SIAM J. COMPUT.
Vol. 14, No. 4, November 1985

DECOMPOSING A POLYGON INTO SIMPLER COMPONENTS*

J. MARK KEIL†

Abstract. The problem of decomposing a polygon into simpler components is of interest in fields such as computational geometry, syntactic pattern recognition, and graphics. In this paper we consider decompositions which do not introduce Steiner points. The simpler components we consider are convex polygons, spiral polygons, star-shaped polygons and monotone polygons. We apply a technique for improving the efficiency of dynamic programming algorithms in order to achieve polynomial time algorithms for the problems of decomposing a simple polygon into the minimum number of each of the component types. Using the same technique we are able to exhibit polynomial time algorithms for the problems of decomposing a simple polygon into each of the component types while minimizing the length of the internal edges used to form the decomposition. When the polygons are allowed to contain holes many of the problems become NP-hard.

Key words. polygon decomposition, convex polygon, star-shaped polygon, spiral polygon, monotone polygon, dynamic programming

1. Introduction.

Let P be a simple polygon in the plane having vertices $v_1, v_2, \cdots v_n$ clockwise on its boundary. There are many ways to decompose such a polygon into simpler components [15], [26]. The simpler components we will consider are convex polygons, spiral polygons, star-shaped polygons and monotone polygons. Using a dynamic programming approach we will develop polynomial time algorithms for decomposing a polygon into the minimum number of each of these simple components. Using the same technique we are able to exhibit polynomial time algorithms for the problems of decomposing a simple polygon into each of the component types while minimizing the length of the internal edges used to form the decomposition.

There are several motivations for considering polygon decomposition problems. An arbitrary polygonal shape can be recognized more easily once its component parts have been identified. This property can be used in pattern recognition schemes [9], [23], [26]. In computational geometry, a problem can often be solved on a general polygon by applying efficient specialized algorithms to the component parts of the decomposed polygon. Other application areas for polygonal decompositions include database systems [20] and graphics [21].

Decomposing a simple polygon into nonoverlapping component parts can be done with or without introducing additional vertices which are commonly called Steiner points. A decomposition which does not increase the number of vertices is preferable when the components are to be processed further. Figure 1.1 shows decompositions of simple polygons into the minimum number of convex polygons with and without Steiner points. The algorithms we describe in this paper do not introduce Steiner points.

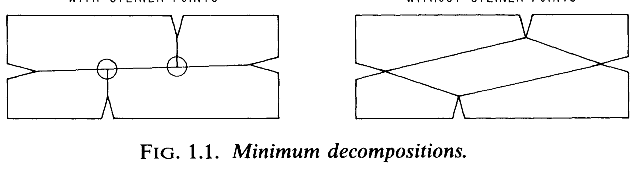

FIG. 1.1. *Minimum decompositions.*

* Received by the editors October 1, 1982, and in final revised form March 20, 1984.

† Department of Computer Science, University of Toronto, Toronto, Ontario, Canada M5S 1A7. This work was supported by the Natural Sciences and Engineering Research Council of Canada.

If the interior angle at a vertex is reflex the vertex is called a notch. Experimental observation from graphics [21] and pattern recognition [23] shows that, in practice, the number of notches in a polygon is much smaller than the number of vertices. We let N denote the number of notches in a polygon and we describe and analyze decomposition algorithms with respect to both n and N.

Before we consider the polygon decomposition problems we make the following definitions. A *Convex Polygon P* is a simple polygon in which any two boundary points can be joined by a segment that lies completely within P. A *Spiral Polygon* is a simple polygon whose boundary chain contains at most one concave subchain. A *Monotone Polygon* is a simple polygon in which there exists two extreme vertices in a preferred direction such that they are connected by two polygonal chains monotonic in the preferred direction. A *Star-shaped Polygon* is a simple polygon in which the entire polygon is visible from at least one fixed (possibly interior) point of the polygon.

In § 2 we develop our dynamic programming approach. We use it in § 3 to develop a polynomial time algorithm for the problem of decomposing a simple polygon into the minimum number of convex components. In § 4 we develop a polynomial time algorithm for decomposing a simple polygon into the minimum number of star-shaped components. We consider minimum edge length decompositions in § 5 and other decomposition problems in § 6.

2. Dynamic programming.

Since their introduction by Bellman [2], dynamic programming (DP) algorithms have been used to solve a wide variety of discrete optimization problems. To apply DP one must represent such a problem by a decision process which proceeds from state to state in a series of stages. At each stage a decision is made that will lead to a state in the next stage at a certain cost. A DP algorithm decomposes the problem into a number of smaller subproblems each having fewer stages than the original problem and it gains its efficiency by avoiding recomputing solutions to common subproblems. For a survey of the use of DP in computer science see [3].

A state consists of variables that describe the condition of the system. A state must contain enough information so that current decisions depend only on the current state and not on the particular history of previous states and decisions. This property is called the state separation property. The state space consists of the set of all possible states in which the system may exist and a valid state space for DP must have the state separation property.

A problem for DP is rarely presented in terms of states and decisions and often such a representation is not obvious. The most difficult part in applying DP is usually the definition of the state space. If too little information is included in a state the state space will not be valid and no correct DP algorithm can be developed. If too much information is included in a state the state space will be large and the DP algorithm will produce an unnecessary amount of computation.

Reducing the size of a valid state space can reduce the time required by a DP algorithm. Karp and Held [13] suggested state space minimization as a good heuristic for speeding up DP algorithms but were unable to provide a method of doing it. Later Ibaraki [12] showed that in general the problem of minimizing the number of states in a valid state space is not decidable.

Elmaghraby [8] introduced the concept of equivalent states which is useful in reducing the size of a valid state space. A policy is a finite ordered sequence of decisions that leads from one state through others to a destination state. Let X be the set of all policies. Then two states s_1 and s_2 are said to be equivalent if and only if the state reached from s_1 after policy x is the same state reached from s_2 after policy x for all permissible policies $x \in X$. In a minimum state space there will be no pair of equivalent states.

To reduce the size of a valid state space Elmaghraby [8] suggests constructing classes of equivalent states. Once these classes have been constructed only one representative member from each class need be kept. This method has not been widely used because it is difficult to find these classes of equivalent states. In this paper we systematically use the equivalence class method of state reduction to exhibit polynomial time algorithms for each of the decomposition problems we consider.

Let us look at the problem of decomposing a polygon into the minimum number of convex polygons and attempt to develop a DP algorithm for it. To do this we need some definitions which are illustrated in Fig. 2.1. Two points of a polygon P are said to be ***visible*** if the line segment joining them lies wholly inside P. A ***subpolygon*** P_{ij} of P with vertices $v_i, v_{i+1}, \cdots v_j$ exists when $1 <= i < j-1 <= n-1$, and v_iv_j are visible. The ***base convex polygon*** C_{ij} of a minimum decomposition (MD) D of P_{ij} is that convex polygon of D that contains the edge v_iv_j in its boundary. We call v_iv_j the ***base edge*** of P_{ij}. A triangle T_{imj} can ***merge*** with a convex decomposition A of P_{im} if $C_{im}(A) \cup T_{imj}$ is a convex polygon. The ***size*** of a MD of P_{ij} is the number $|D|$ of convex polygons in a MD of P_{ij}.

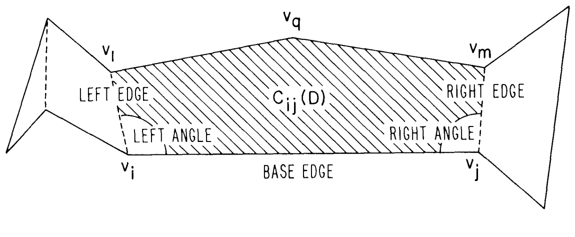

FIG. 2.1. *Definitions.* T_{imj} *merges left with any* MD A *of* P_{im} *with* $C_{im}(D) = v_iv_lv_qv_m$.

