) $\Rightarrow$ (1) by taking an embedding of H , deleting each vertex $w \in W$ of degree 2 with $N(w) = \{a, b\}$ and adding the edge (a, b) . Then, we delete every vertex $w \in W$ with $N(w) = \{a, b, c, d\}$ and add the kite (a, b, c, d) . By deleting parallel edges we get a kite-augmented 1-planar graph.

The other direction ((1) $\Rightarrow$ (3)) works by taking an embedding of G , adding a vertex w of degree 4 for each kite (a, b, c, d) with $N(w) = \{a, b, c, d\}$ and adding a vertex w' for each non-crossing edge (a, b) with $N(w') = \{a, b\}$. $\square$

Having simplified the problem of recognizing 4-map graphs to recognizing kite-augmented 1-planar graphs, Brandenburg uses an algorithm $\mathcal{B}$ based on the algorithm $\mathcal{A}$ for hole-free 4-map graphs from Chen et al.. Regarding $\mathcal{A}$, Brandenburg proves that if there is a separating triple, then there are also separating 4-cycles, meaning that there is no case in $\mathcal{A}$ where a separating triple is used. Moreover, it is proven that at the point where algorithm $\mathcal{A}$ searches for a MC_5 , there is no such clique left, also resulting in a case that is never used in $\mathcal{A}$. Based on these results, Brandenburg adjusts the list of separators for the algorithm $\mathcal{B}$. He uses some generalized version of the separators of algorithm $\mathcal{A}$ called (k, j) -separators.

Definition 3.5 ([Bra19b]). Let G be a connected graph. A pair (S, T) of k vertices S and j edges T is a (k, j) -separator of G if $G - S - T$ has at least two components. Note that the edges of T and all edges incident to a vertex in S are removed from $G - S - T$. We omit T and call S a k -separator if $j = 0$ and call S a separating k -cycle if $G[S]$ is a cycle. A (k, j) -separator (S, T) is minimal if no proper subset (S', T') with $S' \subseteq S$ and $T' \subseteq T$ separates G .

The following order of separators is used:

- Apply all possible 1-separators.
- Apply all possible 2-separators.
- Apply all possible 3-separators.
- Apply all possible separating edges.
- Apply all possible $(3, 1)$ -separators $(\{a, b, c\}, \{(x, y)\})$ with an unmarked edge (a, b) and $\mathcal{E}[(a, b)] = \{x, y\}$.
- Apply all possible 4-separators.
- Apply all possible $(3, 2)$ -separators $(\{a, b, c\}, \{(x, y), (u, v)\})$ with unmarked edges $(a, b), (a, c)$ and $\mathcal{E}[(a, b)] = \{x, y\}, \mathcal{E}[(a, c)] = \{u, v\}$.
- Apply all possible $(4, 1)$ -separators $(\{a, b, c, d\}, \{(x, y)\})$ with unmarked edges $(a, b), (a, c)$ and $\mathcal{E}[(a, b)] = \{x, y\}$.

This is considered useful, since every separator is minimal when being used. In Figure 3.5, for example, $(\{b, i, r\}, \{(l, n), (l, j)\})$ is a $(3, 2)$ -separator and $\{f, e, b\}$ is a 3-separator. Then, the algorithm checks for K_5 subgraphs and ends if it finds one. Next, by doing a case distinction over all K_4 -subgraphs, the graph can again be decomposed. Lastly, the algorithm checks the result and returns a correct embedding.

In the same paper, this algorithm is improved by introducing another algorithm $\mathcal{C}$, which only uses 1-, 2- and 3-separators, separating edges, 4- and 5-separators and then searches for maximal 4-cliques in this order. So, the three (k, j) -separators are replaced with the 5-separator. The running time is quickly determined by the running time of each step of the algorithm. In linear time, it works to compute 1- and 2- separators and their decomposition [HT73], preprocessing, planarity tests and operations on MC_4 s at the end. In addition, 3-, 4- and 5-separators can be computed in quadratic time [HRG00] as well as separating edges. Via applying a linear amount of separators, we get a total running time of $O(n^3)$.

Brandenburg further characterizes 5-map graphs in a similar way to 4-map graphs, namely as *2-fan crossing graphs* [Bra19a]. An edge has a *j-fan crossing* if it is crossed exactly j times by edges with one identical end point (this endpoint being the "center of the fan"). A graph is *k-fan-crossing* if it has a drawing such that no edge has a *j-fan crossing* with $j > k$. A 2-fan crossing graph is *clique-augmented* if every single edge crossing induces a 4-clique and every 2-fan crossing induces a 5-clique. More exactly, if there is a 4-clique (a, b, c, d) if two edges (a, c) and (b, d) have a 1-fan crossing and if there is a 5-clique (a, b, c, d, e) if an edge (a, b) has a 2-fan crossing by the edges (c, d) and (c, e) [Bra19a]. Compare with Figure 3.7.

Theorem 3.6 ([Bra19a]). For a graph $G = (V, E)$ the following are equivalent:

- (1) G is a 5-map graph.
- (2) $G = H^2(B)$ for a planar bipartite graph $B = (V, W, F)$ with vertices of degree at most five in W .

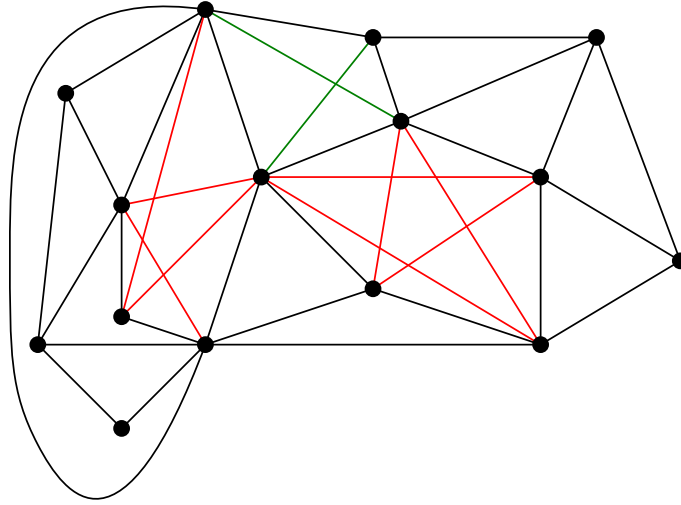


Figure 3.7: A clique-augmented 2-fan crossing graph G , with 1-fan crossings (green) that induce 4-cliques and 2-fan crossings (red) that induce 5-cliques.

(3) G is a clique-augmented 2-fan-crossing graph.

Proof Sketch. The equivalence (1) $\iff$ (2) follows directly from Theorem 2.3. So, it suffices to show that (2) holds if and only if (3) holds. For (2) $\Rightarrow$ (3) Brandenburg deletes the vertices $w \in W$ and adds edges to build the clique. For the other direction he uses the clique-augmented property to make 4-cliques out of the 1-fan crossings and 5-cliques out of the 2-fan crossings [Bra19a]. Take as example the graph in Figure 3.7: Each of the red and green regions can easily be exchanged with a witness point and vice versa. $\square$

Even with this classification of 5-map graphs, no algorithm for the recognition is known.

Another problem addressed by Brandenburg is about the thickness of a book-embedding of k -map graphs. A p -page book embedding of a graph G is a linear ordering of the vertices and an embedding of the edges into p pages, such that no two edges $(a, b), (c, d)$ embedded into the same page cross, which would mean that $a < c < b < d$. The book thickness is the minimum number of pages in all book embeddings [Bra20]. An example is illustrated in Figure 3.8, where we can see that the graph K_4 can be embedded into two pages, while the graph K_5 can not be embedded into two pages. The "pages" are in this case the half plane above the vertices and the half plane below the vertices.

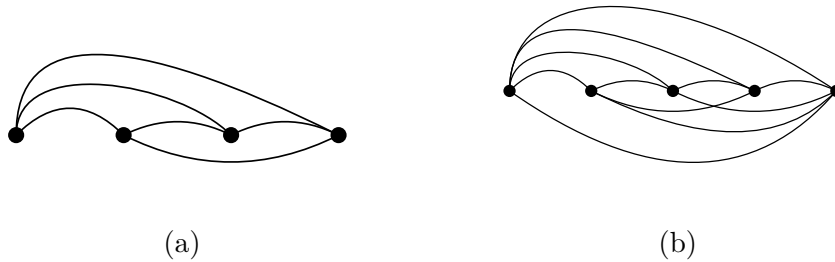


Figure 3.8: (a) A 2-page book embedding of the graph K_4 and (b) a K_5 , which can not be embedded in 2 pages.

Brandenburg shows that given a k -map graph on n vertices and its map, in $O(kn)$ time, a book embedding with $6\lfloor k/2 \rfloor + 5$ pages can be computed. To reach that, he uses results of Bekos et al. [BDLG⁺20] by considering k -framed graphs, graphs that have a planar base structure with faces of degree at most k and additional edges, which may only cross in the interior of the faces of the planar skeleton. A *partial k -framed* graph is a subgraph of a k -framed graph. Also, G is well-formed if for every edge (u, v) of G the intersection of $M(u)$ and $M(v)$ is either a single point or a single curve segment. Regarding the connection between k -map graphs and k -framed graphs, the main results of Bekos et al. [BDLG⁺20] are:

- Every well-formed hole-free k -map graph is k -framed.
- Every k -map graph is partial $2k$ -framed.
- Every k -framed graph is a k -map graph, under the assumption that each face of the planar skeleton induces a clique of size k .

In combination with their following main theorem, this lays the basis for Brandenburg's work.

Theorem 3.7 ([BDLG⁺20]). *The book thickness of a partial k -framed graph is at most $6\lfloor k/2 \rfloor + 5$.*

For Brandenburg's results, firstly, let us look at *planar 2-level subgraphs*. One can decompose each graph into *levels* by taking vertices touching the outer face as level $l = 0$ and achieving level $l + 1$ by deleting the vertices of level l and including the vertices on the new outer face [Yan89]. A 2-level subgraph is a graph where we look at two adjacent levels l and $l + 1$ of the global graph. For example, consider the illustration in Figure 3.9.

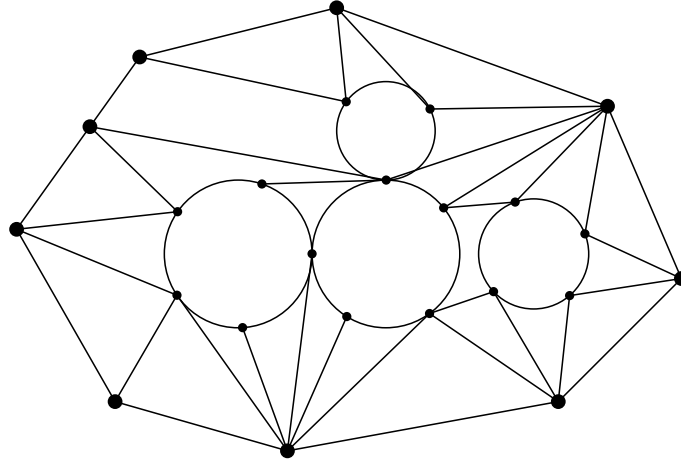


Figure 3.9: A 2-level graph with outer vertices around the outer face and inner vertices in the interior, which are structured in a forest of blocks.

