

# 2D Minimal Graph Rigidity is in NC for One-Crossing-Minor-Free Graphs

Rohit Gurjar ✉🏠

Indian Institute of Technology Bombay, India

Kilian Rothmund ✉

Hochschule Aalen, Germany

Ulm University, Germany

Thomas Thierauf ✉🏠

Ulm University, Germany

---

## Abstract

*Minimally rigid graphs* can be decided and embedded in the plane efficiently, i.e. in polynomial time. There is also an efficient randomized *parallel* algorithm, i.e. in RNC. We present an NC-algorithm to decide whether one-crossing-minor-free graphs are minimally rigid. In the special case of  $K_{3,3}$ -free graphs, we also compute an infinitesimally rigid embedding in NC.

**2012 ACM Subject Classification** Theory of computation → Parallel algorithms; Theory of computation → Problems, reductions and completeness

**Keywords and phrases** Graph Rigidity, Parallel Algorithms, Polynomial Identity Testing, Derandomization

**Digital Object Identifier** 10.4230/LIPIcs.STACS.2026.49

**Related Version** *Full Version*: <https://ecc.weizmann.ac.il/report/2025/103/> [17]

**Funding** *Rohit Gurjar*: Supported by SERB MATRICS grant number MTR/2022/001009.

*Kilian Rothmund*: Supported by DFG grant TH 472/5-2.

*Thomas Thierauf*: Supported by DFG grant TH 472/5-2.

**Acknowledgements** The authors thank the organizers of Dagstuhl Seminar 24381 on *Algebraic and Analytic Methods in Computational Complexity* for inviting them. Part of this research was done during the seminar. We thank Dhara Thakkar for helpful discussions.

## 1 Introduction

Graph rigidity is the combinatorial study of rigidity or flexibility of bar-and-joint frameworks, a set of solid bars connected via hinge joints. Historically, the problem goes back to Euler in 1776, who asked about the rigidity of polyhedrons in 3D. Rigidity of bar-and-joint frameworks, particularly in 2D, has been studied extensively for applications in mechanical engineering like for collision-free robot arm motion planning, or molecular conformations, or network topologies with distance and angle constraints.

Graph rigidity is also a central problem in theoretical computer science. It yields a combinatorial interpretation of certain *polynomial identity testing* (PIT) problems that can be solved efficiently for 2D-rigidity, but the complexity for larger dimensions is wide open, see for example the exposition of Raz and Wigderson [35] or the book of Lovász [29, Chapter 15]. Other interesting aspects are that rigidity defines a matroid and that minimal rigidity reduces to bipartite perfect matching. The latter point motivates the question for the parallel complexity of rigidity.



© Rohit Gurjar, Kilian Rothmund, and Thomas Thierauf;  
licensed under Creative Commons License CC-BY 4.0

43rd International Symposium on Theoretical Aspects of Computer Science (STACS 2026).

Editors: Meena Mahajan, Florin Manea, Annabelle McIver, and Nguyễn Kim Thăng

Article No. 49; pp. 49:1–49:22



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



**Describing the problem.** A bar-and-joint framework can be seen as a graph  $G$ , where vertices are the joints and edges are the bars, together with an embedding  $p : V \rightarrow \mathbb{R}^2$ , for the case of 2D. The vertices of  $G$  are allowed to move continuously subject to the constraint that the distance between adjacent vertices does not change, i.e., the lengths of the bars are fixed. A framework  $(G, p)$  is *rigid* if every such continuous motion of the framework preserves the distance of *all* pairs of vertices, i.e., also of non-adjacent vertices. Otherwise the framework is *flexible*.

A natural question is whether rigidity/flexibility is just the property of the underlying graph  $G$  or whether it also depends on the specific embedding  $p$ . That is, whether rigidity is just determined by the combinatorial structure of the bars, i.e. which tuples of bars are connected at one joint, and does not depend on the specific lengths of the bars. The first answer is that there are examples that show that rigidity depends on the embedding  $p$ . However, for any graph, either almost all of its embeddings are rigid or almost all of them are flexible [16, 5]. Therefore one defines a graph  $G$  to be *rigid* if almost all embeddings  $p$  result in a rigid framework  $(G, p)$ , otherwise  $G$  is *flexible*. Graph rigidity can be equivalently defined based on the notion of *infinitesimal rigidity*, see Section 2.1 and Definition 4.

**Characterizations of minimal rigidity.** A graph is *minimally rigid* if the removal of any edge makes it flexible. Minimally rigid graphs have a combinatorial characterization that is often attributed to Laman, but was already known to Geiringer.

► **Definition 1** (Laman graph). *A graph  $G = (V, E)$  is a Laman graph if it has  $m = 2n - 3$  edges and for all  $S \subseteq V$  with  $|S| \geq 2$ , the subgraph of  $G$  induced by  $S$  has at most  $2|S| - 3$  edges, i.e.*

$$\forall S \subseteq V (|S| \geq 2) \quad |E(S)| \leq 2|S| - 3. \quad (1)$$

► **Theorem 2** ([34], [27]). *A graph  $G$  is minimally rigid iff  $G$  is a Laman graph.*

The Laman condition (1) gives rise to a matroid, the *rigidity matroid* for a graph  $G = (V, E)$ . The ground set is the set of edges  $E$ . A set  $E' \subseteq E$  is *independent*, if graph  $(V, E')$  fulfills (1). The base sets of this matroid are the minimally rigid subgraphs of  $G$  (on  $V$ ).

There is another combinatorial characterization.

► **Theorem 3** ([30]). *A graph  $G = (V, E)$  is minimally rigid iff  $\forall e \in E$  multigraph  $G + e$  is the union of two edge-disjoint spanning trees.*

There is also an iterative way to construct Laman graphs, known as the Henneberg construction [21], where we start with an edge, and then add a node to the graph constructed so far in one of two ways (Theorem 13). This is explained in more detail in Section 2.4.

**Complexity and motivation.** In 2D, rigidity can be solved in polynomial time, namely in time  $\mathcal{O}(n^2)$  [20, 15]. For minimal rigidity, this has been improved to  $\mathcal{O}(n\sqrt{n \log n} + m)$  [15]. We are interested in the *parallel* complexity of minimal rigidity.

Note that the characterization of minimal rigidity in Theorem 2 is similar in nature to Hall's Theorem [19] that characterizes the existence of perfect matchings in bipartite graphs. This can actually be formalized to a reduction from the minimal rigidity problem to the bipartite perfect matching problem [20]. Since the reduction to bipartite perfect matching is efficient in *parallel*, i.e. in NC, the parallel complexity of bipartite perfect matching carries over to minimal rigidity. Bipartite perfect matching is in randomized NC, RNC [33]. The derandomization question, whether one can avoid the randomness without much loss in

efficiency, is of major interest in complexity theory. A few years back, the randomized algorithm has been derandomized to a quasi-NC bound for bipartite perfect matching [14]. This result implies a quasi-NC algorithm for minimal rigidity. One of the major motivations for us to study minimal graph rigidity is that we might have better chances to show that it is in NC than for bipartite perfect matching. That is, minimal rigidity is the litmus test for bipartite perfect matching. Note that a reduction in the other direction is not known. Hence, minimal rigidity might be easier than bipartite perfect matching.

Interestingly, also the complexity questions around general rigidity (i.e., not necessarily minimal), have a status quite similar to those around the bipartite matching problem. Using the matroid property, one can get a polynomial-time algorithm to test if a graph is rigid. Both problems, matching and rigidity, reduce to the polynomial identity testing (PIT) problem [33, 35], and thus they have RNC-algorithms. Its connection with matching and PIT makes graph rigidity a very interesting candidate for studying derandomization.

**Our results.** First, we observe that for planar graphs one can decide minimal rigidity and compute an infinitesimally rigid realization in  $\text{NC}^2$  (Theorem 19). This is based on results in [40, 18, 38]

Our first main result is that for  $K_{3,3}$ -free graphs, we can decide whether a graph is Laman in  $\text{NC}^2$  (Theorem 23) and also compute a rigid realization (Theorem 25).

Also whether a one-crossing-minor-free graph is Laman can be decided in  $\text{NC}^3$  (Theorem 29). However, an NC-algorithm for computing a rigid realization for one-crossing-minor-free graphs or bounded treewidth graphs remains an open problem.

Note that extending results from planar graphs to such minor-free graphs is a technically nontrivial step that has also been done in other contexts, for example, famously by Vazirani [43]: Kasteleyn [24] showed that the number of perfect matchings in planar graphs can be efficiently computed. Then, based on a result of Little [28], Vazirani [43] extended the result to  $K_{3,3}$ -free graphs in  $\text{NC}^2$ . Later, the result was further extended to  $K_5$ -free graphs in  $\text{NC}^3$  [39]. There are many more results in a similar flavor, see for example [10, 42, 26, 25, 3, 8, 7, 12]. Although the literature gives a meta strategy, a result for planar graphs does not automatically imply a result for other minor-free graph classes and new ideas are required.

Following the literature, we decompose the given one-crossing-minor-free graph (or  $K_{3,3}$ -free graph) into planar components or components of bounded treewidth. We observe that there are parallel algorithms to decide rigidity of graphs of bounded treewidth, see Section 2.2. Hence, for planar and bounded treewidth components we can check whether they are Laman. Our main technical tools which allow us to lift the results from planar/bounded treewidth case to  $K_{3,3}$ -free/one-crossing-minor free graphs are as follows.

- Characterization of Laman graphs in terms of Lamanness of certain split graphs obtained by splitting the graph along separating pairs/triples (Lemmas 22, 26).
- For all families of 3-vertex graphs, we construct gadget graphs such that adding the gadget to a graph makes it Laman if and only if adding some member of the family makes it Laman (Lemma 28).
- Combining rigid embeddings of a set of Laman graphs to find a rigid embedding of a Laman graph obtained by joining them using 2-sum operations (Lemma 24).