Let us define the state space by letting the states be of the form s_{ij} where s_{ij} has the interpretation that subpolygon P_{ij} has been decomposed minimally. A decision is then based on a pair of states s_{im} and s_{mj}, $i < m < j$, and leads to the state s_{ij} in the next stage. The minimum cost of a policy that leads to s_{ij} is equal to the size of MD of P_{ij}. A MD of subpolygon P_{ij} is found by taking a MD of P_{im} and P_{mj} for some $i < m < j$ and merging T_{imj} with C_{im} or C_{mj} if possible.

An attempt to use this DP formulation will reveal that the state space is not valid. There can be many MDs of a subpolygon P_{im} and it is not sufficient for a state to contain information about only one of them. Figure 2.2 shows two MDs of P_{im} only one of which merges with T_{imj}.

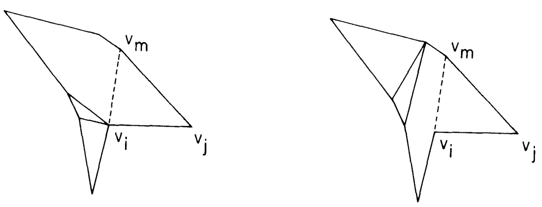

FIG. 2.2. *The* MD *of* P_{im} *shown on the left merges with* T_{imj} *while the* MD *of* P_{im} *shown on the right does not.*

Since storing information about one MD of a subpolygon P_{ij} is insufficient we can try creating a state for each MD of a subpolygon. With this definition the problem encountered due to lack of information has vanished and indeed the state space is valid. We therefore have a correct DP formulation and the only question remaining is the efficiency of the algorithm. Unfortunately, the state space we have defined is very large. In fact, as illustrated in Fig. 2.3 there can be an exponential number of ways of decomposing a simple polygon into convex polygons. Since we are seeking a polynomial time algorithm we must try to reduce the size of the state space.

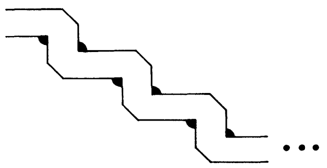

FIG. 2.3. *There are* $2^{n/3-4}$ MD*s of the above polygon. Each notch has an independent choice of two interior edges that will remove it.*

As indicated previously we will use equivalence classes of states. To describe these we make a few more definitions which are illustrated in Fig. 2.1. In a subpolygon P_{ij} the edge preceding v_iv_j in the clockwise representation of $C_{ij}(D)$ is the *right edge* of MD D. The *left edge* of a MD is defined analogously. The *left angle* of a MD D is the angle between the left edge and the base edge of D. The *right angle* of a MD is defined analogously.

The reason why we needed to keep more than one MD of a subpolygon P_{im} was that the MDs varied in their ability to merge with triangles T_{imj}. But it is only the left and right edges of a MD that affect its ability to merge. This suggests that all MDs with a given pair of left and right edges are equivalent under this DP formulation and this equivalence is established in the following lemma.

LEMMA 2.1. *Any members of the class of* MD*s of a subpolygon* P_{im} *with a given pair of left and right edges are equivalent in terms of constructing a* MD *of a valid subpolygon* $P_{ij}(j > m)$ *by merging with* T_{imj}.

Proof. Let A be a MD of P_{im}. T_{imj} can merge left with A if $C_{im}(A) \cup T_{imj}$ is a convex polygon. Since $C_{im}(A)$ and T_{imj} are convex $C_{im}(A) \cup T_{imj}$ is convex if and only if the angles between v_iv_j and the left edge of A and between v_mv_j and the right edge of A are less than 180 degrees. Therefore two MDs of P_{im} with the same left edge and right edge are equivalent in terms of constructing a MD of a valid subpolygon $P_{ij}(j > m)$.

We can therefore reduce the size of the state space by including only one representative from each class of MDs of a subpolygon with a given pair of left and right edges. We have retained enough information to perform merges and the state space remains valid. Since there are only $O(n^2)$ possible pairs of left and right edges of MDs in a subpolygon there are now only a polynomial number of states in the state space and our DP algorithm runs in polynomial time. However we are not yet finished eliminating states and improving the efficiency of the algorithm.

Even in the reduced state space there are states that will not be used in constructing the optimal solution. The following lemma helps identify some of them.

LEMMA 2.2. *No* MD D *of* P *contains an interior edge which connects two vertices of* P *which are not notches.*

Proof. Straightforward.

This means we need not consider states associated with subpolygons P_{ij} where neither v_i nor v_j is a notch. A subpolygon is called a *valid subpolygon* when either $i = 1$

or $j = n$ or at least one of v_i and v_j is a notch. In the rest of this section and in § 3 when we say subpolygon we will mean valid subpolygon. We can remove those states that are not associated with valid subpolygons. The following lemma will enable us to identify classes of states that can be replaced in a valid state space by one representative state.

LEMMA 2.3. *If* MD A *of* P_{im} *has left angle* ϕ_1 *and right angle* θ_1 *and* MD B *of* P_{im} *has left angle* ϕ_2 *and right angle* θ_2 *where* $\phi_1 \leqq \phi_2$ *and* $\theta_1 \leqq \theta_2$, *then* MD B *merging with* T_{imj} *where* P_{ij} *is a valid subpolygon* $(j > m)$, *implies that* A *also merges with* T_{imj}.

Proof. Straightforward.

If two states are associated with the MDs A and B as in Lemma 2.3 the state associated with MD B need not be kept in the state space. To find the MDs that can be similarly eliminated we first define an equivalence relation R over the set of MDs of a subpolygon P_{ij} so that all those MDs of P_{ij} with right angle θ will belong to the equivalence class $R(\theta)$ of R. A MD in $R(\theta)$ with the minimum left angle is called a *left minimum* of $R(\theta)$. Using Lemma 2.3 we now seek the smallest set of MDs of P_{ij} that will be sufficient to represent the right angle equivalence classes. To begin with, Lemma 2.3 implies that any MD that is not a left minimum of one of the classes $R(\theta)$ is not essential. Normally a class of MDs, $R(\theta)$, will be represented by its left minimum. However if two left minimum MDs A and B have angles as in Lemma 2.3 then MD A is used to represent both $R(\theta_1)$ and $R(\theta_2)$. A set of MDs with only these essential right angle class representatives is said to have the *right representative* (RR) property.

In the algorithm a set X with the RR property is stored in a binary search tree so that a MD with a given right angle can be found in $O(\log(\text{size}(X)))$ time. The definition of the RR property implies that a set X with the RR property sorted in increasing order of right angle will also be sorted in decreasing order of left angle. The *left representative* (LR) property is defined analogously.

By identifying equivalent classes of states and eliminating unnecessary states we have substantially reduced the size of the state space. In the next section we will describe the DP algorithm in more detail.

3. Convex decomposition algorithm.

Of all the polygon decomposition problems, the problem of decomposing a polygon into the minimum number of convex components has received the most attention. Chazelle and Dobkin [4], [5], [6] were the first to exhibit a polynomial time algorithm for a minimum polygonal decomposition problem. Their $O(n+N^3)$ time algorithm decomposes a polygon into convex parts and allows Steiner points.

