SIAM J. COMPUT.
Vol. 13, No. 2, May 1984

002

COMPUTATION OF MATRIX CHAIN PRODUCTS. PART II*

T. C. HU† AND M. T. SHING†

Abstract. This paper considers the computation of matrix chain products of the form $M_1 \times M_2 \times \cdots \times M_{n-1}$. If the matrices are of different dimensions, the order in which the matrices are computed affects the number of operations. An optimum order is an order which minimizes the total number of operations. Some theorems about an optimum order of computing the matrices have been presented in Part I [SIAM J. Comput., 11 (1982), pp. 362–373]. Based on those theorems, an $O(n \log n)$ algorithm for finding the optimum order is presented here.

1. Introduction. In Part I of this paper [6], we have transformed the matrix chain product problem into the optimum partitioning problem and have stated several theorems about the optimum partitions of an n-sided convex polygon. Some theorems in Part I can be strengthened and are stated here (the detailed proofs are in [7]).

THEOREM 1. *For every choice of $V_1, V_2, \cdots$ (as prescribed in Part* I), *if the weights of the vertices of the n-gon satisfy the following condition,*

$$w_1 = w_2 = \cdots = w_k < w_{k+1} \leqq \cdots \leqq w_n$$

for some k, $3 \leqq k \leqq n$, then every optimum partition of the n-gon contains the k-gon $V_1 - V_2 - \cdots - V_k$. Furthermore, if $k = 2$ in the above condition, i.e. $w_1 = w_2 < w_3 \leqq w_4 \leqq \cdots \leqq w_n$, then every optimum partition of the n-gon must contain a triangle $V_1 V_2 V_p$ for some vertex V_p with weight equal to w_3.

Note that if $w_1 = w_2 < w_3 < w_4 \leqq \cdots \leqq w_n$, then every optimum partition must contain the triangle $V_1 V_2 V_3$ since there is a unique choice of V_3.

Now, whenever we have three or more vertices with weights equal to w_1 in the n-gon, we can decompose the n-gon into subpolygons by forming the k-gon in the first part of Theorem 1. The partition of the k-gon can be arbitrary, since all vertices of the k-gon are of equal weight. For any subpolygon with two vertices of weights equal to w_1, we can always apply the second part of Theorem 1 and decompose the subpolygon into smaller subpolygons. Hence, we have only to consider the polygons with a unique choice of V_1; i.e., each polygon has only one vertex with weight equal to w_1.

Because of the above theorem, Theorems 1 and 3 of Part I can be generalized as follows.

THEOREM 2. *For every choice of $V_1, V_2, \cdots$ (as prescribed in Part* I), *if the weights of the vertices satisfy the condition*

$$w_1 < w_2 \leqq w_3 \leqq \cdots \leqq w_n,$$

then $V_1 - V_2$ and $V_1 - V_3$ exist in every optimum partition of the n-gon.

THEOREM 3. *Let V_x and V_z be two arbitrary vertices which are not adjacent in a polygon, and V_w be the smallest vertex from V_x to V_z in the clockwise manner ($V_w \neq V_x$, $V_w \neq V_z$), and V_y be the smallest vertex from V_z to V_x in the clockwise manner ($V_y \neq V_x$, $V_y \neq V_z$). This is shown in Fig.* 1. *Assume that $V_x < V_z$ and $V_y < V_w$. The necessary condition for $V_x - V_z$ to exist as an h-arc in any optimum partition is*

$$w_y < w_x \leqq w_z < w_w.$$

* Received by the editors August 4, 1981, and in final revised form February 7, 1983. This research was supported in part by the National Science Foundation under grant MCS-77-23738 and in part by the U.S. Army Research Office under grant DAAG29-80-C-0029.

† University of California at San Diego, La Jolla, California 92093.

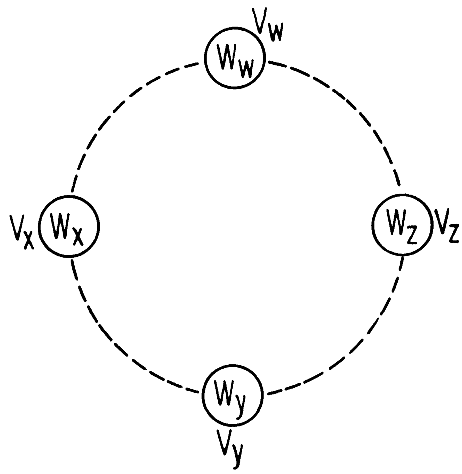

FIG. 1

We shall use "the l-optimum partition" to mean "the lexicographically smallest optimum partition." Based on these theorems, we now present algorithms for finding the unique l-optimum partition.

Using the same notation as in Part I of this paper [6], we can assume that we have uniquely labelled all vertices of the n-gon. A partition is called a *fan* it is consists of only v-arcs joining the smallest vertex to all other vertices in the polygon. We shall denote the fan of a polygon $V_1 - V_b - V_c - \cdots - V_n$ by Fan $(w_1|w_b, w_c, \cdots, w_n)$. The smallest vertex V_1 is called the *center* of the fan.

We define a vertex as a *local maximum* vertex if it is larger than its two neighbors and define a vertex as a *local minimum* vertex if it is smaller than its two neighbors. A polygon is called a *monotone* polygon if there exist only one local maximum and one local minimum vertex. We shall first give an $O(n)$ algorithm for finding the l-optimum partition of a monotone polygon and then give an $O(n \log n)$ algorithm for finding the l-optimum partition of a general convex polygon.

2. Monotone basic polygon. In this section, let us consider the optimum partition of a monotone polygon, i.e. a polygon with only one local minimum vertex and one local maximum vertex. It follows from Theorems 1 and 2 that we can consider a monotone basic polygon only. (A polygon having V_1 adjacent to V_2 and V_3 by sides is called a *basic* polygon.) The understanding of this special case is necessary in finding the optimum partition of a general convex polygon.

Consider a monotone basic n-gon $V_1 - V_2 - V_c - \cdots - V_3$, the fan of the polygon is denoted by

$$\text{Fan}\,(w_1|w_2, w_c, \cdots, w_3)$$

where the smallest vertex V_1 is the center of the fan.

The definition of a fan can also be applied to subpolygons as well. For example, if V_2, V_3 are connected in the basic n-gon and V_2 becomes the smallest vertex in the $(n-1)$-sided subpolygon, the partition formed by connecting V_2 to all vertices in the $(n-1)$-gon is denoted by

$$\text{Fan}\,(w_2|w_c, \cdots, w_3).$$

LEMMA 1. *If none of the potential h-arcs appears in the l-optimum partition of the n-gon, the l-optimum partition must be the fan of the n-gon.*

Proof. Omitted. See [7] for details. □

A potential h-arc will dissect a polygon into two parts, and the subpolygon which contains the larger vertices is called the *upper subpolygon*. Let $V_i - V_j$ and $V_p - V_q$ be two potential h-arcs of any n-gon. We say that $V_p - P_q$ is *above* (or *higher than*) $V_i - V_j$ (and $V_i - V_j$ is *below*, or *lower than*, $V_p - V_q$) if the upper subpolygon of $V_i - V_j$ contains the upper subpolygon of $V_p - V_q$.

Let P be the set of all potential h-arcs in a monotone basic n-gon. P can have at most $n - 3$ arcs.

LEMMA 2. *For any two arcs in P, say $V_i - V_j$ and $V_p - V_q$, we must have either $V_i - V_j$ above $V_p - V_q$ or $V_p - V_q$ above $V_i - V_j$.*

Proof. See [7] for details. □

We can actually show this ordering of potential h-arcs pictorially by drawing a monotone basic polygon in such a way that the local maximum vertex is always at the top and the local minimum vertex is at the bottom. Then a potential h-arc $V_p - V_q$ is physically above another potential h-arc $V_i - V_j$ if the upper subpolygon of $V_i - V_j$ contains the upper subpolygon of $V_p - V_q$. From the definition of the upper subpolygon and the monotone property, we can see that $\max(w_i, w_j) < \min(w_p, w_q)$ if $V_p - V_q$ is above $V_i - V_j$.

Consider the monotone basic n-gon which is shown symbolically in Fig. 2. V_n is the local maximum vertex and $V_i - V_j$, $V_p - V_q$ are potential h-arcs of the monotone basic n-gon. The subpolygon $V_i - \cdots - V_p - V_q - \cdots - V_j$ which is formed by two potential h-arcs $V_p - V_q$ and $V_i - V_j$ and the sides of the n-gon from V_i to V_p and from V_q to V_j in the clockwise direction is said to be *bounded above* by the potential h-arc $V_p - V_q$ and *bounded below* by the potential h-arc $V_i - V_j$, or simply as the subpolygon between $V_i - V_j$ and $V_p - V_q$ for brevity.

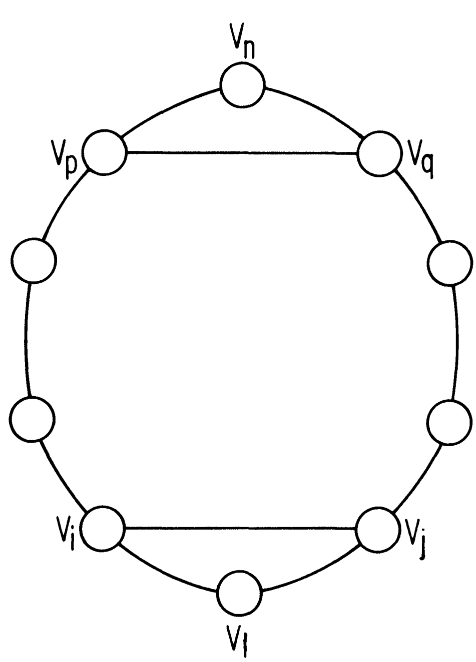

FIG. 2

LEMMA 3. *Any subpolygon which is bounded by two potential h-arcs of the monotone basic n-gon is itself a monotone polygon.*

Proof. See [7] for details. □

LEMMA 4. *Any potential h-arc of a subpolygon bounded above and below by two potential h-arcs of the monotone basic n-gon is also a potential h-arc of the monotone basic n-gon.*

Proof. See [7] for details. □

We can now summarize what we have discussed. If there is no h-arc in the l-optimum partition of a monotone basic n-gon, the l-optimum partition must be a fan. Otherwise, the h-arcs in the l-optimum partition are all layered, one above another. If we consider the local maximum vertex V_n and the local minimum vertex V_1 as two degenerated h-arcs, then the l-optimum partition of a monotone basic n-gon will contain one or more monotone subpolygons, each bounded above and below by two h-arcs and the l-optimum partition of each of these monotone subpolygons is a fan. Then, in finding the l-optimum partition of a monotone basic polygon, we have only to consider those partitions which contain one or more potential h-arcs and each of the subpolygons between two potential h-arcs is partitioned by a fan.

