An optimal algorithm for a parallel cutting problem

John Ginsburg
Department of Mathematics and Statistics
University of Winnipeg
Winnipeg, MB R3B 2E9

Bill Sands*
Department of Mathematics and Statistics
University of Calgary
Calgary, AB T2N 1N4

ABSTRACT. Suppose we are given a set of sticks of various integer lengths, and that we have a knife that can cut as many as w sticks at a time. We wish to cut all the sticks up into pieces of unit length. By what procedure should the sticks be cut so that the total number of steps required is minimum? In this paper we show that the following natural algorithm is optimal: at each stage, choose the w longest sticks (or all sticks of length > 1 if there are fewer than w of them) and cut them all in half (or as nearly in half as possible).

Suppose you have a number of sticks of celery, of various integer lengths, which you want to chop up into pieces of length one. Suppose further that you have a knife which can cut at most w sticks at once. You want to cut up the celery using as few cuts as possible; how do you do it?

In this paper we show that the most reasonable algorithm is the correct one. Namely, *at each stage, choose the w longest sticks and cut them all in half (or as near as possible).* This means that sticks of even length will be cut exactly in half while sticks of odd length will be cut into two pieces differing by one; also, if at any stage there are *less than w* sticks of length greater than 1 then all of them are cut.

*Research done while the second author was visiting the University of Winnipeg

ARS COMBINATORIA 57(2000), pp. 87-95

To illustrate this algorithm, imagine we have a knife which can cut up to 3 sticks at a time, and that we have two sticks to cut, whose lengths are 9 and 6. On the first step , both sticks are cut , producing four sticks of lengths 5,4,3,3. On the second step , the biggest three of these four are cut in half , producing sticks of lengths 3,2,2,2,2,1,3. Then the largest three of these sticks are cut in half giving sticks of lengths 2,1,1,1,2,2,2,1,2,1. On the fourth step , three of these length 2 sticks are cut into two units , and on the fifth step the remaining two sticks of length 2 are cut. Thus the algorithm takes five steps to cut the original pair of sticks into units. While there are other ways to cut the two sticks into units in five steps , we note that some ways of cutting the sticks require more steps than others. For example , our first step might consist of cutting the two sticks into pieces of lengths 6,3 and 3,3. We then might cut units off the ends of the equal sized sticks producing sticks of lengths 6,2,1,2,1,2,1 and then 6,1,1,1,1,1,1,1,1,1 after two more cuts. We could then cut the 6 into two 3's. Then two more steps would be needed to cut the 3's down to units. In all, this method for cutting the sticks requires six steps.

This algorithm is of course an example of the usual binary "divide and conquer" algorithms (e.g., [1], §2.6). However, it might be interesting to observe that our result is *not true* if our knife has "varying width". For example, suppose that we have to chop up just one stick of celery of length 6, and that we only have a knife of width 1, except that for our third cut we can use a knife of width 3. Then the above algorithm would take four cuts (6 to 3,3 to 2,1,3 to 1,1,1,2,1 and we're done on the next step), whereas it can be done in three cuts another way (6 to 4,2 to 2,2,2 and done on the next step).

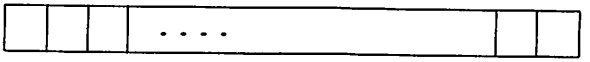
Here is a possible application of our result. Imagine that we are given a collection of linear strips or reels each consisting of consecutive units of information joined side by side. Suppose that , in order to read the information stored in a particular unit on a strip , we detach that unit and examine it. This creates two linear strips , together containing one less unit of information. Our cutting problem then is equivalent to having w "readers" which can read in parallel , and each of which can examine (by detaching) one unit on one linear strip.

Recent work in the mathematical literature has considered other types of cutting problems not directly related to our work , for example the old problem of how to cut a cake fairly (for example , see [2] and [5]), and the problem of deleting the minimum number of edges from a graph to obtain two equally sized subgraphs (for example see [3] and [4]). We have not seen the celery cutting problem[1] in the literature.