Note that obtaining a parallel algorithm for minimal rigidity does not give the same for rigidity, as there is no parallel reduction known. In fact, finding an NC (or quasi-NC) algorithm for 2D graph rigidity is open, even for planar graphs.

**2 Preliminaries**

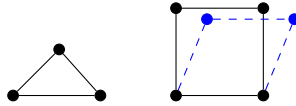
For  $n \in \mathbb{N}$ , we denote  $[n] = \{1, 2, \dots, n\}$ . We use standard complexity classes, in particular, the classes  $\text{NC}^k$  that consist of uniform boolean circuit families with bounded fan-in, polynomial size, and depth  $\mathcal{O}(\log^k n)$ . The corresponding randomized classes are denoted by  $\text{RNC}^k$ . A slight extension is quasi- $\text{NC}^k$  that is defined similarly to  $\text{NC}^k$ , but with circuits of quasi-polynomial size  $2^{\log^{\mathcal{O}(1)} n}$ .

For a graph  $G = (V, E)$  we denote  $n = n(G) = |V|$  as the number of vertices and  $m = m(G) = |E|$  as the number of edges of  $G$ . For a set  $S \subseteq V$ , the edges of  $G$  that are within  $S$  are denoted by  $E(S)$ . A graph  $G$  is *planar* if it can be drawn in the Euclidean plane such that the edges intersect only at the endpoints. The *crossing number* of a graph  $G$  is the minimum number  $c$  such that  $G$  has an embedding in the plane with  $c$  edge crossings. For example planar graphs have crossing number 0 and  $K_{3,3}$  and  $K_5$  have crossing number 1. Planar graphs are minor-closed. However, this does not hold in general for graphs with crossing number  $\geq 1$ : There are examples of graphs  $G$  with crossing number 1 that have minors  $H$  with crossing number 2. To capture such graphs  $H$  as well, Robertson and Seymour [37] defined a graph  $H$  to be *single-crossing* or *one-crossing*, if  $H$  is the minor of a graph  $G$  with crossing number  $\leq 1$ . With this definition, one-crossing graphs are closed under minors. However, in the literature, often the simpler definition is used that one-crossing graphs are those with crossing number  $\leq 1$ .

For graphs  $G, H$ , we say that  $G$  is *H-free*, if  $H$  is not a minor of  $G$ . Graph  $G$  is *one-crossing-minor-free*, if there exists a one-crossing graph  $H$  such that  $G$  is  $H$ -free. Hence, planar,  $K_{3,3}$ -free, and  $K_5$ -free graphs are all one-crossing-minor-free.

**2.1 Infinitesimal rigidity of frameworks**

In a *bar-and-joint framework* we are given a graph  $G = (V, E)$  and an embedding  $p : V \rightarrow \mathbb{R}^2$  in the plane. The edges of the graph are considered as bars and the vertices as joints that are free to move continuously. However, the length of the bars does not change. The framework is *flexible*, if there is a continuous motion such that the distance of some vertices changes. Otherwise, the framework is *rigid*. Figure 1 shows examples.



**Figure 1** The triangle is rigid whereas the rectangle is flexible with the motion indicated in blue. The distance of the diagonally opposite vertices changes with the motion.

The problem whether a given framework  $(G, p)$  is flexible is NP-hard [1]. It is complete for the existential theory of reals [2]. We consider a restricted version of the problem called *infinitesimal rigidity* that can be solved efficiently.

Let  $G = (V, E)$  where  $V = [n]$  and  $|E| = m$ . Let  $p(i) = (x_i, y_i) \in \mathbb{R}^2$  be the coordinates of vertex  $i \in V$ . For an edge  $e = (i, j) \in E$ , the square of its length is

$$|e|^2 = (x_i - x_j)^2 + (y_i - y_j)^2. \tag{2}$$

Consider a smooth motion by treating the coordinates as functions  $x_i(t)$  and  $y_i(t)$  in time  $t$ , such that  $(x_i(0), y_i(0)) = (x_i, y_i)$ . The condition for the motion is that  $|e|^2$  does not change, i.e. is constant. Hence, when we look at the derivative of (2) w.r.t.  $t$ , we get

$$2(x_i - x_j)(x'_i - x'_j) + 2(y_i - y_j)(y'_i - y'_j) = 0. \tag{3}$$

We get  $m$  such equations, one for every edge in  $G$ . We combine them in matrix-vector form. That is, we define the *rigidity matrix*  $R = R(G, p)$  as a  $m \times 2n$  matrix, with  $n$  columns for the  $x$ -part of the nodes and  $n$  columns for the  $y$ -part. Let  $R_e$  be the row for edge  $e = (i, j) \in E$  in  $R$ . We define the  $k$ -th entry  $R_{e,k}$  as

$$R_{e,k} = \begin{cases} x_i - x_j, & \text{for } k = i, \\ x_j - x_i, & \text{for } k = j, \\ y_i - y_j, & \text{for } k = n + i, \\ y_j - y_i, & \text{for } k = n + j, \\ 0, & \text{otherwise.} \end{cases}$$

The derivatives we put in the *velocity vector*  $v$ ,

$$v = (x'_1, x'_2, \dots, x'_n, y'_1, y'_2, \dots, y'_n)^T.$$

Then (3) becomes

$$Rv = 0. \tag{4}$$

For example, consider the triangle graph on three nodes 1, 2, 3. Then  $R$  is a  $3 \times 6$  matrix,

$$R = \begin{pmatrix} x_1 - x_2 & x_2 - x_1 & 0 & y_1 - y_2 & y_2 - y_1 & 0 \\ 0 & x_2 - x_3 & x_3 - x_2 & 0 & y_2 - y_3 & y_3 - y_2 \\ x_1 - x_3 & 0 & x_3 - x_1 & y_1 - y_3 & 0 & y_3 - y_1 \end{pmatrix}$$

Any nonzero vector  $v$  that fulfills (4) corresponds to an *infinitesimal motion*. However, there are three trivial motions that are always possible: a shift  $s_x$  of all vertices in  $x$ -direction or  $s_y$  in  $y$ -direction, and a rotation  $r$  around the origin,

$$\begin{aligned} s_x &= (1, 1, \dots, 1, 0, 0, \dots, 0)^T, \\ s_y &= (0, 0, \dots, 0, 1, 1, \dots, 1)^T, \\ r &= (-y_1, -y_2, \dots, -y_n, x_1, x_2, \dots, x_n)^T. \end{aligned}$$

Hence, the nullspace of  $R$  has dimension at least 3, and therefore, the rank of  $R$  can be at most  $2n - 3$ .

► **Definition 4** (Infinitesimal rigidity). *A framework  $(G, p)$  is infinitesimally rigid if*

$$\text{rank}(R(G, p)) = 2n - 3.$$

*A graph  $G$  is (infinitesimally) rigid<sup>1</sup> if there exists an embedding  $p$  such that the framework  $(G, p)$  is infinitesimally rigid.*

The definition for a graph  $G$  to be rigid given here is equivalent to the definition given in the Introduction [16, 5].

Clearly, the condition for the rank can only be achieved when we have  $m \geq 2n - 3$  edges. The rank of a matrix can be computed in  $\text{NC}^2$  [32], thus, infinitesimal rigidity for a given embedding  $p$  can be tested in  $\text{NC}^2$ . If  $p$  is *not* given, the rigidity problem becomes a PIT-question. In 2D, it can be solved in polynomial time.

We remark that the notions of rigidity and infinitesimal rigidity of a framework  $(G, p)$  are *not* equivalent: If  $(G, p)$  is infinitesimal rigid, it is also rigid. But when  $\text{rank}(R(G, p)) < 2n - 3$ , the framework might still be rigid. This is because a nonzero velocity vector  $v$  with  $Rv = 0$  only guarantees that the edges of  $G$  maintain their lengths in direction  $v$ . It does not ensure that two non-adjacent nodes change their distance. Examples where this happens are the trivial motions, but there are also examples with non-trivial motions.

<sup>1</sup> In the literature it is common to skip the adjective *infinitesimally* when considering a graph  $G$ .

## 2.2 Rigidity of graphs with bounded treewidth

A *tree decomposition* of a graph  $G = (V, E)$  is a tree  $\mathcal{T}$  with a set of nodes  $\mathcal{B}$ , the *bags*, where each  $B \in \mathcal{B}$  is a subset of  $V$ , such that the following conditions hold.

1.  $\bigcup_{B \in \mathcal{B}} B = V$ ,
2.  $\forall (u, v) \in E \exists B \in \mathcal{B} \ u, v \in B$ ,
3.  $\forall v \in V \ \mathcal{B}_v = \{ B \in \mathcal{B} \mid v \in B \}$  forms a subtree of  $\mathcal{T}$ .

The *width* of the tree decomposition is  $w = \max_{B \in \mathcal{B}} |B| - 1$ . The *treewidth* of  $G$  is the minimum width over all tree decompositions of  $G$ . A class of graphs  $\mathcal{G}$  has *bounded treewidth*, if there is a constant  $c$ , such that all graphs in  $\mathcal{G}$  have treewidth bounded by  $c$ .

Courcelle [9] showed that when a graph property is expressible in monadic second-order logic (MSO-logic), then it can be decided in linear time when the input graph has bounded treewidth. Elberfeld, Jakoby and Tantau [11] showed a logspace-version of Courcelle's Theorem. The characterization in Theorem 3 can be used to express rigidity in MSO-logic (see the full version [17] of this paper). Since a graph  $G = (V, E)$  is rigid iff it has a spanning minimally rigid subgraph, we get a MSO-predicate for  $G$  being rigid. It follows that whether a graph is Laman or rigid is in L, and hence in  $\text{NC}^2$ , for graphs with bounded treewidth.

