Combinatorial and Computational Geometry
MSRI Publications
Volume **52**, 2005

The Carpenter's Ruler Folding Problem

GRUIA CĂLINESCU AND ADRIAN DUMITRESCU

Abstract. A carpenter's ruler is a ruler divided into pieces of different lengths which are hinged where the pieces meet, which makes it possible to fold the ruler. The carpenter's ruler folding problem, originally posed by Hopcroft, Joseph and Whitesides, is to determine the smallest case (or interval on the line) into which the ruler fits when folded. The problem is known to be NP-complete. The best previous approximation ratio achieved, dating from 1985, is 2. We improve this result and provide a fully polynomial-time approximation scheme for this problem. In contrast, in the plane, there exists a simple linear-time algorithm which computes an exact (optimal) folding of the ruler in some convex case of minimum diameter. This brings up the interesting problem of finding the minimum area of a convex *universal case* (of unit diameter) for all rulers whose maximum link length is one.

1. Introduction

The carpenter's ruler folding problem is: Given a sequence of rigid rods (links) of various integral lengths connected end-to-end by hinges, to fold it so that its overall folded length is minimum. It was first posed in [Hopcroft et al. 1985], where the authors proved that the problem is NP-complete using a reduction from the NP-complete problem PARTITION (see [Garey and Johnson 1979; Cormen et al. 1990]). A simple linear-time factor 2 approximation algorithm, as well as a pseudo-polynomial $O(L^2n)$ time dynamic programming algorithm, where L is the maximum link length, where presented in [Hopcroft et al. 1985] (see also [Kozen 1992]). A physical ruler is idealized in the problem, so that the ruler is allowed to fold onto itself and lie along a line segment whose length is the size of the case, and thus no thickness results from the segments which lie on top of each other.

The decision problem can be stated as follows. Given a ruler whose links have lengths $l_1, l_2, \ldots, l_n$, can it be folded so that its overall folded length is at most k? Note that different orderings of the links can result in different minimum case

Keywords: approximation scheme, carpenter's ruler, folding problems, universal case.

lengths. For example if the ruler has links of lengths 6, 6 and 3 in this order, the ruler can be folded into a case of length 6, but if the links occur in the order 6, 3 and 6, the optimal case-length is 9.

Our first result (Section 2) improves the 19-year old factor 2 approximation:

THEOREM 1. *There exists a fully polynomial-time approximation scheme for the carpenter's ruler folding problem.*

A fully polynomial-time approximation scheme (FPTAS) for a minimization problem is a family of algorithms A_ε, for all $\varepsilon > 0$, such that A_ε has running time polynomial in the size of the instance and $1/\varepsilon$, and the output of A_ε is at most $(1+\varepsilon)$ times the optimum [Garey and Johnson 1979].

In Section 3, we study a natural, related question: the condition that the folding must lie on a line is relaxed, by considering foldings in the plane with the objective of minimizing the diameter of a convex case containing the folded ruler. Here foldings allow for a free reconfiguration of the joint angles, with the proviso that each link of the ruler maintains its length (the shape of the case is unconstrained). In contrast with the problem on the line, this variant admits an easy exact (optimal) solution which can be computed in linear time, using exact arithmetic.

This brings up the interesting problem of finding the minimum area of a convex case (of unit diameter) for all rulers whose maximum link length is one. A closed curve of unit diameter in the plane is said to be a *universal case* for all rulers whose maximum link length is one if each such ruler admits a planar folding inside the curve. Our results are summarized in:

THEOREM 2. *There exists an $O(n)$ algorithm for the carpenter's ruler folding problem in the plane with lengths $l_1, l_2, \ldots, l_n$, which computes a folding in a convex case of minimum diameter $L = \max(l_1, \ldots, l_n)$. The minimum area A of a convex universal case (of unit diameter) for all rulers whose maximum link length is one satisfies*

$$\frac{3}{8} \le A \le \frac{\pi}{3} - \frac{\sqrt{3}}{4}.$$

The lower bound is $\frac{3}{8} = 0.375$ and the upper bound is ≈ 0.614. We believe the latter is closer to the truth.