We will call the vertices on level l *outer vertices* and the ones on level $l + 1$ *inner vertices*. Then, the inner vertices are structured in *blocks*, which are cycles in the interior level $l + 1$. In the global graph these cycles may contain vertices of level higher than $l + 1$ in the interior. If a block contains only a single vertex or an edge with its vertices, we call it an *elementary* block. There may be vertices contained in two blocks, which are called *cutvertices*. The blocks are structured in a forest-structure. One can look at any two adjacent layers of the global graph in this way, since there is no direct connection between level l and level $l + 2$. This also makes it practicable for a book embedding, because the 2-level graph containing

level l and $l + 1$ can be embedded in the same book pages as the 2-level graph containing level $l + 2$ and $l + 3$ without risking any crossing. This leads to two categories of pairs of levels: "odd" level pairs, for example the level pairs $(1, 2)$ and $(3, 4)$, and "even" level pairs, for example $(0, 1)$ and $(2, 3)$. Inside each category it is possible to "recycle" the pages of a book embedding.

By decomposing our k -map graph in two pairs of two levels, taking the block-structure to get a useful linear ordering for the book embedding and with the result that any 2-level framed multigraph H can be embedded in $3\lfloor k/2 \rfloor + 3$ pages if H is k -framed [Bra20], it is easy to conclude that we need at most the double amount of pages to embed both categories of pairs of levels. One page can even be saved [Bra20]. So, in total, for k -map graphs we have an upper bound of $6\lfloor k/2 \rfloor + 5$ pages.

3.3 Parametrized Problems on Map Graphs

In this section, we consider parametrized problems on map graphs. A problem is called *parametrized* if it assigns a parameter k to every input. It is called *fixed-parameter tractable* (FPT) if there is an algorithm that solves the problem in time $f(k) \cdot n^{\mathcal{O}(1)}$, where the function f is independent from the input size n . This means it is efficiently computable for a fixed parameter. Some of the most common parametrized problems on a map graph $G = (V, E)$ are:

- **Vertex Cover:** Is there a set $S \subseteq V$ with $|S| \leq k$ such that for every edge $(a, b) \in E$ there is $a \in S$ or $b \in S$?
- **k -Path:** Is there a simple path of length k ?
- **Feedback Vertex Set:** Is there a set $S \subseteq V$ with $|S| \leq k$ such that $G - S$ has no cycle?
- **Graph-Coloring:** Can we assign each vertex one of k colors, such that no two vertices with the same color are adjacent?

For eight such parametrized problems, results of Fomin et al. [FLP⁺26] give the fastest algorithms yet. Along with Vertex Cover, Feedback Vertex Set and Longest Path, they give algorithms with a running time of $2^{\mathcal{O}(\sqrt{k} \log k)} \cdot n^{\mathcal{O}(1)}$ for the following problems.

- **Connected Vertex Cover:** Is there a connected set $S \subseteq V$ with $|S| \leq k$ such that for every edge $(a, b) \in E$ there is $a \in S$ or $b \in S$?
- **Connected Feedback Vertex Set:** Is there a connected set $S \subseteq V$ with $|S| \leq k$ such that $G - S$ has no cycle?
- **$\mathcal{F}$ -Deletion:** Given a family $\mathcal{F}$ of connected graphs that contains at least one planar graph, is there a set S with $|S| \leq k$ such that $G - S$ does not contain a graph of $\mathcal{F}$ as minor?
- **k -Cycle:** Is there a cycle of length k ?
- **k -Cycle Packing:** Are there at least k vertex-disjoint cycles?

The two main concepts used for these algorithms are *tree decompositions* and *bidimensionality*.

Definition 3.8 ([FLP⁺26]). *A tree decomposition of a graph G is a pair $\mathcal{T} = (T_{\mathcal{T}}, \beta_{\mathcal{T}})$, where $T_{\mathcal{T}}$ is a rooted tree and $\beta_{\mathcal{T}}$ is a function from $V(T_{\mathcal{T}})$ to $2^{V(G)}$ that satisfies the following three conditions.*

$$(a) \bigcup_{x \in V(T_{\mathcal{T}})} \beta_{\mathcal{T}}(x) = V(G).$$

- (b) For every edge $\{u, v\} \in E(G)$, there exists $x \in V(T_{\mathcal{T}})$ such that $\{u, v\} \subseteq \beta_{\mathcal{T}}(x)$.
- (c) For every vertex $v \in V(G)$, the set of nodes $\{t \in V(T_{\mathcal{T}}) : v \in \beta_{\mathcal{T}}(t)\}$ induces a (connected) subtree of $T_{\mathcal{T}}$.

The width of $\mathcal{T}$ is $\max_{x \in V(T_{\mathcal{T}})} |\beta_{\mathcal{T}}(x)| - 1$. Each set $\beta_{\mathcal{T}}(x)$ is called a *bag*. Moreover, $\gamma_{\mathcal{T}}(x)$ denotes the union of the bags of x and its descendants. The treewidth of G is the minimum width among all possible tree decompositions of G , and it is denoted by $\text{tw}(G)$.

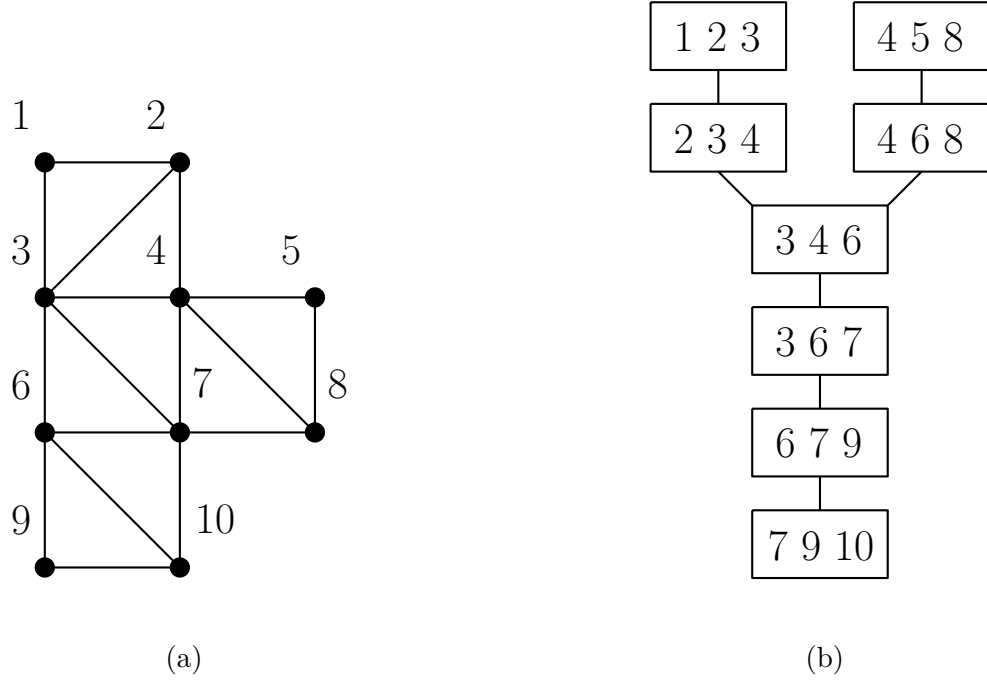


Figure 3.10: (a) A graph G and (b) a tree decomposition of width 2 of G (an example from [BKS20]).

A tree decomposition is called *nice* if every bag can be labeled with one of 4 labels. With such a special form the design of dynamic programming (DP) algorithms gets simplified. The formal definition is the following.

Definition 3.9 ([FLP⁺26]). A tree decomposition $\mathcal{T} = (T_{\mathcal{T}}, \beta_{\mathcal{T}})$ of a graph G is *nice* if for the root r of $T_{\mathcal{T}}$ it holds that $\beta_{\mathcal{T}}(r) = \emptyset$, and each node $v \in V(T_{\mathcal{T}})$ is of one of the following types.

- **Leaf:** v is a leaf in $T_{\mathcal{T}}$ and $\beta_{\mathcal{T}}(v) = \emptyset$. This bag is labelled with **leaf**.
- **Forget vertex:** v has exactly one child u , and there exists a vertex $w \in \beta_{\mathcal{T}}(u)$ such that $\beta_{\mathcal{T}}(v) = \beta_{\mathcal{T}}(u) \setminus \{w\}$. This bag is labelled with **forget**(w).
- **Introduce vertex:** v has exactly one child u , and there exists a vertex $w \in \beta_{\mathcal{T}}(v)$ such that $\beta_{\mathcal{T}}(v) \setminus \{w\} = \beta_{\mathcal{T}}(u)$. This bag is labelled with **introduce**(w).
- **Join:** v has exactly two children, u and w , and $\beta_{\mathcal{T}}(v) = \beta_{\mathcal{T}}(u) = \beta_{\mathcal{T}}(w)$. This bag is labelled with **join**.

For their algorithms, Fomin et al. need even more restrictions on the tree decomposition than being nice. They use a $(\ell, \mathcal{D})$ -NFewCliTD $\mathcal{D}'$, which is a tree decomposition $\mathcal{D}'$ derived from a tree decomposition $\mathcal{D}$ of the witness graph H with width less than ℓ such

that every bag of $\mathcal{D}'$ includes at most ℓ pizzas. Also some new labels, **redundant** and **fake introduce**, are added to the ones from a nice tree decomposition. In total, this leads to the following result, introducing the concept of bidimensionality.

Lemma 3.10 ([FLP⁺26]). *Given a map graph G with a corresponding planar bipartite graph H and an integer $\ell \in \mathbb{N}$, in time $\mathcal{O}(n^2)$, one can either correctly conclude that H contains an $\ell \times \ell$ grid as a minor, or compute a nice tree decomposition $\mathcal{D}$ of B of width less than 5ℓ and a $(5\ell, \mathcal{D})$ -NFewCliTD of G .*

Using this bidimensionality, Fomin et al. find no-instances fast via the grid as a minor and they use dynamic programming to find a solution in the tree-decomposition case. The dynamic programming starts at leafs of the tree, maintaining one table for each bag of the tree decomposition. Then, the dynamic program goes upwards in the tree and updates the tables depending on which label the next node has. Thus, the global problem has been simplified to local subproblems and, because of the bounded treewidth, the problem gets FPT.

For example, consider the problem Feedback Vertex Set for map graphs. Let $\ell = 3(\sqrt{k} + 1)$. Let (G, H, k) with graph G , its witness H and an integer k be an instance of Feedback Vertex Set. Firstly, let us assume the neighbourhood of each witness vertex in H is at most of size $k + 2$, because else we get a clique in G of size at least $k + 3$, which leads to a cycle, even with k vertices deleted. Afterwards, observe that if B contains a $3(\sqrt{k} + 1) \times 3(\sqrt{k} + 1)$ grid as a minor, then (G, H, k) is a no-instance. This is the case, because it follows of the grid that H contains $k + 1$ vertex-disjoint cycles of length at least 8. On those cycles, every second vertex is a witness vertex because of the bipartition-property of H . That leads to $k + 1$ vertex disjoint cycles of length at least 4 in the original graph G resulting in a NO-instance. Because of Lemma 3.10, we otherwise have a nice tree decomposition $\mathcal{D}$ of H of width less than $15(\sqrt{k} + 1)$ and a $(15(\sqrt{k} + 1), \mathcal{D})$ -NFewCliTD $\mathcal{D}'$. This results in an upper bound on cliques in $\mathcal{D}'$ of $15(k + 2)(\sqrt{k} + 1)$. Combining the fact that every clique needs to contain at most $k + 2$ vertices, and that to solve the Feedback Vertex Set problem, we need to delete at least all but two vertices of each clique, there are at most $\mathcal{O}(k^2)$ choices for each clique to get a solution. Using this, the proof bounds the number of states of the dynamic program to $2^{\mathcal{O}(\sqrt{k} \log k)}$, resulting in the standard dynamic program for Feedback Vertex Set on graphs with bounded treewidth.