► **Theorem 5.** *Rigidity of graphs with bounded treewidth can be decided in  $\text{NC}^2$ .*

## 2.3 Structural Decompositions

Let  $G = (V, E)$  be a graph. A set  $S \subseteq V$  with  $|S| = k$  is called a *k-separating set*, if  $G - S$  is not connected. Let  $G_1, G_2, \dots, G_\ell$  be the connected components of  $G - S$ . The *split graphs with respect to S* are the graphs  $G'_1, G'_2, \dots, G'_\ell$ , where  $G'_i$  is the graph obtained by taking the subgraph of  $G$  induced by  $V(G_i) \cup S$ , and adding edges within  $S$  such that all pairs of vertices in  $S$  are adjacent, for  $i = 1, 2, \dots, \ell$ . The edges in the split graphs that are within  $S$  are called *virtual edges* (even those edges in  $S$  which already exist in  $G$ ). A graph  $G$  is called *k-connected* if there is no  $(k - 1)$ -separating set in  $G$ .

A *k-separating set* is called *articulation point* for  $k = 1$ , *separating pair* for  $k = 2$ , and *separating triple* for  $k = 3$ .

Laman graphs are clearly connected, actually even 2-connected.

► **Lemma 6** ([23, Lemma 2.6]). *Laman graphs are 2-connected.*

In particular, every node of a Laman graph has degree at least two.

**3-connected components.** Hopcroft and Tarjan [22] describe an algorithm to decompose a 2-connected graph  $G = (V, E)$  into *triconnected components* by splitting  $G$  successively along all separating pairs in  $G$ . Actually, it suffices to split along *3-connected separating pairs*, where a separating pair  $\{u, v\}$  in  $G$  is called *3-connected*, if there are three vertex disjoint paths between  $u$  and  $v$  in  $G$ .

The components computed by this algorithm are of three types: 3-connected split graphs of  $G$ , simple cycles, or 3-bonds. A *3-bond* is a multi-graph with two vertices and three edges between them.

It is known that the triconnected components of  $G$  are uniquely determined, i.e. independent of the order of the separating pairs in which we do the splitting.

► **Lemma 7** ([31, 22]). *The triconnected components of a 2-connected graph are unique.*

The decomposition leads to the *triconnected component tree*: There is a node for each triconnected component and each 3-connected separating pair of  $G$ . There is an edge between triconnected component node  $C$  and separating pair node  $\{u, v\}$ , if  $u, v \in C$ .

**4-connected components.** We also need to further decompose 3-connected graphs along separating triples into 4-connected components. The split components of two separating triples might overlap and thus we cannot simply split along all separating triples. For example, in a  $K_{3,3}$ -graph both sides form separating triples and we cannot split along both. For an efficient splitting procedure with respect to parallel computation see [42] or [12].

The decomposition again leads to a tree, the *4-connected component tree*. In this tree we have vertices for the separating triples and for the 4-connected components. In addition, there is a vertex representing a 3-bond component for every edge  $(u, v)$  from  $G$ , where  $u, v$  are part of a separating triple. Two vertices in the 4-connected component tree are adjacent if one of them corresponds to a separating triple and the other one to a 4-connected component or a 3-bond sharing vertices with the triple.

**Complexity.** Graph reachability problems can be solved in nondeterministic logspace, NL. In undirected graphs, even in deterministic logspace, L [36]. We have

$$\text{NC}^1 \subseteq \text{L} \subseteq \text{NL} \subseteq \text{NC}^2.$$

In the following lemma, we list some known results along these lines.

► **Lemma 8.** *Let  $G$  be an undirected graph. The following problems can all be solved in  $\text{NC}^2$ .*

1. *Compute the articulation points and the connected components of  $G$ .*
2. *When  $G$  is 2-connected, compute the 3-connected separating pairs, the triconnected components, and the triconnected component tree of  $G$ .*
3. *When  $G$  is 3-connected, compute the separating triples, the 4-connected components, and the 4-connected component tree of  $G$ .*

The component trees can have large depth. As we want to process them in a bottom-up fashion in logarithmic time, we need to identify long paths and treat them separately. Let  $T$  be a rooted tree and let  $v$  be a vertex in  $T$ . Then  $T(v)$  is the subtree of  $T$  rooted at  $v$ . A child  $u$  of  $v$  is a *large child* if  $|T(u)| > |T(v)|/2$ . A *large child path* in  $T$  is a maximal path such that every vertex along the path is a large child of its parent.

► **Lemma 9** ([39]). *Let  $T$  be a tree with  $n$  nodes and  $p$  be a simple path in  $T$ . Then*

- *the number of large child paths on  $p$  is  $\leq \log n$ ,*
- *the number of nodes on  $p$  that are not large children is  $\leq \log n$ ,*
- *all large child paths in  $T$  can be computed in  $\text{NC}^2$ .*

By the first two items in Lemma 9, the number of large child paths in  $T$  is polynomially bounded in  $n$ . Then the last item follows because we can compute the number of nodes in subtrees of  $T$  in L and hence, also in  $\text{NC}^2$ .

**Structure theorems.** The reason why we are considering the above decompositions is that for  $K_{3,3}$ -free,  $K_5$ -free, and one-crossing-minor-free graphs, we end up in components that are planar or of bounded treewidth.

► **Theorem 10** ([4]). *Every 3-connected component of a  $K_{3,3}$ -free graph is either planar or  $K_5$ .*

For  $K_5$ -free graphs, we end up in planar components or a special constant size graph.

► **Theorem 11** ([44]). *Every 3-connected component of a  $K_5$ -free graph is either the Wagner graph, which is the Möbius ladder on 8 vertices, or its 4-connected components are all planar.*

For one-crossing-minor-free graphs, we get planar or bounded treewidth components.

► **Theorem 12** ([37]). *For every one-crossing graph  $H$  there is a constant  $w_H$  such that the 4-connected components of a  $H$ -free graph are either planar or have treewidth  $\leq w_H$ .*

## 2.4 Henneberg sequences

Laman graphs can be constructed iteratively via *Henneberg extensions* [21]. The starting point in a sequence of Henneberg extensions is a graph with two nodes connected by an edge. Let  $G$  be the graph constructed so far. There are two ways to add a new node  $v$  to  $G$ :

- A *Henneberg extension of type 1* connects  $v$  with two arbitrary vertices of  $G$ .
- A *Henneberg extension of type 2* takes an existing edge  $(u, w)$  in  $G$  and replaces it with edges  $(u, v)$  and  $(v, w)$ , and connects  $v$  to an arbitrary third vertex in  $G$ .

► **Theorem 13** ([41]). *A graph  $G = (V, E)$  is Laman iff it can be constructed by a sequence of Henneberg extensions that starts with an arbitrary edge  $e \in E$ .*

Henneberg sequences can also be reversed in the following way.

► **Lemma 14** ([23, Lemma 2.8]). *Let  $G = (V, E)$  be a Laman graph with  $|V| \geq 3$  and  $v \in V$ .*

1. *If  $\deg(v) = 2$ , then  $G - v$  is Laman.*
2. *If  $\deg(v) = 3$ , then  $G - v + (u, w)$  is Laman for some pair  $u, w$  of neighbors of  $v$ .*

Using reversed Henneberg operations one can generalize Theorem 13: One can start the Henneberg sequence for a Laman graph with *any* two nodes  $u$  and  $v$  and edge  $(u, v)$ , even if  $(u, v)$  is not present in  $G$ .

► **Lemma 15** ([18, Lemma 2]). *A Laman graph  $G = (V, E)$  has a Henneberg construction that starts from edge  $(u, v)$ , for any two nodes  $u, v \in V$ .*

The proof for Lemma 15 roughly works by applying reversed Henneberg steps to the graph until we end up with the edge  $(u, v)$ . We further generalize Lemma 15 by showing the existence of a Henneberg sequence that starts with a triangle on any three vertices  $T = \{u, v, w\}$  that contain at least one edge in  $G$ . To do so, we first make a technical observation for the case that all vertices not in  $T$  have large degree.

► **Lemma 16.** *Let  $G = (V, E)$  be a Laman graph,  $|V| \geq 4$ , and  $T = \{u, v, w\} \subseteq V$  such that every  $x \in V' = V - T$  has  $\deg(x) \geq 4$ . Then we have*

- $\deg(u) = \deg(v) = \deg(w) = 2$ ,
- $|E(T)| = 0$ ,
- $G - T$  is a connected graph.

**Proof.** The sum of all node degrees of  $G$  is at least  $\deg(u) + \deg(v) + \deg(w) + 4(n - 3)$ . By the degree sum formula, we therefore have

$$\deg(u) + \deg(v) + \deg(w) + 4(n - 3) \leq 2m.$$

Since  $m = 2n - 3$ , we conclude that

$$\deg(u) + \deg(v) + \deg(w) \leq 6.$$

Since  $G$  is Laman, we also have  $\deg(u), \deg(v), \deg(w) \geq 2$ . This implies the first item.

To show the second item, let  $n' = |V'|$  and  $m' = |E(V')|$ . Let  $E(V', T)$  be the edges between  $V'$  and  $T$ . Since the nodes in  $T$  have degree 2, we have  $|E(V', T)| \leq 6$ . We argue that  $|E(V', T)| = 6$ , and hence  $|E(T)| = 0$ .

The sum of all node degrees in  $V'$  is at least  $4n' - |E(V', T)|$ . By the degree sum formula, we therefore have

$$4n' - |E(V', T)| \leq 2m'.$$

Since  $G$  is Laman, we also have  $m' \leq 2n' - 3$ , and therefore

$$|E(V', T)| \geq 6.$$

To argue that  $G - T$  is connected, assume that there are two components in  $G - T$ , say on nodes  $V_1$  and  $V_2$ , respectively. For each component we can make the same estimates as for  $V'$  above to show the second item. Hence, we would get that  $|E(V_1, T)|, |E(V_2, T)| \geq 6$ . But this contradicts the first item. ◀

We use Lemma 16 to show that a Henneberg sequence for a Laman graph  $G$  can start with a triangle on any three nodes, as long as there is at least one edge between the nodes in  $G$ . This generalizes a result by Haas et al. [18, Lemma 3].