Other folding problems with links allowed to cross have been studied, for example in [Hopcroft et al. 1984; Kantabutra 1992; Kantabutra 1997; Kantabutra and Kosaraju 1986; van Kreveld et al. 1996], while linkage folding problems for noncrossing links have been investigated for example in [Connelly et al. 2003; Streinu 2000]. For other *universal cover* problems, such as the *worm problem*, see [Croft et al. 1991; Klee and Wagon 1991] and the references therein.

2. Proof of Theorem 1

We present two approximation schemes: one based on trimming the solution space and one based on rounding and scaling. We start with notation and observations which apply to both algorithms.

A folding F of the ruler can be specified by the position on the line of the first (free) endpoint of the ruler (i.e., the free endpoint of the first link) and a binary string of length n in which the i-th bit is -1 or 1 depending on whether the i-th segment is folded to the left or right of its fixed endpoint (view this as a sequential process). We call this binary string the *folding vector*.

For a given folding F, let the interval $I_F = [a_F, b_F]$ be the smallest closed interval which contains it (i.e., it contains all the segments of the ruler). We refer to it as the *folding interval*. See also Figure 1.

Figure 1. A carpenter's ruler with segments of length 1, 3, 2 and 4 folded so that it fits into a case of length 5 (left). Its folding vector is $(-1, 1, 1-1)$. Another folding into a case of same length (right). Its folding vector is $(1, -1, 1-1)$.

Denote by OPT the minimum folded length for a ruler whose lengths are $l_1, l_2, \ldots, l_n$. A trivial lower bound — on which the 2-approximation algorithm is based — is OPT $\geq L$, where $L = \max(l_1, l_2, \ldots, l_n)$ is the maximum rod length. We further exploit this observation and the 2-approximation algorithm given in [Hopcroft et al. 1985].

Observation 1. *An optimal solution can be computed by fixing the first segment at* $[0, l_1]$ (*with the free endpoint of the first link at* 0), *and then computing all foldings that extend it, whose intervals have length at most* $2L$ (*thus are included in the interval* $[-2L + l_1, 2L]$).

Proof. Consider an optimal solution. Clearly the first segment can be fixed at any given position of its free endpoint and at any of the two possible orientations. Since there exist approximate solutions whose folding intervals have length at most $2L$, foldings with larger intervals do not need to be considered (are not optimal). □

One can also see that the observation can be somewhat strengthened, since in fact, any of the links can be fixed at a given position and orientation.

Observation 2. *An optimal solution can be computed by first fixing one segment of length* L (*if more exist, select one arbitrarily*) *at* $[0, L]$ *and then computing all foldings that extend it, whose intervals have length at most* $2L$ (*thus are included in the interval* $[-L, 2L]$).

Consider a folding F, whose vector is $(\varepsilon_1, \ldots, \varepsilon_n)$, for a given ruler $l_1, l_2, \ldots, l_n$. For $i = 1, \ldots, n$, let the *partial folding* F_i of the ruler $l_1, l_2, \ldots, l_i$ be that whose folding vector is $(\varepsilon_1, \ldots, \varepsilon_i)$.

For a folding F whose interval is $[a, b]$, clearly the endpoint x of the last segment also lies in the same interval, i.e., $x \in [a, b]$. We say that F has *parameters* a, b and x, or that F is given by a, b and x.

A FPTAS based on trimming the solution space. We now describe the first algorithm which we note, has some similarity features with the fully polynomial-time approximation scheme for the subset-sum problem [Ibarra and Kim 1975] (see also [Cormen et al. 1990] for a more accessible presentation). Let ε be the approximation parameter, where $0 < \varepsilon < 1$. For simplicity assume that $m = 8n/\varepsilon$ is an integer. Set $\delta = L\varepsilon/(2n)$. Consider the partition of the interval $[-2L, 2L]$ into m *elementary intervals* of length δ, given by $[-2L + j\delta, -2L + (j+1)\delta)$, for $j = 0, \ldots, m-1$, except that the last interval in this sequence, for $j = m-1$, is closed at both ends. For simplicity of exposition, we consider the interval $[-2L, 2L]$ instead of the interval $[-2L+l_1, 2L]$ mentioned in Observation 1 (and then the expression of m above is an overestimate). An *interval triplet* denoted (I_a, I_b, I_x), is any of the m^3 ordered triples of elementary intervals.

