Movement Problems for 2-Dimensional Linkages

J.E. Hopcroft

D.A. Joseph

S.H. Whitesides

August 1982

TR 82-515

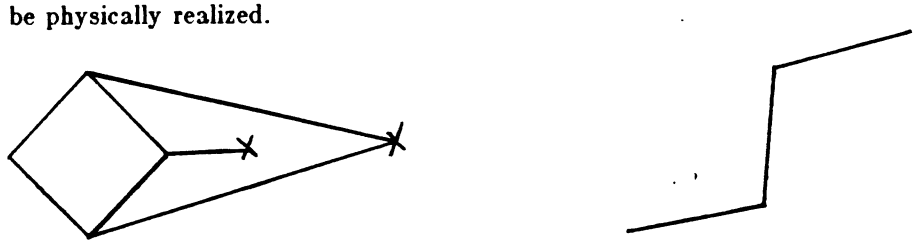
Department of Computer Science
Cornell University
Ithaca, New York 14853

1. Introduction

This paper is concerned with the motion of linkages from the computational complexity point of view. The research was motivated by earlier work in robotics, particularly that of Lozano-Perez and Wesley [LW-79], Lozano-Perez [L-80], Reif [R-79] and Schwartz and Sharir [S-81, S-82]. There are two natural ways in which linkage movement problems arise in robotics. First, a linkage can model a robot arm. A frequently encountered model consists of a sequence of links connected together consecutively at movable joints. Second, linkages can also model hinged objects being moved by an arm or other type of manipulator. In both cases, it is essential to plan collision-avoiding paths of motion, as the manipulator and the object it is moving are generally required to lie within regions whose boundaries are determined by walls and the presence of other objects in the work space.

A *linkage* is a collection of rigid rods called *links* (see Figure 1.1). The endpoints of various links are connected by joints, each joint connecting two or more links. The links are free to rotate about the joints. In a planar linkage, links are allowed to cross over one another, and the linkage may be fastened to the plane so that the locations of certain joints are fixed (the fixed joints are indicated by "x"'s).

In a physical realization of a planar linkage, each link could move in a separate plane parallel to the ground. If links were joined together or to the ground by pins, then a link in one plane might collide with a pin joining links in two other planes. However, it is not difficult to design simple devices that function like pins but that do not interfere with the motions of the linkage. Thus the mathematical model in which links cross over one another and in which the locations of some joints are fixed can be physically realized.



(a) (b)

Figure 1.1 (a) A linkage (b) An arm

An *arm* is a simple type of linkage consisting of a sequence of links joined together consecutively with the location of one end fixed.

Suppose that an arm is required to stay inside some bounded region R of the plane that is connected, but not necessarily simply connected. Then it is a natural question to ask whether new links can be adjoined to the arm in such a way that the original links in the arm automatically stay inside R. The new links may move outside R, and some of the links may have an endpoint fixed in the plane. The key requirement is that no motion of the arm inside R be prevented by the addition of the new links. Clearly this can be done if R is a circular region. In Figure 1.2 we show an arm constrained by a circular region C with and without the circle being physically present.

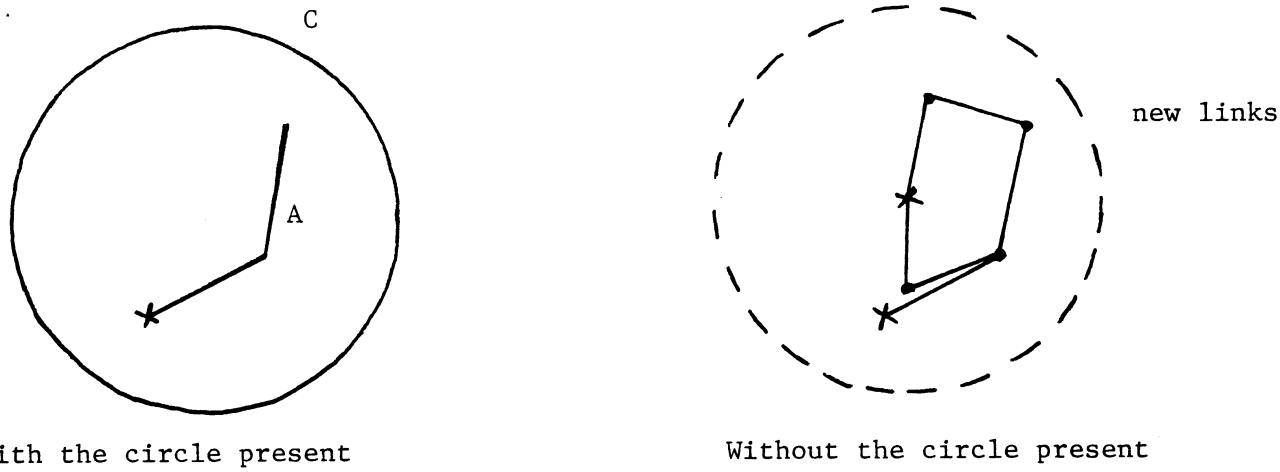


Figure 1.2 An arm constrained by a circle

The reason that we are interested in reductions of this sort is that the motions of the new linkage can be studied without reference to the region R. The first main result of our paper is that for any connected, but not necessarily simply connected, region R whose boundary consists of a finite set of straight line segments and any linkage L positioned within R, there is a reduction of the type we have just described. What's more the new linkage has a description whose size is polynomial in the size of the description of the original linkage and region R and the description can be computed in polynomial time.

Our second result is that the reachability question for planar linkages is PSPACE-hard. In other words, given an initial configuration of an arbitrary planar linkage L, a joint J in that linkage,

and a point p in the plane, the question of whether L can be moved so that J reaches p is PSPACE-hard.

The main technique used throughout the paper is to build complex linkages by connecting together simpler special purpose linkages. Some of these simplier linkages date from the 19th century and are described in Section 2. These include Peaucellier's straight line motion device, a linkage containing a joint whose locus is *exactly* a straight line segment and linkages that translate and rotate vectors and multiply distances.

Section 3 contains an easy demonstration that a linkage required to move inside a bounded convex polygonal region R can be embedded in a more complex linkage that enforces the boundary constraint for the original linkage. The extension of this result to a linkage L constrained to move inside a nonconvex bounded region R with straight line boundaries appears in Section 4. We obtain this result by triangulating the complement of R in its convex hull H, designing a linkage that contains a joint whose locus is a triangle, and then using this device to build a linkage that can keep a link entirely outside a triangle. By keeping each link of L outside each triangle in $H-R$ while requiring each joint of L to remain inside H, we keep L inside R. We do this in such a way that the motion of L is not restricted in any other way.

Section 5 contains our other main result-that the reachability question for planar linkages is PSPACE-hard. We obtain this result by designing a linkage that can simulate a linear bounded automaton LBA. The result should be compared to Reif's result [R-79] that in 3-dimensional space, the reachability problem is PSPACE-hard even for a simple, hinged, tree-like linkage required to move in a nonconvex region.

2. Simple linkages

2.1 Overview

This section describes planar linkages that perform certain tasks. After a discussion in Section 2.2 of Peaucellier's straight line motion linkage, we show in Section 2.3 how to use this device to build linkages that can translate and rotate vectors. Then in Section 2.4 we use these devices to give a modified version of Kempe's construction of a linkage that "solves" a multivariable polynomial

equation [K-1876]. This linkage has certain joints whose positions represent values of variables $x_1,\ldots,x_n$, and the only constraint that the linkage puts on the motion of these joints is that the implied values of the x_i stay within given bounded domains and satisfy a given polynomial equation.