Instead of using tree decomposition, Demaine et al. [DFHT05] used *branch decomposition* to find an algorithm for the (k, r) -**center problem** for map graphs. The problem asks if one can choose at most k vertices of the graph G such that every vertex of G has a distance of at most r to the closest of those k centers. Branch decompositions are again based on a tree structure. A tree is called *ternary* if each node has at most three children.

Definition 3.11 ([RS91]). *A branch-decomposition of a hypergraph G is a pair (T, τ) where T is a ternary tree and τ is a bijection from the set of leaves of T to $E(G)$. The order of an edge e of T is the number of vertices v of G such that there are leaves t_1, t_2 of T in different components of $T \setminus e$, with $\tau(t_1), \tau(t_2)$ both incident with v . The width of (T, τ) is the maximum order of the edges of T , and the branch-width $\beta(G)$ of G is the minimum width of all branch-decompositions of G (or 0 if $|E(G)| \leq 1$, when G has no branch-decompositions).*

Demaine et al. provided an $f(k, r)n^{\mathcal{O}(1)}$ algorithm, with f independent from n (FPT). In their algorithm they proceed by first looking at the branch-width of G to find the first no-instances and then using dynamic programming to compute the actual centers if they exist.

Since 2001, there are results regarding the independent set problem. Chen [Che01] delivered an approximation algorithm to compute a maximal independent set with or without assigned weights to the vertices. This algorithm works recursively in quadratic time but are considered impractical because of an impractical tree-decomposition being used.

Another standard problem for graphs is graph-coloring. For planar graphs there is the famous four-color theorem, which is proven by computing a set of "unavoidable configurations" and checking all of those with a computer [AH78]. It says that every vertex in a planar graph can be colored with one of four colors such that no two vertices with the same color are adjacent. For map graphs it is not that simple. A graph coloring trivially always needs at least the amount of colors as the size of the maximal clique in the graph. Since for maps graphs those can be arbitrarily large, there is no general solution. For k -map graphs, we know that the maximum clique size is $\lfloor 3k/2 \rfloor$ [CGP02]. For $k \in \{3, 4\}$ this holds because those are planar respectively 1-planar graphs, for which we have the four-color theorem respectively the six-color theorem [Bor84]. For $k > 4$ no results are known.

3.4 Recognition of Map Graphs with Outerplanar Witnesses

In 2017, Mnich, Rutter and Schmidt presented a linear time recognition algorithm of map graphs with outerplanar witness [MRS18]. A connected graph is called *outerplanar* if it has at least 3 vertices which can be embedded in the plane, so that all its vertices are adjacent to the outer face. A graph is outerplanar if and only if it contains no K_4 -minor and no $K_{2,3}$ -minor [CH67].

An important concept to recognize map graphs with outerplanar witness are *SPQR-trees*:

Definition 3.12 ([MRS18]). *For a graph G , an SPQR-tree is a tree T for which each node $x \in V(T)$ has an associated multigraph G_x , called skeleton of x , and one of the following four types:*

- *S-node: then G_x is a cycle on at least three vertices.*
- *P-node: then G_x is a multigraph with two vertices and at least three edges.*
- *Q-node: then G_x is a multigraph with two vertices and two parallel edges.*
- *R-node: then G_x is a 3-connected graph.*

In Figure 3.11, a *SPQR-tree* T with the different kinds of nodes is illustrated. Each edge (x, y) of T associates one edge of G_x with one edge of G_y . These edges are called *virtual*.

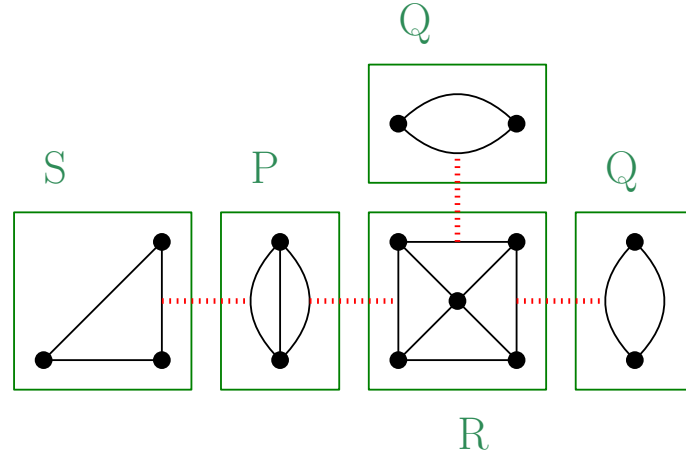
Mnich et al. designed an algorithm which takes a graph G as input and returns if G is a map graph with outerplanar witness and even calculates the outerplanar witness graph if the answer is yes.

At first, they simplify to biconnected components in the following way:

Lemma 3.13 ([MRS18]). *A map graph has a planar (outerplanar, tree) witness if and only if all of its biconnected components have planar (outerplanar, tree) witnesses, respectively.*

The idea of the proof is that if we have a planar (outerplanar, tree) witness for every biconnected component, we can just connect them via the cutvertices (1-separators) of G . In the other direction, if we have a planar (outerplanar, tree) witness H for G , every induced subgraph is planar (outerplanar, tree) and we get the planar (outerplanar, tree) witnesses of the biconnected components $G' = (V', E')$ by taking $H[V']$.

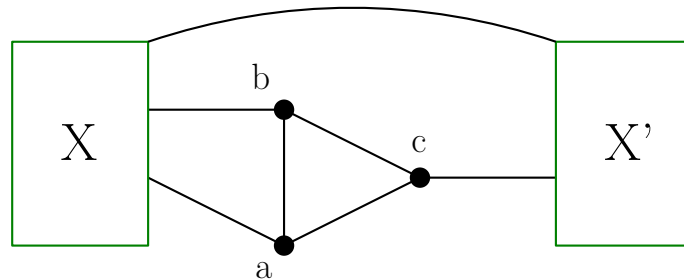
The first important result, following almost directly out of Lemma 3.13 is the following theorem:


 Figure 3.11: A $SPQR$ -tree T with edges of T drawn in red.

Theorem 3.14 ([MRS18]). *A map graph has a tree witness if and only if each of its biconnected components is a clique.*

Proof. Given Lemma 3.13, to prove Theorem 3.14, it suffices to show that a biconnected map graph has a tree witness if and only if it is a clique: A clique always has a tree witness with the pizza. For the other direction, assume that we have a biconnected graph $G = (V, E)$ with tree witness H and G is not a clique. Since H can not be a pizza, it has at least three levels, resulting in always having two adjacent non-leaf vertices, with exactly one of them being a witness vertex and one of them being a vertex of $v \in V$. Then, v is a cut vertex which contradicts the assumption of being biconnected. $\square$

Regarding outerplanar witnesses, we need a few more steps. A clique C is called *represented* by a witness point w if there is a vertex w in the witness graph with $N(w) = C$. By proving that for a graph G with outerplanar witness H each maximal clique C of size at least 4 is represented by one witness point, and naturally having the same for cliques of size 2, the subsequent step is to look at cliques of size 3. For those cliques, Mnich et al. prove that for a graph G that maximizes the number of maximal cliques of size 3 that are drawn as a pizza, every non-pizza maximal 3-clique $\{a, b, c\}$ contains a 2-separator $\{a, b\}$ that separates c from every other maximal clique that contains $\{a, b\}$. This can be imagined as in Figure 3.12, with the outerplanarity getting contradicted, since b may not touch the outer face anymore.


 Figure 3.12: A non-pizza 3-clique $\{a, b, c\}$ if the separator $\{a, b\}$ does not separate X and X' .

As a consequence, it can be proven that for a biconnected map graph G with an outerplanar witness H , every 2 separator separates exactly two components. At this point, it is possible

to decompose the biconnected components of G into $SPQR$ -trees. The witness for the graph $G = (V, E)$ is then constructed by starting with a graph $H = (V, \emptyset)$. Firstly, for each R -node R of one of the $SPQR$ -trees, if the skeleton of R is a clique of non-virtual edges, add a witness point w to H with $N(w) = V(R)$. Else, we have no outerplanar witness. Then, for each S -node S whose skeleton is a clique of size 3 with some real edge, add a witness point w to H with $N(w) = V(S)$. Lastly, for every edge $e = (a, b) \in E$ which has no common neighbour, add a witness point w with $N(w) = \{a, b\}$. If the construction succeeded, we have a witness graph which can be tested for outerplanarity. If the test fails or the construction did not succeed, G is no map graph with outerplanar witness. This algorithm runs in $\mathcal{O}(|V| + |E|)$ time [MRS18].

3.5 Recognition of Map Graphs of Bounded Treewidth

In 2024, Angelini et al. introduced a FPT algorithm to recognize map graphs parametrized by treewidth [ABDL⁺24]. Their first important result is the following. A graph is called *quadrangulated* if each face is bounded by exactly four edges. An example is illustrated in Figure 3.13.

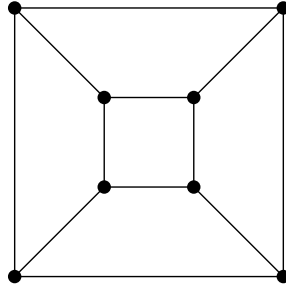


Figure 3.13: A quadrangulated graph.

Theorem 3.15 ([ABDL⁺24]). *A graph is a hole-free map graph if and only if it admits a quadrangulated compact witness that is biconnected.*

Proof Sketch. [[ABDL⁺24]]

" $\Leftarrow$ "

Given a witness H , by Theorem 2.3 we already know that G is a map graph, so, we need to show that it is hole-free. Then, for each face f of H , add one vertex inside f and connect it to all boundary vertices of f . The result is a triangulated graph with faces that are all adjacent to exactly one non-witness vertex. Therefore, we build the map by occupying for each nation exactly the triangulated faces that touch the associated vertex in H . The result is a hole-free map of G .

" $\Rightarrow$ "

We already know that a map graph is hole free, hence, the faces of its witness have exactly 4 or 6 vertices on their boundary [CGP02]. So, we need to rule out the case with 6 boundary vertices. But, in order to be hole free, the 3 nations assigned to the vertices on the boundary need to "reach out" in the middle of this face and form some intersection point where all three meet. By using the right construction, one can always choose such a representation, resulting in a quadrangulated compact witness that is biconnected. $\square$

Let $\mathcal{T} = (T_{\mathcal{T}}, \beta_{\mathcal{T}})$ be a nice tree decomposition of width $t = \omega - 1$. So, each bag $\beta_{\mathcal{T}}(x)$ contains at most ω vertices. Also, let $T_{\beta_{\mathcal{T}}(x)}$ denote the subtree of $T_{\mathcal{T}}$ rooted at $\beta_{\mathcal{T}}(x)$. Then, $G_{\beta_{\mathcal{T}}(x)} = (V_{\beta_{\mathcal{T}}(x)}, E_{\beta_{\mathcal{T}}(x)})$ is the induced subtree of the vertices of the bags of $T_{\beta_{\mathcal{T}}(x)}$

and $H_{\beta_{\mathcal{T}}(x)} = (V_{\beta_{\mathcal{T}}(x)}; W_{\beta_{\mathcal{T}}(x)}; E'_{\beta_{\mathcal{T}}(x)})$ a compact witness of $G_{\beta_{\mathcal{T}}(x)}$. A vertex $v \in V_{\beta_{\mathcal{T}}(x)}$ is called *anchor vertex* if $v \in \beta_{\mathcal{T}}(x)$, a vertex $w \in W_{\beta_{\mathcal{T}}(x)}$ is called *anchor vertex* if $N(w) \subseteq \beta_{\mathcal{T}}(x)$. Further, we call a face f of $H_{\beta_{\mathcal{T}}(x)}$ *active* if it has only one vertex on its boundary (then, the whole witness consists only of one vertex) or if there are at least two anchor vertices on its boundary. For a face f , the *active boundary* is the boundary of f without all non-anchor vertices on it. For the active boundary, any non-anchor vertex v is *shortcutted* by deleting it and its adjacent edges on the boundary of f and adding an edge between the vertices adjacent to v on the boundary of f . Then, the *embedding sketch of $H_{\beta_{\mathcal{T}}(x)}$ with respect to $\beta_{\mathcal{T}}(x)$* (short: sketch) is the embedded graph $S(H_{\beta_{\mathcal{T}}(x)}, \beta_{\mathcal{T}}(x))$ including only vertices and edges of active boundaries. Compare to Figure 3.14. Note that, because of the shortcuts, a sketch does not have to be bipartite.