The algorithm iteratively computes a set of partial foldings $\mathcal{F}_i$ of the ruler $l_1, l_2, \ldots, l_i$, for $i = 1, \ldots, n$, so that at most one partial folding per interval triplet is maintained at the end of the i-th iteration. A partial folding whose folding interval is $[a, b]$, and the endpoint of the last segment at x is associated with the interval triplet (I_a, I_b, I_x), where $a \in I_a$, $b \in I_b$ and $x \in I_x$. If at step i more partial foldings per interval triplet are computed, all but one of them are discarded; the one selected for the next step is chosen arbitrarily from those computed.

$\mathcal{F}_1$ consists of one (partial) folding, given by $a'_1 = 0$, $b'_1 = l_1$, $x'_1 = l_1$. Let $i \geq 2$. In the i-th iteration, the algorithm computes from the set $\mathcal{F}_{i-1}$ of partial foldings of the first $i-1$ links, all the partial foldings of the first i links that extend foldings in $\mathcal{F}_{i-1}$, and whose intervals are included in the interval $[-2L, 2L]$ (there are at most $2|\mathcal{F}_{i-1}|$ of these). It then "trims" this set to obtain $\mathcal{F}_i$, so that if an interval triplet has more partial foldings associated with it, exactly one is maintained for the next iteration. Clearly, $|\mathcal{F}_i| \leq m^3$ at the end of the i-th iteration, for any $i = 1, \ldots, n$. Note that this bound holds during the execution of each iteration as well. After the last iteration n, the algorithm outputs a folding of the ruler (one in $\mathcal{F}_n$) whose interval has minimum length.

Let now F be an optimal folding as specified in Observation 1, whose vector is $(\varepsilon_1, \ldots, \varepsilon_n)$. We have $\varepsilon_1 = 1$. For $i = 1, \ldots, n$, let the partial folding F_i have the (folding) interval $[a_i, b_i]$ and the endpoint of the last segment at $x_i \in [a_i, b_i]$. We have $a_1 = 0$, $b_1 = l_1$ and $x_1 = l_1$, and also $x_i = \sum_{j=1}^{i} \varepsilon_j l_j$, for $i = 1, \ldots, n$.

LEMMA 1. *For $i = 1, \ldots, n$, the algorithm computes a partial folding $F'_i \in \mathcal{F}_i$ of the ruler $l_1, l_2, \ldots, l_i$, whose interval is $[a'_i, b'_i]$ and the endpoint of the last segment is at x'_i, so that*

$$\begin{aligned} &\text{(A)} && |a_i - a'_i| \le i\delta, \\ &\text{(B)} && |b_i - b'_i| \le i\delta, \\ &\text{(X)} && |x_i - x'_i| \le i\delta. \end{aligned}$$

PROOF. We proceed by induction. The basis $i = 1$ is clear. Let $i \ge 2$, and assume that a partial folding F'_{i-1} of the ruler $l_1, l_2, \ldots, l_{i-1}$, is computed by the algorithm after $i - 1$ iterations, as specified. We thus have

$$\begin{aligned} |a_{i-1} - a'_{i-1}| &\le (i-1)\delta, \\ |b_{i-1} - b'_{i-1}| &\le (i-1)\delta, \\ |x_{i-1} - x'_{i-1}| &\le (i-1)\delta. \end{aligned}$$

The partial folding F_i (corresponding to F) has parameters

$$\begin{aligned} a_i &= \min(a_{i-1}, x_{i-1} + \varepsilon_i l_i), \\ b_i &= \max(b_{i-1}, x_{i-1} + \varepsilon_i l_i), \\ x_i &= x_{i-1} + \varepsilon_i l_i. \end{aligned}$$

Consider the partial folding F''_i obtained from F'_{i-1} (i.e., which extends F'_{i-1}) so that its i-th bit in the folding vector is ε_i (the same as in F_i). Note that the algorithm computes F''_i in the first part of iteration i (before trimming). Its parameters are

$$\begin{aligned} a''_i &= \min(a'_{i-1}, x'_{i-1} + \varepsilon_i l_i), \\ b''_i &= \max(b'_{i-1}, x'_{i-1} + \varepsilon_i l_i), \\ x''_i &= x'_{i-1} + \varepsilon_i l_i. \end{aligned}$$