The linkage for solving a polynomial equation plays an important role in both the main results. We use it to keep links outside of triangular regions when we show how to build boundary constraints into a linkage in Section 4. We also use it to synchronize the motions of the LBA simulator given in Section 5. Two important subtleties arise in designing special purpose linkages.

First, we often want to construct a linkage L having a joint J whose locus is some specified set of points. It is important to understand that in such a case, L must be able to move to all points in the set but to *no* other points. Historically, some linkages that have been proposed for performing certain tasks have been faulty because, while they are able to move in some desirable way, they can also move to "configurational singularities" at which they can begin undesired motions. Hence we include correct versions of the linkages that we use.

The second important subtlety is this. Suppose that the locus of some joint J in linkage L is a set of points S and that the locus of some joint J' in linkage L' is a set S'. Now suppose that J and J' are identified. It is not necessarily true that the joint $J=J'$ can then reach all points in $S \cap S'$. Indeed $S \cap S'$ need not be connected! The crucial observation here is that the new linkage can move so that the coordinates of J are given by $(x(t),y(t))$, where x and y are continuous functions of time, if and only if L and L' can move separately so that both the coordinates of J and the coordinates of J' are also given by $(x(t),y(t))$.

2.2 Peaucellier's straight line motion linkage

In 1864 Peaucellier [P-1864] designed a linkage, shown in Figure 2.2.1, that converts circular motion to linear motion. Links AD, AB, DC and BC have equal length as do links EA and EC. The length of FD equals the distance from E to F. The locations of joints E and F are fixed points in the plane, but the linkage is allowed to rotate about these points. As it does, the joint B traces out the line segment XY. This can be seen by observing two facts. First, joints D and B always lie on a ray through E. Second, the distances h, r and t shown in Figure 2.2.2 satisfy

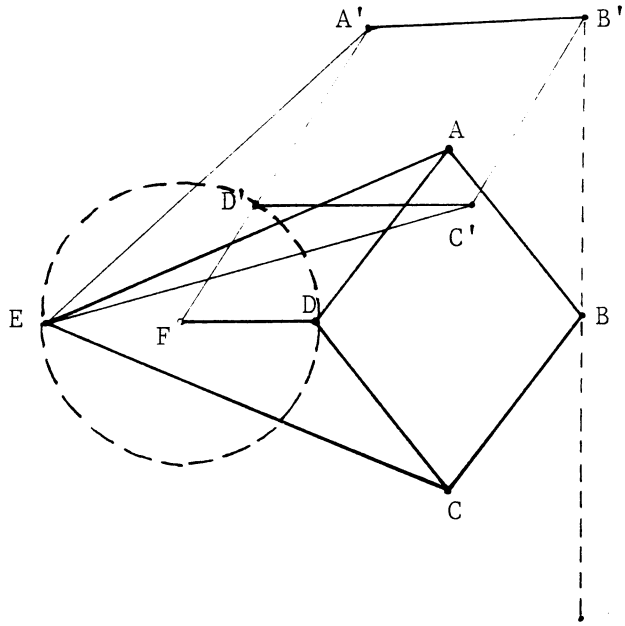


Figure 2.2.1 The Peaucellier straight line motion linkage

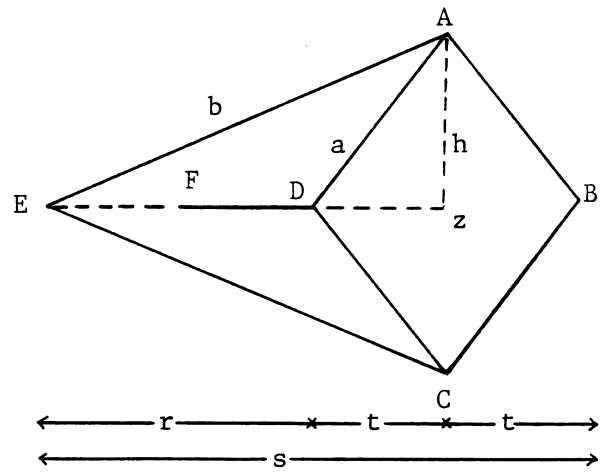


Figure 2.2.2 Consideration of triangles EAZ and DAZ shows that $h^2=a^2-t^2=b^2-(r+t)^2$

$h^2=a^2-t^2=b^2-(r+t)^2$, where r is the distance between D and E. Consequently the distance s between E and B is such that rs is equal to the constant b^2-a^2. (Here h, r, s and t are functions of the position of D.) Hence, this device can be thought of as performing the well known mapping called "inversion with respect to a circle" [E-63]. In this mapping, the image of a point $p=(r,\theta)$ is

the point $p' = (r' \theta)$ where rr' is some given constant. It is known that this mapping takes circles to circles, where a straight line is regarded as a circle of infinite radius. Suppose that the joint E of the Peaucellier device is at the origin of the polar coordinate system and that the given constant is b^2-a^2. Then the device computes the images of the points that D can reach. Since the circle of radius $|FD|$ about F goes through the origin, this circle is mapped to a straight line, in particular the line through X and Y. The points X and Y represent the extremes that B can reach.

Notice that the relative lengths of the links is not too important provided that the linkage can be assembled as shown in Figure 2.2.1 with E,F,D and B on a straight line. In order to argue that the Peaucellier device works correctly, we must demonstrate that joint B cannot reach joint D. For if this could occur, the joint B could leave the line segment and trace out part of the circle that D traces. Similarly we must demonstrate that joint A cannot reach joint C. Joint B cannot reach joint D since the line XY does not intersect the circle of radius $|FD|$ centered at F. Joint A cannot reach joint C, since as D moves counter clockwise, say, the angle FDA straightens and prevents further motion. In this instance, E,D and B are on a straight line with A and C on opposite sides. The reader is referred to [E-63] for a more detailed discussion.

If instead of constraining the joint D to remain on the circle about F, we allow it to move inside the circle, (this is done by adding an additional joint G to the midpoint of the link FD) then the joint B can reach points in the half plane to the right of XY. It is important that D not be allowed to move so far right that A and C collapse together. This is done by adding two links restricting B to a circular region of radius EX centered at E. The joint B can follow any curve in the semicircular region that is the intersection of the half plane to the right of XY with the circle of radius EX centered at E. (See Figure 2.2.3)

Notice that the region, R, of points reachable by the modified Peaucellier linkage is fairly large compared to the length of the links used in the linkage. That is, given a polygonal region R the description of a Peaucellier linkage whose locus of points includes R is polynomially related to the description of R.

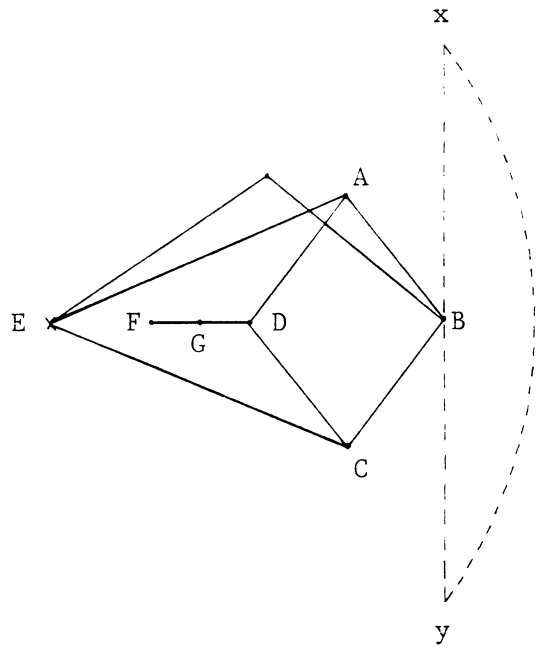


Figure 2.2.3 The region of points reachable by the modified Peaucellier linkage.