Since there are at most $n-3$ nondegenerated potential h-arcs in a monotone basic n-gon, there will be at most 2^{n-3} such partitions and we can divide all these partitions into $(n-2)$ classes by the number of nondegenerated potential h-arcs a partition contains. These classes are denoted by $H_0, H_1, \cdots, H_{n-3}$ where the subscript indicates the number of nondegenerated potential h-arcs in each partition of that class.

There is no potential h-arc in the partitions in the class H_0. Hence the class consists of only one partition, namely the fan

$$\text{Fan}\,(w_1|w_2, \cdots, w_3).$$

In the class H_1, each partition has one nondegenerated potential h-arc. Once the potential h-arc is known, the rest of the arcs must all be vertical arcs forming two fans, one in each subpolygon.

Two typical partitions in H_1 of a monotone basic polygon are shown in Fig. 3. In Fig. 3a, there is one nondegenerated potential h-arc, $V_c - V_i (V_c < V_i)$. The upper

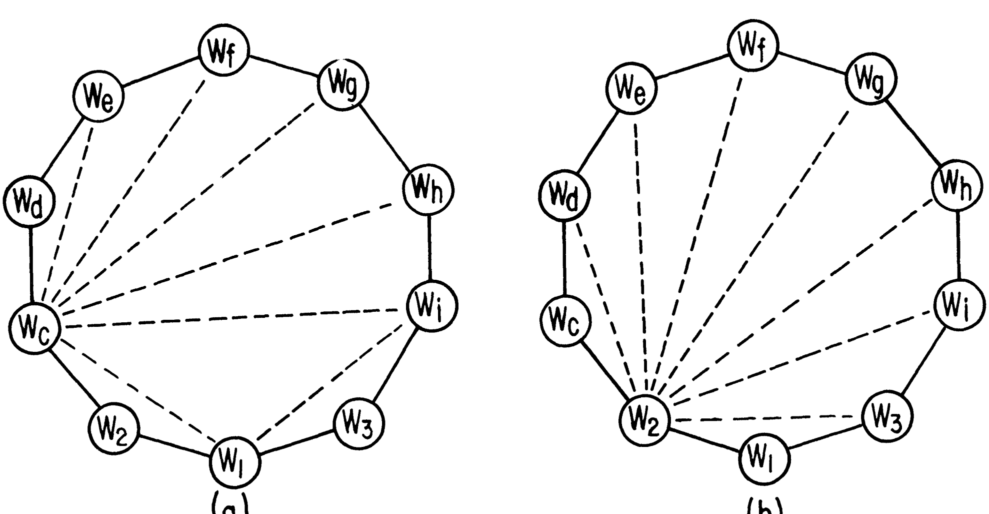

FIG. 3. *Two typical partitions in H_1 of a monotone* 10*-gon.*

subpolygon is a fan

$$\text{Fan}\,(w_c|w_d, \cdots, w_i)$$

and the lower subpolygon is a fan

$$\text{Fan}\,(w_1|w_2, w_c, w_i, w_3).$$

In Fig. 3b, there is one potential h-arc, $V_2 - V_3$, and the upper subpolygon is a fan

$$\text{Fan}\,(w_2|w_c, \cdots, w_3)$$

and the lower subpolygon is a degenerated fan, a triangle.

The cost of the partition in Fig. 3b is

$$\begin{aligned} &w_1w_2w_3 + w_2(w_cw_d + w_dw_e + w_ew_f + w_fw_g + w_gw_h + w_hw_i + w_iw_3) \\ &\qquad = w_1w_2w_3 + w_2(w_c : w_3), \end{aligned} \tag{1}$$

where $w_c : w_3$ is the shorthand notation of the sum of adjacent products from w_c to w_3 in the clockwise direction.

Note that the cost of H_0 of the polygon shown in Fig. 3 is

$$\text{Fan}\,(w_1|w_2, \cdots, w_3) = w_1(w_2 : w_3). \tag{2}$$

The condition for (1) to be less than (2) is

$$\frac{w_2 \cdot (w_c : w_3)}{(w_2 : w_3) - w_2 \cdot w_3} < w_1.$$

Similarly, the condition for the partition in Fig. 3a to be less than H_0 is

$$\frac{w_c \cdot (w_d : w_i)}{(w_c : w_i) - w_c \cdot w_i} < w_1. \tag{3}$$

We say that a partition is said to be *l-optimal* among the partitions in a certain class (or several classes) if it is the lexicographically smallest partition among all the partitions with minimum cost in that class (or several classes). Hence, the l-optimum partition is l-optimal among all partitions in the classes H_0, $H_1, \cdots$, and H_{n-3}.

Now, assume that the l-optimal partition among all the partitions in $H_1, H_2, \cdots, H_{n-3}$ contains only one potential h-arc $V_i - V_k$, as shown in Fig. 4. (Note that $V_i - V_k$ will exist in this partition as an h-arc.) This partition will be the l-optimum partition of the monotone basic n-gon if it costs less than that of the fan in H_0. The condition that the partition with $V_i - V_k$ as the single h-arc costs less than H_0 is

$$\frac{w_i \cdot (w_j : w_k)}{(w_i : w_k) - w_i \cdot w_k} < w_1 \quad \text{if } w_i \leqq w_k$$

or

$$\frac{w_k \cdot (w_i : w_g)}{(w_i : w_k) - w_i \cdot w_k} < w_1 \quad \text{if } w_k < w_i.$$

Combining the two inequalities above, we have

$$\frac{C(w_i, \cdots, w_k)}{(w_i : w_k) - w_i \cdot w_k} < w_1 \tag{4}$$

where $C(w_i, \cdots, w_k)$ denotes the cost of the optimum partition of the subpolygon $w_i - w_j - \cdots - w_g - w_k$ and is equal to the cost of the fan in this case.

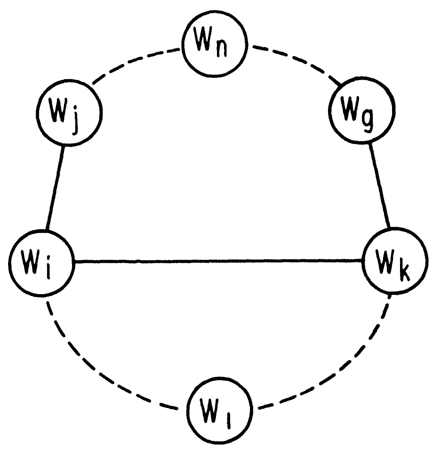

FIG. 4. *A monotone polygon with a single h-arc.*

An h-arc $V_i - V_k$ which divides a polygon into two subpolygons is called a *positive* arc with respect to the polygon if condition (4) is satisfied; i.e., the partition with the arc as the only h-arc and a fan in each of the two subpolygons costs less than the fan in the same polygon. Otherwise, it is called a *negative* arc with respect to the polygon.

When an n-gon is divided into subpolygons, an h-arc is defined as positive in a subpolygon if the cost of partition of the subpolygon with the h-arc as the only h-arc is less than the fan in the subpolygon.

Let us consider a partition with two h-arcs as shown in Fig. 5, and assume that this partition is l-optimal among all partitions in the classes $H_2, H_3, \cdots, H_{n-3}$.

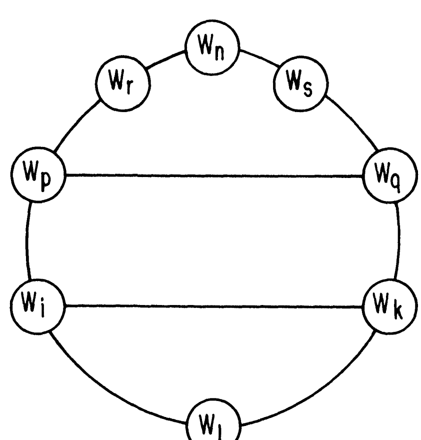

FIG. 5. *A monotone 8-gon with two h-arcs.*

If $V_i - V_k$ is positive with respect to the subpolygon $V_1 - V_i - V_p - V_q - V_k$, then the condition analogous to (4) is

(5a)
$$\frac{C(w_i, w_p, w_q, w_k)}{\{(w_i : w_k) - [(w_p : w_q) - w_p : w_q]\} - w_i \cdot w_k} < w_1.$$

If $V_i - V_k$ is positive with respect to the whole polygon $V_1 - V_i - \cdots - V_n - \cdots - V_k$, then the condition is

$$\text{(5b)} \qquad \frac{C(w_i, w_p, w_r, w_n, w_s, w_q, w_k)}{(w_i : w_k) - w_i \cdot w_k} < w_1.$$

Note that condition (5b) implies (5a).

The condition for the arc $V_p - V_q$ to be positive with respect to the subpolygon $V_i - V_p - V_r - V_n - V_s - V_q - V_k$ is

$$\text{(6a)} \qquad \frac{C(w_p, w_r, w_n, w_s, w_q)}{(w_p : w_q) - w_p \cdot w_q} < \min(w_i, w_k).$$

If the arc $V_p - V_q$ is positive with respect to the whole polygon $V_1 - V_i - V_p - V_r - V_n - V_s - V_q - V_k$, it must satisfy

$$\text{(6b)} \qquad \frac{C(w_p, w_r, w_n, w_s, w_q)}{(w_p : w_q) - w_p \cdot w_q} < w_1.$$

Since $w_1 < \min(w_i, w_k)$, condition (6b) implies (6a).

Here, the presence of $V_i - V_k$ will divide the original polygon into two subpolygons where $V_p - V_q$ appears in the upper subpolygon. If $V_p - V_q$ is a positive arc with respect to the original polygon, then $V_p - V_q$ is certainly positive in the upper subpolygon. But if $V_p - V_q$ is positive in the subpolygon, the arc $V_p - V_q$ may become negative if $V_i - V_k$ is removed; i.e., $V_p - V_q$ becomes negative with respect to the original polygon.

Similarly, if the arc $V_i - V_k$ is positive with respect to a subpolygon, the arc $V_i - V_k$ may become negative if the arc $V_p - V_q$ is removed.