Let the interval triplet which contains F''_i be (I_a, I_b, I_x). The algorithm discards all but one partial folding in this interval triplet, say F'_i, with parameters a'_i, b'_i, x'_i. This implies that

$$\begin{aligned} |a'_i - a''_i| &\le \delta, \\ |b'_i - b''_i| &\le \delta, \\ |x'_i - x''_i| &\le \delta. \end{aligned}$$

The lemma follows once we show that

$$\begin{aligned} &\text{(A}'\text{)} && |a_i - a''_i| \le (i-1)\delta, \\ &\text{(B}'\text{)} && |b_i - b''_i| \le (i-1)\delta, \\ &\text{(X}'\text{)} && |x_i - x''_i| \le (i-1)\delta, \end{aligned}$$

since then, the partial folding F'_i which is computed by the algorithm, satisfies the imposed conditions after step i, e.g. for (A),

$$|a_i - a'_i| \leq |a_i - a''_i| + |a''_i - a'_i| \leq (i-1)\delta + \delta = i\delta.$$

(B) and (X) follow in a similar way.

We will show that (A′) holds by examining four cases, depending on how the minimums for a_i and for a''_i are achieved. The proof of (B') is very similar (with max taking the place of min) and will be omitted.

To prove (A′), recall that

$$|a''_i - a_i| = |\min(a'_{i-1}, x'_{i-1} + \varepsilon_i l_i) - \min(a_{i-1}, x_{i-1} + \varepsilon_i l_i)|.$$

Put $\Delta = |a''_i - a_i|$. We distinguish four cases.

Case 1: $\min(a'_{i-1}, x'_{i-1} + \varepsilon_i l_i) = a'_{i-1}$ and $\min(a_{i-1}, x_{i-1} + \varepsilon_i l_i) = a_{i-1}$. Then using the induction hypothesis,

$$\Delta = |a'_{i-1} - a_{i-1}| \leq (i-1)\delta.$$

Case 2: $\min(a'_{i-1}, x'_{i-1} + \varepsilon_i l_i) = x'_{i-1} + \varepsilon_i l_i$ and $\min(a_{i-1}, x_{i-1} + \varepsilon_i l_i) = x_{i-1} + \varepsilon_i l_i$. Similarly, the induction hypothesis yields

$$\Delta = |x'_{i-1} - x_{i-1}| \leq (i-1)\delta.$$

Case 3: $\min(a'_{i-1}, x'_{i-1} + \varepsilon_i l_i) = a'_{i-1}$ and $\min(a_{i-1}, x_{i-1} + \varepsilon_i l_i) = x_{i-1} + \varepsilon_i l_i$. Note that in this case $\varepsilon_i = -1$. We have two subcases.

Case 3.1: $x_{i-1} - l_i \leq a'_{i-1}$. Recall that $a'_{i-1} \leq x'_{i-1} - l_i$. We have

$$x_{i-1} - l_i \leq a'_{i-1} \leq x'_{i-1} - l_i.$$

Then

$$\Delta = |a'_{i-1} - (x_{i-1} - l_i)| \leq |x'_{i-1} - l_i - (x_{i-1} - l_i)| \leq (i-1)\delta,$$

where the last in the chain of inequalities above is implied by the induction hypothesis.

Case 3.2: $a'_{i-1} \leq x_{i-1} - l_i$. Recall that $x_{i-1} - l_i \leq a_{i-1}$. We have

$$a'_{i-1} \leq x_{i-1} - l_i \leq a_{i-1}.$$

Then

$$\Delta = |a'_{i-1} - (x_{i-1} - l_i)| \leq |a'_{i-1} - a_{i-1}| \leq (i-1)\delta,$$

again by the induction hypothesis.

Case 4: $\min(a'_{i-1}, x'_{i-1} + \varepsilon_i l_i) = x'_{i-1} + \varepsilon_i l_i$ and $\min(a_{i-1}, x_{i-1} + \varepsilon_i l_i) = a_{i-1}$. Note that in this case $\varepsilon_i = -1$. Thus $x'_{i-1} - l_i \leq a'_{i-1}$ and $a_{i-1} \leq x_{i-1} - l_i$. We have two subcases.