Until recently, when Steiner points are disallowed, no polynomial time exact solution was known to the convex decomposition problem thus motivating the development of approximation algorithms. Feng and Pavlidis [9] describe an $O(N^3n)$ time algorithm which does not generally yield a minimum decomposition. Schachter's [25] $O(Nn)$ time decomposition algorithm, based on the Delaunay triangulation, also does not generally yield a minimum decomposition. Recently, Greene [10] has developed an $O(n \log n)$ time algorithm that finds a decomposition that is within 4 times of the minimum decomposition. Note however that any convex decomposition that does not contain unnecessary edges will be within 4 times of the minimum decomposition.

Recently, Greene [10] independently discovered an $O(N^2n^2)$ time exact algorithm for the convex decomposition problem. Our algorithm for that problem runs in $O(N^2n \log n)$ time.

In the previous section we defined the state space for a DP formulation of the convex decomposition problem. In order to complete the algorithm description we make a few definitions. A *reference vertex* is a vertex of P which is either a notch or v_1 or v_n. Nonreference vertices are called *convex vertices.* A valid subpolygon P_{ij} will be one of three types: *type* (a) if both v_i and v_j are reference vertices, *type* (b) if v_i is a reference vertex and v_j is a convex vertex, or *type* (c) if v_i is a convex vertex and v_j is a reference vertex. A *base triangle* T_{imj} of P_{ij} is a triangle $v_iv_mv_j$ $(i<m<j)$ where each of v_iv_m and v_jv_m is either a base edge of a valid subpolygon or an original side of P.

Preprocessing. Some of the work is best done before the main DP procedure begins.

1. Determine which vertices are notches in $O(n)$ time.

2. For each reference vertex x of P determine the set of vertices of P visible from x. For convex vertices store the set of visible reference vertices. Denote such a visibility list sorted by angle about vertex x as $V(x)$. All of this can be done in $O(nN)$ time using the visibility polygon algorithm of El-Gindy and Avis [7].

3. Since each visibility pair $v_iv_j(i<j)$ determined in step 2 is the base edge of a valid subpolygon these valid subpolygons can be sorted in ascending order by the size measure $j-i$ in time $O(nN\log(nN))$.

4. For each valid subpolygon P_{ij} form the set of base triangles. In the $O(N^2)$ subpolygons of type (a) this can be done in $O(n)$ time by computing $T=\{(V_i\cup\{v_{i+1}\})\cap(V_j\cup\{v_{j-1}\})\}$. In the $O(nN)$ subpolygons of type (b) or (c) this can be done in $O(N\log n)$ time by computing T using a binary search.

DP procedure.

1. Consider each valid subpolygon P_{ij} in the order computed in step 3 of the preprocessing.

2. Compute a set XR_{ij} of MDs of P_{ij} with the RR property (and similarly a set XL_{ij} of MDs of P_{ij} with the LR property) as follows.

Let M denote the size of a MD of P_{ij}. Initially set M to n. If $j-i=2$, XR_{ij} will contain a single triangle.

Otherwise, for each base triangle T_{imj} of P_{ij}:

(a) select the MD A from the set XL_{im} of P_{im} with the minimum left angle such that T_{imj} can merge left with $C_{im}(A)$. This is done by binary search. If no such A exists select any MD A of P_{im}. Take the decomposition A selected together with T_{imj} (merged if possible) and a MD B of P_{mj} to form a decomposition D of P_{ij} with right angle (v_jv_m, v_jv_i). If $|D|>M$ discard D as it is not a member of XR_{ij}. If $|D|=M$ then insert D into XR_{ij} as the member with right angle (v_jv_m, v_jv_i). If $|D|<M$ then empty XR_{ij} and insert D as the only member of XR_{ij} and reset M to $|D|$.

(b) select the MD B' from the set XR_{mj} with the minimum right angle that will merge with T_{imj}. Take decomposition B' merged with T_{imj} together with a MD A' of P_{im} to form a decomposition D' of P_{ij}. As in part (a) compare $|D'|$ to M to see if D' belongs in XR_{ij}.

At this point XR_{ij} will consist of a set of MDs with unique right angles. However XR_{ij} may still not have the RR property. To remove MDs from XR_{ij} that are inconsistent with the RR property begin by sorting XR_{ij} by right angle. Set ϕ to the left angle of the MD in XR_{ij} with the smallest right angle. Now scan XR_{ij} in order of increasing right angle. If the MD D under scan has left angle $\phi'\geqq\phi$ delete D from XR_{ij}. If D has left angle ϕ' smaller than ϕ then set ϕ to ϕ'. When the scan is complete XR_{ij} has the RR property. See Fig. 3.1.

3. The subpolygon $P_{1n}=P$ will be the last one considered. Select one member of XL_{1n} or XR_{1n} as a MD of P.

Proof of correctness. We need to show that we correctly compute a set of MDs, with the RR(LR) property, for each subpolygon P_{ij}. We use a proof based on an

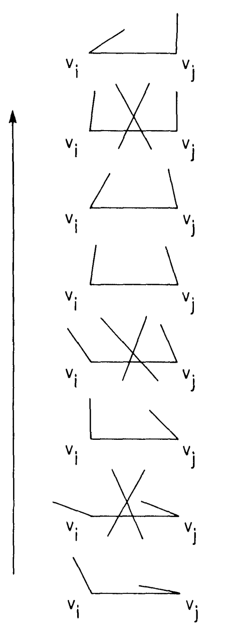

FIG. 3.1. *A set of* MDs *with the RR property results from the scan that removes the crossed* MDs.

induction on the size, $j-i$, of the subpolygons. If $j-i=2$, P_{ij} is a triangle and there is only one MD. Otherwise, as an example, let us consider how we compute a MD of a subpolygon P_{ij} of type (a) with right side v_jv_i. Clearly we need only consider single merges to the left if we use base triangle T_{irj}. If any MD, say A, of P_{ir} merges with T_{irj} then clearly A merged with T_{irj} taken together with a MD of P_{rj} will form a MD of P_{ij}. If no MD of P_{ir} merges with T_{irj} then we may take any MD of P_{ir} together with T_{irj} and a MD of P_{mj} to form a MD of P_{ij}. We are similarly able to compute any MD that we require by single merging at an appropriate base triangle using MDs of smaller subpolygons that have already been calculated. The following lemma shows how an arbitrary MS, as a member of a set with the RR property, can be constructed.

LEMMA 3.1. *Let D be a* MD *in the set XR_{ij} of* MDs *of P_{ij} with the* RR *property. Furthermore when a base triangle T_{imj} exists let XL_{im} be a set of* MDs *of P_{im} with the* LR *property and XR_{mj} be a set of* MDs *of P_{mj} with* RR *property. Then there exists an $O(\log n)$ time algorithm which will construct a* MD *D' of P_{ij} equivalent to D using only v_iv_l (the left side of D), v_jv_r (the right side of D) and XL_{im} and XR_{mj} where T_{imj} is a base triangle of P_{ij} with either $v_m=v_l$ or $v_m=v_r$.*

Proof.

Case 1. If P_{ij} is of type (a) or (b) then T_{imj} exists so that $v_m=v_r$. The decomposition D' required can be found by computing decomposition D' in parts (i) and (ii) and selecting the smaller.

(i) If $C_{ij}(D)$ is a triangle then the MD D' formed by taking T_{imj} together with any MD A of P_{im} and any MD B of P_{mj} will have the same left and right edges as decomposition D and will be, by Lemma 2.1, equivalent to D.

(ii) If $C_{ij}(D)$ is not a triangle let polygon Q be $P_{im}\cup T_{imj}$. Clearly D restricted to Q is a MD of Q. Let A be D restricted to P_{im}. If A is not a MD of P_{im} then any MD of P_{im}, A'', would have the property that $C_{im}(A'')\cup T_{imj}$ is not a convex polygon. But this would imply that there exists a MD of P_{ij} with the same right angle as D but with a smaller left angle. This is a contradiction to XR_{ij} having the RR property. Since

A is a MD of P_{im} when we select the MD A' from XL_{im} in $O(\log n)$ time with the smallest left angle that will merge with T_{imj}, A' will have the same left edge as A by Lemma 2.1. Taking T_{imj} merged with A' together with a MD B of P_{mj} will yield a MD D' of P_{ij} equivalent to D.

