Cutting Rectangles Into Equal Area Pieces

Prosenjit Bose* Jurek Czyzowicz† Dominic Lessard*

1 Introduction

Problem 1 *Let P be an axis parallel rectangle. We want to cut P in k equal area pieces such that the total length of the cuts is minimum.*

The problem depends on how the cuts are made. Although cuts may take a general form, in this paper we restrict our attention to straight line orthogonal cuts. We assume that each cut is complete in that it divides a rectangle into two pieces.

Several variants of this problem have appeared in the literature. Overmars and Welzl [4] studied the problem of cutting a polygon drawn on a piece of paper in the cheapest possible way. Croft, Falconer and Guy [2] studied problems related to tiling and dissection of circles and squares. Frederickson [3] discusses several interesting problems related to geometric dissection. Bose et al.[1] proved many results about cutting squares and circles in equal area pieces. In section 2, we propose solutions to problem 1. In section 3 we present some results from computer simulations and discuss their consequences.

Since the square is the minimum perimeter rectangle enclosing a fixed area, there is a lower bound on the solution of problem 1.

Proposition 1 *[1] The total length of the straight line cuts is greater or equal to $2\sqrt{k}-2$ for the unit square and $2\sqrt{kab}-a-b$ for a rectangle.*

2 Algorithms

The following table gives an overview of the algorithms that were developed to solve problem 1.

Algorithm#	Polygon	Complexity	Report cuts
Algorithm 1	rectangles	$O(4^k)$	yes
Algorithm 2 [1]	squares	$O(1)$	$\Omega(k)$
Algorithm 3	rectangles	$O(k^{3/2})$	yes
Algorithm 4	squares	$O(\sqrt{k})$	$\Omega(k)$
Algorithm 5	rectangles	$O(1)$	$\Omega(k)$

*Research supported in part by NSERC. School of Computer Science, Carleton University, 1125 Colonel By Drive, Ottawa, Ontario, Canada, K1S 5B6. E-mail: (jit,dominic)@scs.carleton.ca

†Research supported in part by NSERC. Département d'Informatique, Université du Québec à Hull, Hull, Quebec. E-mail: czyzowicz@uqah.uquebec.ca

A recursive algorithm can be used to solve problem 1. The inputs are the initial rectangle and the number of equal area pieces to be cut. The output is the optimal set of straight line orthogonal cuts. Note that the first (horizontal or vertical) cut can only be made in $2\lfloor \frac{k}{2} \rfloor$ ways. After the first cut has been made, the algorithm recursively solves the two sub-problems. In this way, every possibility is verified and the optimal solution is returned.

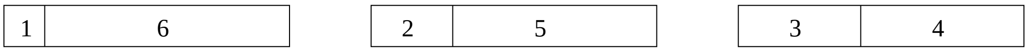


Figure 1: 3 possibilities for a first vertical cut, $k = 7$.

Below we summarize the algorithm:

Optimize(R, k)

Inputs: A rectangle R ($a \times b$) and k. Without loss of generality, assume that $a \leq b$.

Outputs: Set of straight line cuts (or k rectangles).

BEGIN
- $T \leftarrow \phi$. $L \leftarrow \infty$.
- FOR $i = 1$ to $\lfloor \frac{k}{2} \rfloor$
  - $R_1 \leftarrow$ Rectangle$\left(a, \frac{ib}{k}\right)$. $R_2 \leftarrow$ Rectangle$\left(a, \frac{(k-i)b}{k}\right)$.
  - $T_1 \leftarrow$ Optimize(R_1, i). $T_2 \leftarrow$ Optimize(R_2, $k - i$).
  - $l \leftarrow$ total length of the cuts from $T_1 \cup T_2$
  - IF $l < L$ THEN
    - $T \leftarrow T_1 \cup T_2 \cup$ $current\ cut$.
    - $L \leftarrow l + a$ or $(l + b)$.
- Repeat the loop for vertical cuts.
- Return T.

END

Note: $Rectangle(a, b)$ creates a rectangle with dimensions $a \times b$.