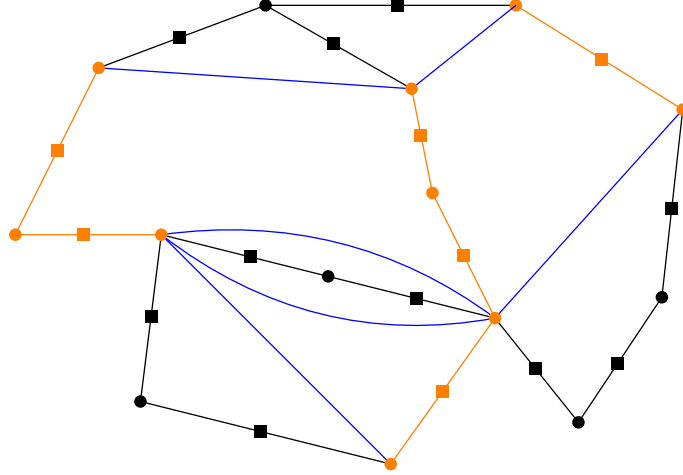


Figure 3.14: An embedding sketch of a witness $H_{\beta_{\mathcal{T}}(x)} = (V_{\beta_{\mathcal{T}}(x)}; W_{\beta_{\mathcal{T}}(x)}; E'_{\beta_{\mathcal{T}}(x)})$ with respect to $\beta_{\mathcal{T}}(x)$. The dots are the vertices of $V_{\beta_{\mathcal{T}}(x)}$ and the squares are the vertices of $W_{\beta_{\mathcal{T}}(x)}$. The anchor vertices are marked orange, the shortcuts are marked blue and the active boundary is everything marked blue and orange.

The algorithm makes progress via a dynamic program on the nice tree decomposition $\mathcal{T} = (T_{\mathcal{T}}, \beta_{\mathcal{T}})$ similar to the previous chapters. During this process, each node of $T_{\mathcal{T}}$ is equipped with a *record* $R_{\beta_{\mathcal{T}}(x)}$, a set of sketches with certain properties, depending on the wanted result.

Definition 3.16 ([ABDL⁺24]). *Let R_X be a record. Consider the following properties:*

- F1** *For every compact witness $H_{\beta_{\mathcal{T}}(x)}$ of $G_{\beta_{\mathcal{T}}(x)}$, $R_{\beta_{\mathcal{T}}(x)}$ contains its sketch $S(H_{\beta_{\mathcal{T}}(x)}, \beta_{\mathcal{T}}(x))$.*
- F2** *For every entry $r \in R_{\beta_{\mathcal{T}}(x)}$, there is a compact witness $H_{\beta_{\mathcal{T}}(x)}$ of $G_{\beta_{\mathcal{T}}(x)}$ such that $r = S(H_{\beta_{\mathcal{T}}(x)}, \beta_{\mathcal{T}}(x))$.*
- F3** *$R_{\beta_{\mathcal{T}}(x)}$ contains no duplicates.*

*The record $R_{\beta_{\mathcal{T}}(x)}$ is called feasible if the Properties **F1**–**F3** hold.*

A feasible record $R_{\beta_{\mathcal{T}}(x)}$ is called k -map feasible if it contains a non-empty subset $R_{\beta_{\mathcal{T}}(x)}^ \subseteq R_{\beta_{\mathcal{T}}(x)}$, called a subrecord, such that:*

- F4** *For every entry $r \in R_{\beta_{\mathcal{T}}(x)}$, it holds that $r \in R_{\beta_{\mathcal{T}}(x)}^*$ if and only if r contains no witness vertex u with $\deg(u) > k$.*

A feasible record $R_{\beta_{\mathcal{T}}(x)}$ is called hole-free feasible if it contains a subrecord $R_{\beta_{\mathcal{T}}(x)}' \subseteq R_{\beta_{\mathcal{T}}(x)}$, such that:

F5 *For every entry $r \in R_{\beta_{\mathcal{T}}(x)}$, it holds that $r \in R'_{\beta_{\mathcal{T}}(x)}$ if and only if r contains no witness vertex u with $\deg(u) > k$ and each complete active boundary of r consists of exactly 4 vertices.*

So, the algorithm starts at the leaf nodes of $T_{\mathcal{T}}$ and calculates their records. Subsequently, it calculates the records of their parents depending on the label of the node by taking the records of the children. A detailed description of the different operations can be found in [ABDL⁺24]. Then, G is a k -map graph respectively a hole-free k -map graph if and only if the algorithm reaches the root of $T_{\mathcal{T}}$ and the record of the root is k -map respectively hole-free feasible [ABDL⁺24]. This algorithm runs in linear time and even returns a witness for yes-instances by storing additional pointers for each record and reconstructing it backwards. Also, a lowest k can be found using this algorithm: Since k is in the range $[1, t + 1]$, one can use a binary search and needs to perform the algorithm $\mathcal{O}(\log t)$ times, which does not change the overall asymptotic running time [ABDL⁺24].

4. Recognizing Map Graphs

Thorup tries to recognize map graphs in polynomial time directly by viewing the solution as a map [Tho98a]. The main strategy is to reduce instances of the problem in various steps to progressively simpler instances. He proves for a subgraph that it is a map graph with an empty, closed region, where the rest of the solution can be embedded. But instead of remembering all representations of the partial solution as maps, he only remembers the possible orders of vertices around that region.

In contrast, we want to find a solution by finding a witness instead of a map, i.e. the goal is to find a witness for the whole graph by finding a witness for a subgraph and try to find a solution for the rest of the graph. Then, this solution will be embedded in a face of the witness of the subgraph.

Since there could be more than one solution, we do not have the concrete drawing of the partial witness, but we give the possible vertex orders of the partial witness around the face in which we want to embed the solution to the smaller problem. To represent these orders, we use a *PQ*-tree.

The idea is to recognize map graphs recursively. We look at a given graph $G = (V, E)$, prove that there is a map of a subgraph of G , which we set as "outer" map, with a (in witness perspective) face, where we can embed a solution of the remaining graph G . The possible orders of the "outer" nations touching these faces are represented in a *PQ*-tree. Of course, we only have to take this much effort if we have a non-planar graph. So, at first, we perform a planarity test, which can be done in linear time [BL76]. If it succeeds, we know it is a map graph, since every planar graph is a map graph. If the planarity test fails, there is more work to be done. We start by defining our recursive problem.

4.1 Local Map Problem

The abstract concept that is used for the recursion is called *Local Mapping Problem* (LMP). Subsequently, we perform various steps on the LMP and need to prove for each step that it is correct and that it does not exclude solutions. We start by giving some definitions. First, for a graph $G = (V, E)$ and a vertex $w \in V$ with neighbours $u, v \in V$, let $R_{u,v}(w)$ be the clockwise ordered neighbourhood of w without u and v starting from u .

Definition 4.1. (*Vertex order around a face regarding V*)

Given an embedding of a planar graph $G = (V \cup W, E)$ and a face f in this embedding and let π be the order of the vertices appearing on a cyclic, clockwise facial walk around f .

Then, obtain π' of π by replacing all $w \in W$ by $R_{u,v}(w)$, where $u \in V$ is the vertex before w in the order and v is the vertex after w in the order. We call π' the vertex order around a face f regarding V and denote it as $s_V(f)$.

These orders will later be the ones stored in a PQ -tree σ . Here, we need to pay attention to boundary segments as a and b in Figure 4.1 as such vertices would appear twice in the vertex order around a face. We call such potential segments *parallel boundary segments* if they connect to a face f only via a set of witness points W_p with $W_p \geq 2$. Similar to Thorup, we assume that there are no parallel segments in the boundary [Tho98a].

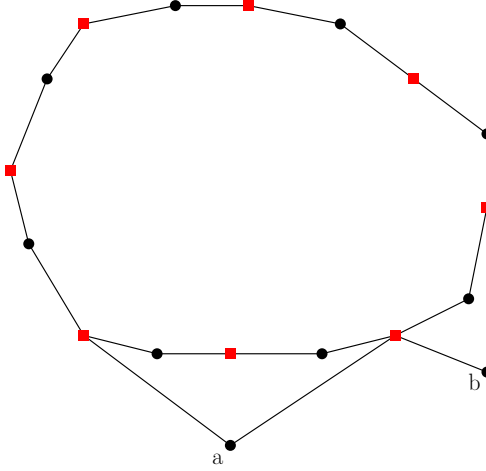


Figure 4.1: A face with parallel segments in the boundary.

We continue with further definitions for witness graphs. For a bipartite planar graph $H = (V; W; E)$ and some set of dummy-witnesses $W_D \subseteq W$, let $H_D = (V; W'; E')$ be the graph without dummy-witnesses and their edges, where $W' = W \setminus W_D$ and $E' = E \setminus E_D$, with E_D being the set of all edges adjacent to vertices in W_D .

For a bipartite graph $H = (V; W; E)$ and some family of subsets $\mathcal{A}$ of V , let W_{A_i3} include for each 3-clique which is drawn as hamantach the three witness vertices $w_1, w_2, w_3 \in W$ with $N(w_j) \in A_i$ and $N(w_j) \neq N(w_k), j \neq k; j, k \in \{1, 2, 3\}$. Also, let $W_{\mathcal{A}} = \{w \in W | \exists A_i : N(w) = A_i\} \cup W_{A_i3}$ and let $E_{\mathcal{A}}$ be the set of all edges adjacent to vertices in $W_{\mathcal{A}}$. Then $H_{\mathcal{A}} = (V; W''; E'')$ with $W'' = W \setminus W_{\mathcal{A}}$ and $E'' = E \setminus E_{\mathcal{A}}$ is the graph ignoring that the elements of $A \in \mathcal{A}$ form cliques.

Definition 4.2. A planar bipartite graph $H = (V; W; E)$ with a set $X \subset V$ is called X -compatible with a vertex order s if the following conditions hold:

- (i) $H_{\mathcal{A}}$ has a face f with $s_{V \setminus X}(f) = s$.
- (ii) $(H_D)^2[V \setminus X] = G[V \setminus X]$.

With these definitions, we are able to define the LMP.

Definition 4.3. (Local Mapping Problem)

Given a graph $G = (V, E)$ together with

- a set of interior vertices $X \subseteq V$,
- a set of boundary vertices $B \subseteq V \setminus X$ with $N(X) \subseteq B$,
- a set of dummy witness vertices W_D , ensuring the form of our LMP without giving rise to new neighborhoods,

- a PC-tree σ with leaf set B , representing the set S of all possible vertex orders s of B around a face f , for which there is a X -compatible witness graph $H_1 = (V \setminus X; W_1 \cup W_D; E_1)$,
- point clusters $\mathcal{A}$, a family of subsets of B , where every $A \in \mathcal{A}$ represents a clique on B , which can not be realized outside of the boundary of f ,
- a set of boundary points $P \subseteq B$, which are vertices adjacent to only one witness vertex on the boundary of f .