► **Lemma 17.** *Let  $G = (V, E)$  be a Laman graph and  $T = \{u, v, w\} \subseteq V$  such that  $|E(T)| \geq 1$ . Then there is a Henneberg sequence for  $G$  that starts with triangle  $(u, v, w)$ .*

**Proof.** We prove the claim by induction on  $n = |V|$ . For  $n = 3$  the claim is trivially true since the only possibility for  $G$  is to be exactly triangle  $(u, v, w)$ . Let  $n \geq 4$ . Let vertex  $x \in V' = V - T$  such that  $\deg(x) \leq 3$ . Note that  $x$  must exist because otherwise, when  $\deg(x) \geq 4$ , for all  $x \in V'$ , then  $|E(T)| = 0$  by Lemma 16, but we have  $|E(T)| \geq 1$ .

We remove  $x$  by a reversed Henneberg step from Lemma 14. Let  $H = G - x$ . The removal operation can only increase the number of edges within  $T$  in  $H$ . Thus, by the induction hypothesis, there is a Henneberg sequence for  $H$  that starts with triangle  $(u, v, w)$ . We extend the sequence by adding  $x$  back to  $H$ . Then the sequence produces  $G$ . ◀

The assumption in Lemma 17 is necessary: There are examples for a graph  $G$  where  $|E(T)| = 0$  and it is *not* possible to construct  $G$  from triangle  $(u, v, w)$ . In Section 5, we will consider the case where  $G$  is 3-connected and  $T$  is a separating triple. Then we can still get a useful statement about the starting point of a Henneberg sequence from Lemma 16 and 17.

► **Corollary 18.** *Let  $G = (V, E)$  be a 3-connected Laman graph with a separating triple  $T = \{u, v, w\}$  and corresponding split graphs  $G_1, G_2, \dots, G_\ell$ , where we have removed all virtual edges from the split graphs that are not edges in  $G$ . Then there is a Henneberg sequence for  $G$  that starts*

- *either with triangle  $(u, v, w)$ ,*
- *or there is a split component, say  $G_1 = (V_1, E_1)$ , that is Laman and  $|E_1(T)| = 0$ , and the sequence initially constructs  $G_1$ .*

**Proof.** The claim follows from Lemma 17 when  $|E(T)| \geq 1$ . So assume that  $|E(T)| = 0$ . Like in the proof of Lemma 17, we remove vertices  $x \in V' = V - T$  with  $\deg(x) \leq 3$  by reversed Henneberg steps, as long as there are such vertices. In case this process introduces an edge within  $T$  at some point, then it will stop with triangle  $(u, v, w)$  as in the proof of Lemma 17. Otherwise, it will stop with a Laman graph  $H = (V_H, E_H)$  with  $|E_H(T)| = 0$ . By Lemma 16, graph  $H - T$  is connected. Hence,  $H$  can be obtained by reverse Henneberg steps from one of the split components  $G_1, G_2, \dots, G_\ell$ , say  $G_1$ . Now a Henneberg sequence for  $H$  can be extended to a sequence for  $G_1$  by adding the vertices of  $G_1$  back that we removed above. Note that  $T \subseteq V_H$ . Hence, adding the remaining vertices of  $G_1$  will not introduce an edge within  $T$ . Now we can add the rest of the vertices back that we removed above. This gives the desired Henneberg sequence for  $G$ . ◀

### 3 Rigid embeddings for planar graphs in $\text{NC}^2$

Streinu [40] and Haas et al. [18] showed that planar Laman graphs can be characterized by planar embeddings in the plane with certain geometric properties. We observe that this characterization can be verified efficiently in parallel. Moreover, the geometric properties can be used to compute an infinitesimally rigid embedding in  $\text{NC}^2$ .

► **Theorem 19.** *Deciding whether a planar graph is Laman is in  $\text{NC}^2$ . Moreover, given a planar Laman graph  $G$ , computing a planar infinitesimally rigid embedding for  $G$  is in  $\text{NC}^2$ .*

Due to the conference page limit, we omit the proof in the proceedings. It can be found in the full version of the paper [17].

### 4 Rigid embeddings for $K_{3,3}$ -free graphs in $\text{NC}^2$

Let  $G$  be a  $K_{3,3}$ -free graph. We want to check whether  $G$  is Laman efficiently in parallel. By Lemma 6, we may assume that  $G$  is 2-connected. By Theorem 10, when we decompose  $G$  into 3-connected components, these components are either planar or  $K_5$ . For planar components we can check if they are Laman by Theorem 19. Hence, what we need is a connection between the Laman properties of  $G$  and its 3-connected components. This is established by the following lemma.

► **Lemma 20.** *Let  $G = (V, E)$  be a 2-connected graph with a separating pair  $\{u, v\}$  and corresponding split graphs  $G_1, G_2, \dots, G_\ell$ , where we have removed all virtual edges from the split graphs that are not edges in  $G$ .*

1. *If  $(u, v) \in E$ , then  $G$  is Laman iff  $G_1, \dots, G_\ell$  are Laman.*
2. *If  $(u, v) \notin E$ , then  $G$  is Laman iff there exists one component, say  $G_1$ , that is Laman, and  $G_2 + (u, v), \dots, G_\ell + (u, v)$  are Laman.*

**Proof.** Let  $G$  be Laman. By Lemma 15, there is a Henneberg construction for  $G$  that starts with edge  $(u, v)$ . Recall that this is independent of whether  $(u, v)$  is an edge in  $G$ . When a new vertex is added in the Henneberg sequence, it belongs to exactly one of the split graphs  $G_i$  and the extension cannot interfere with some vertex from another split graph. Otherwise, the separating pair would not separate the split graphs from each other in  $G$ .

1. If  $(u, v) \in E$ , this gives us Henneberg sequences for all split graphs by subdividing the sequence for  $G$  in its parts for  $G_1, \dots, G_\ell$ , respectively. Hence, they are all Laman.
2. If  $(u, v) \notin E$ , it has been replaced by a type 2 extension adding a vertex in exactly one split graph, say  $G_1$ . Again we subdivide the Henneberg sequence for  $G$ , and get sequences for  $G_1$  and for  $G_2 + (u, v), \dots, G_\ell + (u, v)$ . Therefore, they are all Laman.

For the backward direction, in the first case, we consider Henneberg constructions for  $G_1, G_2, \dots, G_\ell$  that start with edge  $(u, v)$ . Since the edge is present in all the components, it is never used in any of the Henneberg sequences. Hence, we can combine all the Henneberg sequences to a sequence for  $G$ . Hence,  $G$  is Laman.

The second case is similar. Since  $(u, v)$  is not present in  $G_1$ , the edge is used in the sequence for  $G_1$ , but not in any of the sequences for  $G_2 + (u, v), \dots, G_\ell + (u, v)$ . Hence, we can again combine all the Henneberg sequences to a sequence for  $G$ . Hence,  $G$  is Laman. ◀

The above lemma motivates us to define an operation on 2-connected graphs, which we call Laman-split.

► **Definition 21** (Laman-split). For a 2-connected graph  $G = (V, E)$  with a separating pair  $\{u, v\}$ , let  $G_1, G_2, \dots, G_\ell$  be the split graphs obtained after splitting  $G$  along  $\{u, v\}$ , where we have removed all virtual edges that are not edges in  $G$ . The Laman-split of  $G$  along  $\{u, v\}$  are the graphs  $G'_1, G'_2, \dots, G'_\ell$ , where for each  $i \in [\ell]$ ,

$$G'_i = \begin{cases} G_i + (u, v), & \text{if } m(G_i) = 2n(G_i) - 4, \\ G_i, & \text{otherwise.} \end{cases}$$

For a Laman graph, all split graphs  $G_i$  in Definition 21 have either  $2n(G_i) - 4$  or  $2n(G_i) - 3$  edges by Lemma 20. Note that we define the Laman-split also for graphs that are not Laman. In this case, the split graphs can also have other numbers of edges. In such a case,  $G_i$ , and hence  $G$ , are trivially detected as not being Laman.

Recall that by Lemma 7, the standard splitting of 2-connected graphs in triconnected components is unique, i.e. independent of the order of the separating pairs we do the splitting. The following lemma shows when we apply Laman-splits to the components on the way, the resulting Laman components are unique as well.

► **Lemma 22.** Let  $G = (V, E)$  be a 2-connected graph. Then  $G$  is Laman iff there is a way to put the separating pair edges  $(u, v) \notin E$  into the triconnected components of  $G$  such that  $(u, v)$  is in all but one of the components that contain  $u, v$  and that the resulting components are all Laman.

Moreover, in case  $G$  is Laman, this Laman decomposition is unique and can be computed in  $\text{NC}^2$ .

**Proof.** Consider the standard recursive splitting procedure to compute the triconnected component tree of  $G$ . When we have split along a separating pair  $\{u, v\}$ , we can also compute the Laman-split that says in which split components edge  $(u, v)$  should be put. Note that the recursive splitting is always done on the components computed by the standard splitting procedure. The Laman-split is a post-computation on these components that does not affect the recursive splitting. By the characterization given in Lemma 20, we conclude that  $G$  is Laman iff all the components computed by Laman-splits are Laman.

It remains to argue about the uniqueness of the Laman decomposition. This property is crucial for our parallel algorithm to compute the decomposition. So assume that  $G$  is a Laman graph. By Lemma 7, the triconnected components are unique. We argue that also the corresponding components we get from Laman-splits are uniquely determined. That is, whether a separating pair edge is present or not in a component is irrespective of the order of the decomposition.