Case 4.1: $x'_{i-1} - l_i \le a_{i-1}$. Then

$$\Delta = |a_{i-1} - (x'_{i-1} - l_i)| \le |x_{i-1} - l_i - (x'_{i-1} - l_i)| \le (i-1)\delta.$$

Case 4.2: $a_{i-1} \le x'_{i-1} - l_i$. Then

$$\Delta = |(x'_{i-1} - l_i) - a_{i-1}| \le |a'_{i-1} - a_{i-1}| \le (i-1)\delta.$$

This concludes the proof of (A′).

We also clearly have

$$|x_i - x''_i| = |(x_{i-1} + \varepsilon_i l_i) - (x'_{i-1} + \varepsilon_i l_i)| = |x_{i-1} - x'_{i-1}| \le (i-1)\delta,$$

which proves (X') and concludes the proof of the lemma. □

Lemma 1 for $i = n$ implies that the algorithm computes a folding F' of the ruler whose interval is $[a', b']$, so that if F is an optimal folding whose interval is $[a, b]$,

$$|a - a'| \le n\delta = L\varepsilon/2,$$
$$|b - b'| \le n\delta = L\varepsilon/2.$$

Since the algorithm selects in the end a folding whose interval length is minimum, it outputs one whose interval length is not more than

$$|b' - a'| \le |b - a| + \varepsilon L \le (1 + \varepsilon)\mathrm{OPT}.$$

The last in the chain of inequalities above follows from the lower bound $b - a = \mathrm{OPT} \ge L$.

It takes $O(\log L)$ time to compute the three parameters for each partial folding, and $O(\log L)$ space to store this information. Since there are n iterations, and each takes $O(m^3 \log L)$ time, the total running time is $O(nm^3 \log L)) = O(n^4(1/\varepsilon)^3 \log L)$. As each (partial) folding can be stored in $O(n \log L)$ space, the total space is also $O(n^4(1/\varepsilon)^3 \log L)$.

REMARK 1. Using Observation 2, one can modify the algorithm so that $m = 6n/\varepsilon$ (versus $m = 8n/\varepsilon$), which leads to maintaining a somewhat smaller number of interval triplets.

A FPTAS based on rounding and scaling. We apply the rounding and scaling technique, inspired by the method used to obtain an approximation scheme for Knapsack (from [Ibarra and Kim 1975]; see also [Garey and Johnson 1979, pages 135–137]). The algorithm is:

(i) Set

$$\bar{l}_i = \left\lfloor \frac{l_i}{L} 4n \frac{1}{\varepsilon} \right\rfloor.$$

Call the new instance of the carpenter's ruler folding problem with lengths $\bar{l}_i$ the *reduced* instance.

(ii) Use the pseudo-polynomial algorithm in [Hopcroft et al. 1985] to solve exactly the reduced instance. Output the same folding vector.

Note that the maximum length of the reduced instance is $\bar{L} = \lfloor 4n\frac{1}{\varepsilon} \rfloor$ and therefore the running time of the algorithm is

$$O(n \log L + \bar{L}^2 n) = O(n \log L + n^3 (1/\varepsilon)^2).$$

We refine the notation as follows: given folding F whose vector is $(\varepsilon_1^F, \ldots, \varepsilon_n^F)$, set $x_0^F = 0$ and for $i = 1, \ldots, n$, set $x_i^F = \sum_{j=1}^{i} \varepsilon_j^F l_j$. As before $a_F = \min_{i=0}^{n} x_i^F$ and $b_F = \max_{i=0}^{n} x_i^F$, and note that the length of F is $b_F - a_F$. Define $\bar{x}_i^F$, $\bar{a}_F$ and $\bar{b}_F$ in the same way using the length function $\bar{l}$ instead of l. Let

$$q_i := \frac{l_i}{L} 4n \frac{1}{\varepsilon} - \left\lfloor \frac{l_i}{L} 4n \frac{1}{\varepsilon} \right\rfloor.$$

Note that $0 \le q_i < 1$ and $l_i = (\bar{l}_i + q_i) L\varepsilon/(4n)$.