Let's begin with a little notation and terminology which will be useful

[1] not to be confused, incidentally, with the "salary cutting problem" which is currently in vogue at many universities, and remains unsolved.

in our discussion. We shall denote a set of t sticks (actually a multiset) by a t-tuple $S = \{l_1, l_2, \ldots, l_t\}$ where the l_i's are positive integers, not necessarily distinct. The l_i's will be called sticks although they are really lengths of sticks. Any stick of length one is called *trivial*, the others are called *nontrivial*. The algorithm described above , to be proved optimal, will be called the *binary* algorithm.



We think of a stick of length l as consisting of l units joined linearly side-by-side , as illustrated in the figure. We thus speak of the two *ends* of the stick if it has length bigger than 1. When a cut is made to such a stick , it is made between two consecutive units. Any group of k consecutive units in this stick will be called a *piece* of length k of the stick l. In particular , the units themselves are pieces of length 1 , and any two consecutive units form a piece of length 2.

Let $S = \{l_1, l_2, \ldots, l_t\}$ be a set of sticks, and let $\lambda(S) = \sum_{i=1}^{t} l_i$ be the total length of the sticks in S. We suppose that we have a knife which can cut at most w sticks at a time , where w is a given positive integer. Note that , in whatever way we cut S , we select k nontrivial sticks in S , say $l_{i_1}, l_{i_2}, \ldots, l_{i_k}$ where k is some integer with $1 \leq k \leq w$, and each of these sticks l_{i_j} is cut into a pair of sticks whose sum is l_{i_j}. This results in a set of sticks T whose elements are these pairs of sticks , together with all the other sticks of S which were not selected. We say that T *is obtained by cutting* S. Obviously, if T is obtained by cutting S then $\lambda(T) = \lambda(S)$.

Let $S = \{l_1, l_2, \ldots, l_t\}$ be a set of sticks. A *procedure for cutting* S is a sequence $\mathcal{A} = (A_0, A_1, \ldots, A_n)$ of sets where n is some non-negative integer , $A_0 = S$, $A_n = \{1, 1, 1, \ldots, 1\}$, and where A_{i+1} is obtained by cutting A_i for all $i = 0, 1, \ldots, n-1$. We say that n is the number of steps required by the procedure $\mathcal{A}$ on S. Note that, for later convenience, we allow the trivial case $S = \{1, \ldots, 1\}$ where no cuts are needed, so $n = 0$ and $\mathcal{A}$ is just (S).

Now the binary algorithm referred to above can be described as follows. For any set of sticks $S = \{l_1, l_2, \ldots, l_t\}$, not all trivial, we let $B(S)$ be the set of sticks obtained from S as follows. We select the w largest nontrivial sticks in S (breaking ties arbitrarily), or all of the nontrivial sticks in S if there are less than w of them. Each of these selected sticks l_i is replaced by two sticks of lengths $l_i/2$ if l_i is even , and by two sticks of lengths $\lceil l_i/2 \rceil$ and $\lfloor l_i/2 \rfloor$ if l_i is odd. (So $B(S)$ is obtained by cutting S , the w longest pieces of S being cut in half.) We consider the sequence $B_0, B_1, \ldots, B_i, \ldots$ where $B_0 = S$, and where $B_i = B(B_{i-1})$ for all $i > 0$. For some positive integer m we will have $B_m = \{1, 1, 1, \ldots, 1\}$. In the terminology of the preceding paragraph , the binary algorithm applied to S is the procedure $B_0 = S, B_1, \ldots, B_m = \{1, 1, \ldots, 1\}$. We will denote the binary algorithm

by $\mathcal{B}$.

Note that , in applying the binary algorithm to a given set of sticks S , there may be , at various steps , a choice as to which stick or sticks of the same length are to be cut. While the sequence of sets $B_0 = S, B_1, \ldots, B_m = \{1, 1, \ldots, 1\}$ obtained is the same for all such choices (since the elements of these sets are just the lengths of the sticks), we will still speak of particular *implementations* of the algorithm — by this we refer to the particular sticks which are selected at the times when more than one choice is possible.