The preceding discussions can be summarized as:

THEOREM 4. *If an h-arc is positive with respect to a polygon then the arc is positive with respect to any subpolygon containing that arc. If an h-arc is positive with respect to a subpolygon, it may or may not be positive with respect to a larger polygon which contains the subpolygon.*

There are two *intuitive* approaches to finding the l-optimum partition of a monotone basic polygon. The first approach is to put in the potential h-arcs one by one. Each additional potential h-arc will improve the cost until the correct number of h-arcs is reached. Any further increase in the number of h-arcs will increase the cost. To introduce an h-arc into the polygon, we can test each potential h-arc (at most $n-3$) to see if it is positive with respect to the whole polygon. If yes, that positive arc must exist in the l-optimum partition, and the polygon will be divided into two subpolygons, each being a monotone polygon. We can repeat the whole process of testing positiveness of the h-arcs. The trouble is that all these arcs may be negative individually with respect to the whole polygon and yet H_0 may not be the optimum. For example, two arcs $V_i - V_j$ and $V_p - V_q$ may be negative individually with respect to the whole polygon, but the partition with both $V_i - V_j$, $V_p - V_q$ present at the same time may cost less than H_0, as shown in Fig. 6a. This shows that we cannot guarantee an optimum partition simply because no more potential h-arcs can be added one at a time.

The second approach is to put all the potential h-arcs in first, and then take out the potential h-arcs one by one, where each deletion will decrease the cost until the correct number of h-arcs is reached. Any further deletions will increase the cost. Unfortunately, even if all h-arcs are positive with respect to their subpolygon, the partition may not be optimum. In Fig. 6b, each h-arc is positive with respect to its

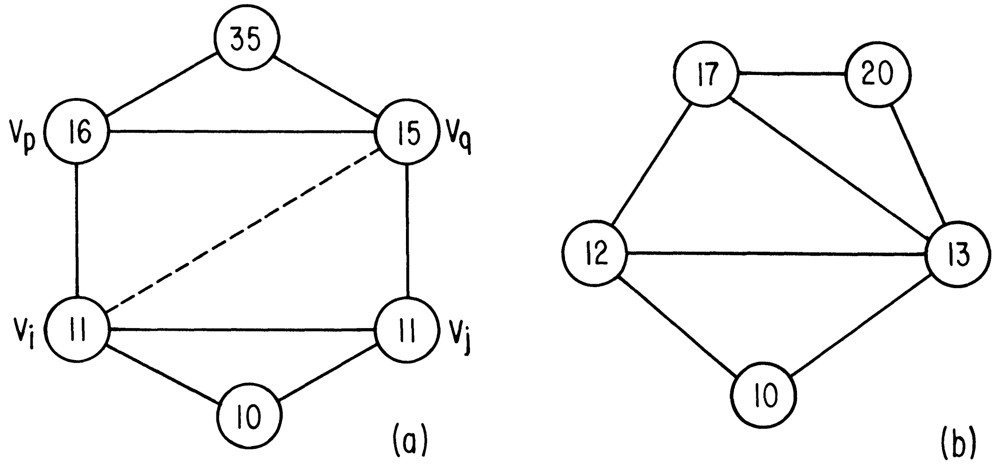

FIG. 6. *Counterexamples for the intuitive approaches.*

local subpolygon, but the partition is not optimum. (Note that positiveness of an h-arc in a quadrilateral is the same as stability. But the idea of stability applies to vertical arcs as well.) This means that we cannot guarantee an optimum partition simply because no h-arc can be deleted one at a time.

Let us outline the idea of an $O(n)$ algorithm for finding the l-optimum partition of a monotone basic polygon. First, we get all the potential h-arcs by the one-sweep algorithm. Then, we start from the highest potential h-arc and process each potential h-arc from the highest to the lowest. For each potential h-arc, we try to get the l-optimum partition of the upper subpolygon above that arc. The l-optimum partition in the subpolygon is obtained by comparing the cost of the l-optimal partition among the partitions of the upper subpolygon which contain one or more potential h-arcs with that of the fan in the upper subpolygon.

If we use the dynamic programming approach to find the l-optimum partition in the upper subpolygon of each potential h-arc, we need $O(n^3)$ operations to find the l-optimum partition of the whole monotone basic n-gon. Fortunately, there are some dependence relationships among these potential h-arcs. Hence, certain subsets of the potential h-arcs will either all exist or all disappear in the l-optimum partition of the monotone polygon. We shall be dealing with potential h-arcs most of the time, so we shall use "arcs" instead of "potential h-arcs" when there is no ambiguity.

Consider the monotone basic polygon shown symbolically in Fig. 7. There are three potential h-arcs, denoted by h_k, h_j and h_i. For any arc h_a, we shall use w_a, w'_a to denote the weights associated with the end vertices of the arc h_a. V_n is the local maximum vertex and V_1 is the local minimum vertex. Without loss of generality, we can assume $w_a \leqq w'_a$ for $a = i, j$ and k. Since we shall deal with subpolygons bounded by two potential h-arcs, let us use h_n for V_n and h_1 for V_1 (i.e., we consider these vertices as degenerated arcs). From Lemmas 1 and 3, the l-optimum partitions of the subpolygons bounded by two potential h-arcs (i.e. the white area of the polygon in Fig. 7) are all fans.

Assume (i) h_k is positive in the subpolygon bounded by h_n and h_j, but h_k is negative in the subpolygon bounded by h_n and h_i;

(ii) h_j is positive in the subpolygon bounded by h_k and h_i, but h_j is negative in the subpolygon bounded by h_k and h_1;

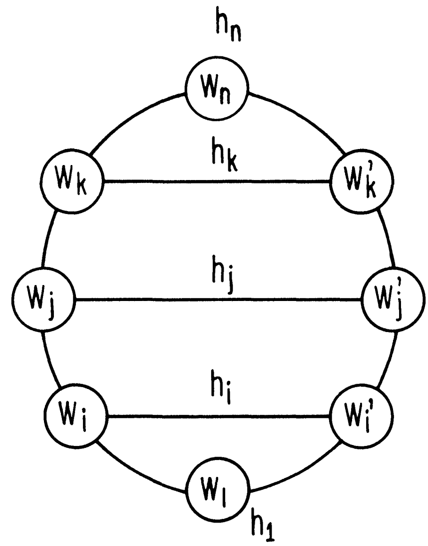

FIG. 7. *An octagon with three potential h-arcs.*

(iii) h_i is positive in the subpolygon bounded by h_j and h_1 only.

Then either the three arcs h_k, h_j, h_i all exist or no h-arcs exists in the optimum partition.

This shows that the existence of an h-arc depends on the existence of another h-arc.

In Fig. 7, the condition for h_k to be positive with respect to the whole polygon is (compare with the condition (5a))

$$\frac{C(w_k, w_n, w'_k)}{(w_k : w'_k) - w_k \cdot w'_k} < w_1. \tag{7}$$

The left-hand side of (7) is denoted by

$$S(h_k \backslash h_n)$$

and is called the *supporting weight* of the arc h_k with respect to the upper subpolygon bounded above by h_n.

The supporting weight of an arc h_k is an indicator of the existence of h_k in a subpolygon. To specify the subpolygon, we have to specify the arc above h_k, e.g. h_n in this case, and an arc below h_k. Once the upper subpolygon of h_k is specified, we can calculate the supporting weight of h_k since the left-hand side of (7) depends only on weights of vertices in the upper subpolygon. To find the arc below h_k which is the lower boundary of the subpolygon, we can use the supporting weight of h_k to test each arc h_i below h_k. (The h_i has two vertices with weights w_i and w'_i.)

If $S(h_k \backslash h_n) < \min(w_i, w'_i)$ then h_k will exist in the subpolygon between h_i and h_n. Otherwise, h_k cannot exist in the subpolygon.

Let h_i, h_j and h_k be three potential h-arcs where h_j lies below h_k and above h_i. Let

$$S(h_i \backslash h_j) = \frac{a}{b} \quad \text{and} \quad S(h_j \backslash h_k) = \frac{c}{d}.$$

Then it follows from the definition of supporting weight that

$$S(h_i \backslash h_k) = \frac{a+c}{b+d}. \tag{8}$$

If $S(h_i\backslash h_j) < S(h_j\backslash h_k)$, we have $S(h_i\backslash h_j) < S(h_i\backslash h_k) < S(h_j\backslash h_k)$. On the other hand, if $S(h_i\backslash h_j) > S(h_j\backslash h_k)$, we have $S(h_i\backslash h_j) > S(h_i\backslash h_k) > S(h_j\backslash h_k)$.

In terms of the supporting weights, we can rewrite the previous conditions (i), (ii) and (iii) as follows:

(i) $w_i < S(h_k\backslash h_n) < w_j$;

(ii) $w_1 < S(h_j\backslash h_k) < w_i$;

(iii) $S(h_i\backslash h_j) < w_1$.

Note that if $S(h_j\backslash h_k) \leqq S(h_k\backslash h_n)$, then it follows from (7) and (8) that $S(h_j\backslash h_k) \leqq S(h_j\backslash h_n) \leqq S(h_k\backslash h_n)$.

Because of conditions (i) and (ii), the l-optimum partition of the subpolygon bounded by h_i and h_n must either be a fan or consist of both h_j and h_k as h-arcs. Hence, in order that both h_j and h_k exist in the l-optimum partition of the subpolygon bounded by h_i and h_n, $S(h_j\backslash h_n)$ must be less than w_i. Suppose $S(h_j\backslash h_n) < w_i$ and $S(h_i\backslash h_j) < w_1$. Then all three arcs h_i, h_j and h_k will exist in the l-optimum partition of the whole polygon if $S(h_i\backslash h_n) < w_1$. If $S(h_i\backslash h_n) \geqq w_1$, then the l-optimum partition will consist of a fan instead.

Define $S(h_n\backslash h_n)$ to be zero. We say that an arc h_k is the *ceiling* of another arc h_i if either condition (i) or conditions (iia), (iib), and (iic) are satisfied:

(i) $h_k = h_n$ if $h_i = h_n$, i.e., h_n is its own ceiling;

or

(ii) a) h_k is above h_i,

b) $S(h_i\backslash h_k) > S(h_k\backslash h_k\text{'s ceiling})$,

c) h_k is the lowest arc which satisfied (iia) and (iib). ("Lowest" means closest to the minimum vertex.)

The ceiling of an arc h_i is the lowest arc (above h_i) which *may* exist in an optimum partition even though h_i does not exist.