Let A be any folding for the original instance and B be an optimum folding for the reduced instance. We have:

$$x_i^B = \sum_{j=1}^{i} \varepsilon_i^B l_i = \sum_{j=1}^{i} \varepsilon_i^B (\bar{l}_i + q_i) \frac{L\varepsilon}{4n} = \frac{L\varepsilon}{4n} \left(\bar{x}_i^B + \sum_{j=1}^{i} \varepsilon_i^B q_i \right).$$

Using $0 \le q_i < 1$, we obtain

$$x_i^B \le \frac{L\varepsilon}{4n} (\bar{x}_i^B + n) \le \frac{L\varepsilon}{4n} \bar{x}_i^B + \frac{L\varepsilon}{4},$$

and therefore

$$b_B \le \frac{L\varepsilon}{4n} \bar{b}_B + \frac{L\varepsilon}{4}.$$

Similarly we have:

$$x_i^B \ge \frac{L\varepsilon}{4n} \bar{x}_i^B - \frac{L\varepsilon}{4},$$

and consequently

$$a_B \ge \frac{L\varepsilon}{4n} \bar{a}_B - \frac{L\varepsilon}{4}.$$

Using the fact that B has optimum length for $\bar{l}$, and the inequality $b_A - a_A \ge L$, we get:

$$b_B - a_B \le \frac{L\varepsilon}{4n} (\bar{b}_B - \bar{a}_B) + \frac{L\varepsilon}{2} \le \frac{L\varepsilon}{4n} (\bar{b}_A - \bar{a}_A) + \frac{\varepsilon}{2} (b_A - a_A). \tag{2–1}$$

Further:

$$\bar{x}_i^A = \sum_{j=1}^{i} \varepsilon_i^A \bar{l}_i = \sum_{j=1}^{i} \varepsilon_i^A \left(\frac{l_i}{L} 4n \frac{1}{\varepsilon} - q_i \right) = \frac{4n}{L\varepsilon} x_i^A - \sum_{j=1}^{i} \varepsilon_i^A q_i \le \frac{4n}{L\varepsilon} x_i^A + n,$$

and therefore

$$\bar{b}_A \le \frac{4n}{L\varepsilon} b_A + n. \tag{2–2}$$

Similarly we have:

$$\bar{x}_i^A = \frac{4n}{L\varepsilon} x_i^A - \sum_{j=1}^{i} \varepsilon_i^A q_i \ge \frac{4n}{L\varepsilon} x_i^A - n,$$

and consequently

$$\bar{a}_A \geq \frac{4n}{L\varepsilon} a_A - n. \tag{2–3}$$

Plugging Equations (2–2) and (2–3) into (2–1) and using again the inequality $b_A - a_A \geq L$, we obtain

$$b_B - a_B \leq \frac{L\varepsilon}{4n}\left(\frac{4n}{L\varepsilon} b_A + n - \left(\frac{4n}{L\varepsilon} a_A - n\right)\right) + \frac{\varepsilon}{2}(b_A - a_A) \leq (b_A - a_A)(1+\varepsilon).$$

If we now let A be an optimal folding for the original instance, we find that $b_B - a_B \leq (1+\varepsilon)\mathrm{OPT}$; this completes the second proof of Theorem 1.

3. Folding in the Plane: Proof of Theorem 2

For the purposes of this section, a folding of the ruler is a polygonal chain of n segments (links), numbered from 1 to n, lying in the plane. Let q_0 be the free endpoint of the first link, and q_1 be its other endpoint. Call $v_1 = \overline{q_0 q_1}$ the vector of link 1. Inductively define ($q_2, \ldots, q_n$ and) $v_2, \ldots, v_n$, the vectors of links $2, \ldots, n$. The *joint angle* between links i and $i+1$ is the angle $\in [0, \pi]$ between v_i and v_{i+1}. The angle is counterclockwise if it describes a left turn, and clockwise if it describes a right turn. Angles of 0 and π are considered both left and right turns.

It is obvious that the diameter of any convex case in which the ruler is folded is at least L, where L is the maximum link length. The following simple linear-time algorithm computes a folding of the ruler, so that all joint angles in $(0, \pi]$ are clockwise (or counterclockwise). The algorithm is certainly implicit in [Hopcroft et al. 1985], where an extensive analysis of reconfiguration problems for rulers confined in discs is made.