Case 2. If P_{ij} is of type (c) then T_{imj} exists so that $v_m = v_l$. Again the decomposition D' required is found by computing decomposition D' in parts (i) and (ii) and selecting the smaller.

(i) If $C_{ij}(D)$ is a triangle the MD D' can be found as in Case 1.

(ii) If $C_{ij}(D)$ is not a triangle in $O(\log n)$ time select a MD B' from XR_{ij} with the minimum right angle that will merge with T_{imj}. The MD D' of P_{ij} formed by taking a MD A of P_{im} together with T_{imj} merged with the MD B' of P_{mj} will have the same left angle as D and at least as small a right angle as D. Since D is in XR_{ij} with the RR property the right edge of D' will in fact equal the right edge of D and again by Lemma 2.2 D' will be equivalent to D.

The next lemma shows how the required set of MDs for each subpolygon can be found.

LEMMA 3.2. *A set XR_{ij} (likewise XL_{ij}) of* MD*s of P_{ij} with the* RR *(likewise* LR*) property can be correctly constructed in $O(n \log n)$ time if P_{ij} is of type* (a) *and in $O(N \log n)$ time if P_{ij} is of type* (b) *or* (c). *The procedure uses only a set XL_{im} of* MD*s of P_{im} with the* LR *property and a set XR_{mj} of* MD*s of P_{mj} with the* RR *property, where m is such that T_{imj} is a base triangle of P_{ij}.*

Proof. Each member decomposition D of XR_{ij} can be found using a different base triangle T_{imj} as in Lemma 3.1. If P_{ij} is of type (a) then $O(\log n)$ work is performed at each of the $O(n)$ base triangles. If P_{ij} is of type (b) or (c) then $O(\log n)$ work is performed at each of the $O(N)$ base triangles. With this method some MDs will be placed into XR_{ij} that are not consistent with the RR property. If we treat XR_{ij} as a set of two-dimensional vectors with first component (180°—right angle) and second component (180°—left angle) then the members of XR_{ij} that are consistent with the RR property will be the maximal elements of XR_{ij} with respect to the natural partial order. The scan procedure described in the algorithm implements the algorithm of Kung, Luccio and Preparata [16] for finding the maxima of a set of vectors. This requires $O(n \log n)$ time if P_{ij} is of type (a) and $O(N \log N)$ time if P_{ij} is of type (b) or (c).

Using the above lemmas we can prove the following.

THEOREM 3.3. *The algorithm finds a* MD *of a simple polygon P in $O(N^2 n \log n)$ time in the worst case.*

Proof. The preprocessing requires $O(N^2 n \log n)$ time. There are $O(N^2)$ subpolygons of type (a). There are $O(nN)$ subpolygons of type (b) or (c). To calculate a set XL_{ij} of MDs of P_{ij} with the LR property and a set XR_{ij} of MDs of P_{ij} with the RR property, as in Lemma 3.2, $O(n \log n)$ time must be spent at subpolygons of type (a) and $O(N \log n)$ time must be spent at subpolygons of type (b) or (c). Altogether $O(N^2 n \log n)$ time is required in the worst case to compute XR_{1n} and XL_{1n}. Since $P_{1n} = P$ any MD in XR_{1n} *or* XL_{1n} is a MD of P.

4. Star-shaped decompositions.

Recall that a star-shaped polygon is a simple polygon in which the entire polygon is visible from at least one (possibly interior) fixed point of the polygon. Avis and Toussaint [1] give an $O(n \log n)$ algorithm that finds a decomposition into at most $n/3$ components and does not use Steiner points. However their algorithm does not generally yield a minimum decomposition. Before we can formulate a DP algorithm for the minimization problem we need a few definitions.

The set of points from which the entire polygon P is visible is called the *kernel* of P. For star-shaped decompositions we need to define two types of merging. Let $S_{im}(A)$ be the base star-shaped polygon of the minimum star-shaped decomposition A of P_{im}. A triangle T_{imj} can ***single-merge*** with MD A of P_{im} if $S_{im}(A) \cup T_{imj}$ is a star-shaped polygon. A triangle T_{imj} can ***double-merge*** with a MD A of P_{im} and a MD B of P_{mj} if $S_{im}(A) \cup T_{imj} \cup S_{mj}(B)$ is a star-shaped polygon.

To formulate a DP algorithm we define the state space by letting the states s_{ij} have the interpretation that subpolygon P_{ij} has been decomposed into the minimum number of star-shaped components. As in the convex case, decisions are based on a pair of states s_{im} and s_{mj}, $i < m < j$, and lead to the state s_{ij} in the next stage. A MD of P_{ij} is found by taking a MD of P_{im} and P_{mj} for some m such that $i < m < j$ and determining whether T_{imj} will double-merge or single-merge with S_{im} or S_{mj}. The size of a MD of P_{ij} can be one less, equal to, or one more than the sum of the sizes of a MD of P_{im} and P_{mj} for some m depending on which merges can take place. After our experience with convex decompositions we are not surprised that the state space in this DP formulation is not valid. As before it is not sufficient for the states to contain information about only one MD of a subpolygon P_{ij}. Figure 4.1 shows two star-shaped MDs of P_{im} only one of which single-merges with T_{imj}.

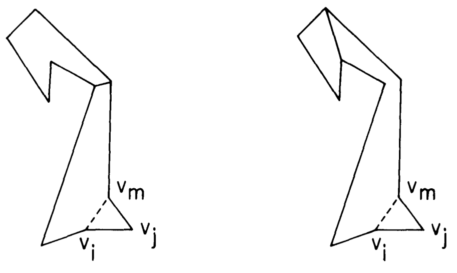

FIG. 4.1. *The* MD *of* P_{im} *shown of the left merges with* T_{imj} *while the* MD of P_{im} *shown on the right does not.*

The surprise comes when we discover that creating a state for each MD of a subpolygon is insufficient for a valid state space. Figure 4.2 shows a subpolygon for which the MD, which consists of a single star-shaped polygon, cannot be found by this DP formulation since the smaller subpolygons defined by a base triangle, as in Fig. 4.2b, are not star-shaped. Even though all MDs of smaller subpolygons are kept and all double and single merges are found, the MD of P_{ij} is not found. As illustrated in Fig. 4.3 there can be an exponential number of ways of decomposing a simple

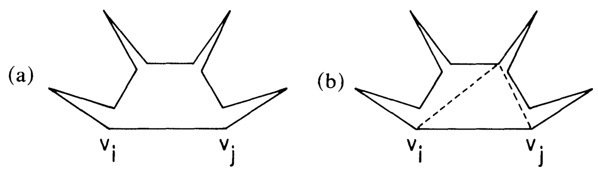

FIG. 4.2

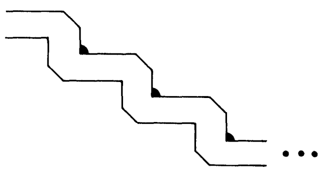

FIG. 4.3. *There are* $2^{n/6-4}$ MD*s of the above polygon.*

polygon into star-shaped polygons. If we are to develop a polynomial time algorithm for this problem we certainly cannot expand the state space further.