We say that an arc h_j is a *son* of another arc h_i if the following conditions are satisfied:

(i) h_j is above h_i (the son is above its father);

(ii) $S(h_j\backslash h_j\text{'s ceiling}) < \min(w_i, w_i')$ where w_i, w_i' are the weights associated to the end vertices of h_i;

(iii) $S(h_i\backslash h_j) \leqq S(h_j\backslash h_j\text{'s ceiling})$; i.e., h_j is *not* a ceiling of h_i;

(iv) h_i is the highest arc which satisfies (i), (ii) and (iii). ("Highest" means closest to the maximum vertex.)

We shall prove in Theorem 6 that:

(i) if the father of any arc h_j exists in the l-optimum partition, then the arc h_j will also exist in the same partition;

(ii) if the father of h_j does not exist in the l-optimum partition, then the arc h_j also does not exist in the same partition.

From the definitions of the ceiling and the father–son relationship, we have the following observations:

(i) Every arc can have *at most* one father but an arc can have many sons. Also, the ancestor–descendant relationship is a transitive relationship. (Note that the ancestor–descendant relationship applies to arcs which are positive with respect to the whole monotone polygon as well.)

(ii) Every arc can have *at most* one ceiling but an arc can be the ceiling of many arcs.

(iii) All the h-arcs in the l-optimum partition of the subpolygon bounded by an arc h_i and its ceiling are descendants of h_j.

(iv) The ceiling of h_j cannot lie below any of the ceilings of h_j's descendants.

In other words, the subpolygon between h_j and its ceiling is nested completely inside the subpolygon bounded by h_j's father and the ceiling of h_j's father. If we treat each subpolygon bounded by an arc h_j and its ceiling as a *block*, then the ancestor–descendant relationship imposes a "nested block structure." For example, if h_k's father is h_j and h_j's father is h_i, then

h_k and its ceiling form the innermost block,

h_j and its ceiling form the middle block, and

h_i and its ceiling form the outermost block.

We shall show that the h-arcs in the l-optimum partition of an inner block exist in the l-optimum partition of the monotone polygon if and only if their ancestors; i.e., the h-arcs, forming the bottoms of the outerblocks, exist.

THEOREM 5. *Let h_j be a potential h-arc. If h_j is present in the l-optimum partition of a monotone polygon, its ceiling h_k will also be present in the l-optimum partition.*

Proof (by contradiction). Suppose there exists an h-arc h_j in the l-optimum partition while its ceiling h_k does not exist in the l-optimum partition. Without loss of generality, we can assume h_j to be the highest arc among those potential h-arcs which are present in the l-optimum partition and violate the theorem. From the definition of supporting weight, i.e. the left-hand side of inequality (7), we have $S(h_j \backslash h_k) < \min(w_j, w_j')$. Let h_c be the lowest h-arc above h_j in the l-optimum partition. The ceiling of h_c must be present in the l-optimum partition and we have $S(h_c \backslash h_c\text{'s ceiling}) < \min(w_j, w_j')$. Since there is no other h-arc between h_j and h_c in the l-optimum partition, the fan is l-optimum in the subpolygon between h_j and h_c. We have the following two cases.

Case 1. If h_c is the ceiling of h_k, we have $S(h_k \backslash h_c) < S(h_j \backslash h_k) < \min(w_j, w_j')$. Hence, the partition with h_k and its descendants as h-arcs costs less than the fan in the subpolygon between h_j and h_c, and we have a contradiction.

Case 2. If h_c is not the ceiling of h_k, we have the following two subcases.

Case 2a. Suppose h_c has a father which lies between h_j and h_c. It follows from the definition of the father–son relationship that $S(h_c\text{'s father} \backslash h_c) \leqq S(h_c \backslash h_c\text{'s ceiling}) < \min(w_j, w_j')$. Hence, the partition with h_c's father and its descendants costs less than the fan in the subpolygon bounded by h_j and h_c, and we have a contradiction.

Case 2b. Now h_c is not the ceiling of h_k and has no ancestor between h_j and h_c. Then among the potential h-arcs which lie between h_j and h_c, there exists a set of arcs $h_d, h_e, \cdots, h_f, h_k$ such that

h_c is the ceiling of h_d,
h_d is the ceiling of h_e,
$\vdots$
h_f is the ceiling of h_k,
h_k is the ceiling of h_j,

and none of these arcs exists in the l-optimum partition. It follows from the definition of a ceiling that

$$S(h_d \backslash h_c) < S(h_e \backslash h_d) < \cdots < S(h_k \backslash h_f) < S(h_j \backslash h_k) < \min(w_j, w_j').$$

Now, the partition with h_d and all its descendants as h-arcs costs less than the fan in the subpolygon bounded by h_j and h_c, and we have a contradiction. In fact, using the same argument, we can show that the arcs $h_d, h_e, \cdots, h_f, h_k$ and all the descendants of these arcs should be in the l-optimum partition of the monotone polygon. □

THEOREM 6. *The sons of an arc h_j will exist in the l-optimum partition of a monotone polygon if and only if h_j is present in the l-optimum partition.*

Proof. (i) Instead of proving the "only if" part of the theorem directly, we will prove, by contradiction, that the existence of any son of h_j implies the existence of h_j in the l-optimum partition.

Among all the potential h-arcs in the monotone polygon, let h_j be the highest arc which is not present in the l-optimum partition of the polygon even though it has one or more sons present in the l-optimum partition. Among all the sons of h_j, let h_k be the lowest son which is present in the l-optimum partition. Finally, among all the potential h-arcs below h_j, let h_i be the highest h-arc which is present in the l-optimum partition. Hence, the l-optimum partition in the subpolygon bounded by h_i and h_k must be a fan. It follows from Theorem 5 that h_k's ceiling also exists in the l-optimum partition and we have $S(h_k \backslash h_k\text{'s ceiling}) < \min(w_i, w_i')$. Otherwise, the l-optimum partition in the subpolygon bounded by h_i and h_k's ceiling should be a fan and h_k as well as its descendants cannot be present in the l-optimum partition. From the definition of the father–son relationship, we know that $S(h_j \backslash h_k) \leqq S(h_k \backslash h_k\text{'s ceiling}) < \min(w_i, w_i')$. This means that in the subpolygon bounded by h_i and h_k, the partition consisting of h_j and its descendants as h-arcs costs less than the fan. This contradicts our assumption that the fan is l-optimum in the subpolygon bounded by h_i and h_k.

(ii) We shall prove the "if" part of the theorem directly by contradiction. Among all the potential h-arcs in the monotone polygon, let h_k be the highest arc which is not present in the l-optimum partition of the polygon even though its father h_j is present in the l-optimum partition. Among all the potential h-arcs present in the l-optimum partition, let h_c be the lowest h-arc above h_k and let h_b be the highest h-arc below h_k in the l-optimum partition as shown in Fig. 8. Hence, the l-optimum partition in the subpolygon bounded by h_b and h_c must be a fan. Note that h_c must be a ceiling of h_k because h_k is the highest arc not satisfying the necessary condition of the theorem. Otherwise, h_c is a descendant of h_k, and by part (i) of this proof, h_k will exist in the l-optimum partition of the polygon. The arc h_b must either be h_j itself or lie above h_j. Hence, we have $\min(w_b, w_b') \geqq \min(w_j, w_j')$. By the definition of the father–son

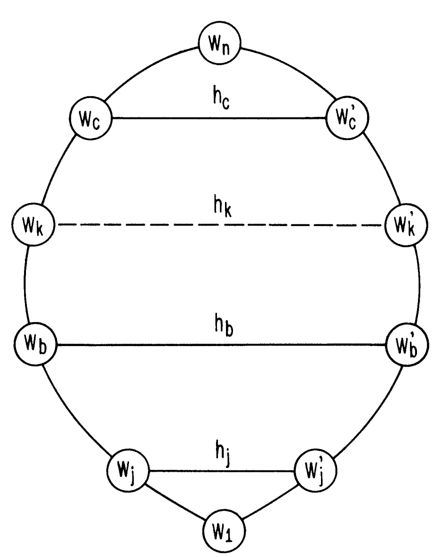

FIG. 8

relationship, we have $S(h_k\backslash h_c) < \min(w_j, w_j') \leqq \min(w_b, w_b')$. This means that in the subpolygon bounded by h_b and h_c, the partition consisting of h_k and its descendants is cheaper than the fan. This contradicts our assumption that the fan is l-optimum in the subpolygon bounded by h_b and h_c. □

COROLLARY 1. *The descendants of any arc h_j will exist in the l-optimum partition of a monotone polygon if and only if h_j exists in the l-optimum partitions.*

Proof. The corollary follows from Theorem 6. □

It follows from Corollary 1 that if a potential h-arc h_j is present in the l-optimum partition of a monotone polygon, all its descendants, all its ancestors and all potential h-arcs which have some ancestors common to those of h_j will be present in the l-optimum partition.

THEOREM 7. *Let h_i and h_j be two potential h-arcs such that h_j is above h_i and the l-optimum partition in the subpolygon bounded by h_i and h_j is a fan. If $S(h_j\backslash h_j$'s ceiling$) \geqq \min(w_i, w_i')$, then h_j and all its descendants cannot exist in the l-optimum partition of any subpolygon bounded above by h_n and below by any potential h-arc not higher than h_i.*

Proof (by contradiction). Assume that there exist such two potential h-arcs but that h_j is present in the l-optimum partition of a subpolygon bounded above by h_n and below by a potential h-arc lower than h_i. Without loss of generality, let h_j be the lowest arc among all the potential h-arcs which are present in the l-optimum partition and which satisfy the assumption. Hence, none of the potential h-arcs between h_i and h_j can exist in the l-optimum partition. Let h_b be the highest potential h-arc below h_j in the l-optimum partition. Since h_b can either be h_i itself or a potential h-arc below h_i, we have $\min(w_b, w_b') \leqq \min(w_i, w_i') \leqq S(h_j\backslash h_j$'s ceiling). The partition with h_j and all its descendants costs more than the fan in the subpolygon bounded by h_b and h_j's ceiling and we have a contradiction. □

Using Theorem 6, we can start from an innermost block and work our way out. Suppose we have located the ceiling of a potential h-arc h_i. Then we can treat h_i and all the sons (and descendants) of h_i as a unit; i.e., all h_i's sons are *condensed* into h_i. Let h_b be the potential h-arc immediately below h_i in the monotone polygon. The l-optimum partition in the subpolygon bounded by h_b and the ceiling of h_i must consist of either h_i and all its descendants as h-arcs or of a fan, depending on whether $S(h_i\backslash h_i$'s ceiling$) < \min(w_b, w_b')$ or $S(h_i\backslash h_i$'s ceiling$) \geqq \min(w_b, w_b')$. If the fan is cheaper, we can delete h_i and all its descendants since none of these arcs can appear as h-arcs in the l-optimum partition of the polygon (Theorem 7).