Let $M = (X, B, \sigma, P, \mathcal{A}, W_D)$ be the local mapping problem. Its goal is to find a witness graph $H_2 = (X \cup B; W_2; E_2)$ such that $G[X] = ((H_2)_D)^2[X]$ that satisfies the following conditions:

- (i) For $x \in X$ and $b \in B$, x and b have a common neighbour in $W_2 \setminus W_D$ if and only if $(x, b) \in E(G)$.
- (ii) The vertex order $s_{X \cup B}(o)$ around the outer face o is a legal ordering of σ .
- (iii) For $A \in \mathcal{A}$, all vertices $a \in A$ form a clique.
- (iv) Boundary points $p \in P$ are vertices having only one neighbour $t \in W_2 \setminus W_D$.
- (v) For every $w_2 \in W_2 \setminus W_D$ touching o , there is a $w_1 \in W_1 \setminus W_D$ touching f with $N(w_1) \subseteq N(w_2)$.

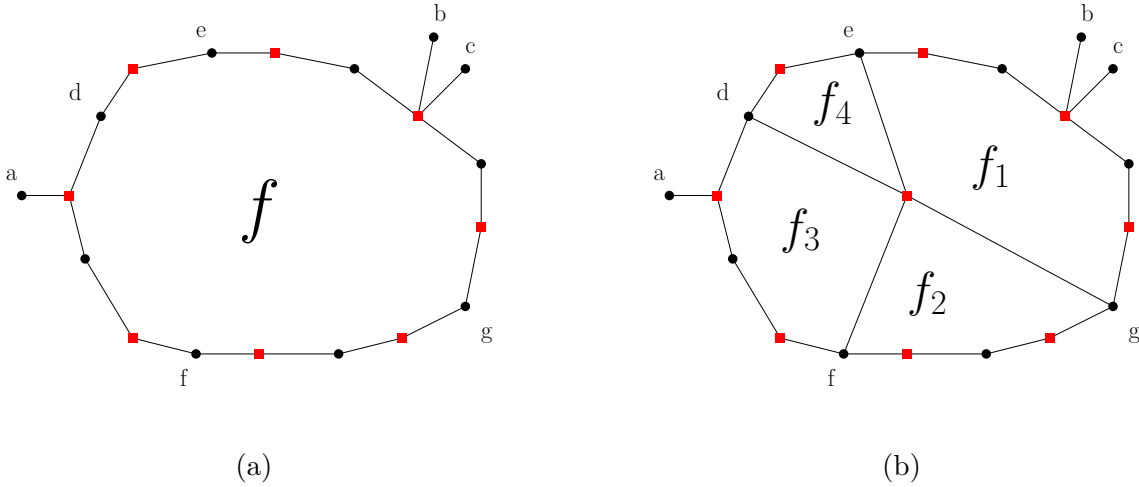


Figure 4.2: Setting of a LMP with the black dots being the set B of boundary vertices around the face f , (a) once ignoring that $A = \{d, e, f, g\}$ needs to form a clique, (b) once respecting that it needs to be connected.

An example setting of a LMP is illustrated in Figure 4.2(a). The goal is to embed our interior vertices X and some witness vertices into the face f , such that the half-square of the result is the graph $G[X \cup B]$. Note that here, the order of the vertices around f seems fixed, but in our LMP, the possible vertex orders are given by σ . In the example, the vertices a, b, c are represented as boundary points. Since every vertex in P has to be represented as a boundary point, we have $P \subseteq \{a, b, c\}$. Regarding $\mathcal{A}$: an $A \in \mathcal{A}$ might be for example $A = \{d, e, f, g\}$. Those vertices are a clique in G , but are not yet realized as a clique in the outer witness graph. The result is that we have to connect them through f , splitting f into several so called *bubbles* f_i as visualized in Figure 4.2(b).

Mostly, such a clique A actually needs to meet in a single witness point in f or on its boundary. It can not be a hamantach or a pizza with crust except for $|A| = 3$, which we show in the following.

Lemma 4.4. *A clique $A \in \mathcal{A}$ of a LMP can only be represented as a hamantach or pizza with crust if $|A| = 3$. Else, A always has to be represented as a pizza.*

Proof. Since we can not connect A outside of the face f , we need to be able to draw A outerplanar. There are the following cases:

Case 1: $|A| < 3$. For a hamantach or a pizza with crust you need at least three vertices.

Case 2: $|A| > 3$.

Case 2.1: A is drawn as a hamantach: Consider a clique $A = \{d, e, f, g\}$ and let d, e be the vertices on the same side of the hamantach, compare Figure 4.3(a). If you now identify f and g with each other, since no edges can go around the witness vertices inside our face, we get a graph $K_{2,3}$, which is not outerplanar [CH67].

Case 2.2: A is drawn as a pizza with crust: Again, consider a clique $A = \{d, e, f, g\}$ and let d, e, f be the pizza-part of our pizza with crust, compare with Figure 4.3(b). If we again identify the pairs (f, g) and (d, e) with each other, we get a $K_{2,3}$ once more, which is not outerplanar [CH67].

Case 3: $|A| = 3$. Here, a pizza with crust is identical with a hamantach. It looks like Figure 4.3(c). In that regard, it actually makes a difference to connecting the clique as a pizza, as, for instance, the connected component drawn in Figure 4.4 could not be realized if the clique was drawn as pizza. $\square$

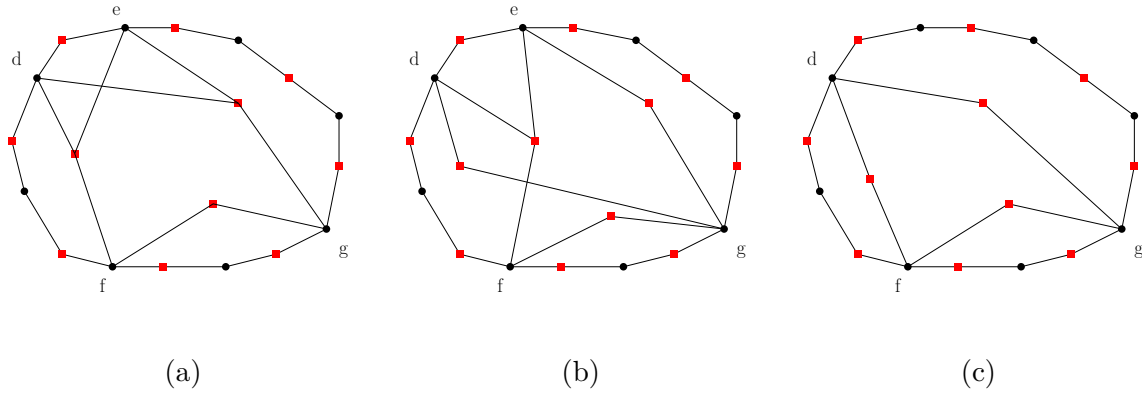


Figure 4.3: A 's in witness perspective: (a) A hamantach with $|A| > 3$, (b) a pizza with crust and (c) a hamantach for $|A| = 3$.

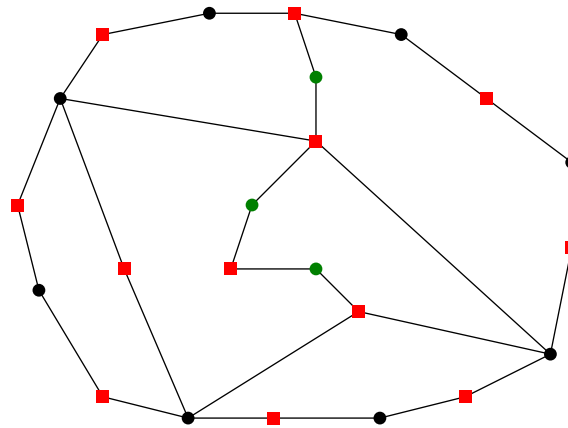


Figure 4.4: Hamantach with $|A| = 3$.

In witness perspective, the here defined LMP is quite similar to Thorup's map perspective, but there are also some differences. For example, Thorup let the cliques A always intersect in exactly one point and does not allow the clique to form a hamantach. Also, we decided

to introduce dummy-witnesses to always have a closed face in which we can embed the solution.

4.2 Initiate the LMP

To begin, we want to choose a boundary set B which is not too small, such that the recursion makes sense. Therefore, let us reduce the problem to the case where $|B| \geq 4$. To do so, we first look at cuts C of size ≤ 3 as Thorup did. If $G|X$ is connected, $G//X$ denotes the graph obtained from G by contracting X to a single vertex x .

Lemma 4.5 ([Tho98a]). *Suppose C is a cut of size ≤ 3 such that $G \setminus C$ contains at least two non-empty connected components X and X' with $N(X) = N(X') = C$. Then, G can be mapped if and only if both $G//X$ and $(G|(X \cup C \cup X'))//X'$ can be mapped. If $|C| = 1$, it suffices that $G \setminus X$ and $G|(X \cup C)$ can be mapped.*

Proof. We can always paste together the solution of $G//X$ and $(G|(X \cup C \cup X'))//X'$, since all orders of three boundary vertices are equivalent. One order π of size at most 3 can always be connected without crossing to another order π' of size at most 3 or its flipped version $\bar{\pi}'$. $\square$

By applying the lemma until no such cut is left, we receive biconnected components $G^* = (V^*, E^*)$. Since our graph is not planar, we have at least one clique with more than 3 vertices. Suppose we can guess a pizza Z with more than three slices, then we can use Z as boundary B for our initial local mapping problem with $X = V^* \setminus Z$ and no constrictions on the ordering of our PQ -tree σ , since we have a clique.

To guess a pizza with at least four slices, firstly, we have to calculate all maximal cliques with a size of at least 4. This works in polynomial time and returns a polynomial number of cliques [CGP98].

Now, we guess for each maximal clique Q that it is a pizza, then use it as a boundary and try to solve the local mapping problem. If there is no solution for any such Q , we guess for each Q , that it is a pizza with crust, trying to solve the problem considering all maximal cliques Q minus one vertex. If there is still no solution (for the graph to be a map graph), every non-planar clique Q must be part of a hamantach.

If a hamantach H representing a clique Q has one side with only one vertex, we would have found the solution considering all maximal cliques minus one face. Thus, every side of the hamantach has at least two vertices. If now one vertex which is not part of the hamantach touched one of the corners, this would create a pizza of size at least 5, which we would have found before. So, each connected component of $G^* \setminus H$ is neighbouring at most three vertices in H since it is not touching any corner of H . This leads to a case distinction over the amount of components touching three vertices of H , how and if we can represent our graph G as a map.

- If more than two components touch three vertices of H , G is not a map graph.
- If two components touch three vertices of H we have one set of three vertices "outside" and one set "inside" (symmetrical, so equivalent if inside or outside). Then, check if other components (touching two faces) can be added.
- If one component touches three vertices of H , we set those three vertices as inside of the hamantach (symmetrical). Then, check if each pair of vertices of the hamantach H , which is a neighbourhood of a connected component of $G^* \setminus H$ and is not already considered as inside of the hamantach, can be placed next to each other on one side of the hamantach or both on the outside of the hamantach.

- If no component touches three faces of H it suffices to check if each pair of faces of H , which is a neighbourhood of a component can be placed next to each other on one side of the hamantach or both on the inside or outside respectively.