We will now define a set KER of potential kernel points. We begin by identifying segments that could contain a side of the boundary of the kernel of the base star-shaped polygon of a MD of a subpolygon. If v_a and v_b are vertices of polygon P then there may exist a segment of the line through v_a and v_b, that lies entirely within P, that contains v_b and exactly one other boundary point v_c of P so that v_b lies between v_a and v_c. We denote such a segment l_{ab} when it exists. That is, l_{ab} is the segment drawn inside P beginning at v_b, in the direction away from v_a, until the boundary of P is reached. We define $L=\{l_{ab} | v_b$ is a notch and v_a is a vertex of P visible from $v_b\}$. The segments in L may intersect with each other or with sides of P or both. Let KER denote the set of all such intersections together with the vertices of P as in Fig. 4.4. With these definitions we can present the following lemma which allows us to achieve a valid polynomial sized state space.

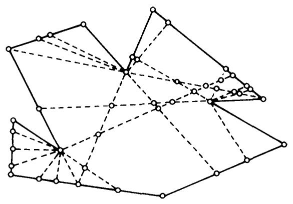
FIG. 4.4

LEMMA 4.1. *The kernel of any star-shaped base polygon of a* MD D *of a subpolygon* P_{ij} *contains a point in the set* KER. *The set* KER *contains at most* $O(n^2N^2)$ *points.*

Proof. The kernel of a base star-shaped polygon $S_{ij}(D)$ of a MD D of P_{ij} is the intersection of the halfplanes defined by the sides of $S_{ij}(D)$. The sides of the kernel of $S_{ij}(D)$ will either be contained in sides of $S_{ij}(D)$ or in segments extending from these sides (i.e. segments in the set L). If $S_{ij}(D)$ is convex then vertex v_i, a point in KER, is contained in the kernel of $S_{ij}(D)$. Otherwise at least one side of the kernel of $S_{ij}(D)$ is contained in a segment l_{ab} in the set L. If vertex v_b, the endpoint of l_{ab} that is a vertex of P, lies in the kernel of $S_{ij}(D)$ we are done as v_b is a point in KER. If vertex v_b is not in the kernel of $S_{ij}(D)$ then a side of the kernel adjacent to the side contained in l_{ab} must also be contained in a segment in L. The point of intersection between these segments will be a vertex of the boundary of the kernel of $S_{ij}(D)$ and also a point in KER.

The set L contains $O(nN)$ segments. There will be at most $O(n^2N^2)$ intersections amongst segments in L and $O(nN)$ intersections between segments in L and the sides of P. Therefore KER will contain at most $O(n^2N^2)$ points.

To achieve a valid state space we introduce pseudo star-shaped polygons. A pseudo star-shaped subpolygon PS_{ij} of P based along v_iv_j has the property that there exists a point x in P and not in PS_{ij} so that every point of PS_{ij} can be seen from x through v_iv_j. We will allow star-shaped MDs of subpolygons P_{ij} to contain pseudo star-shaped polygons based along v_iv_j. Figure 4.2b shows pseudo star-shaped polygons that will double merge to form the correct MD of the polygon. A MD of a subpolygon P_{ij} can be found using our DP formulation if all pseudo star-shaped MDs of P_{im} and P_{mj}, $i<m<j$, are computed.

Lemma 4.1 allows us to compute only $O(n^2N^2)$ star and pseudo star-shaped decompositions for each subpolygon. For each subpolygon P_{ij} and each point x in

KER we find the smallest star-shaped or pseudo star-shaped decomposition D that either contains x as a kernel vertex of the base polygon of D or uses x to see the pseudo star-shaped base polygon of D. We call such a decomposition, (MD_x), the MD of P_{ij} viewed from x. By associating a state with these MDs we have achieved a valid polynomial sized state space. We have again reduced the size of a state space by finding a small number of representative states.

Figure 4.5 shows a polygon whose only star-shaped MD contains an interior edge which connects two vertices of P which are not notches. Therefore, unlike the convex case, we must consider all subpolygons.

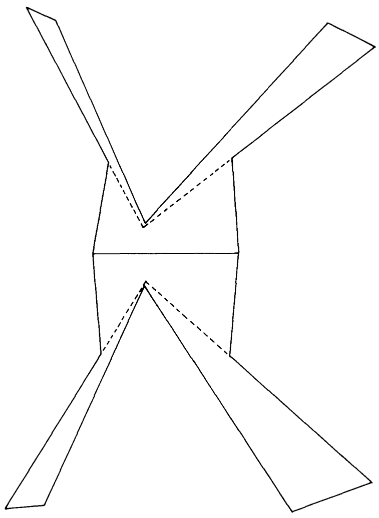
FIG. 4.5

Armed with the state space definition we can now describe the star-shaped decomposition algorithm.

Preprocessing.

1. Form the set KER of potential kernel vertices.

2. For each point $x \in$ KER store a list of the vertices of P that are visible from x.

3. For each vertex v of P determine the set of vertices of P that are visible from v and store them in a sorted list.

4. Since each visibility pair v_iv_j $(i<j)$ determined in step 3 is the base edge of a subpolygon these subpolygons can be sorted by the size measure $j-i$ in time $O(n^2 \log n)$.

5. For each subpolygon P_{ij} of P form the set of base triangles with base edge v_iv_j.

DP *procedure.*

1. Consider each subpolygon P_{ij} in the order computed in step 4 of the preprocessing.

2. Compute the set of MDs of P_{ij} as viewed from each member of KER.

If $j-i=2$, P_{ij} will consist of a single triangle. Points in KER that lie in the triangle have real MDs of P_{ij}. Points in KER that do not lie in the triangle but are visible from v_i, v_{i+1} and v_j have pseudo MDs of P_{ij}.

Otherwise for each base triangle T_{imj} of P_{ij}, for each point x in KER:

(a) First check for double merges. If both MD_x of P_{im} and MD_x of P_{mj} exist then they are merged to form a candidate for MD_x of P_{ij}.

(b) If a double-merge for x is not possible at T_{imj}, we check for single-merges. The MD_x of P_{im} single merges with T_{imj} if x is visible from v_j. To form this MD_x of P_{ij} we take the MD_x of P_{im} merged with T_{imj} together with the smallest real MD of P_{mj}. Single merges with MDs of P_{mj} are analogous.

(c) Finally, if we can perform no merges a candidate for MD_x of P_{ij} exists if x lies in T_{imj} or x is visible from v_i, v_m and v_j through edge v_iv_j. This candidate is formed by taking the smallest real MD of P_{im} together with T_{imj} and the smallest real MD of P_{mj}.

As the various candidates for the MD_x of P_{ij} are considered only the smallest is kept. Also the smallest real MD of P_{ij} will be needed when no merge past v_iv_j is performed.

3. The subpolygon $P_{1n} = P$ will be the last one considered. The smallest real MD of P_{1n} is kept as a minimum star-shaped decomposition of P.

THEOREM 4.2. *The above algorithm correctly computes the minimum star-shaped decomposition of a simple polygon.*

Proof. Lemma 4.1 implies that it is sufficient to show that we compute MD_x of P_{ij} correctly, for all $x \in \text{KER}$, for all subpolygons P_{ij}. We shall proceed by induction on the size $(j-i)$ of a subpolygon.

If $j-i=2$, P_{ij} will consist of a single triangle and clearly MD_x of P_{ij} is computed correctly if it exists. Also P_{ij} is its own smallest real MD.