We will show below that the number of steps taken by the binary algorithm in cutting any set of sticks S is no more than the number of steps required by any other procedure for cutting S. Observe that the case when $w = 1$ is trivial : in this case , since exactly one cut is made at any step of any procedure , it follows that *every* procedure requires exactly $\sum_{i=1}^{t}(l_i - 1)$ steps in cutting a set of sticks $S = \{l_1, l_2, \ldots, l_t\}$. Thus we will assume that $w > 1$ from now on. It is also worth noting that , even though the first step of the binary algorithm , applied to a set of sticks S , produces the lexicographically smallest set of sticks that can be obtained from S in one cutting step , this does not lead to a direct proof of optimality. It is possible for another cutting procedure to produce a lexicographically smaller set of sticks after some number of steps than the binary algorithm does in the same number of steps. As an example of this ,suppose we take $w = 2$, and let $S = \{11\}$. Three steps of the binary algorithm produces the sets $\{6, 5\}, \{3, 3, 3, 2\}, \{3, 2, 1, 2, 1, 2\}$. Another procedure for cutting S produces $\{7, 4\}, \{4, 3, 2, 2\}, \{2, 2, 2, 1, 2, 2\}$. The third set from the binary procedure is lexicographically larger than the third set produced by the second procedure , since it has a bigger largest element (we thank Bill Martin for this observation).

Our proof of the optimality of the binary algorithm is an induction based on the idea of a prevailing pair which we now describe.

Consider an implementation of the binary algorithm $\mathcal{B}$ for cutting a set of sticks $S = \{l_1, l_2, \ldots, l_t\}$. Let l_i be a nontrivial stick in S , and let a and b be two consecutive units of the stick l_i. The pair $\{a, b\}$ will be called a *prevailing pair* for this implementation of $\mathcal{B}$ if the following three conditions are satisfied :

(i) at any step of the procedure , any stick s which contains the pair $\{a, b\}$ is only cut if all other sticks then existing which have the same length as s are cut as well;

(ii) whenever a stick of odd length containing the pair $\{a, b\}$ is cut , the pair $\{a, b\}$ remain together in the bigger of the two pieces;

(iii) the cut which splits the two units a and b from one another occurs by cutting the length 2 stick $\{a, b\}$ (this stick having been obtained at some point in the procedure).

Lemma 1. *Let $S = \{l_1, l_2, \ldots, l_t\}$. Let l_i be a nontrivial stick in S, and let a and b be two consecutive units at one end of l_i. Then there is an implementation of the binary algorithm on S in which $\{a, b\}$ is a prevailing pair.*

Proof: We just implement the procedure so that conditions (i) and (ii) above are satisfied at every step. Since a and b will always be together at one end of any piece containing them , this is certainly possible to do , and a and b will not be split from one another until they occur together as a stick of length 2. □

It can further be shown that, if the stick l_i in Lemma 1 is a *longest* stick in S , then the two units a and b are not cut apart until the final step of the procedure. We will not need this fact, however.

In discussing a procedure for cutting a set of sticks S, we will find it useful to use the following term. For any stick s which occurs in the procedure , we refer to the *ancestors* of s ; by this we mean the sticks occurring earlier in the procedure which contain s.

Lemma 2. *Let $S = \{l_1, l_2, \ldots, l_t\}$, and let a and b be consecutive units at one end of the stick of length l_1. Suppose that $\{a, b\}$ is a prevailing pair in a particular implementation of the binary algorithm $\mathcal{B}$. Let $S^* = \{l_1 - 1, l_2, \ldots, l_t\}$ be the set of sticks obtained from S by considering the pair $\{a, b\}$ to be a single unit at the end of the stick l_1. Let $\mathcal{B}^*$ be the procedure on S^* obtained from $\mathcal{B}$ in the obvious manner, the only change being that when the pair $\{a, b\}$ is split apart by $\mathcal{B}$ we do not make this cut but leave a and b together. Then $\mathcal{B}^*$ is an implementation of the binary algorithm acting on S^*.*

Proof: We only need to observe that

(i) at each stage, w sticks will be cut whenever there are at least that many nontrivial sticks available, and otherwise all nontrivial sticks will be cut. For the only cut in $\mathcal{B}$ which is not made in $\mathcal{B}^*$ is the one cutting the prevailing pair, and by condition (i) above this cut is not made in $\mathcal{B}$ unless all other nontrivial sticks at that stage are cut as well. Furthermore , this implies that , in $\mathcal{B}^*$, it is always the largest pieces which are cut at any stage.