4.3 Reducing the Boundary

We now want to look at the case if $N(X) \neq B$. The goal is to reduce the boundary such that $N(X) = B$. We can trivially delete boundary vertices that have no neighbours in the interior and are not part of a clique $A \in \mathcal{A}$. For boundary segments, one can imagine that it is like taking an abbreviation and connecting the neighbouring two boundary segments via a dummy witness. As for the boundary points, we can just delete them, since if they are not part of any $A \in \mathcal{A}$, they are already connected to every neighbour of them outside of our boundary. In this context, Thorup states the following. Note that, here, we use Thorups definition of a LMP, which is similar but, for example, does not use dummy-witnesses.

Lemma 4.6 ([Tho98a]). *Let $M = (X, B, \sigma, P, \mathcal{A})$ be a local mapping problem. There is no legal mapping of M if one of the following two conditions is satisfied:*

- (i) *There is an $A \in \mathcal{A}$ with $P \cap A \cap N(X) \neq \emptyset$ and $A \not\subseteq N(X)$.*
- (ii) *There is an $A \in \mathcal{A}$ such that $A \not\subseteq N(X)$ and $|A \cap N(X)| > 2$.*

Otherwise, there is a mapping of M if and only if there is one for

$$(X, (B \cap N(X)), \sigma|_{(B \cap N(X))}, P \cap N(X), \mathcal{A}|_{(B \cap N(X))}),$$

where $\mathcal{A}|_{(B \cap N(X))} = \{A \cap N(X) \mid A \in \mathcal{A}\}$.

Proof. [Tho98a] Concerning (i), we know that there is a single point face $p \in A \cap N(X)$. However, then all faces in A meet in p , so a face in X incident to p must be incident to all of A , contradicting $A \not\subseteq N(X)$. The same is true for (ii), for the faces in A meet in a single point, so there can be at most two of them not being single points. Hence, from $|A \cap N(X)| > 2$, it follows that X is incident to all of A , contradicting that $A \not\subseteq N(X)$. $\square$

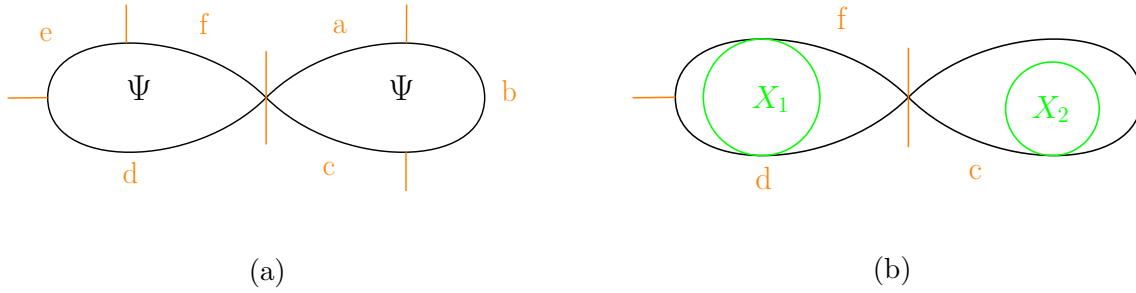
Although it gets clear that there is no solution if (i) or (ii) holds, we unfortunately do not get a proof for the "otherwise"-statement. Also, note that Thorup states in [Tho98a], " ρ may intersect itself and in this way give rise to more intersections of boundary faces", which leads to the following counterexample:

Let $M = (X, B, \sigma, P, \mathcal{A})$ with

- $X = \{x_1, x_2\}$,
- $B = \{a, b, c, d, e, f\}$,
- σ being one Q-node as root with all of B in alphabetic order as children,
- $P = \emptyset$,
- $\mathcal{A} = \{\{f, a, d, c\}\}$,
- $G[X] = (V_X, E_X)$ with $V_X = \{x_1, x_2\}$ and $E_X = \{(x_1, d), (x_1, f), (x_2, c)\}$.

The setting is illustrated in Figure 4.5(a).

Obviously, for this LMP, there is a legal mapping, see in Figure 4.5(b). But (ii) of Thorups's lemma is fulfilled by having an $A = \{f, a, d, c\}$ with $A \not\subseteq N(X)$ since $a \notin N(X)$ and $|A \cap N(X)| = |\{d, c, f\}| = 3 > 2$.


 Figure 4.5: (a) Setting and (b) solution of LMP M .

For boundary vertices $b \notin N(X)$ included in such an A , it seems not to be that simple to get rid of them as long as we might be dealing with disconnected components in the interior X . As soon as we have reduced our LMP to interiors with connected components X_i for every $N(X_i)$ and A we have $|N(X_i) \cap A| \leq 2$ or $|N(X_i) \cap A| = |A|$. In the first case, we obviously do not delete any vertex of A , since every vertex of A is in the neighbourhood of one X_i . In the second case, our connected component appears only in one bubble of f , with the vertices of A clustered in the boundary. Then A is already connected as a clique and we can delete A of $\mathcal{A}$ for this LMP and delete all vertices in B which are not in X_i . Hence, the next step is to simplify the LMP to LMPs with connected components and then to look at connected components having a whole A in their neighborhood and try to simplify them to LMPs within single bubbles.

4.4 Disconnected Interior Components

Let us now take a look at connected components of the interior X and how they interact with each other and with B . The goal of this section is to firstly adjust the PQ -tree in a way that it only represents vertex orders that allow each connected component $X_i \subset X$ to have a valid solution and secondly to give a PQ -tree for every such component to get a LMP for each X_i such that the global LMP is solvable if and only if every LMP of a connected component is solvable.

A σ -ordering π of B crosses two sets $N(X_1)$ and $N(X_2)$ if $x_1 \prec_\pi x_2 \prec_\pi x_3 \prec_\pi x_4$ with $x_1, x_3 \in N(X_1)$ and $x_2, x_4 \in N(X_2)$. Clearly, we have to avoid such crossings to get to a planar witness graph. Also, let $\mathcal{N} = \{N(X_i) | X_i \text{ is connected component of } X\}$, being exactly the set where two elements may not cross. Now, we do not have deleted every vertex $b_i \in B$ with $b_i \notin N(X)$, but have some of those boundary vertices $b_i \in A \in \mathcal{A}$ left. Those vertices b_i clearly all belong to some A , which is not completely included in a neighbourhood of a connected component of the interior. This leads to two options for every $A \in \mathcal{A}$: Either being completely contained in a neighbourhood $N(X_i)$ of a connected component X_i of X , which will be solved by solving the LMP for the component. Or not being completely contained in a neighbourhood $N(X_i)$ of a connected component X_i of X , where we need to pay attention that the vertices of A can be connected in the interior. This can be achieved by handling those A as an independent neighbourhood, which may not be crossed by another set. So, we adjust Thorup's definition of $\mathcal{N}$ in the following way: $\mathcal{N} = \{N(X_i) | X_i \text{ is connected component of } X\} \cup \{A \in \mathcal{A} | \forall X_i : A \not\subseteq N(X_i)\}$. This would also work for our "special" case of an A with $|A| = 3$ being a hamantach, since if a connected component only touches two vertices of A , it does not matter how we draw A and otherwise the whole A is already in the neighbourhood of our component and this component is the only one being in the "middle" bubble of the hamantach.

Thorup states here, again without complete proof, the following. Note that, again, we use Thorup's definition of a LMP, which can be easily transferred to our version.

Lemma 4.7 ([Tho98b]). *Let $M = (X, B, \sigma, P, \mathcal{A})$ be a local mapping problem with X dividing into disjoint connected components $X_1, \dots, X_k$. There is no legal mapping of M if one of the following conditions is satisfied:*

- *For some X_i , $(\sigma, P, \mathcal{A})|_{N(X_i)}$ is undefined.*
- *There is an $A \in \mathcal{A}$, with $a_1, a_2, a_3 \in A$, such that for $i = 1, 2, 3$, there is a X_j with $a_i \in N(X_j)$ and $A \not\subseteq N(X_j)$.*
- *There are $a_1, a_2, a_3 \in B$, and X_i and X_j , $i \neq j$, such that for $i = 1, 2, 3$, $a_i \in N(X_i) \cap N(X_j)$.*

Else, there is a solution to M if there exists a non-crossing σ -ordering π such that all $M_i = (X_i, N(X_i), \pi|_{N(X_i)}, P|_{N(X_i)}, \mathcal{A}|_{N(X_i)})$ have a solution.

Again, it is clear that we do not have a legal mapping if one of the conditions is fulfilled, but the "Else"-case stays unclear and unproved.

To find such non-crossing σ -orderings π and the PQ -trees for the LMPs to the connected components, Thorup presents the following theorem and corollary:

Theorem 4.8 ([Tho98b]). *Given a PQ -tree σ with leaf set B , and a family $\mathcal{N}$ of subsets of B , in polynomial time, we can decide if there is a σ -ordering π of B that does not cross any two sets in $\mathcal{N}$. Moreover, we can then find a PQ -tree σ_N for each $N \subseteq B$ satisfying the following conditions:*

- *If an ordering π is a σ -ordering that does not cross any two sets in $\mathcal{N}$, then for each $N \in \mathcal{N}$, $\pi|_N$ is a σ_N -ordering.*
- *Conversely, if for each $N \in \mathcal{N}$, π_N is a σ_N -ordering of N , there exists a σ -ordering π that does not cross any two sets in $\mathcal{N}$, and such that for each $N \in \mathcal{N}$, $\pi|_N = \pi_N$ or $\pi|_N = \overline{\pi_N}$.*

Corollary 4.9 ([Tho98b]). *Given a local map problem with disconnected interior, in polynomial time, we can either decide that there is no solution, or for each component X_i we can find a PQ -tree σ_{N_i} , so that the local map problem can be solved if and only if each (X_i, σ_{N_i}) can be solved.*

At first, one can see that if we take the original PQ -tree, connect neighboured children of Q -nodes via edges, add one vertex n_i per $N_i \in \mathcal{N}$ and add an edge (n_i, x) for each $x \in N_i$, we get a graph that is planar if and only if there is a valid solution to the LMP M .

To prove the theorem, an algorithm gets introduced. In that regard, some definitions are necessary. We say a set $N \subseteq B$ *intersects* a node v in a tree σ if N contains a leaf descending from v . Let $\mathcal{N}_v$ be the family of $N \in \mathcal{N}$ that intersect v .

To find the PQ -tree σ_{N_i} for each N_i respectively for each connected component X_i , we look at the vertices of σ from the bottom upwards layer by layer:

Given a LMP M and a set $\mathcal{N}$, go through all non-leaf nodes v of σ , such that you handle v only if all of its non-leaf children are already handled.

If there is no $N \in \mathcal{N}$ such that for the vertex v we have $\text{lca}(N) = v$, we mark v as handled. Else, let $\mathcal{N}_v^+$ be the set of all $N \in \mathcal{N}$ with $\text{lca}(N) = v$. Let

$$N^* = \{\bigcup N \mid N \in \mathcal{N}_v \setminus \mathcal{N}_v^+\} \cap L(v).$$

If $N^* \neq \emptyset$, we add c^* as new leaf child to v on one end and we add c^* to N^* . Now, we run the following algorithm with root v , the induced subtree σ_v under v and $\mathcal{N}' = \{N^*\} \cup \mathcal{N}_v^+$.

Let C be the set of children of v and $C_i \subseteq C$ the children intersected by a set $N_i \in \mathcal{N}'$ and let $X_{c,c'}(\mathcal{N}) = \{N \in \mathcal{N} \mid N \text{ intersects } c \text{ and } c'\}$.