Now, what we have to do is to find an innermost block to start our computations. After obtaining the list of potential h-arcs of the monotone polygon using the one-sweep algorithm, we know that the degenerated arc h_n is the ceiling of the highest potential h-arc in the list, and this potential h-arc does not have any descendants. So, we should start from the highest potential h-arc and work our way down the list of potential h-arcs.

We now give two examples to illustrate the concepts, notation and algorithm. Then a formal description of the algorithm will be given.

Consider a monotone basic polygon with five potential h-arcs, $h_6, h_5, \cdots, h_2$ where h_6 is the highest arc as shown symbolically in Fig. 9. Let $w_i \leqq w_i'$ for $i = 2, 3, \cdots$. The maximum vertex, which lies above h_6, has the weight w_7 and the minimum vertex, which lies below h_2, has the weight w_1. We can regard w_7 (and w_1) as a degenerated arc and use h_7 to represent w_7 (and h_1 to represent w_1).

Example 1. There are two possible candidates for the l-optimum partition in the subpolygon bounded by h_5 and h_7. We shall use $C(\underline{h_5}, h_6, \overline{h_7})$ to denote the cost

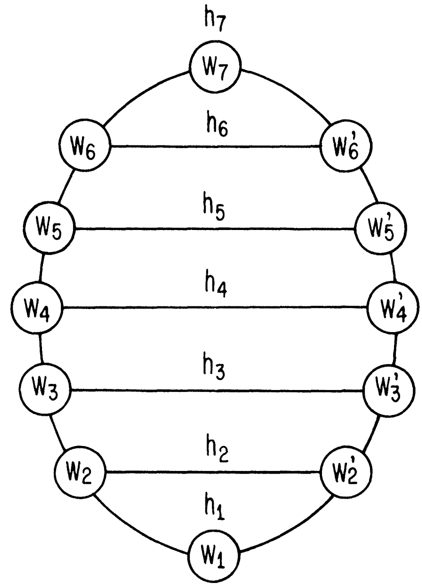

FIG. 9. *A 12-gon with 5 h-arcs.*

of the partition with h_6, and $H_0(\underline{h_5}, \overline{h_7})$ to denote the cost of the fan in the subpolygon. Similarly, we shall use $C(\underline{h_2}, h_5, h_6, \overline{h_7})$ to denote the cost of the partition with h_5 and h_6 as the only 2 h-arcs in the subpolygon bounded by h_2 and h_7. Note that there is a bar underneath the h-arc which forms the bottom of the subpolygon and a bar above the h-arc which forms the top of the subpolygon.

The necessary computations and results of the comparisons are shown in Table 1.

If $S(h_2\backslash h_7) < w_1$, the partition with h_2, h_3, h_4, h_5 and h_6 as h-arcs will be l-optimum in the polygon. Otherwise, the fan $H_0(\underline{h_1}, \overline{h_7})$ will be l-optimum.

Now, let us consider a more complicated example.

Example 2. Consider the 6 potential h-arcs shown in Fig. 9. Assume that we have the computations and results shown in Table 2.

If $S(h_2\backslash h_7) < w_1$, the partition with h_2, h_5 and h_6 as h-arcs is l-optimum. Otherwise, the fan $H_0(\underline{h_1}, \overline{h_7})$ will be l-optimum.

Let us give the algorithm for finding the l-optimum partition of a monotone basic polygon.

ALGORITHM M

(I) Get all the potential h-arcs of the polygon by the one-sweep algorithm [6]. (All these arcs form a vertical list, with the highest arc closest to the maximum vertex V_n and the lowest arc closest to the minimum vertex V_1.)

(II) Process the potential h-arcs one by one, from the top to the bottom. Let h_j be the potential h-arc being processed, let h_k be the potential h-arc immediately above h_j, and let h_i be the potential h-arc immediately below h_j in the monotone polygon. (If h_j is the highest potential h-arc in the polygon, h_k will be the degenerate arc h_n; if h_j is the lowest potential h-arc in the polygon, h_i will be the degenerated arc h_1.) Note that by the time we start processing h_j, we have already obtained the l-optimum partition of the subpolygon between h_j and h_n. We have also located the ceilings of every h-arc in the l-optimum partition of this subpolygon. When we process h_j, we

TABLE 1

Computations	Observations	Remarks
1. $S(h_6\backslash h_7)$	$w_4 < S(h_6\backslash h_7) < w_5$	h_7 is the ceiling of h_6: $S(h_6\backslash h_7) < w_5 \Rightarrow C(\underline{h_5}, h_6, \overline{h_7}) < H_0(\underline{h_5}, \overline{h_7})$
2. $S(h_5\backslash h_6)$	$w_3 < S(h_5\backslash h_6) < w_4$	$S(h_5\backslash h_6) < S(h_6\backslash h_7) \Rightarrow h_6$ is a son of h_5; condense h_6 into h_5 and calculate $S(h_5\backslash h_7)$
3. $S(h_5\backslash h_7)$	$w_3 < S(h_5\backslash h_7) < w_4$	h_7 is the ceiling of h_5; $S(h_5\backslash h_7) < w_4 \Rightarrow C(\underline{h_4}, h_5, h_6, \overline{h_7}) < H_0(\underline{h_4}, \overline{h_7})$.
4. $S(h_4\backslash h_5)$	$w_2 < S(h_4\backslash h_5) < w_3$	$S(h_4\backslash h_5) < S(h_5\backslash h_7) \Rightarrow h_5$ is a son of h_4; condense h_5 into h_4 and calculate $S(h_4\backslash h_7)$
5. $S(h_4\backslash h_7)$	$w_2 < S(h_4\backslash h_7) < w_3$	h_7 is the ceiling of h_4; $S(h_4\backslash h_7) < w_3 \Rightarrow C(\underline{h_3}, h_4, h_5, h_6, \overline{h_7}) < H_0(\underline{h_3}, \overline{h_7})$
6. $S(h_3\backslash h_4)$	$w_1 < S(h_3\backslash h_4) < w_2$	$S(h_3\backslash h_4) < S(h_4\backslash h_7) \Rightarrow h_4$ is a son of h_3; condense h_4 into h_3 and calculate $S(h_3\backslash h_7)$
7. $S(h_3\backslash h_7)$	$w_1 < S(h_3\backslash h_7) < w_2$	h_7 is the ceiling of h_3; $S(h_3\backslash h_7) < w_2 \Rightarrow C(\underline{h_2}, h_3, h_4, h_5, h_6, \overline{h_7}) < H_0(\underline{h_2}, \overline{h_7})$
8. $S(h_2\backslash h_3)$	$S(h_2\backslash h_3) < w_1$	$S(h_2\backslash h_3) < S(h_3\backslash h_7) \Rightarrow h_3$ is a son of h_2; condense h_3 into h_2 and calculate $S(h_2\backslash h_7)$
9. $S(h_2\backslash h_7)$	?	

TABLE 2

Computations	Observations	Remarks
1. $S(h_6\backslash h_7)$	$w_1 < S(h_6\backslash h_7) < w_2$	h_7 is the ceiling of h_6; $S(h_6\backslash h_7) < w_5 \Rightarrow C(\underline{h_5}, h_6, \overline{h_7}) < H_0(\underline{h_5}, \overline{h_7})$
2. $S(h_5\backslash h_6)$	$S(h_6\backslash h_7) < S(h_5\backslash h_6) < w_2$	$S(h_5\backslash h_6) > S(h_6\backslash h_7) \Rightarrow h_6$ is the ceiling of h_5; $S(h_5\backslash h_6) < w_4 \Rightarrow C(\underline{h_4}, h_5, \overline{h_6}) < H_0(\underline{h_4}, \overline{h_6})$
3. $S(h_4\backslash h_5)$	$w_2 < S(h_4\backslash h_5) < w_3$	$S(h_4\backslash h_5) > S(h_5\backslash h_6) \Rightarrow h_5$ is the ceiling of h_4; $S(h_4\backslash h_5) < w_3 \Rightarrow C(\underline{h_3}, h_4, \overline{h_5}) < H_0(\underline{h_3}, \overline{h_5})$
4. $S(h_3\backslash h_4)$	$w_1 < S(h_3\backslash h_4) < w_2$	$S(h_3\backslash h_4) < S(h_4\backslash h_5) \Rightarrow h_4$ is a son of h_3; condense h_4 into h_3 and calculate $S(h_3\backslash h_5)$
5. $S(h_3\backslash h_5)$	$w_2 < S(h_3\backslash h_5) < w_3$	$S(h_3\backslash h_5) > S(h_5\backslash h_6) \Rightarrow h_5$ is the ceiling of h_3; $S(h_3\backslash h_5) > w_2 \Rightarrow C(\underline{h_2}, h_3, h_4, \overline{h_5}) > H_0(\underline{h_2}, \overline{h_5})$; both h_3 and h_4 cannot exist in the l-optimum partition and should be deleted from the list of potential h-arcs; we should then check to see if the fan is cheaper in the subpolygon bounded by h_2 and h_6; $S(h_5\backslash h_6) < w_2 \Rightarrow C(\underline{h_2}, h_5, \overline{h_6}) < H_0(\underline{h_2}, \overline{h_6})$
6. $S(h_2\backslash h_5)$	$S(h_2\backslash h_5) < w_1$	$S(h_2\backslash h_5) < S(h_5\backslash h_6) \Rightarrow h_5$ is a son of h_2; we should condense h_5 into h_2 and calculate $S(h_2\backslash h_6)$
7. $S(h_2\backslash h_6)$	$S(h_2\backslash h_6) < w_1$	$S(h_2\backslash h_6) < S(h_6\backslash h_7) \Rightarrow h_6$ is a son of h_2; we should condense h_6 into h_2 and calculate $S(h_2\backslash h_7)$.
8. $S(h_2\backslash h_7)$	?	

first locate the ceiling of h_j and condense all h_j's descendants into h_j. Then we obtain the l-optimum partition of the subpolygon between h_i and h_n by deleting those blocks of arcs which cannot exist in the l-optimum partition of the subpolygon between h_i and h_n.