We will frequently use the Peaucellier linkage to constrain a joint J of some other linkage to a line. This can be done by identifying the joint J with the joint of the Peaucellier linkage that moves on a line. When we do this identification, we say that J is moving in a *slot*. The geometry and positioning of the Peaucellier linkage determine the length and position of the slot.

Also observe that the two points of the Peaucellier device that are normally attached to the plane could instead be attached to a rigid structure made up of links that is free to move in the plane. In this situation the slot itself has allowable motions. (See Figure 2.2.4.)

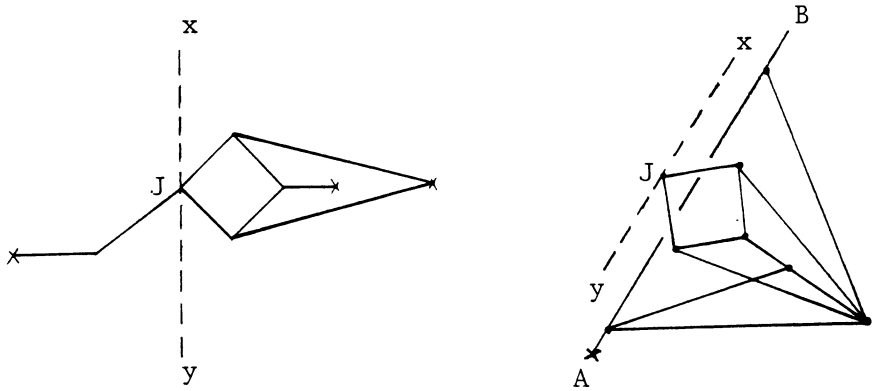


Figure 2.2.4 (a) An arm in a slot (b) A slot on a platform

2.3 Translators and rotators

We will need linkages to perform certain basic tasks. Since many of the previously published constructions have deficiencies of the sort described earlier, we include correct versions of these linkages. We do not attempt to construct the simplest linkage for a task, but rather one that is conceptually easy to understand and to prove correct. Throughout this section, we assume that R is a given bounded planar region.

The first device we construct is a translator. A *translator* is a linkage such that the only restriction on the movement of four of its joints S,T,U and V in the region R is that the position of T relative to S remains the same as the position of V relative to U. Alternatively, any three of these joints can be moved freely, and the position of the fourth joint is uniquely determined by the above relation and the position of the other three.

The linkage consisting of four parallelograms shown in Figure 2.3.1 is a natural candidate for a translator. Joints S, T and U can be moved to any three points in the plane provided the distance between S and T does not exceed a+ b and the distance between S and U does not exceed c+ d. At first it appears that the position of V relative to U is always the same as the position of T relative to S, i.e., that the vector ST is equal to the vector UV. The difficulty is that one or more of the parallelograms may convert to a contraparallelogram (see Figure 2.3.2), and thus other motions are possible.

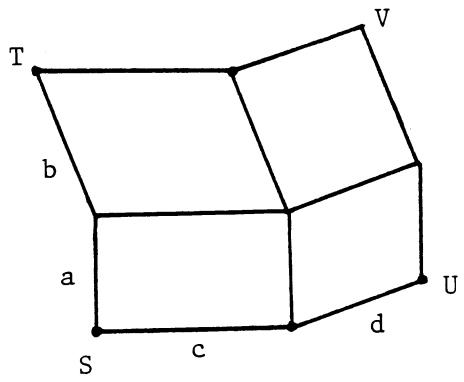


Figure 2.3.1. A faulty translator

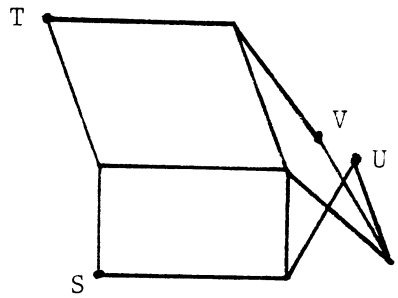


Figure 2.3.2 Conversion of a parallelogram to a contraparallelogram

One might attempt to overcome this difficulty by attach to each diagonal of the parallelograms a sufficiently short two-link segment. This would keep a parallelogram from straightening. Unfortunately, this also prevents the movement of T to S when S is held fixed. and this motion is essential in a construction of Kempe's that we use. We solve the problem by using a more complex device involving nine parallelograms. The linkage shown in Figure 2.3.3 will be part of this device. The lengths of the links A_1B_1 ,B_1C_1 and C_1D_1 can be chosen long enough so that no matter where A_1 is positioned inside the bounded region R, D_1 can move freely in R while A_1 is kept fixed and C_1 is constrained to move on a line l through A_1 by means of a *slot* (see Section 2.2). In fact, if the links are sufficiently long, then the slot can be constructed so that the angles between l and links A_1B_1 and B_1C_1 does not exceed 30° no matter how D_1 moves in R. (Of course the joints B_1 and C_1 are outside R, but this does not concern us, as we will never need to attach them to the joints of a linkage required to stay inside R.) Also note that the angle between C_1D_1 and l can be kept to at

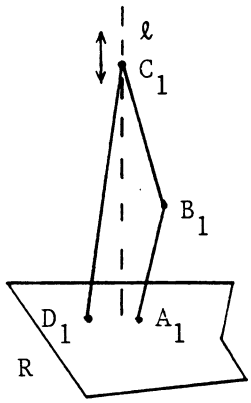


Figure 2.3.3 Keeping links nearly "vertical". (The slot along l and the two link connection between A_1 and D_1 are not shown.)

most 30° by the addition of a two-link segment connecting A_1 to D_1 and having length equal to the diameter of R. A similar linkage with joints A_1, A_2, A_3 and A_4 can be constructed so that A_4 can move freely in R while the angles between A_1A_2, A_2A_3 and A_3A_4 and another line l' through A_1 are kept within 30°. It is convenient to choose l' perpendicular to l since the links in the segments $A_1, \ldots, D_1$ and $A_1, \ldots, A_4$ will appear as sides of parallelograms in our nine-parallelogram translator. The fact that these links can be kept nearly parallel to l and l' will prevent any of the nine parallelograms from straightening. In this way, we avoid the flaw in the faulty four-parallelogram translator.

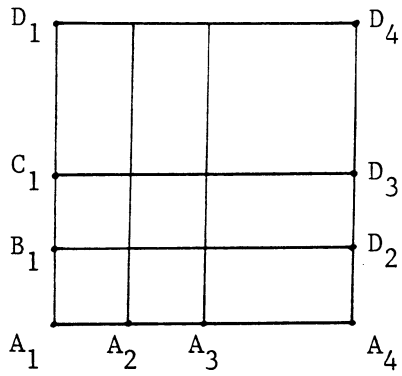


Figure 2.3.4 The predecessor of a translator.

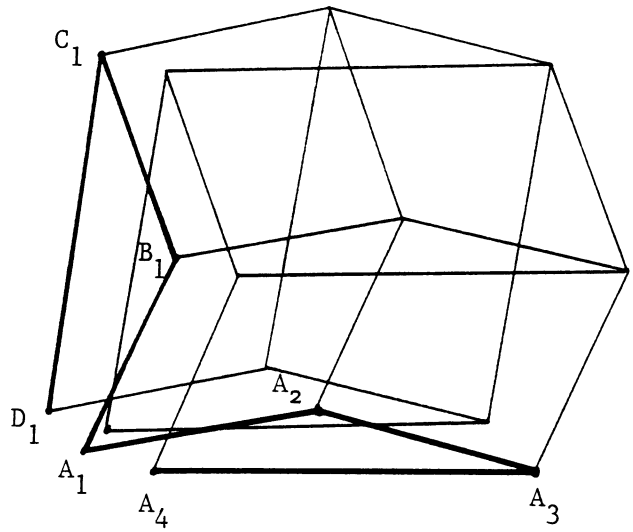


Figure 2.3.5 A translator. (The constraining devices attached to segments $A_1,\ldots,D_1$ and $A_1,\ldots,A_4$ are not shown.)