Fix arbitrarily a disk D of diameter L, whose boundary is the circle C. Fix the first free endpoint of the ruler (i.e., the free endpoint of the first link) at some point p_0 of C. For $i = 1, \ldots, n$, iteratively fix the next point of the ruler (i.e., the next endpoint of its i-th link) at one of the at most two intersection points of C with the circle with center at p_{i-1} and radius l_i. One can also select the appropriate intersection point at each step, so that all joint angles in $(0, \pi]$ are clockwise (or counterclockwise). An illustration appears in Figure 2. Consider now the closed convex curve R, of unit diameter, obtained from a *Reuleaux triangle*, by replacing one of the circular arcs with a straight segment, as in Figure 3. (A Reuleaux triangle can be obtained from an equilateral triangle ABC by joining each pair of its vertices by a circular arc whose center is at the third vertex; see [Yaglom and Boltyanskiĭ 1961].) The above algorithm can be modified to compute a folding of a ruler with maximum link length 1 inside R: Fix the first free endpoint of the ruler at some point p_0 of the circular arc AB. Iteratively fix the next point of the ruler at some intersection point (it exists!) with the open curve BAC. The area of R is $\frac{1}{3}\pi - \frac{1}{4}\sqrt{3} \approx 0.614$, as claimed.

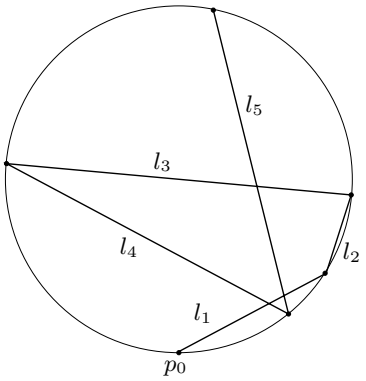

Figure 2. A carpenter's ruler with five links folded so that it fits in a circular case of diameter L, where $L = l_3$ is the maximum link-length. All joint angles in this folding are counterclockwise (i.e., left turns).

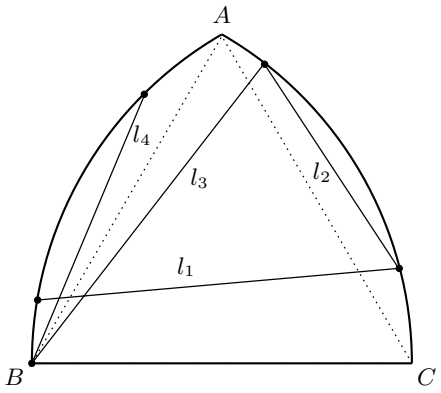

Figure 3. The closed curve R obtained from a Reuleaux triangle, and a ruler with four links folded inside; the length of l_3 is 1.

It remains to prove the lower bound in Theorem 2. Consider a 3-link ruler $ABCD$ with lengths $AB = 1$, $BC = x < 1$ and $CD = 1$, where the choice of the length $x = \frac{1}{2}(\sqrt{7} - 1) \approx 0.8229$ of the middle link is explained below. We will show that the area of any convex case for it is at least $\frac{3}{8}$. In any folding in which the unit length links do not intersect, the diameter of the case exceeds one. Assume therefore that they intersect (see Figure 4). The area of $BCAD$ (i.e., the convex hull of the four endpoints of the links) is

$$\frac{ab\sin\alpha}{2} + \frac{(1-a)(1-b)\sin\alpha}{2} + \frac{a(1-b)\sin\alpha}{2} + \frac{(1-a)b\sin\alpha}{2} = \frac{\sin\alpha}{2},$$

where $\alpha = \widehat{BOD}$.

For a given x, the area is minimized when either $A = D$ so that the folding forms an isosceles triangle (small x), or when AD is parallel to BC and $AD = 1$

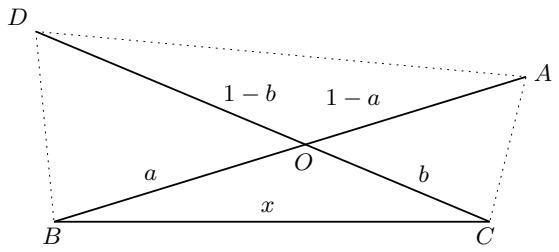

Figure 4. A ruler with link-lengths 1, x and 1.