Let us assume v is a P -node. Now, for each $N_i \in \mathcal{N}'$, do:

If $|C_i| \geq 3$:

-
- 1 Set $\mathcal{N}_i := \mathcal{N}'$ and $\sigma_i := \sigma$.
 - 2 $\forall c \in C \setminus C_i$ replace $\mathcal{N}_c$ with $\bigcup \mathcal{N}_c$ in $\mathcal{N}_i$
 - 3 $\forall N \in \mathcal{N}_i, \forall c \in C$, cluster $N \cap L(c)$ in σ_i
 - 4 $\forall X_{c,c'}(\mathcal{N}_i \setminus N_i)$ cluster $L(c) \cap (\{N_i\} \cup X_{c,c'}(\mathcal{N}_i \setminus N_i))$ and $L(c') \cap (\{N_i\} \cup X_{c,c'}(\mathcal{N}_i \setminus N_i))$ in σ_i
 - 5 $\forall N \neq N_i$ not intersecting two $c, c' \in C$: Delete N from $\mathcal{N}_i$
 - 6 Set $\sigma_i = \sigma_i \cup \mathcal{N}_i$
 - 7 Cluster all $X_{c,c'}(\mathcal{N}_i \setminus N_i)$ in σ_i
 - 8 Set $\sigma_{N_i} := \sigma_i \mid N_i$
-

If $|C_i| = 2$, Thorup proposes to merge the the trees below the two children $c, c' \in C_i$ and proceed depending on the result.

If this is possible, we get a PQ -tree σ_{N_i} for each $N_i \in \mathcal{N}_v^+$ and a tree σ_{N^*} for the set N^* . Then, one can continue with the next vertex of σ , but with the subtree under v replaced by σ_{N^*} without c^* and $\mathcal{N} := \mathcal{N} \setminus \mathcal{N}_v^+$.

The intuition of the algorithm is the following: In step (1), we initialize. In step (2), we look at the children c of v , which are not intersected by the set N_i . Then, the set $\mathcal{N}_c$, which is the union of all $N \in \mathcal{N}_c$, needs to be clustered at the end, since they all intersect c without any other neighbourhood interfering. Because of Lemma 2.7, the clustering in this superset is independent of the global clustering and we can look at $\{\bigcup N \mid N \in \mathcal{N}_c\}$ instead of each $N \in \mathcal{N}_c$ individual. In step (3), we cluster each neighbourhood N beneath each child c . This is necessary because the only other option would be that there are complete neighbourhoods N' in between the elements of N , but then, we would have already handled and deleted those neighbourhoods N' because $\text{lca}(N') = c \neq v$. In step (4), we cluster every set of neighbourhoods N under each child c , if those neighbourhoods intersect c and a second children c' . This seems necessary due to the "arms" of the connected components related to each N running parallel. Step (5) ensures that there is still no neighbourhood $N \in \mathcal{N}_i$ with $\text{lca}(N) = c \neq v$ left, step (6) ensures that the tree also has none of those neighbourhoods left. In step (7), we cluster the each set of of neighbourhoods that intersect the same two children globally and in step (8), we return the PQ -tree for our LMP which only look at the connected component associated with N_i .

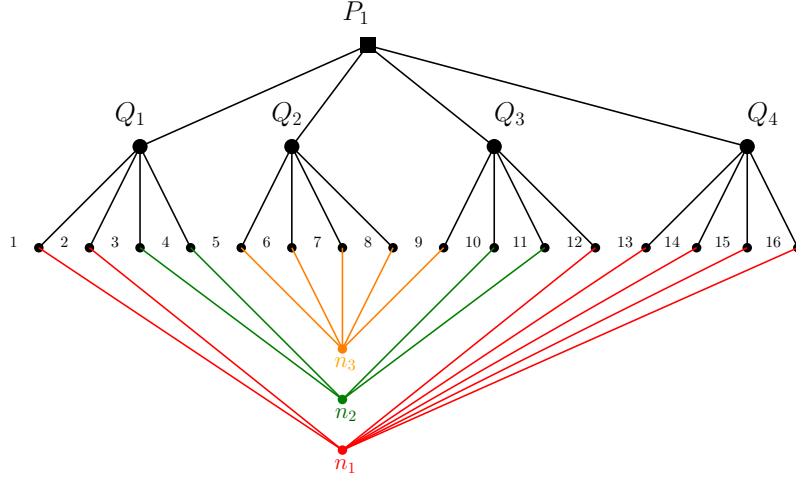
Unfortunately, the algorithm has three problems: being inprecise with the steps for the case $|C_i| = 2$, missing the case that v is a Q -node and a mistake in the algorithm for the case that v is a P -node and $|C_i| = 3$. For the latter, we can show a simple example, which contradicts line 4 of the algorithm:

Take the PQ -tree σ_e illustrated in Figure 4.6. Consider

$$\mathcal{N} = \{N_1 = \{1, 2, 12, 13, 14, 15, 16\}, N_2 = \{3, 4, 10, 11\}, N_3 = \{5, 6, 7, 8, 9\}\}.$$

Clearly, we can find a solution for this example, since if we connect neighboured children of the Q -nodes in the drawing, we instantly get a planar drawing.

Here we have the case that we do not have to handle all the Q -nodes as there is no N_i with $i \in \{1, 2, 3\}$ such that $\text{lca}(N_i) = Q_j$ for $j \in \{1, 2, 3, 4\}$. Therefore, the next node to look at is the P -node. We then have $\mathcal{N}_{P_1}^+ = \{N_1, N_2, N_3\}$ and $N^* = \emptyset$, which leads to $\mathcal{N}' = \mathcal{N}_{P_1}^+$. Let us start with trying to find σ_{N_1} , a PQ -tree with leaf set B_i for some inner connected component X_i with $N(X_i) = B_i$, representing every possible vertex-order on the boundary of X_i . Since $|C_1| = |\{Q_1, Q_3, Q_4\}| = 3$, we get to the above described case with v being a


 Figure 4.6: PQ -tree σ_e with marked neighbourhoods.

P -node and $|C_i| \geq 3$.

For initialisation, we set $\mathcal{N}_1 = \mathcal{N}' = \mathcal{N}_{P_1}^+$ and $\sigma_1 = \sigma_e$. Then, in the second step, we have to replace $\forall c \in C \setminus C_i$ every $\mathcal{N}_c$ with $\bigcup \mathcal{N}_c$ in $\mathcal{N}_i$. The only $c \in C$ affected is Q_2 , but since there is only one neighbourhood beneath Q_2 , there is nothing to unite. In step three, we need to cluster every neighbourhood underneath every child of P_1 , which is already the case. In step 4, if we have $X_{Q_2, Q_3}(\mathcal{N}_1 \setminus N_1) = \{N_3\}$. We now have to cluster $L(Q_2) \cap (\{N_1\} \cup X_{Q_2, Q_3}(\mathcal{N}_1 \setminus N_1))$ and $L(Q_3) \cap (\{N_1\} \cup X_{Q_2, Q_3}(\mathcal{N}_1 \setminus N_1))$, but the second one means, that we have to cluster $L(Q_3) \cap (N_1 \cup N_3)$, which is not possible since we can not cluster leaf 9 and leaf 12. So, we have the case where we have a valid solution, but the algorithm fails, which leads us to conclude that the presented algorithm is wrong.

4.5 Split Pizzas and Further Steps in the Algorithm

At this point, the idea and the global algorithm to recognize map graphs seems already to have too many mistakes for a quick fix, but let us assume we have reduced our problem to cases where $B = N(X)$ and X is connected.

Recall that a *bubble* be a part of the face f of the LMP, being bordered by the boundary and edges of some interior construct that splits the overall face. If we map in X in more than one bubble in our outer witness graph H , those bubbles need to be neighbored, with a witness point between them having an edge to at least one vertex in each bubble. We call such witness points *split points* and the set of all neighbours of a split point *split pizza*. We call a split pizza *proper* if in cyclic order around the split point the vertices of B in this order are divided in at least two groups. Note that the bubbles where we want to embed X can not be connected like a hamantach since we cannot connect the A s as a hamantach. The only exeption is when $|A| = 3$, which still does not lead to an hamantach-form of the bubbles.

Theorem 4.10 ([Tho98a]). *If X is connected, a set of safe pizzas and rice balls including all possible proper split pizzas is the set of cliques $F \cup S$, $F \subseteq X$, $S \subseteq B$ satisfying the following conditions:*

1. F is a maximal clique in X and $|F| > 1$.
2. S is the set of all vertices in B that neighbour all vertices in F and $|S| > 1$.
3. $S \cup F$ is a clique.

4. If we remove all edges (f_1, f_2) between vertices in F for which there is not a single vertex in $B \setminus S$ touching both f_1 and f_2 , then X splits into connected components $Y_1, \dots, Y_k$, $k > 1$, such that for any $i \neq j$, we have $N(Y_i) \cap N(Y_j) \cap B \subseteq S$.

The proof is done in two steps.

Lemma 4.11. *Every proper split pizza satisfies (1)-(4).*

Proof.

1. $|F| > 1$, since we have at least two interior vertices in our pizza to separate the boundary vertices in the pizza in order to get a proper split pizza. If F is no maximal clique in X , the maximal clique would be a pizza with crust or a hamantach. The first one would create a disconnected boundary and the second one is not possible because then, the bubbles need to form a hamantach, which we excluded.
2. Since a proper split pizza is a pizza and we include every vertex touching the central witness of the pizza to our split pizza, S is correct and for a split pizza to be proper, we have $|S| \geq 2$.
3. Holds, since it is a pizza.
4. Since we do not allow parallel boundary segments, there can not be a boundary vertex touching two bubbles of interior vertices without touching the split pizza.

□

Lemma 4.12. *Any clique $F \cup S$ classified by theorem 4.10 is a pizza or there is a contractible curve μ turning $F \cup S$ into a pizza.*

The idea of the proof is to make a case distinction over how F and how $F \cup S$ looks like. Since cliques are proven to be either a pizza, a pizza with crust, a hamatach or a planar clique, we get to the following cases:

Case 1: F is a planar clique, $F \cup S$ is a planar clique

Here, $|F| < 3$, since $|S| > 1$ and $|F \cup S| \leq 4$. Since $|F| > 1$, we know $|F| = |\{f_1, f_2\}| = 2$. So $F \cup S$ has to be a rice ball. One of the two f_i with $i \in \{1, 2\}$ always has to be the "middle vertex" of the rice ball, since else the two boundary segments would be parallel. Now, no boundary vertex $b \in B \setminus S$ can touch both f_1 and f_2 because it would be a parallel segment. Hence, there can not be a inner component X_i which touches f_1 and f_2 , otherwise (4) would not be fulfilled. So, we have two sets X_1, X_2 of the interior: A set X_2 , which always can be embedded in the face of the rice ball not touching f_1 and a set X_1 , which always can be embedded in the face of the rice ball not touching f_2 . Now we have two faces not occupied by B or X , where we can lay through our contractible curve as illustrated in Figure 4.7(a).