This is trivial in case a separating pair  $\{u, v\}$  is connected by an edge  $(u, v)$  in  $G$ . Then all components will have the edge  $(u, v)$  as well by Lemma 20. We only have to argue for the case that  $\{u, v\}$  is *not* connected by an edge in  $G$ . In this case, the edge  $(u, v)$  will be present in all but one of the components by Lemma 20. We argue that the component without edge  $(u, v)$  is uniquely determined.

Consider the triconnected component tree  $\mathcal{T}$ . We argue via induction on the number of component nodes in  $\mathcal{T}$ . If  $\mathcal{T}$  has just one component node, then there is no separating pair and the claim is trivial.

In the inductive step, let  $\mathcal{T}$  have more than one component node. Let  $C$  be a component node with a separating pair  $\{u, v\}$  such that *all* other split components at  $\{u, v\}$  are leaves in  $\mathcal{T}$ . In the leaf components  $\{u, v\}$  is the only separating pair. Hence, it is uniquely defined whether the separating pair edge should be present in a leaf component or not, so that

## 49:12 Minimal Graph Rigidity in NC for OCMF Graphs

it has the right number of edges to be Laman. Therefore the same holds for the parent component  $C$  by Lemma 20. Note also that the presence or absence of a separating pair edge  $(u, v)$  in  $C$  is not affected when  $C$  is further split along a different separating pair.

Now we can prune the leaf components considered above from  $\mathcal{T}$  and get a tree with a smaller number of component nodes where we can apply the induction hypothesis.

For the complexity bound, we describe a parallel procedure to obtain the Laman components. First we compute all triconnected components in  $\text{NC}^2$  (Lemma 8). To determine where to put the separating pair edges, we do a Laman-split of  $G$ , for every separating pair  $\{u, v\}$  in parallel. That is, we treat each separating pair as the starting point of a Laman decomposition of  $G$ . Thereby we will put the edges correctly in the respective components by the uniqueness property: For any triconnected component  $H$  that contains  $\{u, v\}$ , we add the separating pair edge  $(u, v)$  to  $H$ , if after Laman-split in  $G$  the component  $H'$  that contains  $H$  has edge  $(u, v)$ . ◀

Now we have all the tools to decide efficiently in parallel whether a given  $K_{3,3}$ -free graph is Laman.

► **Theorem 23.** *Given a  $K_{3,3}$ -free graph  $G$ , we can decide whether  $G$  is Laman in  $\text{NC}^2$ .*

**Proof.** Given a 2-connected  $K_{3,3}$ -free graph  $G$ , we apply the algorithm from Lemma 22 to compute its Laman components. Here we might already detect that  $G$  is not Laman when the Laman-split yields some component where the number of edges does not match the number according to Lemma 20. Otherwise, we have that  $G$  is Laman iff all the components are Laman. Note that the components are planar graphs or subgraphs of  $K_5$  by Theorem 10, because separating pair edges only replace virtual edges, and hence do not affect planarity. Thus, we can apply Theorem 19 for all components in parallel to check if they are all Laman. All the subroutines used are in  $\text{NC}^2$ . ◀

The decision algorithm splits the graph  $G$  along separating pairs until all components are planar and then checks that these components are Laman. We observed in Section 3 that we can also compute rigid embeddings for the planar components. To find a rigid embedding of  $G$ , we now want to reassemble the embeddings of the components appropriately.

In Lemma 24, we make the assumption that the two nodes of a separating pair are mapped to the same pair of points in all the components, respectively. We show that then we directly have a rigid embedding for the whole graph  $G$ .

► **Lemma 24.** *Let  $G = (V, E)$  be a Laman graph with a separating pair  $\{u, v\}$ . Let  $G_1, G_2, \dots, G_\ell$  be the Laman components obtained after a Laman-split of  $G$  along  $\{u, v\}$ . Let  $p_1, p_2, \dots, p_\ell$  be infinitesimal rigid embeddings of the components such that*

$$\begin{aligned} p_1(u) &= p_2(u) = \dots = p_\ell(u), \\ p_1(v) &= p_2(v) = \dots = p_\ell(v), \\ p_1(u) &\neq p_1(v), \end{aligned}$$

*so that the common embedding  $p = \cup_{i=1}^\ell p_i$  is well defined. Then  $p$  is an infinitesimally rigid embedding of  $G$ .*

**Proof.** We prove the claim for  $\ell = 2$ . For larger  $\ell$ , we can iterate the argument, combining two graphs in every round. By Lemma 20, edge  $(u, v)$  is *not* contained in at most one of the components. Hence, we may assume that  $(u, v)$  is contained in  $G_2$ .

When we combine the rigidity matrices  $R_1 = R(G_1, p_1)$  and  $R_2 = R(G_2, p_2)$  as shown in Figure 2, we essentially get the rigidity matrix  $R = R(G, p)$ .



## 49:14 Minimal Graph Rigidity in NC for OCMF Graphs

Note that  $M$  is  $4 \times 3$  matrix and  $\text{rank}(M) = 3$  since we assume that  $(x_u, y_u) \neq (x_v, y_v)$ . Hence, the kernel of  $M^T$  has dimension 1. It follows that  $a_{u,v}$  must be a multiple of  $b_{u,v}$ . Hence, vector  $a$  is a multiple of the row-vector  $(u, v)$  of  $R$ . But recall that  $a$  is in the row-span of  $R'_2$  and row-vector  $(u, v)$  is linearly independent of  $R'_2$ . Hence, we must have  $a = 0$ . Therefore, the rigidity matrix  $R$  has full rank.  $\blacktriangleleft$

To get a rigid embedding of a  $K_{3,3}$ -free Laman graph, it now remains to show how to achieve the assumption of Lemma 24.

► **Theorem 25.** *Given a  $K_{3,3}$ -free Laman graph  $G$ , we can compute an infinitesimally rigid embedding in  $\text{NC}^2$ .*

**Proof.** We follow the algorithm from Theorem 23 and apply Lemma 22 to decompose  $G$  into planar Laman components  $C_1, C_2, \dots, C_k$ . Then we compute infinitesimally rigid embeddings  $p_1, p_2, \dots, p_k$  for the components in parallel by Theorem 19.

The vertices that belong to some separating pair occur in several components. We will construct new rigid embeddings  $q_1, q_2, \dots, q_k$  that will map different copies of any such vertex to the same point, and leave all other vertices unchanged. Then  $q_1, q_2, \dots, q_k$  will fulfill the assumption of Lemma 24, and  $q = \bigcup_{i=1}^k q_i$  will be a rigid embedding for  $G$ .

For  $v \in V$ , let

$$S_v = \{i \in [k] \mid v \in V(C_i)\}.$$

For every  $v \in V$  where  $|S_v| > 1$ , we construct a pair of univariate polynomials  $(a_v(t), b_v(t))$  that interpolates the points  $\{p_i(v) \mid i \in S_v\}$ . That is, we compute the interpolation polynomials such that  $(a_v(i), b_v(i)) = p_i(v)$ , for every  $i \in S_v$ . Then we replace the coordinates of such a vertex  $v$  by  $(a_v(t), b_v(t))$  in each component. That is, we define embeddings  $p_{i,t}(v)$ , for  $i \in [k]$  and  $v \in V(C_i)$ ,

$$p_{i,t}(v) = \begin{cases} (a_v(t), b_v(t)), & \text{if } |S_v| > 1, \\ p_i(v), & \text{otherwise.} \end{cases}$$

Note that a component  $C_i$  can have several separating pair nodes and we replace their coordinates by different polynomials, respectively, but all in the same variable  $t$ . The interpolation guarantees that  $p_{i,t}(v)$  agrees with  $p_i(v)$  for  $t = i$ , for every  $v \in V(C_i)$ ,

$$p_{i,i}(v) = p_i(v). \tag{5}$$

Consider the rigidity matrices  $R(C_i, p_{i,t})$ , where some of the entries are polynomials in  $t$ . Our goal is to find a value for  $t$  such that all matrices  $R(C_i, p_{i,t})$  have full rank. Let  $R_i$  be a non-singular  $(2n_i - 3)$ -square submatrix of  $R(C_i, p_i)$  and define  $R_{i,t}$  as the corresponding submatrix of  $R(C_i, p_{i,t})$ . Since  $\det(R_i) \neq 0$  and  $R_i = R_{i,i}$  by (5), we have that  $\det(R_{i,t})$  is a non-zero polynomial. Hence, the product

$$A(t) = \det(R_{1,t}) \det(R_{2,t}) \cdots \det(R_{k,t})$$

is a non-zero polynomial too. For the degree of  $A(t)$  note that  $\deg(a_v), \deg(b_v) = |S_v| - 1 \leq n$ . Therefore

$$\deg(\det(R_{i,t})) \leq n(2n_i - 3) < 2n^2.$$

Hence, for  $A(t)$  we get

$$\deg(A(t)) < k2n^2 \leq 2n^3.$$

It follows that we can find a  $t_0 \in [2n^3 + 1]$  such that  $A(t_0) \neq 0$ . Now we define  $q_i = p_{i,t_0}$ , for all  $i \in [k]$ . By construction,  $q_i$  is still a rigid embedding of  $C_i$ . Then  $q = \bigcup_{i=1}^k q_i$  is a rigid embedding for  $G$  by Lemma 24.

For the complexity, note that polynomial interpolation and evaluation is in  $\text{NC}^2$  [13]. ◀

## 5 Deciding Lamanness of one-crossing-minor-free graphs in $\text{NC}^3$

In this section, we give an  $\text{NC}^3$  algorithm for deciding whether a  $K_5$ -free graph is Laman, or even more general, whether a one-crossing-minor-free graph is Laman. We use Theorem 11, respectively Theorem 12, and further decompose the graph at separating triples into 4-connected components. We first show how the Laman property is preserved in the components. This can be seen as a generalization of Lemma 20 for separating pairs to separating triples.