Theorem 1 *Optimize(R, k) solves problem 1 and runs in $O(4^k)$ time.*

Proof: The number of elementary operations is given by

$$T(k) = 2 \sum_{i=1}^{\lfloor k/2 \rfloor} (T(i) + T(k-i) + c_1) \text{ where } c_1 \text{ is a small constant}$$

From this we deduce the following:

$$\begin{aligned} T(k) - T(k-1) &\leq 2T(\lfloor k/2 \rfloor) + 2T(k-1) + 2c_1 \\ T(k) &\leq 3T(k-1) + 2T(\lfloor k/2 \rfloor) + 2c_1 \\ &\leq 4T(k-1) \end{aligned}$$

This implies that the algorithm above is $O(4^k)$. ∎

The time complexity of the algorithm can be slightly improved by retaining optimal solutions of sub-problems in a table in order to avoid a recomputation of the same problem multiple times. However, even with this table the algorithm remains exponential.

Theorem 2 *There exist instances where the algorithm described above runs in $\Omega(2^{k^{1-\epsilon}})$ time for any $\epsilon > 0$.*

Proof: Note that a problem instance is specified by three parameters: the number of pieces, the length of the rectangle and the height of the rectangle. To prove the lower bound, we show that the algorithm needs to verify many unique problem instances.

Let $m = c\lfloor \frac{k}{\log k} \rfloor$ be the number of prime numbers smaller than or equal to k for some constant c (prime numbers theorem [5]). Let $P_1 > P_2 > ... > P_m$ represent these primes. There are 2^{m-1} different subsets containing an even number of such primes. For each such subset, we show how to construct a unique problem instance.

Without loss of generality, let $P_{s1}, P_{s2}, \ldots, P_{s2i}$ represent the 2i primes in decreasing order of one such subset. Figure 2 shows an example of how these primes are used to construct a unique instance. We alternatively apply horizontal and vertical cuts until we cut out a rectangle with area $1/k$. This rectangle will be a unique problem instance and the prime numbers determine where the cut occurs. The first vertical cut occurs such that we have P_{s1} pieces to the right and $k - P_{s1}$ pieces to the left. Notice that the rectangle to the right has dimensions $(P_{s1}/k \times 1)$. The next cut is a horizontal cut of the latter rectangle with P_{s2} pieces above. This rectangle has dimensions $(P_{s1}/k \times P_{s2}/P_{s1})$. The next rectangle has P_{s3} pieces to the right and dimensions $(P_{s1}P_{s3}/P_{s2}k \times P_{s2}/P_{s1})$. This process continues until we have one rectangle in the top right corner. This rectangle will have the following dimensions:
$\left(\frac{P_{s1}P_{s3}...P_{s(2i-1)}}{P_{s2}P_{s4}...P_{2i}k} \times \frac{P_{s2}P_{s4}...P_{s2i}}{P_{s1}P_{s3}...P_{s(2i-1)}}\right)$.

Since the dimensions of the rectangle are determined by the primes, each instance is unique. The number of subsets is 2^{m-1} which implies the result since

$$\lim_{k\to\infty} \frac{k^\epsilon}{\log k} = \lim_{k\to\infty} \ln(10)\epsilon k^\epsilon = \infty$$

for any $\epsilon > 0$. ∎

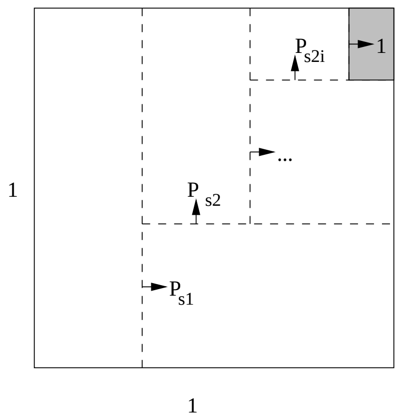


Figure 2: Cuts of a unit square

Proposition 2 *[1] There exists an approximation algorithm that finds a solution that is no more than $1 + \frac{1}{2(\sqrt{k}-1)}$ times the optimal solution in $O(1)$ time[1]. $\Omega(k)$ time is required to report the cuts.*

[1] Assuming that square root of numbers are computed in constant time.

3 Results

In order to gain a better understanding of the structure of optimal solutions, simulations of algorithm Optimize(R, k) were made[2]. From the simulations, we noticed some recurring patterns permitting us to make the following conjectures.