To construct the main body of the translator, begin with the nine-parallelogram linkage shown in Figure 2.3.4. The three link segments connecting A_1,B_1,C_1 and D_1 and A_1,A_2,A_3 and A_4 are not yet constrained as described above. Notice (by applying the parallelogram law of vector addition) that it is possible to move these segments independently of each other without breaking links or creating contraparallelograms, although parallelograms may straighten. As long as no contraparallelograms are created, the position of D_1 relative to A_1 is the same as the position of D_4 relative to A_4. Now move D_1 and A_4 (and hence D_4) to A_1, as shown in Figure 2.3.5, and then attach the constraining devices described in the discussion of Figure 2.3.3 to A_1,B_1,C_1 and D_1 and to A_1,A_2,A_3 and A_4. Joints D_1 and A_4 can still move freely in R, but the links in the segments $A_1,\ldots,D_1$ and $A_1,\ldots,A_4$ must remain nearly parallel to l and l', preventing the formation of contraparallelograms.

The next device we construct is called a rotator. A *rotator* is a linkage such that the only restriction on the movement inside of R of three of its joints A, I and H is that the distance from A to I be equal to the distance from A to H. In this construction, we begin with the quadrilateral linkage $ABCD$ shown in Figure 2.3.6a. The lengths of the sides of $ABCD$ satisfy $|AD| = |AB| < |CD| = |CB|$. Then we constrain C to a slot through A. We want to insure that the slot through A always bisects the angle DAB. We also want to insure that AD and AB can rotate freely about A, so it is necessary that $ABCD$ be able to straighten to allow links AB and AD to cross over each other. However, when B coincides with D, B and D must not be allowed to simultaneously move off the line AC in the same direction. If this happens the slot through A would no longer bisect the angle DAB. To solve this problem, we construct another quadrilateral $AGFE$ with link lengths satisfying $|AG| = |AE| < |FG| = |FE|$ and also, $|AE| + |EF| > |AD| + |DC|$. Then we constrain F to move in the slot through A in which C moves. Finally, we join the quadrilaterals by adding links ED and BG. Now D and B can rotate freely about A (for an appropriately designed slot), but $ABCD$ must be straight whenever B and D coincide. Hence, B and D cannot move to the same side of the slot through A and the slot remains the bisector of angle DAB.

Now we attach platforms to AD and AB (as shown in Figure 2.2.4), and slots that coincide with AD and AB. We then add links IJ and HJ, where I is constrained to move in the slot along

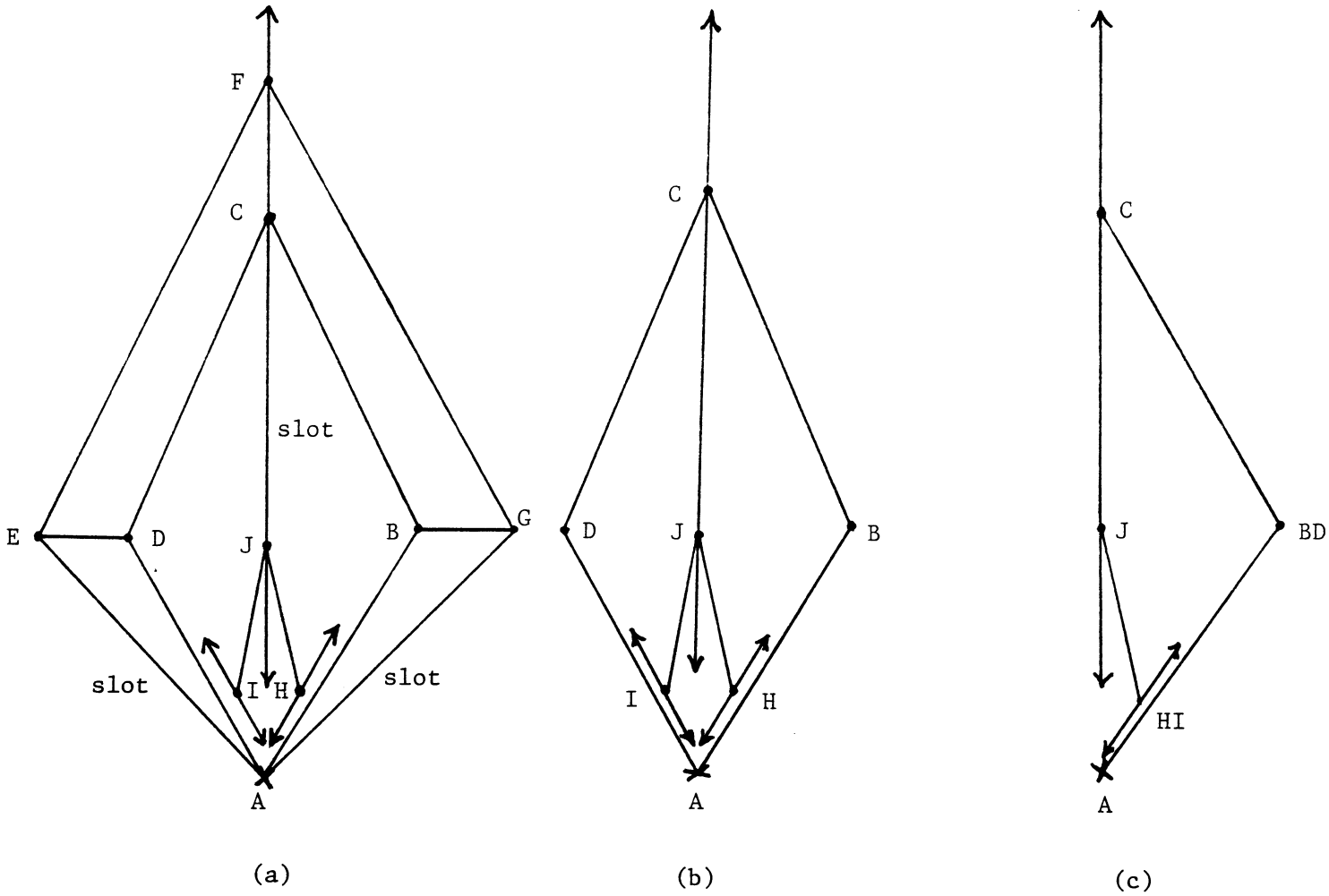


Figure 2.3.6 A distance rotator. Without the addition of the quadrilateral $AGFC$, D could move to B and the two superimposed joints could then move to the same side of the slot through A and C. Hence the slot through A and C would no longer bisect angle DAB.

AD, H is constrained to move in the slot along AB, and J is constrained to the slot in which C and F move. Since triangles AIJ and AHJ are always congruent, the distance between A and I must equal the distance between A and H. Note that this is the only constraint on the motion of I and H. This completes the construction of a rotator.

