Infinitesimally Locked Linkages with Applications to Locked Trees

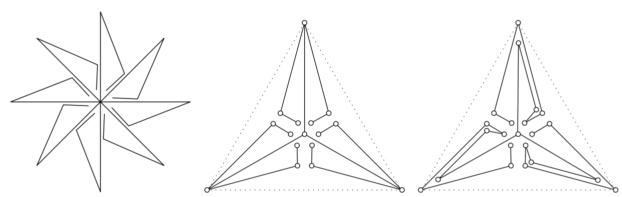
Robert Connelly* Erik D. Demaine† Günter Rote‡

Abstract. Recently there has been much interest in linkages (bar-and-joint frameworks) that are *locked* or *stuck* in the sense that they cannot be moved into some other configuration while preserving the bar lengths and not crossing any bars. We propose a new algorithmic approach for analyzing whether planar linkages are locked in many cases of interest. The idea is to examine degenerate frameworks in which multiple edges converge to geometrically overlapping configurations. We show how to study whether such frameworks are locked using techniques from rigidity theory, in particular first-order rigidity and equilibrium stresses. Then we show how to relate locked degenerate frameworks to locked frameworks that closely approximate the degenerate frameworks. Our motivation is that most existing approaches to locked linkages are based on approximations to degenerate frameworks. In particular, we show that a previously proposed locked tree in the plane [1] can be easily proved locked using our techniques, instead of the tedious arguments required by standard analysis. We also present a new locked tree in the plane with only one degree-3 vertex and all other vertices degree 1 or 2. This tree can also be easily proved locked with our methods, and implies that the result about opening polygonal arcs and cycles [2] is the best possible.

1 Linkages. A *linkage* is a graph together with an assignment of lengths to edges; each edge is called a rigid *bar*. A *polygonal arc*, *polygonal cycle*, or *polygonal tree* is a linkage whose graph is a single path, cycle, or tree, respectively. A *configuration* of a linkage in $\mathbb{R}^d$ is a mapping of the vertices to points in $\mathbb{R}^d$ that satisfies the bar-length constraints. A configuration is *(strongly) simple* if only incident bars intersect, and then only at the common endpoint. A *motion* is a continuum of configurations, that is, a continuous function mapping the time interval $[0, 1]$ to configurations; often, each configuration is required to be simple. The *configuration space* of a given subset of configurations (e.g., simple configurations) is the space in which points correspond to configurations and paths correspond to motions.

We focus here on *planar linkages* embedded in $\mathbb{R}^2$. In this case, the linkage also specifies the combinatorial planar embedding because this cannot change by a motion that avoid crossings. It is known that the configuration space of simple planar configurations is not always connected for a polygonal tree [1], and is always connected for a polygonal arc, polygonal cycle, or indeed any disjoint union of nonnesting polygonal arcs and cycles [2, 3]. The key distinction is that arcs and cycles have maximum degree 2, but a tree may have vertices of higher degree.

Two questions naturally arise from these results. First, how many high-degree vertices are necessary, and how high must the degrees be, to make a tree have a disconnected configuration space? We settle this question by proving that the maximum-degree-2 result [2] is tight: a single degree-3 vertex can prevent opening. See Figures 1(b) and 1(c) for the two-step construction. Second, and more generally, how can we tell whether a linkage has a connected configuration space? We present a method for designing examples that can be proved by a polynomial-time algorithm to have a disconnected configuration space, and furthermore to be *locked* in a stronger sense. This result does not settle the algorithmic decision problem, but solves many cases of interest. In particular, we use this result in our solution to the first problem.



(a) From [1] (b) New 1 (c) New 2

Figure 1: Locked planar polygonal trees.

2 Self-Touching Linkages. Here we begin the exploration of the analogous linkage problems when bars are allowed to touch, and even lie along each other, but not properly cross. This notion requires topological information in addition to the geometric embedding, to keep track of which vertices are on which side of which other bars. More precisely, a *self-touching configuration* of a linkage consists of a configuration (the geometry) together with, for each vertex and bar that are not incident in the graph of the linkage but are nonetheless touching geometrically, a *sidedness specification* of on which side of the bar the vertex is.

3 Locked Linkages. There are two natural notions of being "locked." (1) We call a self-touching linkage *locked* if the configuration space has multiple connected components. (2) We call a self-touching configuration *locked within* ε if no path in the configuration space (motion) can get outside of a surrounding ball of radius ε. (Distance is measured according to some reasonable metric on the configuration space.) The second definition is stronger for suffi-

*Department of Mathematics, Cornell University, Ithaca, NY 14853, U.S.A., connelly@math.cornell.edu

†MIT Laboratory for Computer Science, 200 Technology Square, Cambridge, MA 02139, U.S.A., edemaine@mit.edu

‡Institut für Informatik, Freie Universität Berlin, Takusstraße 9, D-14195 Berlin, Germany, rote@inf.fu-berlin.de

ciently small ε.

We also introduce a stronger notion of being locked. First, a *δ-perturbation* of a self-touching configuration is a repositioning of the vertices within disks of radius δ that is consistent with the sidedness specification. The key aspect of a perturbation is that it allows the bar lengths to change slightly (each by at most 2δ). In particular, for every self-touching configuration and for every $\delta > 0$, there is a δ-perturbation that is a simple configuration.