(ii) each cut in $\mathcal{B}^*$ cuts each stick "in half". This is clearly true for non-ancestors of the stick $\{a, b\}$. Any ancestor of $\{a, b\}$ of even length $2r$ in $\mathcal{B}$ becomes a stick of odd length $2r - 1$ in $\mathcal{B}^*$. Since in $\mathcal{B}$ this stick is cut into two sticks of length r (one containing the prevailing pair), in $\mathcal{B}^*$ the corresponding stick will be cut into two sticks of lengths r and $r - 1$. When an ancestor of $\{a, b\}$ of odd length is cut in $\mathcal{B}$, $\{a, b\}$ is always part of the larger piece in $\mathcal{B}$. This corresponds in $\mathcal{B}^*$ to cutting an even length stick into two equal halves. □

Here is an illustration of the induced binary algorithm described in Lemma 2. Let us take a set of two sticks of lengths 19 and 5 and suppose we can cut up to five sticks at a time , so that $w = 5$. We choose a pair of units at the end of the stick of length 5, and we implement the algorithm $\mathcal{B}$ so that this pair is prevailing. As we run through the steps of the procedure, the stick which contains this pair will be denoted with an asterisk. So to begin with we have the set of sticks $S = B_0 = \{19, 5^*\}$. We cut both sticks in half to get $B_1 = \{10, 9, 3^*, 2\}$. We then cut all four sticks of B_1 in half to get $B_2 = \{5, 5, 5, 4, 2^*, 1, 1, 1\}$. Then the five biggest sticks in B_2 are cut to give $B_3 = \{3, 2, 3, 2, 3, 2, 2, 2, 1, 1, 1, 1, 1\}$. At this point our prevailing pair has been split apart. We again cut the five biggest sticks to get

$$B_4 = \{2, 1, 1, 1, 2, 1, 1, 1, 2, 1, 2, 2, 2, 1, 1, 1, 1, 1\}.$$

Cutting five of the remaining 2's gives

$$B_5 = \{2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1\}$$

and cutting the last 2, we end with

$$B_6 = \{1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1\}.$$

In applying the algorithm $\mathcal{B}^*$, each of the sticks marked with an asterisk is thought of as a stick with length one less. When the prevailing pair is cut by $\mathcal{B}$, the stick containing it has become a 1 (as seen by $\mathcal{B}^*$), and this cut is just ignored by $\mathcal{B}^*$. This special 1 just sits there the rest of the way. Thus $\mathcal{B}^*$ is the following procedure (the special unit corresponding to the prevailing pair is indicated with an asterisk):

$$S^* = B_0^* = \{19, 4\}; \qquad B_1^* = \{10, 9, 2, 2\};$$

$$B_2^* = \{5, 5, 5, 4, 1^*, 1, 1, 1\}; \qquad B_3^* = \{3, 2, 3, 2, 3, 2, 2, 2, 1^*, 1, 1, 1\};$$

$$B_4^* = \{2, 1, 1, 1, 2, 1, 1, 1, 2, 1, 2, 2, 2, 1^*, 1, 1, 1\};$$

$$B_5^* = \{2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1^*, 1, 1, 1\};$$

$$B_6^* = \{1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1^*, 1, 1, 1\}.$$

If we ignore the asterisk on the 1, this is just the binary algorithm applied to $\{19, 4\}$ with $w = 5$.