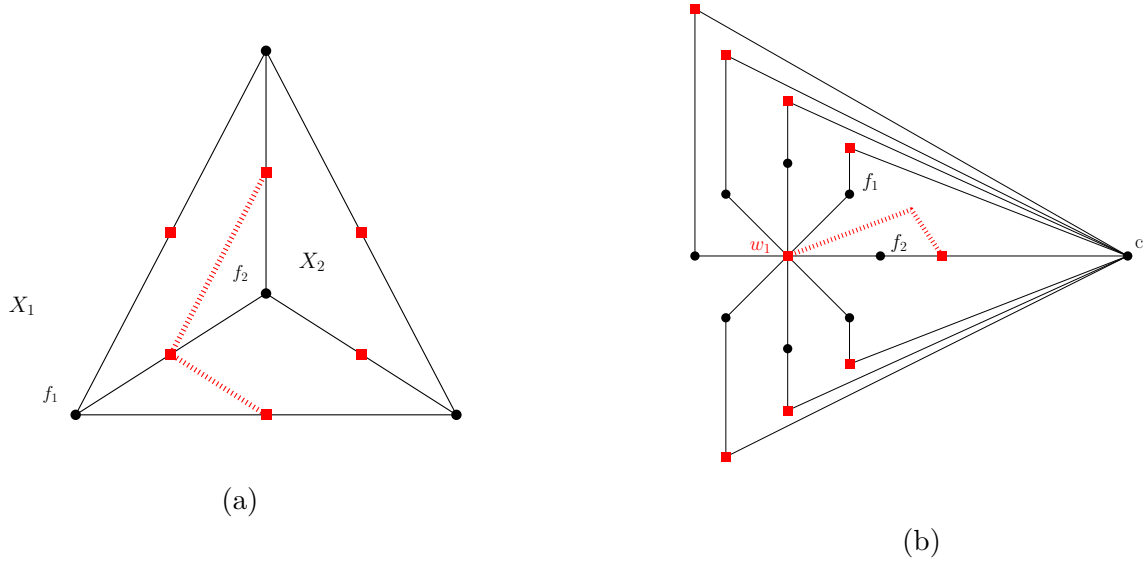


Figure 4.7: Contractible curves in cliques in map graphs: (a) Case 1 of the case distinction and (b) case 4 of the case distinction.

Case 2: F is a planar clique, $F \cup S$ is not a planar clique

Here, $|F| < 4$, since else there can not be a $s \in S$ embedded such that it is neighboured to all of F . So, F is automatically a pizza, a pizza with crust or a hamantach and we get to one of the following cases.

Case 3: F is a pizza, $F \cup S$ is a pizza

Then $F \cup S$ is already a pizza.

Case 4: F is a pizza, $F \cup S$ is a pizza with crust

Let w_1 be the witness point neighboured to all of F and to all of S but one vertex $c \in S$, which is the crust. Two elements of F must always lay next to each other, else we have a parallel segment to c . Now, assume we do not have a contractible curve. Then, there is a pair f_1, f_2 , that lies next to each other and it has to be connected independent of w_1 to "block" the contractible curve from the center. This connection must be built of vertices of X , since else we would have a parallel segment to c . But this also leads to f_1 and f_2 having a common neighbour $b \in B \setminus S$ in order to not contradict (4). This b then is parallel to the other connection of f_1 and f_2 , which leads to a contradiction. So we always have a contractible curve as illustrated in Figure 4.7(b).

Case 5: F is a pizza, $F \cup S$ is a hamantach

In this case, the side of the hamantach with only vertices of S has to be a single vertex, otherwise we would have parallel segments. Now, we have a pizza with crust and with that we get to case 4.

If F is a pizza with crust or a hamantach, the cases and sub-cases are getting a bit more complicated since there are more options to "block" contractible curves. But even letting this part of the proof unknown, we can see that this proof relies on the fact that there are no parallel segments in the boundary. The question to be discussed is if we really can avoid such segments, since there is nowhere a reasoning why they would not exist.

Again, let us assume this all works and we can find all split pizzas and recurse on the separated components after splitting those cliques. Under this circumstances, Thorup

presents various further steps to proceed in the algorithm.

The next step is to find interior 1-cuts in X . If such a cut exists, Thorup proposes to simplify the LMP to two LMPs, one with one side of the 1-cut compromised to a single vertex, the other one by doing the same with the other side of the cut.

To proceed in in such a LMP with biconnected interior X , we select one interior vertex $q \in X$ with the most neighbours in B . Let Q be the set with vertices having exactly the same neighbours in B as q . Now, by looking at $X \setminus Q$, we again decompose the interior in various bubbles as illustrated in Figure 4.8.

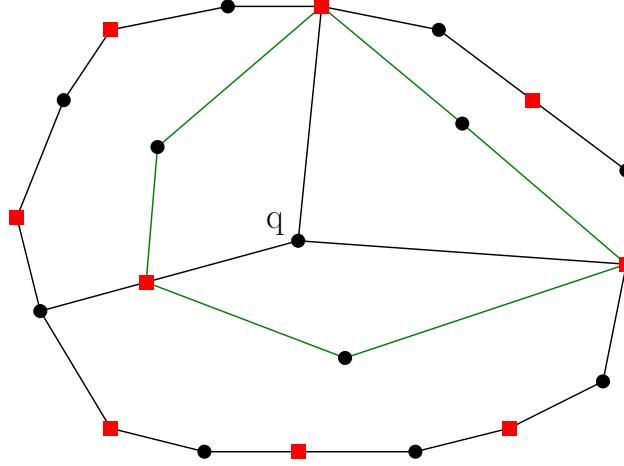


Figure 4.8: Interior set Q with cycle around it.

This might look different depending on the size of Q and the size of $A = B \cap N(Q)$, which Thorup proposes to solve via a case distinction. Here, the intersections of Q and B work as split points of those bubbles, but it can occur that we have cycles around Q as shown in Figure 4.8, making it more complicated to split than at the split points before. In total, Thorup tries to simplify the problem via such sets Q until the interior is easy enough to embed, which would lead to a solution of the LMP.

5. Conclusion

In this thesis, we provided a broad overview over map graphs, hole-free map graphs, k -map graphs as well as algorithms on them. A special focus was layed on polynomial-time recognition algorithms, mainly the one for general map graphs. While, for example, for recognizing 4-map graphs and hole-free 4-map graphs, the way to go is to simplify the problem to recognizing kite-augmented, respectively, triangulated 1-planar graphs, the presented idea of an algorithm to recognize general map graphs goes a more direct way. Either one tries to directly find a map as Thorup's version of the algorithm proposes, or one can try to find a witness graph as we did. The algorithms for all of those recognition problems have some parallels, namely to separate the graph in a useful way and try to solve the problem for the smaller graphs. Also, each algorithm analyzes the (maximal) cliques and their structure in the resulting subgraphs to properly embed the cliques in the plane. In contrast to the (hole-free) 4-map graph case, the general case is still unresolved at this point. Even in Thorup's promising ideas, many details are missing or wrong. We presented what arose from investigating these ideas in more depth and pointed to concrete inconsistencies.

Nevertheless, the ideas still seem promising. Therefore, we proposed ways to amend the errors we encountered and introduced the view of witness graphs, which might lead to a more combinatorial approach to the recognition algorithm.

Other open questions are, on one hand, parametrized problems as l -color coloring for k -map graphs with $k > 4$. To solve such problems, the basis and first results in [FLP⁺26] seem quite promising. On the other hand, recognizing k -map graphs with $k > 6$ as well as 5- and 6-map graphs remains open as well. However, for the latter [Bra19a] already gives first ideas towards an algorithm.

Bibliography

- [ABDL⁺24] Patrizio Angelini, Michael A Bekos, Giordano Da Lozzo, Martin Gronemann, Fabrizio Montecchiani, and Alessandra Tappini. Recognizing map graphs of bounded treewidth. *Algorithmica*, 86(2):613–637, 2024.
- [AH78] Kenneth Appel and Wolfgang Haken. The four-color problem. In *Mathematics today twelve informal essays*, pages 153–180. Springer, 1978.
- [BDLG⁺20] Michael A Bekos, Giordano Da Lozzo, Svenja Griesbach, Martin Gronemann, Fabrizio Montecchiani, and Chrysanthi Raftopoulou. Book embeddings of non-planar graphs with small faces in few pages. *arXiv preprint arXiv:2003.07655*, 2020.
- [BKS20] Bubacarr Bah, Jannis Kurtz, and Oliver Schaudt. Discrete optimization methods for group model selection in compressed sensing. *Mathematical Programming*, 190, 06 2020.
- [BL76] Kellogg S. Booth and George S. Lueker. Testing for the consecutive ones property, interval graphs, and graph planarity using pq-tree algorithms. *Journal of Computer and System Sciences*, 13(3):335–379, 1976.
- [Bor84] Oleg V. Borodin. Solution of Ringel’s problems on the vertex-face coloring of plane graphs and on the coloring of 1-planar graphs. *Diskretnyi Analiz*, 41:12–26, 1984. (in Russian).
- [Bra19a] Franz J Brandenburg. Characterizing 5-map graphs by 2-fan-crossing graphs. *Discrete Applied Mathematics*, 268:10–20, 2019.
- [Bra19b] Franz J Brandenburg. Characterizing and recognizing 4-map graphs. *Algorithmica*, 81(5):1818–1843, 2019.
- [Bra20] Franz J Brandenburg. Book embeddings of k-map graphs. *arXiv preprint arXiv:2012.06874*, 2020.
- [CGP98] Zhi-Zhong Chen, Enory Grigni, and Christos H Papadimitriou. Planar map graphs. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 514–523, 1998.
- [CGP02] Zhi-Zhong Chen, Michelangelo Grigni, and Christos H Papadimitriou. Map graphs. *Journal of the ACM (JACM)*, 49(2):127–138, 2002.
- [CGP06] Zhi-Zhong Chen, Michelangelo Grigni, and Christos H Papadimitriou. Recognizing hole-free 4-map graphs in cubic time. *Algorithmica*, 45(2):227–262, 2006.
- [CH67] Gary Chartrand and Frank Harary. Planar permutation graphs. In *Annales de l’institut Henri Poincaré. Section B. Calcul des probabilités et statistiques*, volume 3, pages 433–438, 1967.

- [Che01] Zhi-Zhong Chen. Approximation algorithms for independent sets in map graphs. *Journal of Algorithms*, 41(1):20–40, 2001.
- [DFHT05] Erik D Demaine, Fedor V Fomin, MohammadTaghi Hajiaghayi, and Dimitrios M Thilikos. Fixed-parameter algorithms for (k, r) -center in planar graphs and map graphs. *ACM Transactions on Algorithms (TALG)*, 1(1):33–47, 2005.
- [FLP⁺26] Fedor V Fomin, Daniel Lokshtanov, Fahad Panolan, Saket Saurabh, and Meirav Zehavi. Decomposition of map graphs with applications: Fv fomin et al. *Discrete & Computational Geometry*, pages 1–39, 2026.
- [HRG00] Monika R Henzinger, Satish Rao, and Harold N Gabow. Computing vertex connectivity: New bounds from old techniques. *Journal of Algorithms*, 34(2):222–250, 2000.
- [HT73] John E Hopcroft and Robert Endre Tarjan. Dividing a graph into triconnected components. *SIAM Journal on computing*, 2(3):135–158, 1973.
- [HT74] John Hopcroft and Robert Tarjan. Efficient planarity testing. *Journal of the ACM (JACM)*, 21(4):549–568, 1974.
- [MRS18] Matthias Mnich, Ignaz Rutter, and Jens M Schmidt. Linear-time recognition of map graphs with outerplanar witness. *Discrete optimization*, 28:63–77, 2018.
- [RS91] Neil Robertson and P.D Seymour. Graph minors. x. obstructions to tree-decomposition. *Journal of Combinatorial Theory, Series B*, 52(2):153–190, 1991.
- [Tho98a] Mikkel Thorup. Map graphs in polynomial time. In *Proceedings 39th Annual Symposium on Foundations of Computer Science (Cat. No. 98CB36280)*, pages 396–405. IEEE, 1998.
- [Tho98b] Mikkel Thorup. Map graphs in polynomial time. Available at <http://www.diku.dk/~mthorup/PAPERS/map.ps.gz>, 1998.
- [Yan89] Mihalis Yannakakis. Embedding planar graphs in four pages. *Journal of Computer and System Sciences*, 38(1):36–67, 1989.