Now a self-touching configuration is *strongly locked* if, for every $\varepsilon > 0$, there is a $\delta > 0$ such that every δ-perturbation is locked within ε. In particular, all sufficiently small simple perturbations are locked. Thus the definition of strongly locked configurations provides a connection between the less-intuitive notion of self-touching configurations and the more commonly studied notion of simple configurations.

4 Rigidity. The notions of rigidity and infinitesimal rigidity of configurations of linkages are well-understood when the bars are permitted to cross each other; see e.g. [2]. For simple configurations of linkages whose bars are not permitted to cross, the no-crossing constraint automatically holds for a short interval of time, so the notions of rigidity and infinitesimal rigidity remain the same.

For self-touching linkages, however, the no-crossing constraint introduces additional restrictions at the very beginning of motion; see Figure 2. These constraints specify that a vertex $\mathbf{p}_k$ touching the relative interior of the bar $\mathbf{p}_i\mathbf{p}_j$ must remain on the bar or move to the correct side. In other words, the area of the triangle $\mathbf{p}_i\mathbf{p}_j\mathbf{p}_k$ cannot decrease or increase, depending on whether $\mathbf{p}_k$ is specified to be left or right of $\mathbf{p}_i\mathbf{p}_j$. This constraint can be viewed as a *zero-length strut* connecting $\mathbf{p}_k$ and the corresponding point on the bar $\mathbf{p}_i\mathbf{p}_j$.

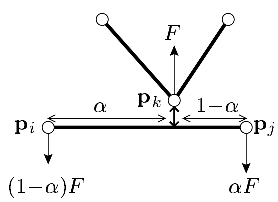


Figure 2: Zero-length strut and proportional distribution of stress F.

The dual notion of an infinitesimal motion (in the linear-programming sense) is an *equilibrium stress*. A *stress* assigns a real number to each bar and a nonnegative real number to each zero-length strut. Intuitively, if the stress is negative, then the bar pulls on both its endpoints by a force proportional to the stress; and if the stress is positive, then the bar or strut pushes against the two ends by the same amount. For zero-length struts, a force perpendicular to $\mathbf{p}_i\mathbf{p}_j$ pushes $\mathbf{p}_k$ and $\mathbf{p}_i\mathbf{p}_j$ apart by pushing directly on $\mathbf{p}_k$ and by distributing force proportionally to $\mathbf{p}_i$ and $\mathbf{p}_j$ according to the relative distance of $\mathbf{p}_k$ along the bar $\mathbf{p}_i\mathbf{p}_j$. A stress is *in equilibrium* if the forces at each vertex sum to the zero vector.

The *bar version* of a self-touching configuration results from replacing each zero-length strut constraint with a zero-length bar constraint. In other words, any vertex $\mathbf{p}_k$ on the relative interior of the bar $\mathbf{p}_i\mathbf{p}_j$ must remain on that bar, keeping the area of the triangle $\mathbf{p}_i\mathbf{p}_j\mathbf{p}_k$ zero.

Lemma 1 *If a self-touching configuration has a nonzero equilibrium stress, and the bar version of the configuration is rigid, then the configuration is infinitesimally rigid.*

5 Main Result. Our main result is the following connection between rigid and locked linkages:

Theorem 1 *If a self-touching linkage is infinitesimally rigid, then it is strongly locked.*

6 Proving a Configuration to be Strongly Locked. Using the tools above, one can follow the following outline for proving that a particular configuration is strongly locked. First, model the configuration as a small perturbation of a self-touching configuration of a linkage with slightly different edge lengths. Second, check that the bar version of the self-touching configuration is rigid. This step is normally quite easy because the zero-length bars restrict motions severely, often creating rigid triangles. Third, check via linear programming that there is a nonzero equilibrium stress. If the answer to the second and third parts is "yes," then the self-touching configuration is strongly locked, and hence the original configuration is strongly locked.

The key aspect of this approach is that the third step is automated. However, the third step is conservative: if it reports that it found no nonzero equilibrium stress, the linkage may nevertheless be infinitesimally rigid. The difficulty is with a vertex on the reflex side of another vertex, where the corresponding stress from the zero-length struts cannot be specified by a linear constraint. Fortunately, often this difficulty does not arise (as in Figure 1(a)), and when it arises, it can often be overcome (as in Figure 1(c)).

References.

[1] T. Biedl, E. Demaine, M. Demaine, S. Lazard, A. Lubiw, J. O'Rourke, S. Robbins, I. Streinu, G. Toussaint, and S. Whitesides. A note on reconfiguring tree linkages: Trees can lock. *Discrete Appl. Math.*, 2001. `http://www.arXiv.org/abs/cs.CG/9910024`.

[2] R. Connelly, E. D. Demaine, and G. Rote. Straightening polygonal arcs and convexifying polygonal cycles. In *Proc. 41st IEEE Sympos. Found. Comp. Sci.*, pp. 432–442, 2000.

[3] I. Streinu. A combinatorial approach to planar non-colliding robot arm motion planning. In *Proc. 41st IEEE Sympos. Found. Comp. Sci.*, pp. 443–453, 2000.