Lemma 3. *Let $S = \{l_1, l_2, \ldots, l_t\}$. Apply the binary algorithm $\mathcal{B}$ to S, yielding the sequence of sets $B_0 = S, B_1, B_2, \ldots$. Let $\mathcal{A}$ be any other cutting procedure applied to S, given by the sequence of sets $A_0 = S, A_1, A_2, \ldots$. Then there is no integer k such that*

$$B_k = \{1, 1, \ldots, 1, 2, 2, \ldots, 2, 3\} \qquad \text{and} \qquad A_k = \{1, 1, \ldots, 1, 2, 2, \ldots, 2\},$$

where B_k has at most $w-1$ 2's.

Proof: Suppose such a k exists. We first prove by induction on i that, for every $i \leq k$, the set B_{k-i} has a unique longest stick and its length is $2^{i+1}+1$, and there are at most w sticks in B_{k-i} whose lengths are $\geq 2^i+1$. This statement is true for $i=0$ by assumption.

Now assume that the statement is true for some $i < k$. Since the two largest elements of B_{k-i} have sum at most $(2^{i+1}+1)+2^{i+1} = 2^{i+2}+1$, there can be no element larger than this in B_{k-i-1}. If the stick p of length $2^{i+1}+1$ in B_{k-i} was also present as a stick in B_{k-i-1}, then because it was not cut in forming B_{k-i} there must have been at least w other sticks in B_{k-i-1} of lengths at least $2^{i+1}+1$ which were cut, and these together with p would have resulted in at least $w+1$ sticks in B_{k-i} of lengths at least 2^i+1, contradicting the induction hypothesis. Thus the stick p must have come from a stick of length exactly $2^{i+2}+1$ in B_{k-i-1}. By the uniqueness of p there cannot be two sticks of this length in B_{k-i-1}. Similarly, if there were more than w sticks of lengths at least $2^{i+1}+1$ in B_{k-i-1}, these would give rise to more than w sticks in B_{k-i} of lengths at least 2^i+1, contradiction. This proves the statement.

Next we observe that every stick in A_{k-i} has length at most 2^{i+1}. This is true when $i=0$, and since the longest stick in A_{k-i-1} is at most twice the length of the longest stick in A_{k-i}, the statement follows by induction.

The preceding paragraphs imply that A_{k-i} is not equal to B_{k-i} for any i, contradicting $A_0 = B_0$. □

Now we are ready to prove our main result.

Theorem. *Let $S = \{l_1, l_2, \ldots, l_t\}$, not all trivial. Apply the binary algorithm $\mathcal{B}$ to S, yielding the sequence of sets $B_0 = S, B_1, B_2, \ldots, B_m = \{1,1,\ldots,1\}$. Let $\mathcal{A}$ be any other cutting procedure applied to S, given by the sequence $A_0 = S, A_1, A_2, \ldots, A_n = \{1,1,\ldots,1\}$. Then $m \leq n$ and for each $i \in \{0,\ldots,m\}$ the number of sticks in B_i is at least as big as the number of sticks in A_i. (In particular, $m \leq n$ says that the binary algorithm is optimal.)*

Proof: The proof is by induction on $\lambda(S)$. It is clearly true if all the l_i's are equal to 1 or 2. Now choose some $S = \{l_1, l_2, \ldots, l_t\}$ and assume the result holds for all sets of sticks of smaller total length. Choose a stick s of length two which was cut at the last step of $\mathcal{A}$, so that s was one of the sticks of A_{n-1}. Locate the original stick of S which is an ancestor of s; we can assume it is l_1. By Lemma 1, we can assume that $\mathcal{B}$ has been implemented on S in such a way that there is a prevailing pair $\{a,b\}$ at one end of the stick l_1. As in Lemma 2, let $S^* = \{l_1-1, l_2, \ldots, l_t\}$ be the set of sticks obtained from S by considering the pair $\{a,b\}$ to be a single unit at one end of the stick l_1 , and let $\mathcal{B}^*$ be the procedure described in

Lemma 2. This is an implementation of $\mathcal{B}$ on S^*, yielding the sequence of sets $B_0^* = S^*, B_1^*, B_2^*, \ldots, B_{m'}^* = \{1, 1, \ldots, 1\}$. Note that each B_i^* differs from B_i only in the length of the appropriate ancestor of $\{a, b\}$, until the step when $\mathcal{B}$ cuts the piece $\{a, b\}$ itself, after which each B_i^* contains one less 1 than the corresponding B_i does. Also, $m' = m$ unless $\{a, b\}$ is the **only** piece cut at this step; this must therefore occur at the last step, i.e., B_{m-1} contains the stick $\{a, b\}$ and no other nontrivial pieces. In this case m' will equal $m - 1$.