While ($h_j \neq$ the degenerated arc h_1) do
Begin
1. [To locate the ceiling of h_j].
While $S(h_j \backslash h_k) \leqq S(h_k \backslash h_k\text{'s ceiling})$ do
Begin
a. Comment: Now, h_k is a son of h_j.
b. We will combine h_k and all its descendants into h_j and calculate the combined supporting weight $S(h_j \backslash h_k\text{'s ceiling})$.
c. Replace h_k by h_k's ceiling; i.e., h_k is always used to denote the lowest h-arc above h_j which is not yet combined into h_j.
End.
2. [To delete those blocks of arcs which cannot exist in the l-optimum partition of the subpolygon between h_i and h_n].
While $C(\underline{h_i}, h_j \text{ and } h_j\text{'s descendants}, \overline{h_j\text{'s ceiling}}) \geqq H_0(\underline{h_i}, \overline{h_j\text{'s ceiling}})$;
i.e., $S(h_j \backslash h_j\text{'s ceiling}) \geqq \min(w_i, w_i')$. Do
Begin
a. Delete h_j and all its descendants from the list of potential h-arcs.
b. Replace h_j by the ceiling of h_j; i.e., h_j is always used to denote the arc immediately above h_i in the subpolygon between h_i and h_n.
End.
3. [Prepare to process next arc].
Replace h_k by h_j, h_j by h_i and h_i by the arc immediately below h_i in the list of potential h-arcs.
End.

(III) Output the l-optimum partition consisting of the arcs which remain in the list of potential h-arcs after Step II as h-arcs.
Then stop.

THEOREM 8. *The partition produced by Algorithm* M *is l-optimum.*

Proof. We have shown in Part I of this paper [6] that all h-arcs present in the l-optimum partition of the polygon are potential h-arcs, and all potential h-arcs are included in the list obtained by the one-sweep algorithm. We claim that (i) whenever Algorithm M finishes Step II.1, the ceiling of h_j is correctly located, (ii) whenever Algorithm M finishes Step II.2, the arcs which have been deleted by Algorithm M cannot exist in the l-optimum partition of the subpolygon bounded above by h_n and below by an arc lower than h_i, and (iii) the partition consisting of all the potential h-arcs remaining above h_i as h-arcs is l-optimum in the subpolygon bounded by h_i and h_n after Step II.2. (If the claim is true, the partition output by Algorithm M will be l-optimum in the monotone polygon.)

We shall prove the claim by induction on the number of h-arcs above an arc h_j.

It is easy to see that the claim is true when $h_j =$ the highest arc in the list of potential h-arcs.

Suppose the claim is true for all potential h-arcs above some arc h_j. Let h_i be the arc immediately below h_j in the list of potential h-arcs. Just before Algorithm M starts processing h_j, all the potential h-arcs which remain above h_j exist as h-arcs in the

l-optimum partition of the subpolygon between h_j and h_n. We can divide these arcs into two groups: (i) those which are descendants of some other arcs in the subpolygon, and (ii) those which have no ancestor in the subpolygon.

It follows from the definition of the father–son relationship that only arcs in group (ii) can be sons of h_j. Let the set of arcs in group (ii) be $h_t, h_{t-1}, \cdots, h_p, h_{p-1}, \cdots, h_{j+2}, h_{j+1}$ such that h_n is above h_t, h_t is above $h_{t-1}, \cdots, h_p$ is above $h_{p-1}, \cdots, h_{j+2}$ is above h_{j+1} and h_{j+1} is above h_j. Note that there exists no other h-arc between h_{j+1} and h_j in the l-optimum partition of the subpolygon. Since none of these arcs has an ancestor in the subpolygon, we must have

$$\begin{aligned} &h_n \text{ as the ceiling of } h_t,\\ &h_t \text{ as the ceiling of } h_{t-1},\\ &\vdots\\ &h_p \text{ as the ceiling of } h_{p-1},\\ &\vdots\\ &h_{j+2} \text{ as the ceiling of } h_{j+1}. \end{aligned}$$

It follows from the definition of the ceiling that

$$S(h_{j+1}\backslash h_{j+2}) > \cdots > S(h_{p-1}\backslash h_p) > \cdots > S(h_{t-1}\backslash h_t) > S(h_t\backslash h_n).$$

Since h_{j+1} is the lowest h-arc in the l-optimum partition of the subpolygon bounded by h_j and h_n, we have

$$\min(w_j, w'_j) > S(h_{j+1}\backslash h_{j+2}) > \cdots > S(h_t\backslash h_n).$$

Now, if $S(h_j\backslash h_{j+1}) \leqq S(h_{j+1}\backslash h_{j+2})$, all four conditions of the father–son relationship are satisfied and Algorithm M will correctly condense h_{j+1} and its descendants into h_j. Using the same argument repeatedly, we conclude that Algorithm M correctly locates the ceiling of h_j at the end of Step II.1. Whenever the potential h-arc h_j and its descendants are removed in Step II.2, the conditions in Theorem 7 are satisfied. Hence h_j and its descendants cannot exist in the l-optimum partition of any subpolygon bounded above by h_n and below by a potential h-arc lower than h_i. Now, at the end of Step II.2, we can again divide the potential h-arcs remaining above h_i into two groups:

(i) those which are descendants of some other arcs in the subpolygon, and
(ii) those which have no ancestor in the subpolygon.

Let h_j be the h-arc immediately above h_i after Step II.2. The arc h_j must be the lowest arc in group (ii). It follows from the definition of ceiling that for any arc h_k above h_j in group (ii), we have

$$\min(w_i, w'_i) > S(h_j\backslash h_j\text{'s ceiling}) > S(h_k\backslash h_k\text{'s ceiling}).$$

From Theorem 6, if any of the arcs in group (ii) does not exist in the l-optimum partition, all its descendants in group (i) will not exist in the l-optimum partition. Suppose the partition consisting of all the potential h-arcs remaining above h_i as h-arcs is not l-optimum in the subpolygon between h_i and h_n. Then some of these potential h-arcs in group (ii) and their descendants should not exist in the l-optimum partition. Assume that h_k is the highest potential h-arc remaining above h_i after Step II.2, but h_k should not exist in the l-optimum partition. Let h_b be the highest h-arc below h_k in the l-optimum partition. Hence, the fan should be l-optimum in the subpolygon between h_b and h_k's ceiling. Since $S(h_k\backslash h_k\text{'s ceiling}) < \min(w_i, w'_i) \leqq \min(w_b, w'_b)$, the partition with h_k and its descendants as h-arcs in the subpolygon bounded by h_b and h_k's ceiling is always cheaper than the fan, and we have a contradiction.

Hence, the claim is true, and the partition output by Algorithm M is l-optimum. □

In order for Algorithm M to run efficiently, we need a data structure which enables us to calculate the supporting weights, to keep track of the ceiling of each potential h-arc and to update the list of potential h-arcs easily. One way to implement Algorithm M is to place all potential h-arcs obtained in Step I in a linear linked list, with the highest arc at the head of the list and the lowest arc at the tail of the list. Each of these potential h-arcs, say h_i, is associated with a record variable with the following fields:

(i) the label of the end vertex which is closer to V_1 in the clockwise direction;
(ii) the label of the other end vertex;
(iii) the ceiling of h_i;
(iv) the list of sons of h_i;
(v) the cost of the l-optimum partition in the subpolygon between h_i and its ceiling, i.e. the numerator of $S(h_i \backslash h_i$'s ceiling);
(vi) the quantity $(w_i : w_j + w_j \cdot w'_j + w'_j : w'_i) - w_i \cdot w'_i$ where w_i, w'_i are weights of the end vertices of the potential h-arc h_i and w_j, w'_j are the weights of the end vertices of h_i's ceiling, i.e. the denominator of $S(h_i \backslash h_i$'s ceiling) (it is obtained by subtracting the product $w_i \cdot w'_i$ from the sum of the adjacent products from w_i to w'_i around the subpolygon $w_i - \cdots - w_j - w'_j - \cdots - w'_i$); and
(vii) the supporting weight $S(h_i \backslash h_i$'s ceiling).

Note that only the first three fields of each potential h-arc are defined at the end of Step I, the other four fields of each potential h-arc are set to the correct value when the potential h-arc is being processed in Step II. Since the sums of adjacent products of the form $w_i : w_j$ are used repeatedly in calculating the cost of the fan between two adjacent potential h-arcs and the denominators of the supporting weights, we can eliminate a lot of repeated calculations by initializing the elements of an array CP to

$$\mathrm{CP}[1] = 0 \quad \text{and} \quad \mathrm{CP}[i] = w_1 : w_i \quad \text{for } 2 \leqq i \leqq n.$$

Then the sum of the adjacent products $w_i : w_j$ can be obtained from $\mathrm{CP}[j] - \mathrm{CP}[i]$.

As we process the arcs in the list of potential h-arcs one by one from the top to the bottom, we shall remove a potential h-arc from the list if (i) the arc is found to be a son of another potential h-arc in Step II.1, or (ii) the partition with the arc and all its descendants is not l-optimum in some subpolygon in Step II.2. Let h_k be an arc which is removed from the list in Step II.1 and let h_j be its father. After h_k is removed from the list of potential h-arcs, it will be added to the list of h_j's sons, i.e. the fourth field of h_j. Then, we have to calculate the supporting weight $S(h_j \backslash h_k$'s ceiling). The numerator of $S(h_j \backslash h_k$'s ceiling) can be obtained by adding the numerator of $S(h_k \backslash h_k$'s ceiling) in the fifth field of h_k to the numerator of $S(h_i \backslash h_j)$. Similarly, the denominator of $S(h_j \backslash h_k$'s ceiling) can be obtained by adding the denominator of $S(h_k \backslash h_k$'s ceiling) in the sixth field of h_k to the denominator of $S(h_j \backslash h_k)$. Hence, we can calculate $S(h_j \backslash h_k$'s ceiling) in a constant amount of time. Note that whenever Algorithm M finishes Step II.2, only those potential h-arcs which are present in the l-optimum partition of the subpolygon between h_i and h_n and yet have no ancestors above h_i remain above h_i in the list of potential h-arcs.

THEOREM 9. *Algorithm* M *runs in* $O(n)$ *time*.