3.1 Properties

Conjecture 1 *The approximation algorithm presented in [1] is optimal.*

Conjecture 2 *The optimal solution of problem 1 contains at most two types of rectangles.*

Two examples of this are shown in figures 3 and 4. Note that in figure 4 we have an example showing that the optimal solution is not unique.
Other properties:

1. There's always a complete cut separating two types of rectangle.
2. For any vertical or horizontal line drawn on an optimal solution from cutting a square, no one of those line will intersect more than $\lceil\sqrt{k}\rceil$ pieces.

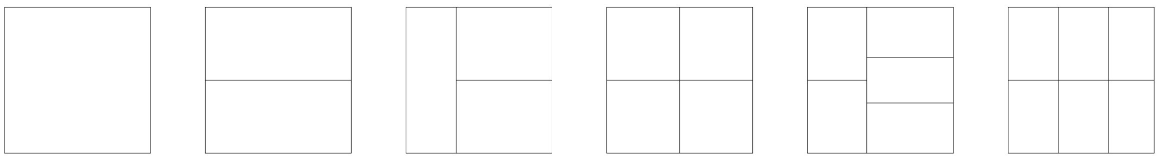


Figure 3: Squares cut in 2, 3, 4, 5 and 6 pieces optimally.

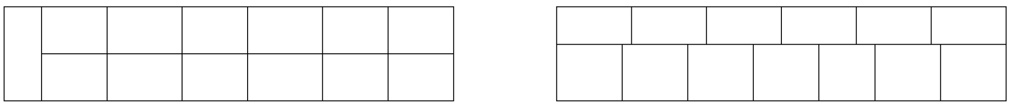


Figure 4: Two rectangles (1×6) cut in 13 pieces optimally.

3.2 Other Algorithms

Theorem 3 *If Conjecture 2 is true then there exists an algorithm solving problem 1 in $O(k^{3/2})$ time.*

Proof: The algorithm is similar to Optimize(R, k). (same inputs and same outputs.) Conjecture 2 allows us to remove the two recursive calls. Once the first cut has been made, each of the two resulting rectangles must be cut into pieces that are the same, otherwise we will have more than two types of rectangles in the final solution.

[2]The reader will find many results of these simulations on the following web site:
`http://www.scs.carleton.ca/~dominic/english/cg.html`.

To cut a rectangle into k pieces of the same type, h horizontal cuts and v vertical cuts are made such that $h \times v = k$. Therefore, we need to compute all factors of k. There are $O(\sqrt{k})$ such factors.
The time complexity is therefore

$$T(k) \;=\; 2\sum_{i=1}^{\lfloor \frac{k}{2} \rfloor} \left(c_2 + 2c_1\sqrt{k}\right) + 3 \;\leq\; k\left(c_2 + 2c_1\sqrt{k}\right) + 3 \;\leq\; c_2 k + 2c_1 k^{3/2} + 3$$

which implies that the algorithm runs in $O(k^{3/2})$ time. ■

The next table shows the optimal length of a unit square and a rectangle 7×19 cut into k equal area pieces by the two algorithms. The results[3] seem to support the two conjectures.

	Unit square			7×19 Rectangle		
k	Lower bound	$O(4^k)$ algo.	$O(k^{3/2})$ algo.	Lower bound	$O(4^k)$ algo.	$O(k^{3/2})$ algo.
2	$2\sqrt{2}-2$	1	1	6.619	7	7
3	$2\sqrt{3}-2$	$\frac{5}{3}$	$\frac{5}{3}$	13.94997	14	14
4	2	2	2	20.13025	21	21
5	$2\sqrt{5}-2$	2.6	2.6	27.57519	28	28
10	$2\sqrt{10}-2$	4.4	4.4	46.93833	47	47
15	$2\sqrt{15}-2$	5.8	5.8	63.33085	971/15	971/15
20	$4\sqrt{5}-2$	7	7	77.15038	779/10	779/10
25	8	8	8	89.32563	2238/25	2238/25

Conjecture 3 *There exists an algorithm solving problem 1 in $O(\sqrt{k})$ time if the input is a square.*