Now we combine a translator that keeps the relative position of T to S equal to the relative position of V to U (but does not otherwise restrict their motions inside R) with a rotator that keeps the distance between A and H equal to the distance between A and I (but does not otherwise constrain their motions inside R). We do this simply by identifying A with U and H with V. The result is a linkage containing four joints $S, T, A=U$ and I whose motions inside R must satisfy only

one requirement, that the distance between S and T be equal to the distance between $A=U$ and I. We call this device a *distance copier.*

A distance copier can be used to construct an angle adder. An *angle adder* is a linkage containing four equal-length links OA, OB, OC and OD whose motions are constrained only by the requirement that angle AOD be equal to angle AOB plus angle AOC. We will only need a device that correctly adds angles AOB and AOC when angle AOC is less than π. To construct such a device, we take four equal-length links OA, OB, OC and OD and attach a distance copier to A, B, C and D that keeps the distance between A and C equal to the distance between B and D. Then to ensure that angle AOC is added to angle AOB rather than subtracted from it, we add two links OE and EB to form a triangle with a right angle at O. Now we connect E to D with a two-link segment of length $|EB|$. See Figure 2.3.7. These additional links constrain D to be on the correct side of the line OC.

2.4 Linkages for multiplication

In the late 1800's Kempe [K-1876] showed how to construct linkages to "solve" multivariable polynomial equations. We will make important use of a modified version of his construction. Given a set of variables $x_1,x_2,...,x_n$ with bounded domains and a polynomial equation $p(x_1,x_2,...,x_n)=0$, we can design a linkage that will force the x_i to satisfy the equation.

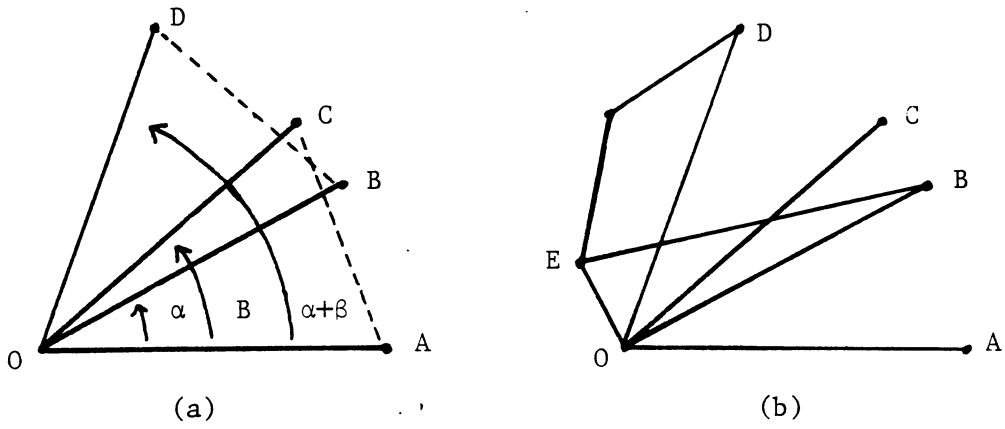


Figure 2.3.7 An angle adder. In b), the right triangle EOB has been added, together with a two-link segment connecting E to D of length $|EB|$.

Consider the links AB and BC of equal length shown in Figure 2.4.1. Joint A is fastened to the plane, and joint C moves in a slot on the x-axis. The position of C represents the value of a variable x whose domain is determined by the slot. The length of the slot is such that B cannot straighten. Additional links, whose description we omit, can be added to ensure that AB remains vertical when $x=0$. Then $x=a\cos\alpha$. Thus for a fixed value of α the variable x can be represented by the angle α.

We can now rewrite the polynomial equation expressing each x_i as $a\cos\alpha_i$. Replace products of cosines by cosines of sums of angles using the formula

$$\cos\alpha\cos\beta = \frac{1}{2}(\cos(\alpha+\beta)+\cos(\alpha-\beta))$$

thereby reducing the equation to the form

$$a_0+\sum a_i\cos\gamma_i=0,$$

where each γ_i is a sum of α_i' s.

Using the technique for adding angles described in the previous section, we can design a linkage that constructs each γ_i from the α' s. Recall that the construction for adding angles works correctly as long as the second summand is in the range $[0, \pi]$, and since B cannot straighten, this condition is satisfied by the α's. The terms $a_i\cos\gamma_i$ can be summed by constructing a sequence of links of lengths $a_0, a_1,\ldots$ connected together at their end points and making each link a_i form the angle γ_i with the horizontal by using a translator. The translator is attached to the end points of a_i, O, and a joint A_i, where OA_i is a link rigidly attached to the moving side of the angle γ_i having the same length as a_i. Finally, the free joint of the last a_i is constrained to a slot on the vertical axis. In this construction, insure that $0<\alpha_i<\pi$ by choosing the constant a to be greater than max $|x_i|$ so that B need not straighten.

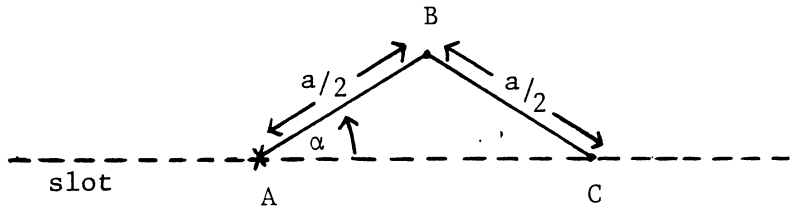
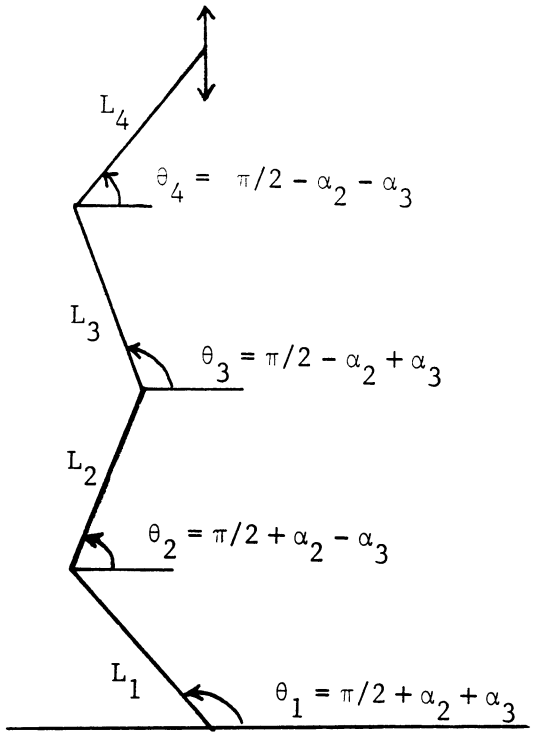
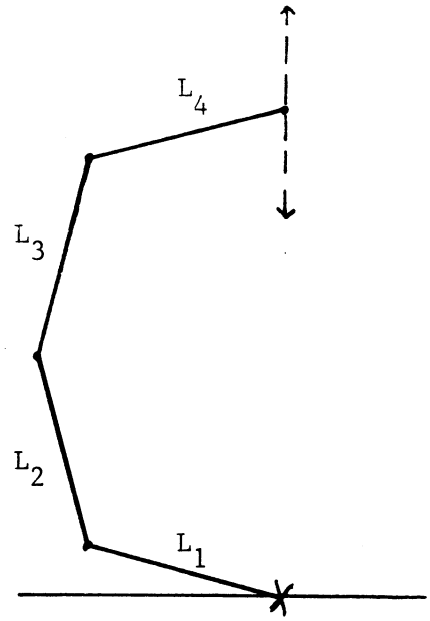


Figure 2.4.1 Representing x by angle α

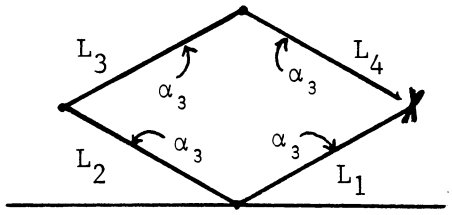
Note that for all motions of the linkages, $p(x_1, \ldots, x_n)=0$. Furthermore, for each choice of x_i's solving the equation, the linkage can move to a configuration that represents this choice. The number of links in the straightforward implementation of Kempe can be exponential in n because the summation may have exponentially many terms. However, we need the Kempe construction to enforce only two particular equations, so this problem does not concern us.