► **Lemma 26.** *Let  $G = (V, E)$  be a 3-connected graph with a separating triple  $T = \{u, v, w\}$  and corresponding split graphs  $G_1, G_2, \dots, G_\ell$ , where we have removed all virtual edges from the split graphs that are not edges in  $G$ . Let  $\Delta_T = \{(u, v), (u, w), (v, w)\}$  be the triangle edges and  $E(T)$  the actual edges of  $G$  in  $T$ .*

*Then  $G$  is Laman iff there is a way to put each  $e \in \Delta_T - E(T)$  in all but one of  $G_1, G_2, \dots, G_\ell$ , such that the resulting components are all Laman.*

**Proof.** The argument goes along the lines of the proof of Lemma 20, extended to triples.

Let  $G$  be Laman. By Corollary 18, there is a Henneberg construction for  $G$  that starts either with triangle  $(u, v, w)$ , or with one component, say  $G_1 = (V_1, E_1)$ , where  $|E_1(T)| = 0$ .

- If the sequence starts with triangle  $(u, v, w)$ , each triangle edge  $e \in \Delta_T - E(T)$  will be subdivided by a type 2 step, adding a vertex in one split component, say  $G_i$ . Hence, to get a Henneberg sequence for all the components, we have to add  $e$  to all of them except  $G_i$ .
- If the sequence starts by constructing  $G_1$ , the rest of the sequence constructs  $G_2, \dots, G_\ell$  by extending from  $T$ , but without using any triangle edges, because  $|E_1(T)| = 0$ . Hence, we get Henneberg sequences for  $G_2 + \Delta_T, \dots, G_\ell + \Delta_T$ .

For the reverse direction, we have Henneberg sequences for all the components, where we have added the edges from  $\Delta_T - E(T)$  to  $G_1, G_2, \dots, G_\ell$ , as described in the lemma. If all components have a Henneberg sequence that starts with triangle  $(u, v, w)$ , then we can combine them to one sequence for  $G$ . If there is component, say  $G_1 = (V_1, E_1)$  with  $|E_1(T)| = 0$ , that cannot be constructed from triangle  $(u, v, w)$ , we start with the sequence for  $G_1$ . By definition, the other components are  $G_2 + \Delta_T, \dots, G_\ell + \Delta_T$ , that have Henneberg sequences starting with triangle  $(u, v, w)$ . These sequences will not use any triangle edge, and hence, we can attach them to the sequence for  $G_1$ . This yields a sequence for  $G$ . ◀

When applying Lemma 26, we will consider different choices of triangle edges that can be added to make a component Laman. For three nodes in  $G$  with no edge between them, there are three choices to add two edges between them. The following lemma states that when two of the choices make the graph Laman, then so does the third one.

► **Lemma 27.** *Let  $G = (V, E)$  be a graph and  $u, v, w \in V$  be three nodes in  $G$  with no edge between them. If  $G + \{(u, v), (u, w)\}$  and  $G + \{(u, v), (v, w)\}$  are Laman then  $G + \{(u, w), (v, w)\}$  is also Laman.*

## 49:16 Minimal Graph Rigidity in NC for OCMF Graphs

**Proof.** Assume that  $G + \{(u, w), (v, w)\}$  is not Laman. Then there must be a subset  $S$  of vertices including at least two triple vertices such that

$$|E[S]| > 2|S| - 3. \quad (6)$$

Then  $S$  will also satisfy (6) in  $G + \{(u, v), (u, w)\}$  or  $G + \{(u, v), (v, w)\}$ , a contradiction. ◀

Consider the 4-connected component tree of a 3-connected graph. Let components  $H_1, \dots, H_\ell$  be leaf nodes in the tree that are attached via a common separating triple to parent component  $H_0$ . The following lemma shows how to prune the leafs in the tree and replace them by a constant size gadget in  $H_0$  such that the Laman property is maintained.

► **Lemma 28.** *Let  $H = (V, E)$  be a graph with a separating triple  $T = \{u, v, w\}$  and corresponding split graphs  $H_0, H_1, \dots, H_\ell$ , where we have removed all virtual edges that are not edges in  $H$ . Assume that the graphs  $H_1, \dots, H_\ell$  are planar or of bounded treewidth (even with all virtual edges). Then there is an  $\text{NC}^2$ -algorithm that*

■ *either computes a constant-size gadget graph  $\Gamma$  on  $T$  such that*

$$H \text{ is Laman} \iff H_0 \cup \Gamma \text{ is Laman}$$

■ *or determines directly that  $H$  is not Laman.*

*Moreover, let  $H'_0$  be graph  $H_0$  plus the edges missing from triangle  $T$ . If  $H'_0$  is planar or of bounded treewidth  $w \geq 5$ , then  $H_0 \cup \Gamma$  is planar or of bounded treewidth  $w$ , respectively.*

*Also, the choice of the gadget in the first item depends only on  $E(T)$  and  $H_1, \dots, H_\ell$ , and not on  $H_0$ .*

**Proof.** Let  $\Delta_T = \{(u, v), (v, w), (u, w)\}$  be the triangle edges on  $T$ . Lemma 26 describes how to put the triangle edges of  $\Delta_T - E(T)$  in split components for graph  $H$  to be Laman. However, this does not uniquely determine the placement of the edges. Therefore we consider all possibilities to put the edges that are consistent with Lemma 26.

Let  $\mathcal{F}_0$  be the family of those sets  $F_0 \subseteq \Delta_T - E(T)$  for which there exist sets  $F_1, F_2, \dots, F_\ell \subseteq \Delta_T - E(T)$  such that

1. each edge in  $\Delta_T - E(T)$  appears in all but one of  $F_0, F_1, \dots, F_\ell$  and
2.  $H_1 + F_1, H_2 + F_2, \dots, H_\ell + F_\ell$  are all Laman.

By Lemma 26, we have

$$H \text{ is Laman} \iff \exists F_0 \in \mathcal{F}_0 \quad H_0 + F_0 \text{ is Laman.} \quad (7)$$

We claim that the family  $\mathcal{F}_0$  can be computed in  $\text{NC}^2$ . To see this observe that the number of possible tuples  $(F_0, F_1, \dots, F_\ell)$  which satisfy item 1 above is  $(\ell + 1)^{|\Delta_T - E(T)|} \leq (\ell + 1)^3$ . For all such tuples, we can check in parallel the Lamanness of  $H_j + F_j$ , for all  $j \in [\ell]$ . Since  $H_1, H_2, \dots, H_\ell$  are planar or of bounded treewidth (with all virtual edges), we can invoke the  $\text{NC}^2$ -algorithm from Theorem 19 or Theorem 5 to check whether  $H_j + F_j$  is Laman.

If  $\mathcal{F}_0 = \emptyset$ , then we can directly say that  $H$  is not Laman. When  $\mathcal{F}_0 \neq \emptyset$ , we construct an appropriate gadget. By (7), it suffices to construct a gadget  $\Gamma$  such that

$$\exists F_0 \in \mathcal{F}_0 \quad H_0 + F_0 \text{ is Laman} \iff H_0 + \Gamma \text{ is Laman.}$$

Recall that we need the construction of  $\Gamma$  to depend only on  $E(T)$ , but to be independent of  $H_0$ . For each family  $\mathcal{F}_0$  and  $E(T)$ , we construct a gadget graph  $\Gamma$  such that for any  $F_0 \subseteq \Delta_T - E(T)$ , we have

$$F_0 \in \mathcal{F}_0 \iff \Gamma \text{ can be obtained from } (T, F_0) \text{ via a sequence of Henneberg steps.} \quad (8)$$

If  $\mathcal{F}_0$  has a unique set  $\mathcal{F}_0 = \{F_0\}$ , then we take  $\Gamma = (T, F_0)$ . Note that by construction, each  $F_0 \in \mathcal{F}_0$  has the same cardinality. Thus,  $F_0$  is always unique when  $|E(T)| = 3$  or  $|E(T)| = 2$ , and may be unique in case  $|E(T)| = 1$  or  $|E(T)| = 0$ . In the following, we consider *all* cases where  $\mathcal{F}_0$  contains at least two sets. For these, we construct gadgets  $\Gamma$  shown in Table 1 that we put into  $H_0$ . We use the notation  $e_1 = (u, v)$ ,  $e_2 = (u, w)$ ,  $e_3 = (v, w)$ . Clearly, the definition of the gadgets is up to vertex relabeling.

Below we explain why property (8) holds for each of the gadgets given in the table. The implication from left-to-right in (8) is given in the description of the gadgets in Figure 3. We argue for the reverse direction. Note that in both types of Henneberg steps, the quantity  $2|V| - |E|$  remains constant. Hence, all graphs  $(T, F_0)$  which lead to the same gadget via Henneberg steps must have the same number of edges.

■ **Table 1** Gadgets for all possibilities of  $\mathcal{F}_0$  and  $E(T)$  where  $|\mathcal{F}_0| \geq 2$ .

Case	$E(T)$	$\mathcal{F}_0$	$\Gamma$
1	$\{e_1\}$	$\{\{e_2\}, \{e_3\}\}$	Figure 3 (a)
2	$\emptyset$	$\{\{e_1, e_2\}, \{e_1, e_3\}\}$	Figure 3 (b)
3	$\emptyset$	$\{\{e_1, e_2\}, \{e_1, e_3\}, \{e_2, e_3\}\}$	Figure 3 (c)
4	$\emptyset$	$\{\{e_1\}, \{e_2\}, \{e_3\}\}$	Figure 3 (a)
5	$\emptyset$	$\{\{e_1\}, \{e_2\}\}$	Not a valid possibility

**Case 1.** The only possible reverse Henneberg steps from the gadget in Figure 3 (a) is to remove the degree 3 node  $x$  and add a triangle edge. Since  $F_0$  is restricted to be a subset of  $\Delta_T - E(T) = \{e_2, e_3\}$ , the only possibility for the resulting graph is either  $(T, \{e_2\})$  or  $(T, \{e_3\})$ .