Proof. It takes $O(n)$ time to sweep around the monotone polygon twice, once to obtain all potential h-arcs in Step I and once to initialize the array CP. There are two while loops in Step II, and it only takes a constant amount of time to execute either while loop once. Whenever the while loop in Step II.1 is executed once, a potential h-arc is removed from the list and condensed into its father. Whenever the while loop in Step II.2 is executed once, a potential h-arc is deleted from the list. Once an arc is removed or deleted from the list, it will never be considered again. Since there are at most $n-3$ arcs in the list obtained in Step I, Algorithm M can execute both while loops at most $n-3$ times. So is takes $O(n)$ time to process all the potential h-arcs in Step II and to output the l-optimum partition in Step III. Hence, Algorithm M runs in $O(n)$ time. □

3. The convex polygon. In this section we shall extend the results in § 2 to the case of a general convex polygon.

There may be several local maximum vertices in a general convex polygon. Let us still draw the polygon in such a way that the global minimum vertex is at the bottom. From Theorem 4 of Part I, we know that all potential h-arcs are still compatible in a general convex polygon. However, unlike those in a monotone polygon, the potential h-arcs no longer form a linear list. Instead, they form a tree, called an *arc-tree*. In Fig. 10a, there is a 12-gon with 6 potential h-arcs, and they are labelled as h_2, h_3, h_4, h_5, h_6 and h_7. (Note that we also obtain $V_4 - V_3$, $V_7 - V_6$ and $V_6 - V_8$ from the one-sweep algorithm. In order to have a simpler example, let us assume that all three of these arcs are unstable and hence are not shown in Fig. 10a.) To get a better feeling of the arc-tree, we can redraw the 12-gon as shown in Fig. 10b. By regarding V_1 as a degenerated arc h_1, V_{12} as a degenerated arc h_8, and V_{11} as a degenerated arc h_9, we have h_1 as the root of the arc tree and the arcs h_8 and h_9 as the leaves.

An arc h_j is *above* another arc h_i (and h_i is *below* h_j) if h_j is in one of the subtrees of h_i. We shall be dealing with subpolygons, each bounded below by a potential h-arc and above by a set of potential h-arcs. We can define the supporting weights of the potential h-arcs in a similar way. For example, the supporting weight of the arc h_2

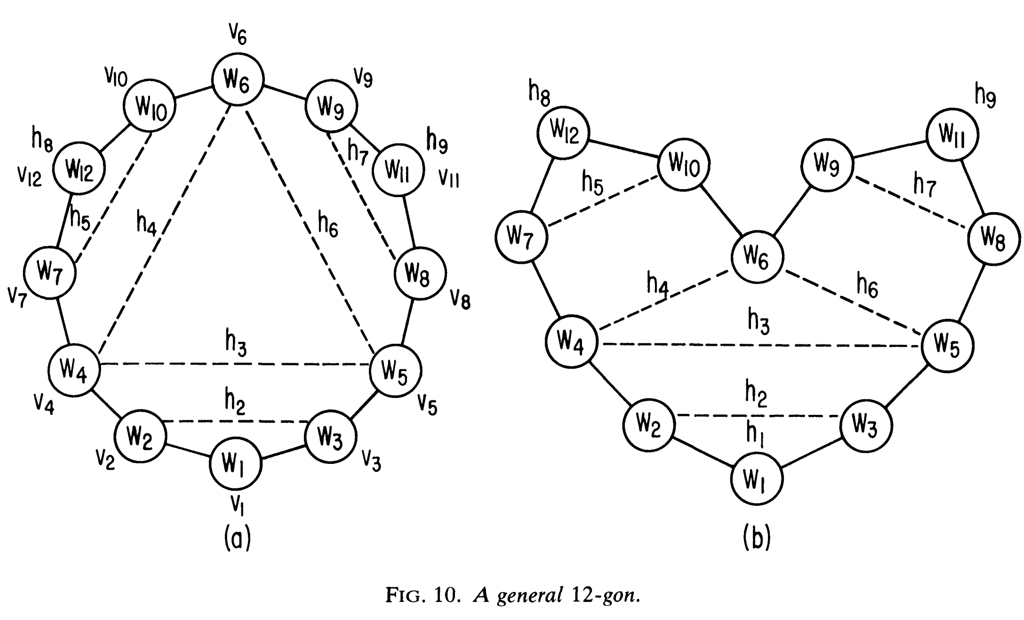

FIG. 10. *A general 12-gon.*

with respect to the subpolygon bounded above by $\{h_4, h_6\}$ in Fig. 10b equals

$$\frac{C(w_2, w_4, w_6, w_5, w_3)}{[w_2 : w_3 - (w_4 : w_6 - w_4 \cdot w_6) - (w_6 : w_5 - w_6 \cdot w_5)] - w_2 \cdot w_3}$$

and is denoted by $S(h_2 \backslash \{h_4, h_6\})$. Again, for any leaf node h_n, we define $S(h_n \backslash \{h_n\})$ to be zero.

We say that a set of potential h-arcs U_i is the *ceiling* of another potential h-arc h_i (or simply h_i's ceiling for short) if either condition (i) or conditions (iia), (iib), (iic) and (iid) are satisfied:

(i) $U_i = \{h_i\}$ if h_i is a leaf node;

or

(ii) for all $h_k \in U_i$,
 a) h_k is above h_i;
 b) $S(h_i \backslash U_i) > (h_k \backslash h_k$'s ceiling);
 c) for all $h_j \in U_i$ such that $h_j \neq h_k$, neither h_j is above h_k nor h_k is above h_j; and
 d) conditions (iia), (iib) or (iic) will be violated if h_k is replaced by any arc below h_k in the subpolygon between h_i and U_i.

We say that an arc h_j is a *son* of another arc h_i if the following conditions are satisfied:

(i) h_j is above h_i (the son is above its father);
(ii) $S(h_j \backslash h_j$'s ceiling$) < \min(w_i, w_i')$ where w_i, w_i' are weights associated to the end vertices of h_i;
(iii) h_j is *not* in the ceiling of h_i; and
(iv) h_i is the highest arc which satisfies (i), (ii) and (iii).

It is easy to see that all the previous discussions on the ceilings and the ancestor–descendant relationships in § 2 still hold under the new definition of ceilings and father–son relationships. Using arguments similar to those used in the proofs of Theorems 5, 6 and 7, we can generalize Theorems 5, 6, 7 and Corollary 1 as follows:

THEOREM 10. *If a potential h-arc h_j exists in the l-optimum partition of a convex polygon, all potential h-arcs in its ceiling will also exist in the l-optimum partition.*

Proof. Omitted. □

THEOREM 11. *The sons of an arc h_j will exist in the l-optimum partition of a convex polygon if and only if h_j is present in the l-optimum partition.*

Proof. Omitted. □

COROLLARY 2. *The descendants of an arc h_j will exist in the l-optimum partition of a convex polygon if and only if h_j exists in the l-optimum partition.*

Proof. The corollary follows from Theorem 11. □

THEOREM 12. *Let X be a set of potential h-arcs above another potential h-arc h_i such that* (i) *for any two arcs $h_j, h_k \in X$, neither h_j is above h_k nor h_k is above h_j if $h_j \neq h_k$, and* (ii) *the l-optimum partition in the subpolygon between h_i and the arcs in X is a fan. Let h_j be a potential h-arc in X such that for any $h_k \in X$, $S(h_j \backslash h_j$'s ceiling$) \geqq S(h_k \backslash h_k$'s ceiling). If $S(h_j \backslash h_j$'s ceiling$) \geqq \min(w_i, w_i')$ where w_i, w_i' are the weights associated with the end vertices of h_i, then h_j and all its descendants cannot exist in the l-optimum partition of any upper subpolygon bounded below by a potential h-arc no higher than h_i.*

Using the facts in Theorems 10, 11, 12 and Corollary 2, we can again start from the potential h-arcs which lie immediately below the leaf nodes and work our way down. The leaf nodes are the ceiling of these arcs. Before we can locate the ceiling of any arc which does not lie immediately below the leaf nodes, we must first process all the arcs above it, i.e. the arcs in its subtrees. Hence, we can do a postorder traversal through the arc tree. When we process a potential h-arc, we first find the l-optimum partition of the subpolygon bounded below by the arc and above by the leaf nodes in its subtrees; then we will locate the ceiling of the potential h-arc. Let us consider the following example.

Example 3. Consider the 12-gon with six potential h-arcs as shown in Figs. 10a and 10b. The necessary computations and the results of comparisons are shown in Table 3.