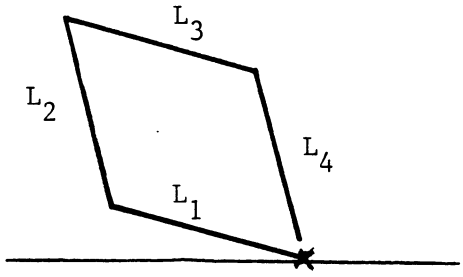
a) $x_1=0$, $\alpha_1=\pi/2$, $\alpha_2=10$, $\alpha_2=30$



b) α_2 increases
$\alpha_1=\pi/2$, $\alpha_2=45$, $\alpha_3=30$



c) $x_2=0$, α_2 reaches π
$\alpha_1=\pi/2$, $\alpha_2=\pi/2$, $\alpha_3=30$



d) α_1 decreases
$\alpha_1=45$, $\alpha_2=\pi/2$, $\alpha_3=30$

Figure 2.4.2 Links for the polynomial $x_1x_2x_3=0$

One of the equations that we are interested in is $x_1x_2x_3=0$. This equation states that one of x_1, x_2 or x_3 must be zero. Figure 2.4.2 is a simplified picture of how this equation can be mechanically solved. The equation reduces to

$$cos(\alpha_1+\alpha_2+\alpha_3)+cos(\alpha_1+\alpha_2-\alpha_3)+cos(\alpha_1-\alpha_2+\alpha_3)+cos(\alpha_1-\alpha_2-\alpha_3)=0.$$

If $x_1=0$, meaning that $\alpha_1=\pi/2$, then the links L_1 and L_4 must be oriented so that $\theta_4=\pi-\theta_1$. Similarly links L_2 and L_3 must be oriented so that $\theta_3=\pi-\theta_2$. If $x_2=0$ then L_1 and L_3 must be parallel as must L_2 and L_4. Both conditions are met when $x_1=0$ and $x_2=0$. At this point the shape of the figure changes from that in (a) and (b) to a parallelogram, shown in (c), that rotates about the origin.

3. Replacing the boundaries for a convex region

Suppose that M is a linkage and R is a closed region whose boundary is a convex polygon. We will show that by adding additional links to M we can constrain M to the region R without destroying any motions of M that were totally within R. However the new links may move outside R.

The region R is the intersection of a finite number of half-planes. By adding constraining linkages to force the original linkage M to lie in each half-plane, we can force the linkage M to lie within the intersection of the half-planes and hence within the convex polygon. For each side of the polygon and each joint J of M we construct a Peaucellier device that constrains J to a semicircular region containing the polygon and having one of the sides of the polygon on the boundary while allowing J to move along any curve in the semicircular region (see Figure 3.1). Clearly, this will constrain the linkage M to remain within R. However, we must show that we have not restricted the allowable motions of M. As pointed out earlier, identifying joint J_1 of one linkage with joint J_2 of another may restrict the movement of J_1 to a region smaller than the intersection of the original reachable regions of J_1 and J_2. In fact the intersection may not even be a connected region. The subtle point that one must consider is that even when the intersection is connected, the joints still may not be able to reach all points in the intersection since the possible paths the joints can follow may not be compatible.

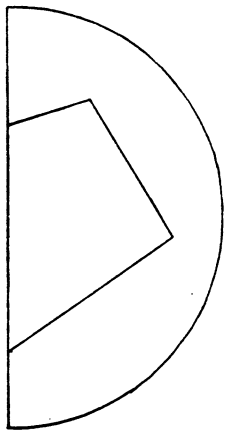


Figure 3.1 Polygon inside semicircular region

However, in this particular case, each Peaucellier device constraining J to a region bounded in part by a side of R allows J to move along any curve. Hence the allowable motions of a joint J of M are not restricted.

The number of Peaucellier devices needed is equal to the product of the number of joints of M and the number of sides of the polygon. The lengths of the links in each Peaucellier device are polynomially related to the lengths of the sides of R. Consequently, the description of the new linkage is polynomial in the size of the description of the original linkage M and region R.

4. Replacing the boundaries for a nonconvex region

4.1 Overview

In this section we show how to incorporate into a linkage L the boundaries of an arbitrary region R, not necessarily simply connected but having straight-line boundaries. Here two problems arise. First, the region is not simply the intersection of half-planes. Second, constraining the endpoints of a link to be in a region does not necessarily constrain the entire link to be in the region. In Figure 4.1.1, even if A and B are constrained to lie within the region R, the link AB may be partially outside the region.

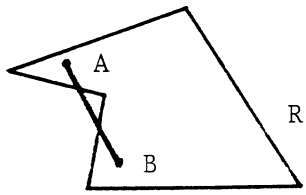


Figure 4.1.1 A link with endpoints in a nonconvex region

To handle these problems, let H be the convex hull of the region R. The region $H-R$ can be quickly partitioned into a small set of triangles, the number of triangles being polynomial in the number of line segments in the boundary. (See Eves [E-63].) If we can exclude a link from a triangle without otherwise restricting its motion, then we can restrict a link to R without restricting its motion. Applying the construction to each link of L will solve the problem.

In Section 4.2 we describe a linkage for tracing a triangle and then use this construction in Section 4.3 for constraining a link to remain outside a triangle.

4.2 A linkage for tracing a triangle

In order to construct a linkage that can reach all points in the closed exterior of a triangle, we begin by constructing a linkage that traces the boundary of a triangle. Suppose that we are given a triangle XYZ. Then we can construct three straight line motion linkages with designated joints A, B and C such that the joints A, B and C move along the segments XY, YZ and XZ respectively (Figure 4.2.1). We would like to construct a fourth linkage with a designated joint D such that D must be at the same position as either A,B or C. Then provided D can move freely subject to the above constraint, we will have constructed a linkage that traces the triangle XYZ.

We force D to be at the same position as either A, B or C by using Kempe's construction as presented in Section 2.4. Let d_1,d_2 and d_3 denote the distances from D to the joints A, B and C

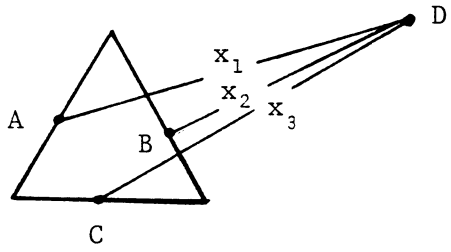


Figure 4.2.1. Forcing D to trace the boundary of a triangle

respectively. Joint D is at the same location as one of A, B or C provided $d_1d_2d_3=0$. For convenience the distances d_1,d_2 and d_3 can be translated to the x-axis by means of distance copiers. Adding the linkage to force $d_1d_2d_3=0$ then completes the construction.

4.3 Constraining a link to remain outside a triangle

We now construct a linkage to constrain the motions of a link so that it can move freely outside a triangular region. Consider the triangle XYZ shown in Figure 4.3.1. The triangle is inside a triangular figure with rounded corners. The distance between parallel edges of the inner triangle and the outer is d. The corners of the outer triangle have been replaced by circular arcs of radius d centered at the vertices of the inner triangle.

Using the construction given in Section 4.2, we can constrain a joint A to the boundary of XYZ and using a similar construction we can constrain a joint B to the boundary of the outer triangular figure. We can connect A and B with a link P of length d. The possible motions of the link P consist of rotating about the inner triangle but always remaining perpendicular to an edge, except at the vertices. At a vertex, the link P can rotate from a position perpendicular to one edge to a position perpendicular to the other.