As an inductive assumption let us assume that MD_x of P_{ij}, for all $x \in \text{KER}$, including the smallest real MD, have been computed correctly for all subpolygons P_{ij} with $j-i<q$. Let us consider the computation of MD_x for a subpolygon P_{ij} with $j-i=q$. Clearly there exists at least one base triangle, T_{imj}, that is contained in the base star or pseudo star-shaped polygon $S_{ij}(MD_x)$. If $S_{ij}(MD_x) = T_{imj}$ then, by the inductive assumption, we will have correctly computed MD_x by performing no merges as in part (c) of step 2 of the algorithm. If either the left side of $S_{ij}(MD_x) = v_iv_m$ or the right side of $S_{ij}(MD_x) = v_mv_j$ then, by the inductive assumption, we will have correctly computed MD_x by single merging as in part (b) of step 2 of the algorithm. If neither v_iv_m or v_mv_j lie in the boundary of $S_{ij}(MD_x)$ then, again by the inductive assumption, we will have correctly computed MD_x by double-merging as in part (a) of step 2 of the algorithm.

We can similarly compute all MD_x, for all $x \in \text{KER}$, for any subpolygon $P_{ij}$with $j-1=q$. That one of the MD_x of P_{ij} is the smallest real MD of P_{ij} is evident from Lemma 4.1. We have therefore shown by induction that all MD_x, for all $x \in \text{KER}$, for all subpolygons P_{ij} will be computed correctly.

Analysis. Step 1 of the preprocessing requires $O(n^2N^2)$ time. Step 5 of the preprocessing requires $O(n^3)$ time. In the DP procedure, given a viewing vertex x a double merge can be found in $O(\log n)$ time at a base triangle if the MDs of a subpolygon are stored sorted by viewing vertex. A single merge between T_{imj} and MD_x of P_{im} can be detected in $O(\log n)$ time by searching for v_j in $V(x)$. Since there are $O(n^2N^2)$ points in KER, $O(n^2N^2 \log n)$ time is spent at each base triangle. Since there are $O(n^3)$ base triangles, altogether the algorithm may require $O(n^5N^2 \log n)$ time to find the minimum star-shaped decomposition.

5. Minimum edge length decompositions.

In some applications a decomposition that minimizes the total length of the internal edges used to form the decomposition is useful. Figure 5.1 shows that such a decomposition can be quite different from a decomposition that minimizes the number of components. Lingas et al. [19] have developed an $O(n^4)$ time algorithm which decomposes a rectilinear polygon into rectangles while minimizing the total internal edge length. Using our DP approach we are able to develop polynomial time algorithms to decompose a simple polygon into each of the component types while minimizing the amount of "ink" necessary.

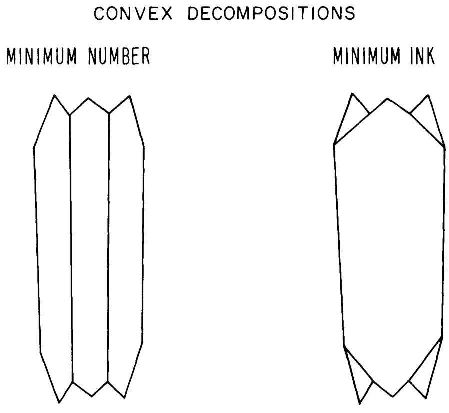

FIG. 5.1. *Convex decompositions.*

5.1. Convex decompositions. Since the convex minimum "ink" decomposition seems to have the widest range of applications we will treat it first. Most of the terms we defined when dealing with the convex minimum number decomposition problem will also be useful here. Refer to Fig. 2.1. We define the *length* of a convex decomposition D to be the sum of the lengths of the internal line segments forming D. A *free MD* of a subpolygon P_{ij} is a convex decomposition D of P_{ij} such that if D' is any other convex decomposition of P_{ij} then length $(D') \geqq$ length (D).

Independently Greene [11] has noticed that his algorithm for the convex minimum number problem [10] can be adapted to yield an $O(N^2n^2)$ time algorithm for the convex minimum edge length problem. By developing our algorithm for that problem with our general DP approach we are able to easily adapt the algorithm for convex components to other simple types of component polygons. To begin we define the state space by letting the states be of the form s_{ij} where s_{ij} has the interpretation that subpolygon P_{ij} has been decomposed minimally. That is, we associate a free MD of P_{ij} with s_{ij}. Decisions are as before and the cost of a policy that leads to s_{ij} is equal to the length of a MD of P_{ij}. As we expect this DP formulation is not valid. Figure 5.2 shows that the free MD of P_{im} cannot merge to form the free MD of P_{ij}.

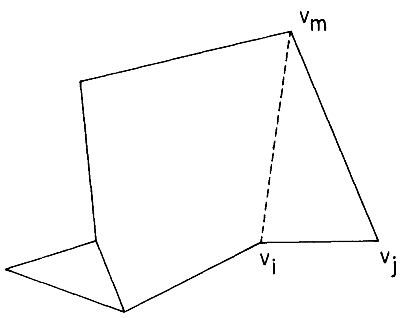

FIG. 5.2

Here we encounter a major difference between the minimum "ink" and the minimum number problems. In the minimum number case if edge v_iv_m was not part of a MD of P_{ij} then one MD of P_{im} would merge with T_{imj}. Figure 5.2 shows that this is not the case with minimum "ink" decompositions. We would therefore gain nothing if we create a state for each free MD of a subpolygon.

Instead we introduce fixed MDs. A *fixed MD* of a subpolygon P_{ij} with a given left edge and right edge is the convex decomposition D of P_{ij} such that if D' is any other convex decomposition of P_{ij} with the same left edge and right edge as D then length $(D') \geqq$ length (D). The usefulness of fixed MDs is established in the following lemma.

LEMMA 5.1.1. *Any members of the class of convex decompositions of a subpolygon P_{im} with a given pair of left and right edges are equivalent in terms of constructing a decomposition of subpolygon P_{ij} $(j > m)$ by merging with T_{imj}.*

Proof. Analogous to that of Lemma 2.1.

Since we are seeking a minimum edge length decomposition it is clear that a state space that includes a state for each fixed MD of a subpolygon is valid. As there are only $O(n^2)$ possible pairs of left and right edges of decompositions in a subpolygon there are only a polynomial number of states in the state space and the resulting DP algorithm runs in polynomial time.

Even in this polynomial state space there are states that will not be used in constructing the optimal solution. Lemma 2.2, which states that no MD of P contains an interior edge which connects two vertices of P which are not notches, remains valid for minimum edge length decompositions. We therefore need only consider states that are associated with a fixed MD of a valid subpolygon.

The following definitions allow us to increase the efficiency of the algorithm by placing an ordering on the states. A set X of fixed MDs of a subpolygon P_{ij} has the *left increasing right fixed* (LIRF) *property* for a given right angle ϕ if it contains all fixed MDs of P_{ij} with the given right angle ϕ subject to the following constraint. If a fixed MD A of P_{ij} with right angle ϕ has left angle θ_1 and a fixed MD B of P_{ij} with right angle ϕ has left angle θ_2 so that $\theta_1 \geqq \theta_2$ then A cannot be a member of X unless length (A) is less than length (B). If X is sorted in increasing order of left angle then it is also sorted in decreasing order of length. The *right increasing left fixed* (RILF) *property* is defined analogously.

We are now ready to describe the DP algorithm in more detail.

Preprocessing. We use the same four preprocessing steps that we used in the algorithm for computing the decomposition of a simple polygon into the minimum number of convex components.

DP *procedure.*

1. Consider each valid subpolygon P_{ij} in the order computed in the preprocessing.

2. Compute a set of fixed MDs of P_{ij} with the LIRF property for each valid right edge of a decomposition of P_{ij} and compute a set of fixed MDs of P_{ij} with the RILF property for each possible left edge of a decomposition of P_{ij}.

These computations are done as follows.

If $j - i = 2$, the only MD is a triangle.

Otherwise for each base triangle T_{imj}.