**Case 2.** Starting from the gadget in Figure 3 (b), the first reverse Henneberg step has to remove  $x$  because it has degree 3. Then the edge we have to add can only be  $e_1 = (u, v)$ . Then the second reverse Henneberg step can only be to remove  $y$ . The edge we can add has to be either  $e_2$  or  $e_3$ . Thus, the resulting graph can only be either  $(T, \{e_1, e_2\})$  or  $(T, \{e_1, e_3\})$ .

**Case 3.** From any two triangle edges we can construct the gadget in Figure 3 (c) as explained in the caption. Hence, starting from the gadget, we can reverse these steps and end up in any two triangle edges.

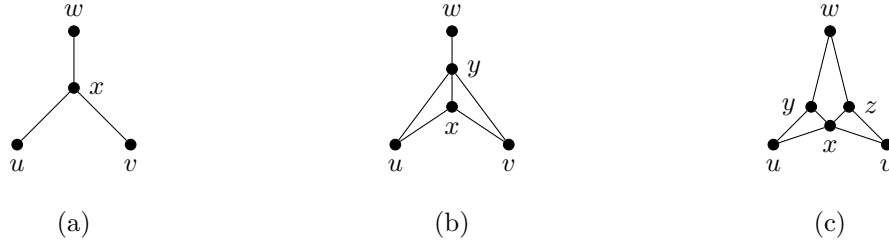
**Case 4.** The only possible reverse Henneberg steps from the gadget in Figure 3 (a) is to remove the degree 3 node  $x$  and add one of the triangle edges.

**Case 5.** We show that  $\mathcal{F}_0 = \{\{e_1\}, \{e_2\}\}$  is not a valid possibility. From the definition of  $\mathcal{F}_0$ , recall that we can have  $F_0 = \{e_1\}$  only when, say,  $F_1 = \{e_2, e_3\}$  and  $F_j = \{e_1, e_2, e_3\}$  for  $j \geq 2$ . Similarly, we can have  $F_0 = \{e_2\}$  only when  $F_1 = \{e_1, e_3\}$ . Then from Lemma 27,  $F_1 = \{e_1, e_2\}$  is also a valid choice for  $H_1$  to be Laman. Hence,  $F_0 = \{e_3\}$  should also be present in  $\mathcal{F}_0$ .

Finally, we argue the last part of the lemma about planarity and bounded treewidth. If component  $H'_0$  is planar, it can be embedded such that the triangle is one face of the embedding. Then we can put any of the gadgets from Figure 3 inside the triangle so that  $H_0 + \Gamma$  is planar.

If the component  $H'_0$  has bounded treewidth  $w$ , consider a tree decomposition  $(\mathcal{T}, \mathcal{B})$  of  $H'_0$  of width  $w$ . In a tree decomposition, every clique must be contained in a common bag [6, Lemma 2.1]. Thus, there must exist a bag  $B \in \mathcal{B}$  that contains the triangle nodes, i.e.,  $T \subseteq B$ .

To get a tree decomposition of  $H'_0 + \Gamma$ , we put an additional bag  $B' = V(\Gamma)$  in  $\mathcal{T}$  that is adjacent to  $B$ . Note that  $|B'| \leq 6$ . Hence, the treewidth of  $H'_0 + \Gamma$  is bounded by  $\max\{w, 5\}$ . The same holds for  $H_0 + \Gamma$  because treewidth does not increase when edges are removed. ◀



■ **Figure 3** Gadgets from Lemma 28. (a) The *or-gadget* that can be obtained from any one of the triangle edges via a Henneberg type 2 step. (b) The *uv-or-gadget* that can be obtained from edge  $(u, v)$  plus any one of the other two triangle edges via two Henneberg type 2 steps. (c) The *2-or-gadget* that can be obtained from any two triangle edges via a Henneberg type 1 step and two type 2 steps.

Now we can prove our main Theorem.

► **Theorem 29.** *Given a one-crossing-minor-free graph  $G$ , we can decide whether  $G$  is Laman in  $\text{NC}^3$ .*

**Proof.** We first decompose the input graph  $G$  into triconnected components in  $\text{NC}^2$  by Lemma 8. Then, in parallel we further decompose each triconnected component into 4-connected components in  $\text{NC}^2$  by Lemma 8 and we identify the large child paths in the 4-connected component trees by Lemma 9. As  $G$  is one-crossing-minor-free, the 4-connected components are planar or of bounded treewidth by Theorem 12.

By Lemma 22, we can decompose  $G$  into components resulting from the triconnected components, such that  $G$  is Laman iff all these components are Laman. Then for every such component  $C$ , in parallel, we decide whether it is Laman as follows. We apply Lemma 26 in a bottom up fashion along the 4-connected component tree of  $C$ . The leaf components in a 4-connected component tree contain a single separating triple and we can decide for what choices of triple edges the component is Laman by Theorem 19 or by Theorem 5. Then we put gadgets into the parent components according to Lemma 28. The gadgets we put into each separating triple in a parent component are only defined by the children components that are attached to the triple. In particular, we can put the gadgets into the parent components by working in parallel for every triple. Note that in case of overlapping separating triples, multi-edges could emerge in a parent component after we have added the gadgets. In this case, we detected that a parent component is not Laman. We continue this in a bottom-up fashion until we reach the root. Note that if we run this procedure as it is, the parallel complexity would be proportional to the depth of 4-connected component tree, which could be large.

Instead, when we reach a large child path along the way in component  $C$ , we deviate from the bottom-up evaluation. Let the large child path consists of components  $P_1, P_2, \dots, P_k$ , where  $P_i$  is the parent component of  $P_{i+1}$  in the 4-connected component tree. Let  $T_i$  be the separating triple between components  $P_i$  and  $P_{i+1}$ , for  $i = 1, 2, \dots, k - 1$ , and  $T_0$  be separating triple between component  $P_1$  and its parent in the tree.

If the above procedure reaches some component of the large child path at a separating triple  $T \neq T_i$ , for all  $i = 0, 1, \dots, k - 1$ , then we put a gadget as described above. Then each path component  $P_i$  is planar or of bounded treewidth and has at most two triples that

have not been replaced by a gadget. Therefore, for each path component in parallel we can apply Theorem 19 or Theorem 5 to check for what choices of edges in the two triples the component is Laman in  $\text{NC}^2$ . We describe how to merge the components  $P_1, P_2, \dots, P_k$  into one component.

**Merging two components.** Let  $H$  be a graph with a separating triple  $T$ . Let  $A, B$  and  $C_1, C_2, \dots, C_h$  be the components obtained when we split  $H$  at  $T$ . Let  $T_A$  and  $T_B$  be two other triples that are present in  $A$  and  $B$ , respectively. Suppose for each component  $C_i$ , we have already computed for which choices of edges in triple  $T$  it is Laman. Similarly, suppose we have computed for which choices of edges in triples  $T_A$  and  $T$ , component  $A$  is Laman, and analogously for the edges in triples  $T$  and  $T_B$ , w.r.t. component  $B$ . Then using the conditions in Lemma 26, we can find out for what edge choices in triples  $T_A$  and  $T_B$ , graph  $H$  is Laman. This can be done in  $\text{NC}^1$  because there is only a polynomial number of possibilities of putting the edges of triple  $T$  in components  $A, B, C_1, \dots, C_h$  by the condition in Lemma 26, which can be checked in parallel. Moreover, the number of edge choices in  $T_A$  and  $T_B$  is constant.

**Merging a path.** We apply the process of merging two components recursively in a binary tree fashion on  $P_1, P_2, \dots, P_k$ . At the bottom layer, we start with applying the above merge procedure on pairs of neighboring path components at their common separating triple  $T_i$ , in parallel. After merging two path components, we get a new component that again has two triples at each end and we have computed the edge choices in these two triples that make the component Laman.

When we have merged all path components into a single component, we find the choices of edges in the triple  $T_0$  for which the split graph in  $C - T_0$  that contains  $P_1$  is Laman. Then we put the corresponding gadget in  $T_0$  and carry on with the bottom up evaluation. Clearly, the above procedure for a large child path is in  $\text{NC}^2$ , as the merge step is in  $\text{NC}^1$ .

If it happens during the bottom up evaluation that a component or a large child path is not Laman for any choice of edges we can conclude that the graph is not Laman and stop the bottom-up evaluation.

Regarding the complexity, note that the  $\text{NC}^2$ -algorithms that we run as subroutines in the bottom up evaluation are the ones from Theorem 19 and 5. By Lemma 9, there are at most  $\log n$  many large child paths on a path from a leaf node to the root in the 4-connected component tree. Thus, the algorithm sequentially runs at most  $\log n$  many  $\text{NC}^2$  algorithms as subroutines and therefore we end up in  $\text{NC}^3$ . ◀

## Open problems

For  $K_{3,3}$ -free Laman graphs we can compute an infinitesimally rigid embedding efficiently in parallel. This is open for the case of one-crossing-minor-free graphs. In fact, it is open even for graphs of bounded treewidth. It is also open for  $K_5$ -free Laman graphs, even though the 4-connected components are all planar. A problem there is that we do not have the analog of Lemma 24 for separating triples. The main open problem is still to show rigidity or minimal rigidity in 2D for arbitrary graphs in  $\text{NC}$ . Note that even for planar graphs, rigidity is not known to be in  $\text{NC}$ .