TABLE 3

Computations	Observations	Remarks
1. $S(h_5\backslash\{h_8\})$	$w_1 < S(h_5\backslash\{h_8\}) < w_2$	The fan is l-optimum in the subpolygon between h_5 and h_8; $\{h_8\}$ is the ceiling of h_5 h_4 is the next arc to be processed.
2. $S(h_4\backslash\{h_5\})$	$w_3 < S(h_4\backslash\{h_5\}) < w_4$	$S(h_5\backslash\{h_8\}) < w_4 \Rightarrow C(\underline{h_4}, h_5, \overline{h_8}) < H_0(\underline{h_4}, \overline{h_8})$ $S(h_4\backslash\{h_5\}) > S(h_5\backslash\{h_8\}) \Rightarrow \{h_5\}$ is the ceiling of h_4 Before we can process h_3, we have to process h_7 first
3. $S(h_7\backslash\{h_9\})$	$w_2 < S(h_7\backslash\{h_9\}) < w_3$	The fan is l-optimum in the subpolygon between h_7 and h_9 $\{h_9\}$ is the ceiling of h_7 h_6 is the next arc to be processed
4. $S(h_6\backslash\{h_7\})$	$w_3 < S(h_6\backslash\{h_7\})$ $< S(h_4\backslash\{h_5\}) < w_4$	$S(h_7\backslash\{h_9\}) < w_5 \Rightarrow C(\underline{h_6}, h_7, \overline{h_9}) < H_0(\underline{h_6}, \overline{h_9})$ $S(h_6\backslash\{h_7\}) > S(h_7\backslash\{h_9\}) \Rightarrow \{h_7\}$ is the ceiling of h_6 h_3 is the next arc to be processed
5. $S(h_3\backslash\{h_4, h_6\})$	$w_2 < S(h_3\backslash\{h_4, h_6\})$ $< S(h_6\backslash\{h_7\})$ $< S(h_4\backslash\{h_5\})$	$S(h_6\backslash\{h_7\}) < S(h_4\backslash\{h_5\}) < w_4$ $\Rightarrow C(\underline{h_3}, h_4, h_6, \overline{h_5}, \overline{h_7}) < H_0(\underline{h_3}, \overline{h_5}, \overline{h_7})$ Both h_4 and h_6 may be sons of h_3 since $S(h_4\backslash\{h_5\}) > S(h_6\backslash\{h_7\})$, test h_4 first to see if h_4 is a son of h_3 $S(h_3\backslash\{h_4, h_6\}) < S(h_4\backslash\{h_5\}) \Rightarrow h_4$ is a son of h_3 Condense h_4 into h_3 and calculate $S(h_3\backslash\{h_5, h_6\})$
6. $S(h_3\backslash\{h_5, h_6\})$	$w_2 < S(h_3\backslash\{h_5, h_6\})$ $< S(h_6\backslash\{h_7\})$	$S(h_3\backslash\{h_5, h_6\}) < S(h_6 \Rightarrow \{h_7\})\backslash h_6$ is a son of h_3 Condense h_6 into h_3 and calculate $S(h_3\backslash\{h_5, h_7\})$.
7. $S(h_3\backslash\{h_5, h_7\})$	$S(h_5\backslash\{h_8\})$ $< w_2 < S(h_7\backslash\{h_9\})$ $< S(h_3\backslash\{h_5, h_7\}) < w_3$	$S(h_3\backslash\{h_5, h_7\}) > S(h_7\backslash\{h_9\}) > S(h_5\backslash\{h_8\})$ $\Rightarrow \{h_5, h_7\}$ is the ceiling of h_3 h_2 is the next arc to be processed.
8. $S(h_2\backslash\{h_5, h_9\})$	$S(h_2\backslash\{h_5, h_9\}) < w_1$	$S(h_3\backslash\{h_5, h_7\}) > w_2$ $\Rightarrow C(\underline{h_2}, h_3, h_4, h_6, \overline{h_5}, \overline{h_7}) > H_0(\underline{h_2}, \overline{h_5}, \overline{h_7})$ h_3, h_4 and h_6 cannot exist in the l-optimum partition and should be deleted from the arc tree Now, h_5, h_7 are the two arcs immediately above h_2 since $S(h_7\backslash\{h_9\}) > S(h_5\backslash\{h_8\})$, test h_7 first to see if h_7 can be deleted from the arc tree $S(h_7\backslash\{h_9\}) > w_2$ $\Rightarrow C(\underline{h_2}, h_7, \overline{h_5}, \overline{h_9}) > H_0(\underline{h_2}, \overline{h_5}, \overline{h_9})$ h_7 should be deleted from the arc tree $S(h_5\backslash\{h_8\}) < w_2$ $\Rightarrow C(\underline{h_2}, h_5, \overline{h_8}, \overline{h_9}) < H_0(\underline{h_2}, \overline{h_8}, \overline{h_9})$ $S(h_2\backslash\{h_5, h_9\}) < S(h_5\backslash\{h_8\}) \Rightarrow h_6$ is a son of h_2 Condense h_5 into h_2 and calculate $S(h_2\backslash\{h_8, h_9\})$
9. $S(h_2\backslash\{h_8, h_9\})$	?	

If $S(h_2\backslash\{h_8, h_9\}) < w_1$, the partition with h_2 and h_5 as h-arcs is l-optimum. Otherwise, the fan $H_0(\underline{h_1}, \overline{h_8}, \overline{h_9})$ will be l-optimum.

From the above example, we have the following observations. Let h_j be the arc being processed and let X be the set of arcs immediately above h_j in the arc tree. By the time we process h_j, we have already obtained (i) the l-optimum partitions of the subpolygons between the leaf nodes and the arcs in X and (ii) the ceilings of all the arcs in X. For any arc h_k in X, the l-optimum partition in the subpolygon bounded below by h_j and above by the arcs in $X - \{h_k\} \cup h_k$'s ceiling must either be a fan or consist of h_k and its descendants as h-arcs depending on whether $S(h_k\backslash h_k\text{'s ceiling}) \geqq \min(w_j, w_j')$ or $S(h_k\backslash h_k\text{'s ceiling}) < \min(w_j, w_j')$, where w_j, w_j' are the weights associated with the end vertices of h_j. If the fan is cheaper, h_k and h_k's descendants will be removed from the arc tree and the set X becomes $X - \{h_k\} \cup h_k$'s ceiling. We can repeat the above process until the l-optimum partition in the subpolygon bounded below by h_j and above by the leaf nodes in the subtrees of h_j is obtained. Since $\max_{h_k \in X} S(h_k\backslash h_k\text{'s ceiling}) < \min(w_j, w_j')$ implies $(\forall h_k \in X)(S(h_k\backslash h_k\text{'s ceiling}) < \min(w_j, w_j'))$, the arc with maximum supporting weight in X should be chosen and tested for possible deletion. Similarly, since $\max_{h_k \in X} S(h_k\backslash h_k\text{'s ceiling}) < S(h_j\backslash X)$ implies $(\forall h_k \in X)(S(h_k\backslash h_k\text{'s ceiling}) < S(h_j\backslash X))$, the arc with maximum supporting weight should also be chosen and tested for possible condensation.

Now, let us give the algorithm for finding the l-optimum partition of a general convex polygon.

ALGORITHM P

(I) Get all the potential h-arcs of the polygon by the one-sweep algorithm [6]. (All these arcs form a tree.)

(II) Append the degenerated arcs to the arc tree obtained in Step I and label all leaf nodes as "processed."

(III) Process the potential h-arcs, one by one, from the leaves to the root. (We cannot process a potential h-arc until all the potential h-arcs in its subtrees have been processed.) Let h_j be the arc to be processed, h_i be the arc immediately below h_j in the arc tree, X be the set of potential h-arcs immediately above h_j in the arc tree, and h_m be an arc in X such that

$$S(h_m\backslash h_m\text{'s ceiling}) = \max_{h_k \in X} S(h_k\backslash h_k\text{'s ceiling}).$$

Repeat
  Begin
1. [To delete those blocks of arcs which cannot exist in the l-optimum partition of the subpolygon between h_j and the leaf nodes in its subtrees.]
   While $S(h_m\backslash h_m\text{'s ceiling}) \geqq \min(w_j, w_j')$ do
     Begin
a. Delete h_m and its descendants from the arc tree.
b. Replace X by $X - \{h_m\} \cup h_m$'s ceiling and then update h_m accordingly.
     End.
2. [To locate the ceiling of h_j.]
   If $h_j \neq h_1$
   then
     While $S(h_j\backslash X) \leqq S(h_m\backslash h_m\text{'s ceiling})$ do
       Begin
a. Comment: h_m is a son of h_j.

b. Combine h_m and all its descendants into h_j and calculate the combined supporting weight

$$S(h_j \backslash X - \{h_m\} \cup h_m\text{'s ceiling}).$$

c. Replace X by $X - \{h_m\} \cup h_m$'s ceiling and then update h_m accordingly.
End.

3. [Prepare to process next arc.]
If $h_j \neq h_1$
then
If h_i has a subtree which has not been processed then pick a subtree of h_i which has not been processed and apply Step II to this subtree recursively
else
Begin
Replace X by the arcs immediately above h_i in the arc tree, h_j by h_i and h_i by the arc immediately below h_i in the arc tree.
End.
End.
Until $(h_j = h_1)$.

(IV) Output the l-optimum partition consisting of the arcs which remain in the arc tree after Step II as h arcs. Then stop.

Using arguments similar to those in the proof of Theorem 8, we have the following theorem.

Theorem 13. *The partition produced by Algorithm* P *is l-optimum.*

Proof. Omitted.

One way to implement Algorithm P is to place all the potential h-arcs obtained in Step I in a linked tree. Each potential h-arc in the arc tree is again associated with a record variable similar to those described in § 2. We shall also initialize the ith element of the array CP to the quantity $w_1 : w_i$ for $2 \leqq i \leqq n$ and set CP[1] to zero. Hence, from our discussions in § 2, we know that we can calculate the supporting weights in a constant amount of time. Since we always test the arc with the largest supporting weight for possible deletion or condensation among all the arcs in X in Step II of the algorithm, we should keep track of the arcs in X and in each ceiling by means of the priority queues. When an arc h_m in X is deleted from the arc tree, we remove h_m from X, then we merge X and the ceiling of h_m into one priority queue. Similarly, when an arc h_m in X is condensed into h_j, we remove h_m from X and add it to the list of h_j's sons, then we merge X and the ceiling of h_m into one priority queue and set the ceiling of h_m, i.e. the third field of h_m, to NIL. Hence, it takes $O(\log n)$ time for each update of X to $X - \{h_m\} \cup h_m$'s ceiling in both Step II.1 and Step II.2.

THEOREM 9′. *Algorithm* P *runs in* $O(n \log n)$ *time.*

Proof. It takes $O(n)$ time to sweep around the monotone polygon twice, once to obtain all potential h-arcs in Step I and once to initialize the array CP. It also takes $O(n)$ time to append the degenerated arcs in the arc tree. There are two while loops in Step III, and it takes $O(\log n)$ time to execute either while loop once. Whenever the while loop in Step III.1 is executed once, a potential h-arc is deleted from the arc tree. Whenever the while loop in Step III.2 is executed once, a potential h-arc is removed from the arc tree and condensed to its father. Once an arc is removed or deleted from the list, it will never be considered again. Since there are at most $n-3$ arcs in the arc tree, Algorithm P can execute both while loops at most $n-3$ times.

So, it takes $O(n \log n)$ time to process all the potential h-arcs in Step III. Finally, it takes $O(n)$ time to output the l-optimum partition in Step IV. Hence, Algorithm P runs in $O(n \log n)$ time. $\square$

4. Conclusions. In this paper, we have presented an $O(n \log n)$ algorithm to find the unique lexicographical smallest optimum partition of a general convex polygon. Both Algorithm M and Algorithm P have been implemented in Pascal [7]. We have also compared Algorithm P with the $O(n^3)$ dynamic programming algorithm and found that Algorithm P runs faster than the dynamic programming algorithm when n is greater than or equal to 7.

REFERENCES

[1] A. V. AHO, J. E. HOPCROFT AND J. D. ULLMAN, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, MA, 1974.

[2] S. S. GODBOLE, *An efficient computation of matrix chain products*, IEEE Trans. Computers, C-22 (1973), pp. 864–866.

[3] H. HOROWITZ AND S. SAHNI, *Fundamentals of Computer Algorithms*, Computer Science Press, Potomac, MD 1978.

[4] T. C. HU AND M. T. SHING, *Computation of matrix chain products*, in 1981 Army Numerical Analysis and Computer Conferences, February 1981.

[5] ———, *An $O(n)$ algorithm to find a near-optimum partition of a convex polygon*, J. Algorithms, to appear.

[6] ———, *Computation of matrix chain products*, I, this Journal, 11 (1982), pp. 362–373.

[7] ———, *Computation of matrix chain products, Part* I, *Part* II, Report STAN-CS-81-875, Stanford Univ., Stanford, CA, Sept. 1981.