(a) Compute a set X of the fixed MDs of P_{ij} with left edge v_iv_m with the RILF property. First all fixed MDs of P_{ij} with left edge v_iv_m are found as follows.

Take a free MD of P_{im} together with

(i) T_{imj} and a free MD of P_{mj} and

(ii) T_{imj} merged with the smallest MD of P_{mj} that will merge for each possible valid right edge of a MD of P_{mj}. The smallest MD in P_{ij} with a given right edge is found using binary search in a set with the LIRF property for the given right edge.

Now that all fixed MDs of P_{ij} with left edge v_iv_m have been placed in a set X we remove some MDs so that X will have the RILF property. We have the MDs in X sorted in increasing order of right angle. Set a variable L to the length of the MD in X with the smallest right angle. X is scanned in increasing order of right angle. If the MD D under scan has length greater than or equal to L we delete D from X. Otherwise we set L to the length of D. We continue the scan until all MDs in X have been examined.

(b) A set of fixed MDs of P_{ij} with right edge v_mv_j with the LIRF property is found similarly.

At this point there may be valid left edges v_iv_m in subpolygons of type (b) for which no set of MDs has been found and similarly valid right edges in subpolygons of type (c) for which no set of MDs have been found. This will happen when v_iv_m and v_mv_j are not both valid edges so that valid base triangle T_{imj} does not exist. Figure 5.3 illustrates the situation.

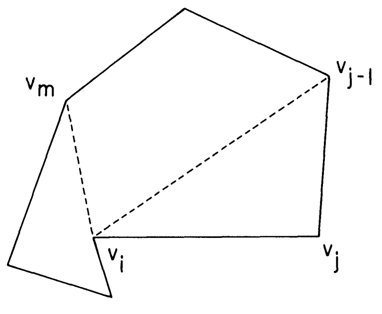

FIG. 5.3. *The only base triangle of P_{ij} is $T_{i(j-1)j}$.*

Consider a valid left edge v_iv_m in a subpolygon of type (b) such that T_{imj} is not a valid base triangle of P_{ij}. No set of MDs with left edge v_iv_m has been explicitly collected. However MDs with left edge v_iv_m have been encountered while collecting MDs with various right edges. Each time a MD with the valid left edge v_iv_m, such that T_{imj} is not a valid base triangle of P_{ij}, is encountered we should set it aside. When all valid base triangles have been processed we will have several sets of MDs corresponding to these stray left edges. These sets can be given the RILF property using the scanning procedure described above. Similarly sets of MDs with the LIRF property are found for valid right edges v_mv_j in subpolygons of type (c) for which T_{imj} is not a valid base triangle. In this way all required MDs of a subpolygon are found.

The free MD of a subpolygon P_{ij} is the smallest of all the fixed MDs found.

3. The subpolygon $P_{1n} = P$ will be the last one considered. A free MD of $P_{1,n}$ is the desired MD of P.

Proof of correctness.

LEMMA 5.1.2. *Let D be a fixed* MD *of a subpolygon P_{ij} with left edge v_iv_l and right edge v_rv_j. Then there exists an $O(\log n)$ time algorithm that forms a fixed* MD *D' of P_{ij} with the same left and right edges as D. The algorithm uses only free* MD*s of P_{il}, P_{lj}, P_{ir} and P_{rj} and a set of fixed* MD*s of P_{ir} with the* RILF *property with left edge v_iv_l and a set of fixed* MD*s of P_{lj} with the* LIRF *property with right edge v_rv_j.*

Proof. Assume P_{ij} is of type (a) or (b). The proof is analogous if P_{ij} is of type (c). When P_{ij} is of type (a) or (b) then T_{imj} exists so that $v_m = v_r$. The decomposition D' required can be found by computing decomposition D' in part i or ii.

(i) If $v_l = v_r$ then $C_{ij}(D)$ is a triangle (i.e. T_{imj}). Clearly the decomposition D' formed by taking a free MD of P_{il} and a free MD of P_{rj} will have the same length, left edge and right edge as D.

(ii) If $v_l \neq v_r$ then C_{ij} is not a triangle and we must consider MDs of P_{im} that merge with T_{imj}. Let A' be the smallest fixed MD of P_{im} with left edge v_iv_l that will merge with T_{imj}. A' is found in $O(\log n)$ time by binary search in the set of MDs of P_{im} with the RILF property with left edge v_iv_l. If A is the restriction of D to $P_{im} \cup T_{imj}$ then clearly the length of A' is at least as small as the length of A. The desired MD D' is formed by taking A' merged with T_{imj} together with a free MD of P_{mj}.

The next lemma shows how the required sets of MDs for each subpolygon can be found.

LEMMA 5.1.3. *There is an algorithm that runs in $O(n^2 \log n)$ time for subpolygons of type* (a) *and in $O(nN \log n)$ time for subpolygons of types* (b) *or* (c) *that correctly finds a set with the* LIRF *property for all valid right edges in P_{ij} and a set with the* RILF *property for all valid left edges of P_{ij}. The procedure uses only a free* MD *of P_{im} and P_{mj} and a set with the* LIRF *property for all valid right edges in P_{mj} and a set with the* RILF *property for all valid left edges of P_{im} where T_{imj} is a base triangle of P_{ij}.*

Proof. Each fixed MD of P_{ij} can be found in $O(\log n)$ time using the algorithm of Lemma 5.1.2. There are $O(n^2)$ fixed MDs in a subpolygon of type (a) and $O(nN)$ fixed MDs in a subpolygon of type (b) or (c). These MDs are organized into sets with the RILF property for valid left edges of P_{ij} and into sets with the LIRF property for valid right edges of P_{ij}. This is done by sorting the MDs in each set and performing and scanning procedure described in the algorithm. A free MD of P_{ij} is found by keeping track of the smallest fixed MD of P_{ij}.

Using the above lemmas we can prove the following.

THEOREM 5.1.4. *The algorithm finds a minimum edge length decomposition of a simple polygon in $O(N^2n^2 \log n)$ time in the worst case.*

Proof. The preprocessing requires $O(N^2 n \log n)$ time. There are $O(N^2)$ subpolygons of type (a) at which $O(n^2 \log n)$ time is spent computing fixed MDs as in Lemma 5.1.3. There are $O(Nn)$ subpolygons of type (b) or (c) at which $O(Nn \log n)$ time is spent computing fixed MDs as in Lemma 5.1.3. Altogether $O(N^2n^2 \log n)$ time is required to compute the fixed and free MDs of each subpolygon. Since $P_{1n} = P$ a free MD of P_{1n} is the desired minimum edge length decomposition of P.

5.2. Star-shaped decompositions. For star-shaped decompositions we define fixed MDs as follows. A fixed MD of a subpolygon P_{ij} with a given kernel point $x \in \text{KER}$ is the star-shaped (or pseudo star-shaped) decomposition D of P_{ij} such that if D' is any other star-shaped (or pseudo star-shaped) decomposition of P_{ij} with the same kernel point x then length $(D') \geq$ length (D).

Lemma 4.1 allows us to compute only $O(N^2n^2)$ fixed star and pseudo star-shaped minimum decompositions of each subpolygon. For each subpolygon we associate one state with each point in KER that can either be a kernel vertex of a base polygon of a fixed MD of P_{ij} or a vertex than can see a fixed pseudo star-shaped base polygon through v_iv_j. The resulting state space is valid and of polynomial size.

Figure 4.5 shows a polygon whose star-shaped minimum edge length decomposition contains an interior edge which connects two vertices which are not notches. Therefore, we again must consider all subpolygons as possible components.

The details and analysis of the star-shaped minimum edge length decomposition algorithm are very similar to those of the star-shaped minimum number decomposition algorithm. Altogether $O(N^2 n^5 \log n)$ time may be required to find a star-shaped minimum edge length decomposition.