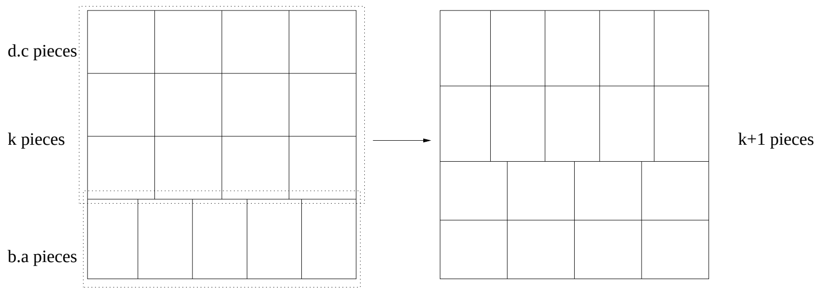
Below we summarize the algorithm:
To compute the optimal solution of cutting a square in k pieces, we need to cut the square in $k-1$ pieces in $O(1)$ time. Cutting into k pieces means that we have to call the following $O(1)$ time function with the optimal solution $k-1$ pieces. The first step is computing the optimal solution for $\lfloor\sqrt{k}\rfloor^2$ pieces, which is the nearest square less than or equal to k. Therefore the first call to the following function will be Constant Optimize(S, $\lfloor\sqrt{k}\rfloor^2$, $\lfloor\sqrt{k}\rfloor$, $\lfloor\sqrt{k}\rfloor$, 0, 0). This functions doesn't compute the square root of any number and it doesn't use the artefact that $k = a^2 + r$ like [1].

Constant Optimize(S, k, a, b, c, d)
Inputs: A square S, number of pieces k.
Optimal solution: $a \times b$ and $c \times d$ pieces. Where $a \leq b$, $c \leq d$ and $b \geq d$. If there is only one type of rectangle in the optimal solution then $c = d = 0$.
Outputs: Optimal configuration of the cuts ($k+1$ equal area pieces): $newa$, $newb$, $newc$ and $newd$.

BEGIN
  IF $c = 0$ THEN
    $newa \leftarrow 1$;
    IF $a = b$ THEN

[3]The simulations used only rational numbers as the dimensions of the rectangles. Then the total lengths of the cuts were also rational numbers, making comparisons more significant.

$newb \leftarrow b+1$; $newc \leftarrow b$; $newd \leftarrow b-1$;
ELSE
$newb \leftarrow b$; $newc \leftarrow newd \leftarrow b-1$;
ELSE IF each entry is different than 1 THEN
$newb \leftarrow b$; $newd \leftarrow d$;
$sum \leftarrow a \times d + b \times c$;
IF $sum = k+1$ THEN
$newa \leftarrow c$; $newc \leftarrow a$;
ELSE
$newa \leftarrow a+1$; $newc \leftarrow c-1$;
ELSE
$newb \leftarrow b$;
IF $c = 1$ THEN
$newc \leftarrow newd \leftarrow 0$;
IF $b \times b = k+1$ THEN
$newa \leftarrow b$;
ELSE
$newa \leftarrow d$;
ELSE
$newa \leftarrow a+1$; $newc \leftarrow c-1$; $newd \leftarrow d$;
Return $\{newa, newb, newc, newd\}$.
END



Theorem 4 *There exists an approximation algorithm solving problem 1 for* $1 \times b$ *rectangles that is no more than* $1 + \frac{b+1}{2\sqrt{kb}-b-1}$ *where* $b \geq 1$. *This algorithm runs in* $O(1)$ *time.* $\Omega(k)$ *time is required to report the cuts.*

Proof:

The following table shows the number of columns (c) that will contain the solution ($c-1$ vertical cuts). r_c are the corresponding ratios.

c	1	2	3	4	5	6	7	...	i
r_c	[0,2[	[2,6[	[6,12[	[12,20[	[20,30[	[30,42[	[42,56[	...	$[i(i-1), i(i+1)]$

Without loss of generality, we assume that the longest side of the rectangle R is vertical. First, the algorithm computes the ratio $\frac{k}{b}$. Then it searches in the previous table the number of columns it has to compute. Once this step is done, it remains to compute how many rectangles (p) will appear in each of the c columns. If the number of pieces to cut is equal to jc where j is an integer, then the solution will contain only on type of rectangle, says j rectangles in each column. Otherwise, it will contain $k \bmod c$ columns with $\lfloor \frac{k}{c} \rfloor + 1$ rectangles and $c - k \bmod c$ columns with $\lfloor \frac{k}{c} \rfloor$ rectangles. The figure 5 shows an example.