We now add two additional links AC and AD at joint A and arms consisting of two links to connect B and D and B and C. The lengths of the arms when fully extended are designed to force the angles BAD and BAC to be in the range $[-\frac{\pi}{2}, \frac{\pi}{2}]$. Thus if A is on edge YZ of the inner triangle, AD and AC are constrained to the half plane determined by the line through Y and Z. Similar

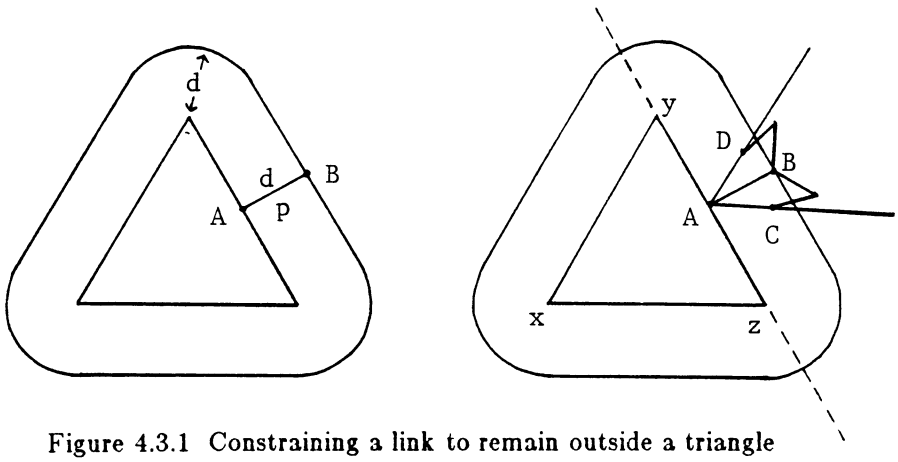


Figure 4.3.1 Constraining a link to remain outside a triangle

statements apply for the other two edges. When A is at a vertex, AC and AD are constrained to the union of two half-planes.

Attached to the links AD and AC are platforms that contain slots coinciding with AD and AC. The end points of the link ST that we wish to exclude from the triangle XYZ move in these slots. Clearly, ST can never enter the triangle since its end points are always in a half-plane whose boundary is a line through A perpendicular to AB. The triangle XYZ is outside this half-plane and thus, by convexity, ST does not intersect the triangle XYZ.

We must show that the motions of ST are not further restricted. When ST is completely contained within a half-plane associated with the edge of the inner triangle, we can fix A at any point on the triangle's edge and move T by rotating D about A and sliding T along AD. The movement of S is obtained by analogous use of AC. When ST leaves the half-plane, we move A to the appropriate vertex and rotate B about A as necessary to keep ST in the half-plane determined by the perpendicular to AB through A.

5. PSPACE-hardness of the reachability problem for linkages

We now show that the reachability problem for planar linkages is PSPACE-hard. That is, given an initial configuration of an arbitrary planar linkage L, a joint J in that linkage, and a point p in the plane, the question of whether L can be moved so that J reaches p is PSPACE-hard.

Our proof consists of showing that there are linkages that are capable of simulating Linear Bounded Automaton (LBA) computations and that the size of the description of a linkage that simulates a given LBA on inputs of length n is linear in n and the size of the description of the LBA. The PSPACE-hardness of the linkage reachability problem then follows from the fact that the acceptance problem for LBA's is PSPACE-complete. For definitions of an LBA and PSPACE see [HU-79].

5.2 Some useful linkages

We begin by building up a collection of simple devices that perform various functions. First, we define a *cell* to be a horizontal slot of some fixed size containing a joint. The joint represents the value of a Boolean variable. The left end of the slot indicates value 0, the right end value 1. Certain cells will be grouped together to form *registers*.

It is convenient to have a device called a *lock* that can be used to force the value of each cell in a register to be 0 or 1 and to prevent the value of any cell from changing during certain time periods. Figure 5.2.1 shows a lock attached to a register. The horizontal rectangular bar is part of the lock. The bar is simply a rigid structure made up of links with joints to which one can attach other links. The bar is attached to a slot so that it can move vertically. The attachment is by two joints so that no rotation is possible. Attached to the bar are a number of vertical slots. Each joint representing a Boolean variable is attached to a link, the other end of which moves in one of the vertical slots. The links are designed so that when the bar is in the lower, unlocked position the Boolean variable joint can move freely in its cell because the other end of the link can move up and down the vertical slot. When the bar is in the upper, locked position each Boolean variable joint is in a 0-1 position. Note that these variable joints cannot move when the bar is up.

In order to coordinate the linkage motions that take place during the simulation of two moves of the LBA, we design a *sequence controller* with five variables s_1, s_2, s_3, l_1 and l_2. Each variable is represented as a joint in a slot. We restrict the values that the variables can assume by adding a Kempe linkage to force

$$[s_1^2 + s_2^2 + s_3^2 + (1-l_1)^2][s_2^2 + s_3^2 + (1-l_1)^2 + (1-l_2)^2] \cdots [s_1^2 + s_2^2 + (1-l_1)^2 + l_2^2] = 0.$$

The consequence of this equation is that the only possible values the variables can assume are those shown in Table 5.2.1. The restriction on the value of the variables allows only one variable to change value at a time, and the variable that can change value is determined by the values of the

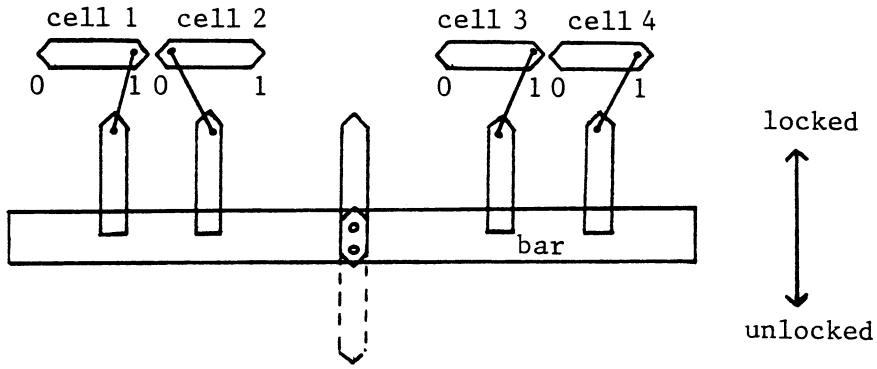


Figure 5.2.1 A lock on cells of a register (in unlocked position).

s_1	s_2	s_3	l_1	l_2
0	0	0	1	–
–	0	0	1	1
1	0	0	–	1
–	0	0	0	1
0	0	–	0	1
0	0	1	–	1
0	–	1	1	1
0	1	1	1	–
0	–	1	1	0
0	0	–	1	0

Table 5.2.1

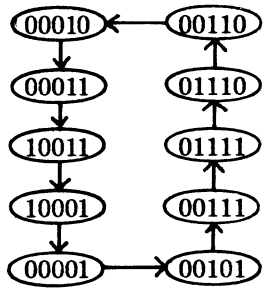


Figure 5.2.2 The allowable sequence of values

remaining variables. As a result, the only allowable sequence of changes from one set of 0-1 values to another is that shown in Figure 5.2.2. Of course, the changes can reverse at any time. We will use the values of these variables to control certain events, thereby sequencing the order in which the events can take place. In particular, the variables l_1 and l_2 will control locks, and the s's will sequence the order in which these locks are opened and closed.

Since we represent values of variables by positions of joints, we will often use the words "joint" and "variable" interchangeably. Also, we will denote a variable and the joint that represents it with the same symbol.