6. Other problems. Recall that a spiral polygon is a simple polygon whose boundary chain contains at most one concave subchain. That is, a spiral polygon has at most one set of adjacent notches. Feng and Pavlidis [9] give an algorithm for decomposing a simple polygon into spiral polygons. Their algorithm does not introduce Steiner points and does not generally yield a minimum decomposition. Our DP formulation for the spiral decomposition problem is similar to that for the convex decomposition problem. In the spiral case, base angles which are notches are allowed. Some double merges may then be necessary as illustrated in Fig. 6.1. Also some of the convex equivalence classes must be subdivided. A class of MDs with a given pair of left and right angles are not all equivalent in their ability to merge. A MD with a convex base polygon can spiral merge with some base triangles where a MD with a nonconvex base polygon could not. These changes do not significantly affect the DP approach and we are able to develop polynomial time algorithms for both the minimum number and minimum "ink" spiral decomposition problems [14].

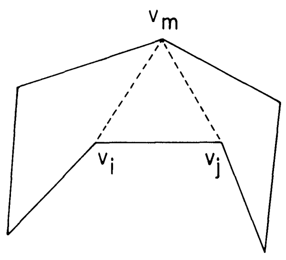

FIG. 6.1

Recall that a monotone polygon contains two extreme vertices in a preferred direction such that they are connected by two polygonal chains monotonic in the preferred direction. Lee and Preparata [17] give an $O(n \log n)$ algorithm for decomposing a simple polygon into monotone polygons without using Steiner points. Their algorithm does not generally yield a minimum decomposition. A valid DP formulation for the monotone decomposition problem could have a state for all monotone MDs of a subpolygon. This state space can be reduced by recognizing that MDs having base polygons monotone with respect to the same directions are equivalent. A convex polygon is monotone with respect to all directions and it is only sides which are adjacent to notches in a polygon that can eliminate directions of monotonicity [24]. The sides of a subpolygon divide the set of all directions into a polynomial number of classes of direction. If a subpolygon is monotone with respect to one direction in a class it is monotone with respect to all directions within that class. Therefore the state space can be further reduced by keeping only one representative MD for each of the classes of direction. Since there are only a polynomial number of such classes a polynomial DP algorithm results. This approach will work for both the minimum number and the minimum "ink" decomposition problems.

The complexity of a convex decomposition problem may increase if we allow the polygon to contain holes that must be avoided. Holes are nonoverlapping "island" simple polygons inside the main polygon. Lingas [18] has shown that if Steiner points are allowed the problem of decomposing a polygon with polygonal holes into the minimum number of convex components is NP-hard. O'Rourke and Supowit [22] show that if Steiner points are allowed and overlapping of components is allowed decomposing a simple polygon with polygonal holes into the minimum number of convex, star-shaped or spiral components is also NP-hard. These proofs can be adapted [14] to prove that decomposing a polygon with polygonal holes into the minimum number of each of convex, star-shaped or spiral components is NP-hard when Steiner points are disallowed. It can be shown using a component design proof that decomposing a polygon with polygonal holes into the minimum number of monotone components is NP-hard and that decomposing a polygon with polygonal holes into convex components using minimum ink is also NP-hard [14].

7. Further research. If the introduction of Steiner points is allowed, most of the polygon decomposition problems remain open. The only result of this type in Chazelle and Dobkin's [4], [5], [6] polynomial time algorithm for the problem of decomposing a simple polygon into the minimum number of convex components.

Acknowledgment. I would like to thank Derek Corneil for many useful discussions on these problems.

REFERENCES

[1] D. Avis and G. T. Toussaint, *An efficient algorithm for decomposing a polygon into star-shaped polygons*, Pattern Recognition, 13 (1981), pp. 395-398.
[2] R. Bellman, *Dynamic Programming*, Princeton Univ. Press, Princeton, NJ, 1957.
[3] K. Q. Brown, *Dynamic programming in computer science*, Computer Science Dept. Report CMU-CS-79-106, Carnegie-Mellon Univ., Pittsburgh, 1979.
[4] B. Chazelle and D. Dobkin, *Decomposing a polygon into its convex parts*, Proc. 11th Annual ACM Symposium on Theory of Computing, 1979, pp. 38-48.
[5] B. Chazelle, *Computational geometry and convexity*, Ph.D. Thesis, Yale Univ., New Haven, CT, 1980.
[6] B. Chazelle and D. Dobkin, *Optimal convex decompositions*, in Computational Geometry, North-Holland, Amsterdam, 1984.
[7] H. El-Gindy and D. Avis, *A linear algorithm for determining visibility from a point in a polygon*, J. Algorithms, 2 (1981), pp. 186-197.
[8] S. E. Elmaghraby, *The concept of "State" in discrete dynamic programming*, J. Math. Anal. Appl., 29 (1970), pp. 523-557.
[9] H. Feng and T. Pavlidis, *Decomposition of polygons into simpler components: feature generation for syntactic pattern recognition*, IEEE Trans. Comput., C-24 (1975), 636-650.
[10] D. Greene, *The decomposition of polygons into convex parts*, Manuscript, Xerox Parc, 1982.
[11] ———, Private communication, 1982.
[12] T. Ibaraki, *Minimal representations of some classes of dynamic programming*, Inform. and Control, 27 (1975), pp. 289-328.
[13] R. M. Karp and M. Held, *Finite-state processes and dynamic programming*, SIAM J. Appl. Math., 15 (1967), pp. 693-718.
[14] J. M. Keil, *Decomposing polygons into simpler components*, Ph.D. Thesis, Univ. Toronto, Toronto, 1983.
[15] J. M. Keil and J.-R. Sack, *Minimum decompositions of polygonal objects*, in Computational Geometry, North-Holland, Amsterdam, 1984.
[16] H. T. Kung, F. Luccio and F. P. Preparata, *On finding the maxima of a set of vectors*, J. Assoc. Comput. Mach., 22 (1975), pp. 469-476.
[17] D. T. Lee and F. P. Preparata, *Location of point in a planar subdivision and its applications*, this Journal, 6 (1977), pp. 594-606.
[18] A. Lingas, *The power of non-rectilinear holes*, Proc. 9th Colloquium Automata, Languages and Programming, Aarhus, 1982.
[19] A. Lingas, R. Pinter, R. Rivest and A. Shamir, *Minimum edge length decompositions of rectilinear figures*, unpublished manuscript, MIT, 1981.

[20] W. LIPSKI, E. LODI, F. LUCCIO, C. MUGNAI AND L. PAGLI, *On two-dimensional data organization* II, Fundamenta Informaticae, 2 (1979).

[21] W. NEWMAN AND R. SPROULL, *Principles of Interactive Computer Graphics*, Second ed., McGraw-Hill, New York, 1979.

[22] J. O'ROURKE AND K. J. SUPOWIT, *Some* NP-*hard polygon decomposition problems*, IEEE Trans. Information Theory, IT-29 (1983), pp. 181-190.

[23] T. PAVLIDIS, *Analysis of set patterns*, Pattern Recognition, 1 (1968), pp. 165-178.

[24] F. P. PREPARATA AND K. J. SUPOWIT, *Testing a simple polygon for monotonicity*, Inform. Proc. Letters, 12 (1981), pp. 161-164.

[25] B. SCHACHTER, *Decomposition of polygons into convex sets*, IEEE Trans. Comput., C-27 (1978), pp. 1078-1082.

[26] G. T. TOUSSAINT, *Pattern recognition and geometrical complexity*, Proc. Fifth International Conference on Pattern Recognition, Miami Beach, 1980, pp. 1324-1347.