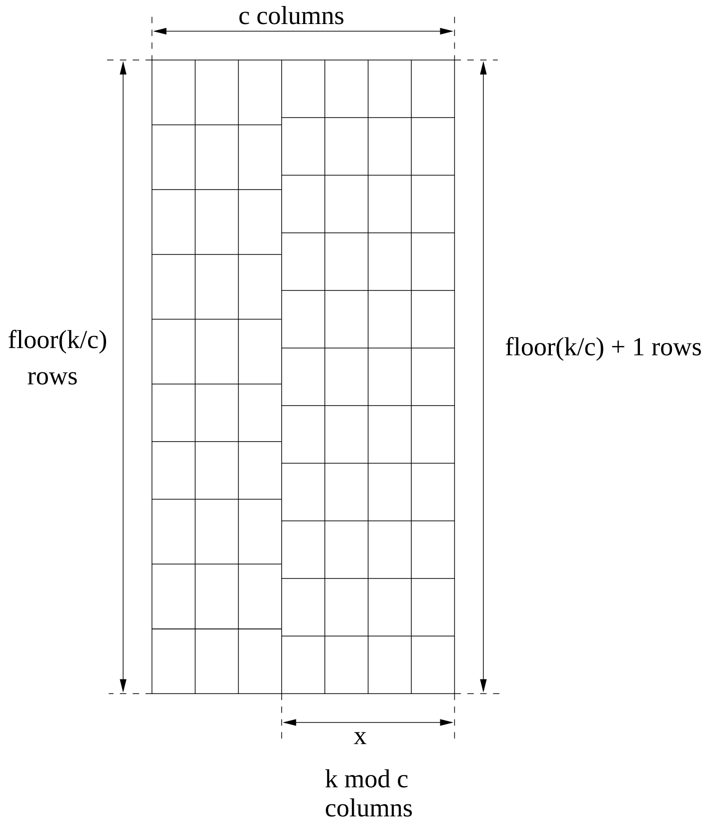


Figure 5: Cutting a rectangle with k rectangles, using c columns.

The total length of the cuts is given by the following formula:

$$C = (c-1)b + (1-x)\left(\lfloor \frac{k}{c} \rfloor - 1\right) + x \lfloor \frac{k}{c} \rfloor$$

Where $x = \frac{\lfloor \frac{k}{c} + 1 \rfloor (k \bmod c)}{k} \leq 1$.

We can compute the number of columns in $O(1)$ time by using the next simple condition. Let r be the ratio and $m = \lfloor \sqrt{r} \rfloor$. If $r \geq m(m+1)$ then we have to build the new solution with $m+1$ columns. Otherwise, we use m columns. To estimate the ratio bound, we use the proposition 1. Then $C_{opt} \geq 2\sqrt{kab} - a - b = 2\sqrt{kb} - b - 1$, because we cut $1 \times b$ rectangles.

$$\frac{C}{C_{opt}} = \frac{(c-1)b + x - 1 + \lfloor \frac{k}{c} \rfloor}{2\sqrt{kb} - b - 1}$$

We divide the proof in two cases:
Case 1: $c = m = \lfloor \sqrt{\frac{k}{b}} \rfloor$.

Also, because $r < m(m+1)$ we get $k < b\lfloor\sqrt{\frac{k}{b}}\rfloor\left(\lfloor\sqrt{\frac{k}{b}}\rfloor+1\right)$.

$$
\begin{aligned}
\frac{C}{C_{opt}} &\leq \frac{\left(\lfloor\sqrt{\frac{k}{b}}\rfloor-1\right)b+\lfloor\frac{k}{\lfloor\sqrt{\frac{k}{b}}\rfloor}\rfloor}{2\sqrt{kb}-b-1} \\
&\leq \frac{b\sqrt{\frac{k}{b}}-b+\frac{k}{\lfloor\sqrt{\frac{k}{b}}\rfloor}}{2\sqrt{kb}-b-1} \\
&\leq \frac{b\sqrt{\frac{k}{b}}-b+\frac{\lfloor\sqrt{\frac{k}{b}}\rfloor\left(\lfloor\sqrt{\frac{k}{b}}\rfloor+1\right)b}{\lfloor\sqrt{\frac{k}{b}}\rfloor}}{2\sqrt{kb}-b-1} \\
&= \frac{\sqrt{kb}+b\lfloor\sqrt{\frac{k}{b}}\rfloor}{2\sqrt{kb}-b-1} \\
&\leq \frac{2\sqrt{kb}}{2\sqrt{kb}-b-1} \\
&= \frac{2\sqrt{kb}-b-1+b+1}{2\sqrt{kb}-b-1} \\
&= 1+\frac{b+1}{2\sqrt{kb}-b-1}
\end{aligned}
$$