A seemingly even more challenging open problem is to consider infinitesimal rigidity in *higher dimensions*. The  $m \times 2n$  rigidity matrix  $R = R_2$  in 2D can clearly be generalized to the  $m \times dn$  rigidity matrix  $R_d$ , for dimensions  $d \geq 3$ . The PIT problem for  $R_2$  is in polynomial time because of the various characterizations we have for rigidity, like Theorems 2, 3, and 13. However, we do not have such characterizations even for  $d = 3$ . A derandomization in polynomial time of the PIT for  $R_3$  is an open problem for decades. See also the exposition of Raz and Wigderson [35] on this topic.

## References

- 1 Timothy G. Abbot. Generalizations of Kempe’s universality theorem. Master’s thesis, Massachusetts Institute of Technology, 2008. Joint work with Reid W. Barton and Erik D. Demaine. URL: <http://web.mit.edu/tabbott/www/papers/mthesis.pdf>.
- 2 Zachary Abel, Erik D. Demaine, Martin L. Demaine, Sarah Eisenstat, Jayson Lynch, and Tao B. Schardl. Who needs crossings?: Noncrossing linkages are universal, and deciding (global) rigidity is hard. *Journal of Computational Geometry*, 16(1):378–452, 2025.
- 3 Rahul Arora, Ashu Gupta, Rohit Gurjar, and Raghunath Tewari. Derandomizing Isolation Lemma for  $K_{3,3}$ -free and  $K_5$ -free Bipartite Graphs. In *33rd Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 47 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 10:1–10:15, 2016. doi:10.4230/LIPIcs.STACS.2016.10.
- 4 Takao Asano. An approach to the subgraph homeomorphism problem. *Theoretical Computer Science*, 38:249–267, 1985. doi:10.1016/0304-3975(85)90222-1.
- 5 Leonard Asimow and Ben Roth. The rigidity of graphs II. *Journal of Mathematical Analysis and Applications*, 68(1):171–190, 1979.
- 6 Hans L. Bodlaender. NC-algorithms for graphs with small treewidth. In *Graph-Theoretic Concepts in Computer Science*, pages 1–10. Springer, 1989. doi:10.1007/3-540-50728-0\_32.
- 7 Erin Chambers and David Eppstein. Flows in one-crossing-minor-free graphs. In *Algorithms and Computation*, pages 241–252. Springer, 2010. doi:10.1007/978-3-642-17517-6\_23.
- 8 Archit Chauhan, Samir Datta, Chetan Gupta, and Vimal Raj Sharma. The Even-Path Problem in Directed Single-Crossing-Minor-Free Graphs. In *49th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 306 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 43:1–43:16, 2024. doi:10.4230/LIPIcs.MFCS.2024.43.
- 9 Bruno Courcelle. The monadic second-order logic of graphs I. *Information and Computation*, 85:12–75, 1990.
- 10 Samir Datta, Prajakta Nimbhorkar, Thomas Thierauf, and Fabian Wagner. Graph Isomorphism for  $K_{3,3}$ -free and  $K_5$ -free graphs is in Log-space. In *Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, volume 4 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 145–156, 2009. doi:10.4230/LIPIcs.FSTTCS.2009.2314.
- 11 Michael Elberfeld, Andreas Jakoby, and Till Tantau. Logspace versions of the theorems of Bodlaender and Courcelle. In *51st IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 143–152, 2010. doi:10.1109/FOCS.2010.21.
- 12 David Eppstein and Vijay V. Vazirani. NC algorithms for computing a perfect matching and a maximum flow in one-crossing-minor-free graphs. *SIAM Journal on Computing*, 50(3):1014–1033, 2021. doi:10.1137/19M1256221.
- 13 Ömer Eğecioğlu, Efstratios Gallopoulos, and Çetin K. Koç. A parallel method for fast and practical high-order Newton interpolation. *BIT Numerical Mathematics*, 30(2):268–288, 1990. doi:10.1007/bf02017348.
- 14 Stephen A. Fenner, Rohit Gurjar, and Thomas Thierauf. Bipartite perfect matching is in quasi-NC. *SIAM Journal on Computing*, 50(3):218–235, 2021. doi:10.1137/16M1097870.
- 15 Harold N. Gabow and Herbert H. Westermann. Forests, frames, and games: Algorithms for matroid sums and applications. *Algorithmica*, 7(1–6):465–497, 1992. doi:10.1007/BF01758774.
- 16 Herman Gluck. *Almost all simply connected closed surfaces are rigid*, volume 438 of *Lecture Notes in Mathematics*, pages 225–239. Springer, 1975.
- 17 Rohit Gurjar, Kilian Rothmund, and Thomas Thierauf. 2D Minimal Graph Rigidity is in NC for One-Crossing-Minor-Free Graphs. Technical Report TR25-103, Electronic Colloquium on Computational Complexity, 2025. URL: <https://eccc.weizmann.ac.il/report/2025/103/>.
- 18 Ruth Haas, David Orden, Günter Rote, Francisco Santos, Brigitte Servatius, Hermann Servatius, Diane Souvaine, Ileana Streinu, and Walter Whiteley. Planar minimally rigid graphs and pseudo-triangulations. *Computational Geometry*, 2005.

- 19 Philip Hall. On representatives of subsets. *Journal of the London Mathematical Society*, 10(37):26–30, 1935.
- 20 Bruce Hendrickson. Conditions for unique graph realizations. *SIAM Journal on Computing*, 21(1):65–84, 1992. doi:10.1137/0221008.
- 21 Lebrecht Henneberg. *Die graphische Statik der starren Systeme*. B. G. Teubner, 1911. URL: <https://books.google.de/books?id=jAIIIAAAAMAAJ>.
- 22 John E. Hopcroft and Robert E. Tarjan. Finding the triconnected components of a graph. Technical Report TR 72 - 140, Cornell University, 1972.
- 23 Bill Jackson and Tibor Jordán. Connected rigidity matroids and unique realizations of graphs. *Journal of Combinatorial Theory, Series B*, 94(1):1–29, 2005. doi:10.1016/j.jctb.2004.11.002.
- 24 Pieter W. Kasteleyn. *Graph theory and crystal physics*, pages 43–110. Academic Press, 1967.
- 25 Samir Khuller. Coloring algorithms for  $K_5$ -minor free graphs. *Information Processing Letters*, 34(4):203–208, 1990. doi:10.1016/0020-0190(90)90161-P.
- 26 Samir Khuller. Extending planar graph algorithms to  $K_{3,3}$ -free graphs. *Information and Computation*, 84(1):13–25, 1990. doi:10.1016/0890-5401(90)90031-C.
- 27 Gerard Laman. On graphs and rigidity of plane skeletal structures. *Journal of Engineering Mathematics*, 4(4):331–340, 1970.
- 28 Charles H. C. Little. *An extension of Kasteleyn's method of enumerating the 1-factors of planar graphs*, volume 403 of *Lecture Notes in Mathematics*, pages 63–72. Springer, 1974.
- 29 László Lovász. *Graphs and Geometry*, volume 65 of *Colloquium Publications*. American Mathematical Society, 2019.
- 30 László Lovász and Y. Yemini. On generic rigidity in the plane. *SIAM Journal on Algebraic Discrete Methods*, 3(1):91–98, 1982. doi:10.1137/0603009.
- 31 Saunders Maclaine. A structural characterization of planar combinatorial graphs. *Duke Mathematical Journal*, 3(3):460–472, 1937.
- 32 Ketan Mulmuley. A fast parallel algorithm to compute the rank of a matrix over an arbitrary field. *Combinatorica*, 7(1):101–104, 1987. doi:10.1007/BF02579205.
- 33 Ketan Mulmuley, Umesh V. Vazirani, and Vijay V. Vazirani. Matching is as easy as matrix inversion. *Combinatorica*, 7(1):105–113, 1987. doi:10.1007/BF02579206.
- 34 Hilda Pollaczek-Geiringer. Über die Gliederung ebener Fachwerke. *ZAMM - Journal of Applied Mathematics and Mechanics*, 7(1):58–72, 1927.
- 35 Orit E. Raz and Avi Wigderson. *Subspace Arrangements, Graph Rigidity and Derandomization Through Submodular Optimization*, pages 377–415. Springer, 2019.
- 36 Omer Reingold. Undirected connectivity in log-space. *Journal of the ACM*, 55(4), 2008. doi:10.1145/1391289.1391291.
- 37 Neil Robertson and Paul Seymour. Excluding a graph with one crossing. *Graph Structure Theory*, pages 669–675, 1993. doi:10.1090/conm/147/01206.
- 38 Jonathan Rollin, Lena Schlipf, and André Schulz. Recognizing Planar Laman Graphs. In *27th European Symposium on Algorithms (ESA)*, volume 144 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 79:1–79:12, 2019. doi:10.4230/LIPIcs.ESA.2019.79.
- 39 Simon Straub, Thomas Thierauf, and Fabian Wagner. Counting the number of perfect matchings in  $K_5$ -free graphs. *Theory of Computing Systems*, 59(3):416–439, 2016. doi:10.1007/S00224-015-9645-1.
- 40 Ileana Streinu. A combinatorial approach to planar non-colliding robot arm motion planning. In *41st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 443–453, 2000. doi:10.1109/SFCS.2000.892132.
- 41 Tiong-Seng Tay and Walter Whiteley. Generating isostatic frameworks. *Structural Topology*, 11:21–68, 1985.
- 42 Thomas Thierauf and Fabian Wagner. Reachability in  $K_{3,3}$ -free Graphs and  $K_5$ -free Graphs is in Unambiguous Logspace. *Chicago Journal of Theoretical Computer Science*, 2, 2014.

## 49:22 Minimal Graph Rigidity in NC for OCMF Graphs

- 43 Vijay V. Vazirani. NC algorithms for computing the number of perfect matchings in  $K_{3,3}$ -free graphs and related problems. *Information and Computation*, 80(2):152–164, 1989. doi: 10.1016/0890-5401(89)90017-5.
- 44 Klaus Wagner. Über eine Eigenschaft der ebenen Komplexe. *Mathematische Annalen*, 114(1):570–590, December 1937.