The next device we need is a gate for NOT and a gate for AND. To obtain negation, we use the distance copier of Section 2.3 to force the distance of a joint from one end of a slot to be the same as the distance of another joint from the opposite end of its slot. Thus when one cell has value 0, the other has value 1 and vice versa.

To construct an AND gate, we force the product of the distances of two joints from the 0-ends of their slots to equal the distance of a third joint from the 0-end of its slot. Let x_1,x_2 and x_3 be these distances. Then when x_1 and x_2 both have 0-1 values, $x_3 = x_1 AND x_2$.

Using these linkages it should be clear that we can construct a linkage to compute any Boolean function. However, to make it easy to check the correct behavior of the linkage we must be careful not to form a loop by using the output of a gate as an input to one of its predecessors. This might cause the linkage to be rigid since the loop might imply a relationship between the rates of motion of

certain joints that would not be satisfied for any nonzero rate. Our design will contain only two loops, and we will use a decoupling mechanism with them to insure that the entire linkage does not jam.

5.3 Simulation of an LBA

One idea for a mechanical simulation of a given LBA, M, is the following. Suppose that we have two registers that can be used for storing instantaneous descriptions (ID's) of M. Since Boolean variables are modeled by joints moving in slots, the contents of a register at a given time will not necessarily be a sequence of 0's and 1's. However, we will design a linkage connecting these two registers so that whenever the contents of both registers are sequences of 0's and 1's (i.e., whenever both registers contain ID's), the ID in one represents the result of a legal move of M from the ID in the other. We would also like the linkage to have the property that as M makes its moves, its ID's appear alternately in one register and then in the other. In this way, we can simulate the operation of an LBA. Since we are only interested in the reachability problem, however, we do not need to build a linkage that actually simulates M; rather, we only need a linkage which is *able* to simulate M. The linkage could make other moves as well, provided that it never moved a certain joint J to a point p representing an accepting state of M by accident. The linkage we are about to construct *can* simulate M, but in addition, it can undo and then redo sequences of moves. Because of this we will assume that M is deterministic and has no move from any accepting state.

We begin the construction with the two registers R_1 and R_2 used to store the ID's of M. Attached to the cells of the registers are two Boolean circuits constructed from NOT and AND gates. The output f_1 is true whenever the ID in register R_2 follows from the ID in register R_1 by one move of the LBA. The output f_2 is true whenever the ID in register R_1 follows from the ID in the register R_2 by one move of the LBA.

The variables l_1 and l_2 in the sequence controller described in Section 5.2 are connected to locks on registers R_1 and R_2, respectively, with l_i=1 when its lock is in the closed position. The variables s_1 and s_2 are connected to the outputs f_1 and f_2 by the linkage in Figure 5.3.1. Joints f_1 and f_2 are free to move when s_1 and s_2 are 0. However s_1 or s_2 can move to value 1 only if f_1 or

f_2, respectively, has value 1.

Initially R_1 holds the configuration of the LBA at time zero, and s_1=0, s_2=0, s_3=0, l_1=1 and l_2=0. This corresponds to the first entry in Table 5.2.2.

We now describe a sequence of events that simulate the behavior of M on a given input. Since l_2 is initially 0, the variables in R_2 can move freely. In particular, they can move to the ID of M after its first move. Then l_2 can move to a locked position, i.e., l_2 can take on the value 1. Note that as variables in R_2 were changing values, f_1 and f_2 were also changing, but this is allowed since s_1=0 and s_2=0. At this point, the sequence controller has advanced to the second state shown in Table 5.2.2 and can now advance to the third state, with s_1=1. This is because f_1 must be 1 since the ID in R_2 follows from the ID in R_1 by one move of M. Hence s_1 can move to 1.

Next R_1 unlocks allowing s_1 to return to zero. (Note that s_i can change to zero independently of f_i's value.) Now the variables in R_1 can change to the next ID of M. Again, f_1 and f_2 must be changing while R_1 is changing, but this is permitted since s_1 and s_2 have value 0. At this point, the variable s_3 can change to 1 and then l_1 can lock. The next step is for s_2 to change value to 1. This is allowed because f_2 has value 1: the configuration in R_1 follows from the configuration in R_2 by one move of M. As soon as s_2 changes value to 1, then l_2 can unlock, and s_2 can change back to 0. Finally, s_3 can change back to 0, completing a cycle of the sequence controller. During the cycle the linkage has simulated two moves of the LBA.

Observe that the simulation may proceed forward or backward. If the simulation proceeds from ID_1 to ID_2 and then reverses, it may back up into an ID other than ID_1 since two ID's may both have the same successor ID. The only concern here is that the simulation might accidentally back into an accepting ID. This can be prevented by modifying the LBA so that no move is possible

Figure 5.3.1 Decoupling mechanism

from any ID with an accepting state and then basing the design of the linkage on the modified LBA. Note that the linkage may back into a configuration corresponding to an ID of M that could not be reached from its initial state. However, since we are only considering deterministic LBA's, the linkage must move forward along the same computational path it has just backed up. Of course its forward progress may be interrupted from time to time by additional backing up and retracing of sequences.

Finally, another Boolean circuit is attached to the two registers R_1 and R_2. The Boolean circuit computes a 1 output whenever one of the registers is locked and contains an accepting ID. The output of this circuit is a joint J. There is a motion of the linkage that moves J to 1 if and only if the *LBA* reaches an accepting ID.

This completes the proof that the reachability problem for planar linkages is PSPACE-hard.

References

[E-63] Eves, Howard. *A survey of geometry.* Allyn and Bacon, Boston, Massachusetts, 1963.

[HCV-52] Hilbert, D. and F. Cohn-Vossen. *Geometry and the Imagination.* Chelsea Publishing Company, New York, New York, 1952.

[HJW-82a] Hopcroft, John, Deborah Joseph and Sue Whitesides. On the Movement of Robot Arms in 2-Dimensional Bounded Regions. Computer Science Department, Cornell University, TR 82-486, April 1982.

[HJW-82b] Hopcroft, John, Deborah Joseph and Sue Whitesides. Determining the Points of a Circular Region Reachable by Joints of a Robot Arm. Computer Science Department, Cornell University, TR 82-416, 1982.

[HU-79] Hopcroft, John and Jeffrey D. Ullman. *Introduction to automata theory, languages, and computation.* Addison-Wesley Publishing Company, Reading, Massachusetts, 1979.

[K-1876] Kempe, A. B. *On a general method of describing plane curves of the nth degree by linkwork.* Proc. London Math. Soc. 1876, Vol.7, pp. 213-216.

[L-80] Lozano-Perez, Tomas. Automatic Planning of Manipulation Transfer Movements. M.I.T. Artificial Intelligence Laboratory, A.I. Memo 606, December 1980.

[LW-79] Lozano-Perez, Tomas, and Michael A. Wesley. *An algorithm for planning collision-free paths among polyhedral obstacles.* Communications of the ACM 1979, Vol. 22, No. 10, pp. 560-570.

[S-81] Schwartz, Jacob T., and Micha Sharir. On the 'Piano Movers' Problem I. The Case of a Two-dimensional Rigid Polygonal Body Moving Amidst Polygonal Barriers. Department of Computer Science, New York University, TR 39, October 1981.

[S-82] Schwartz, Jacob T., and Micha Sharir. On the 'Piano Movers' Problem II. General Techniques for Computing Topological Properties of Real Algebraic Manifolds. Department of Computer Science, New York University, TR 41, February 1982.

[P-1864] Peaucellier, M., Correspondence. *Nouvelles Annales de Mathematiques* 1864, Series 2, Vol. 3, pp. 414-415.

[R-79] Reif, J. Complexity of the Mover's Problem and Generalizations. *Proceedings 20th IEEE Foundations of Computer Science Conf.*, 1979, pp.421-427.