Similarly , by thinking of the two units which make up the stick s as a single unit throughout the procedure $\mathcal{A}$, we see that $\mathcal{A}$ gives rise to a procedure on S^* given by the sequence of sets $A_0^* = S^*, A_1^*, A_2^*, \ldots, A_{n'}^* = \{1, 1, \ldots, 1\}$. As before, the only times A_i can have more pieces than A_i^* is after s has been cut by $\mathcal{A}$. But we chose s so that this doesn't happen until the last step, so the only such example is that A_n has one more 1 than A_n^* does. Also, $n = n'$ unless A_{n-1} contains only one 2, namely s, in which case A_{n-1}^* contains only 1's and $n' = n - 1$.

By induction, since $\lambda(S^*) < \lambda(S)$, we have that $m' \leq n'$, and for each i the number of pieces in B_i^* is $\geq$ the number of pieces in A_i^*. From above, this will imply that $m \leq n$ and the number of pieces in each B_i is $\geq$ the number of pieces in A_i, unless $m - 1 = m' = n' = n$, B_{m-1} contains $\{a, b\}$ and no other nontrivial pieces, and A_{n-1} contains s plus at least one (but at most $w - 1$) other 2's and no other nontrivial pieces. In this latter case , we now observe that the nontrivial elements of $B_{n-1} = B_{m-2}$ can be either exactly $w + 1$ 2's, or exactly one 3 and less than w 2's. For otherwise:

- if there were a stick of length 5 or more in B_{n-1} it would produce a stick of length 3 or more in B_n.
- if there were a stick of length 4 or two sticks of length 3 in B_{n-1}, this would result in at least two 2's in B_n.
- if there were a 3 and w 2's, or if there were at least $w+2$ 2's, in B_{n-1} , this would also produce two 2's in B_n.
- if B_{n-1} consisted of less than $w + 1$ 2's , this would mean that B_n has no nontrivial sticks at all.

But if B_{n-1} contained only some number of 1's and exactly $w + 1$ 2's (one of which must be $\{a, b\}$), B_{n-1}^* would contain 1's and exactly w 2's, while A_{n-1}^* contains 1's and less than w 2's. Since the sum of the lengths of sticks in B_{n-1}^* and in A_{n-1}^* are the same, namely one less than $\lambda(S)$, and since B_{n-1}^* has at least as many pieces as A_{n-1}^*, this is impossible. Therefore the nontrivial sticks in B_{n-1} must consist of one 3 and less than w 2's. This means that B_k and A_k satisfy the conditions of Lemma 3 for $k = n - 1$, which is impossible. □

Acknowledgement. We would like to thank the members of the University of Winnipeg Combinatorics seminar for several helpful discussions on this topic.

References

[1] A.V. Aho, J.E. Hopcroft and J.D. Ullman, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, 1974.

[2] S.Brams, A.Taylor and W. Zwicker, Old and new moving knife schemes, *The Mathematical Intelligencer*, **17**, No.4 (1995), 30–35.

[3] T.N. Bui and A. Peck, Partitioning planar graphs, *Siam J. Comput.* **21**, No.2 (1992), 203–215.

[4] M. Goldberg and Z. Miller, A parallel algorithm for bisection width in trees, *Comput. Math. Appl.* **15**, No.4 (1988), 259–266.

[5] J. Robertson and W. Webb, Approximating fair division with a limited number of cuts, *Journ. Comb. Theory, Series A* **72** (1995), 340–344.