Case 2: $c = m+1 = \lfloor\sqrt{\frac{k}{b}}\rfloor+1$

$$
\begin{aligned}
\frac{C}{C_{opt}} &\leq \frac{\left(\lfloor\sqrt{\frac{k}{b}}\rfloor-1+1\right)b+\lfloor\frac{k}{\lfloor\sqrt{\frac{k}{b}}\rfloor+1}\rfloor}{2\sqrt{kb}-b-1} \\
&\leq \frac{b\sqrt{\frac{k}{b}}+\frac{k}{\sqrt{\frac{k}{b}}}}{2\sqrt{kb}-b-1} \\
&= \frac{2\sqrt{kb}}{2\sqrt{kb}-b-1} \\
&\leq 1+\frac{b+1}{2\sqrt{kb}-b-1}
\end{aligned}
$$

Note that if $k < b$ then the total length of the cuts given by the algorithm is $k-1$. ■

Conjecture 4 *The previous algorithm gives optimal solutions for rectangles with dimensions $1 \times b$, where $b \geq 1$ is an integer and $k \geq 2$.*

There exist instances where the algorithm doesn't give the optimal solution. The figure 6 shows a rectangle (1.0×7.2) cut into 15 pieces. For this rectangle there is no optimal solution which contain complete long cut.

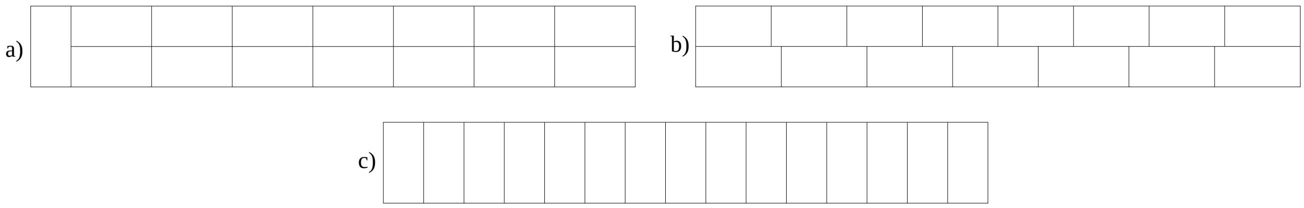


Figure 6: Rectangle (1×7.2), cut into 15 pieces. a) optimal solution: $\frac{205.8}{15}$, b) solution using one long cut: $\frac{206}{15}$ and c) solution without long cut: $\frac{210}{15}$.

4 Conclusion

In this paper we presented algorithms solving the problem 1. We also studied upper and lower bounds on this algorithms. Some properties of optimal solutions were observed. Open problems are prove or disprove the conjectures presented in section 3.

References

[1] BOSE, P.K., CZYZOWICZ, J., KRANAKIS, E., KRIZANC, D. and MAHESHWARI, A. (1998), *Cutting circles and squares in equal area pieces* to appear in FUN98.

[2] CROFT, H.T., FALCONER K.J. and GUY. R.K. (1991), *Unsolved problems in geometry*, Springer-Verlag.

[3] FREDERICKSON, G.N. (1997), *Dissections: Plane and Fancy*, Cambridge University Press.

[4] OVERMARS, M.H. and WELZL E. (1985), *The complexity of cutting paper*, Proc. 1st ACM Symposium on Computational Geometry, Baltimore, Maryland, USA, pp. 316-321.

[5] PARSHIN, A.N. and SHAFAREVICH, I.R. (1995), *Number Theory I*, Encyclopaedia of Mathematical Science, Vol 49. Springer-Verlag, Berlin.