(large x). The area of the isosceles triangle is

$$\sqrt{\left(1+\frac{x}{2}\right)\left(\frac{x}{2}\right)\left(\frac{x}{2}\right)\left(1-\frac{x}{2}\right)}.$$

The area of the trapezoid $BCAD$ is

$$\frac{1+x}{2}\sqrt{1-\left(\frac{1+x}{2}\right)^2}.$$

Now choose x to balance the two areas. A routine calculation gives

$$x=\frac{\sqrt{7}-1}{2},$$

and the corresponding area is 3/8. This completes the proof of Theorem 2.

We conclude with these questions: Is the curve R a convex universal case of minimum area? If not, what is the minimum area of such a universal case? Does convexity of the case make any difference?

Acknowledgement

We thank the anonymous referee (of an earlier version) who suggested the modification of the Reuleaux triangle in Figure 3.

References

[Connelly et al. 2003] R. Connelly, E. D. Demaine, and G. Rote, "Straightening polygonal arcs and convexifying polygonal cycles", *Discrete Comput. Geom.* **30**:2 (2003), 205–239.

[Cormen et al. 1990] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to algorithms*, MIT Press, Cambridge, MA, 1990.

[Croft et al. 1991] H. T. Croft, K. J. Falconer, and R. K. Guy, *Unsolved problems in geometry*, Problem Books in Mathematics, Springer, New York, 1991.

[Garey and Johnson 1979] M. R. Garey and D. S. a. Johnson, *Computers and intractability*, W. H. Freeman and Co., San Francisco, 1979.

[Hopcroft et al. 1984] J. Hopcroft, D. Joseph, and S. Whitesides, "Movement problems for 2-dimensional linkages", *SIAM J. Comput.* **13**:3 (1984), 610–629.

[Hopcroft et al. 1985] J. Hopcroft, D. Joseph, and S. Whitesides, "On the movement of robot arms in 2-dimensional bounded regions", *SIAM J. Comput.* **14**:2 (1985), 315–333.

[Ibarra and Kim 1975] O. H. Ibarra and C. E. Kim, "Fast approximation algorithms for the knapsack and sum of subset problems", *J. Assoc. Comput. Mach.* **22**:4 (1975), 463–468.

[Kantabutra 1992] V. Kantabutra, "Motions of a short-linked robot arm in a square", *Discrete Comput. Geom.* **7**:1 (1992), 69–76.

[Kantabutra 1997] V. Kantabutra, "Reaching a point with an unanchored robot arm in a square", *Internat. J. Comput. Geom. Appl.* **7**:6 (1997), 539–549.

[Kantabutra and Kosaraju 1986] V. Kantabutra and S. R. Kosaraju, "New algorithms for multilink robot arms", *J. Comput. System Sci.* **32**:1 (1986), 136–153.

[Klee and Wagon 1991] V. Klee and S. Wagon, *Old and new unsolved problems in plane geometry and number theory*, vol. 11, The Dolciani Mathematical Expositions, Math. Assoc. America, Washington (DC), 1991.

[Kozen 1992] D. C. Kozen, *The design and analysis of algorithms*, Texts and Monographs in Computer Science, Springer, New York, 1992.

[van Kreveld et al. 1996] M. van Kreveld, J. Snoeyink, and S. Whitesides, "Folding rulers inside triangles", *Discrete Comput. Geom.* **15**:3 (1996), 265–285.

[Streinu 2000] I. Streinu, "A combinatorial approach to planar non-colliding robot arm motion planning", pp. 443–453 in *41st Annual Symposium on Foundations of Computer Science* (Redondo Beach, CA, 2000), IEEE Comput. Soc. Press, Los Alamitos, CA, 2000.

[Yaglom and Boltyanskiĭ 1961] I. M. Yaglom and V. G. Boltyanskiĭ, *Convex figures*, vol. 4, Library of the mathematical circle, Holt, Rinehart and Winston, New York, 1961.

Gruia Călinescu
Department of Computer Science
Illinois Institute of Technology
Chicago, IL 60616
United States
calinesc@iit.edu

Adrian Dumitrescu
Computer Science
University of Wisconsin–Milwaukee
3200 N. Cramer Street
Milwaukee, WI 53211
United States
ad@cs.uwm.edu