FURTHER COMMENTS ON BYKAT'S CONVEX HULL ALGORITHM

Mark H. OVERMARS and Jan van LEEUWEN
Department of Computer Science, University of Utrecht, 3508 TA Utrecht, The Netherlands

Received 10 October 1979; revised version received 17 March 1980)

Algorithm, convex hull, expected time complexity

1. Introduction

In a recent paper, Bykat [4] presented an algorithm for computing the convex hull of a set of n points in the plane and claimed a worst case complexity of O(n log n) steps for it. Soon after, Fournier [8] noted that the analysis was wrong and indicated how the algorithm could be forced to make $\Omega(n^2)$ steps. The same observation was made recently in a short note by Dévai and Szendrényi [6].

The prime objective of this paper is to point out that Bykat's algorithm is identical to Eddy's [7] published one year earlier (as noticed independently by Andrew [1] also) and that the comments of Fournier and of Dévai and Szendrényi, however appropriate, are completely subsumed by the analysis Eddy already provided for the algorithm. The essential idea of the algorithm is reported to have been developed independently by R.W. Floyd before, also for sets of points in higher dimensions (cf. [2]). Eddy [7] proved that the algorithm's runtime on arbitrary sets of n points in the plane is bounded by O(n h), where h is the actual number of extreme points on the convex hull, and reported useful facts about the implementation of his routine and its observed performance in practice. Although the worst case can be bad, experimental evidence supplied by Eddy [7] suggests that the algorithm will do much better on the average.

It appears that the distributions of points on which the Bykat–Eddy algorithm will do badly are very special and unlikely to occur, and a better runtime is to be expected when some assumptions about the distributions to arise in practice are made. After presenting a slightly modified and recursive description of the algorithm, we shall prove that the expected runtime of the Bykat–Eddy algorithm is *linear* when points are chosen randomly from a uniform distribution within some bounded convex region. Within the framework set up by Dévai and Szendrényi [6] it is easy to show that, in fact, the expected runtime of the algorithm is bounded by $(c_0 + 4c_1)n$, where c_0 and c_1 are implementation dependent constants [5]. It follows that the Bykat–Eddy algorithm is provably as good as any other algorithm for convex hull determination in the plane (provided the same assumption on the distribution of points is in effect), including the divide-and-conquer algorithm suggested in [3].

2. Expected time complexity of the Bykat–Eddy algorithm

Let $P = \{p_1, ..., p_n\}$ be a set of points in the plane for which the convex hull must be determined. We assume that each point is given by means of its Cartesian coordinates. The Bykat–Eddy algorithm is based on the following elementary result:

Lemma 1. Given two distinct points L and R, the point p_i that has greatest distance from the line $\overline{LR}$ must be an extreme point of P's convex hull.

It is not hard to show (see e.g. [4]) that a point p_i as referred to in the lemma can always be found in O(n) steps, assuming unit cost for single arithmetic operations. The Bykat–Eddy algorithm employs Lemma 1 in the following manner. We shall only give a brief account of the steps of the algorithm. For details see [4] or [7]. We describe the initial phase first.

{Initialization}
Step 1: Determine the points L and R in P which have smallest and largest x-coordinate, respectively (see Fig. 1). L and R belong to the convex hull of P.
Step 2: Determine the points T and B in P which are at greatest distance 'above' and 'below' the line $\overline{LR}$, respectively (see Fig. 2). T and B again belong to the convex hull of P. For clarity we omit the minor modifications necessary in degenerated cases.
Step 3: Consider the parallelogram G whose vertical sides through L and R run parallel to the y-axis and whose 'horizontal' sides through T and B run parallel to $\overline{LR}$ (see Fig. 3). Note that all elements of P lie within G. Compute the subsets S^1, S^2, S^3 and S^4 of P which are contained in the (usually) 4 triangular regions of $G \backslash \overline{LTRB}$, respectively.
{End of initialization}

It is clear that the 'contour' of the convex hull runs from L to T through S^1, continues on to R through S^2 and so on. We shall determine the parts of the contour by subdividing each triangular region into smaller pieces (which will be triangles again) through which the contour must run, until we get to single points. The procedure BE for it is most easily described recursively (although previous authors did not do so).

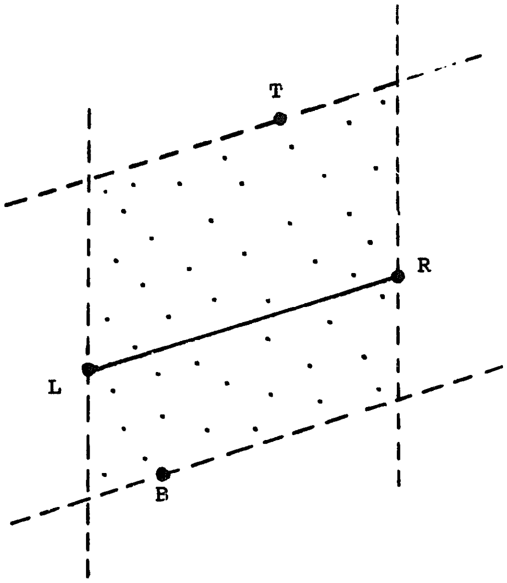

Fig. 2.

procedure BE(S, L, Q, R).
{BE is called when we have separated off a non-empty set of points S and a triangle $\overline{LQR}$ containing S, and we know that the contour of the convex hull must run from L to R through $\overline{LQR}$ (see Fig. 4)}
begin
Step 1: Find the point M in S at greatest distance from $\overline{LR}$. M must belong to the convex hull. Imagine a line $\overline{Q_1Q_2}$ through M parallel to $\overline{LR}$ (see Fig. 4).

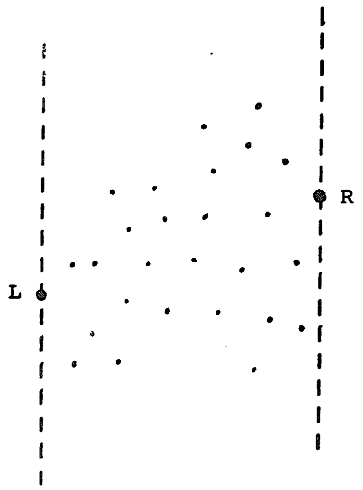

Fig. 1.

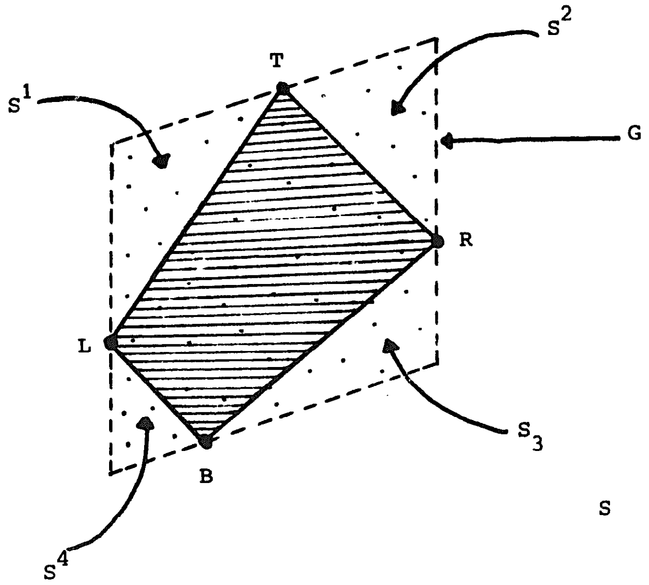

Fig. 3.

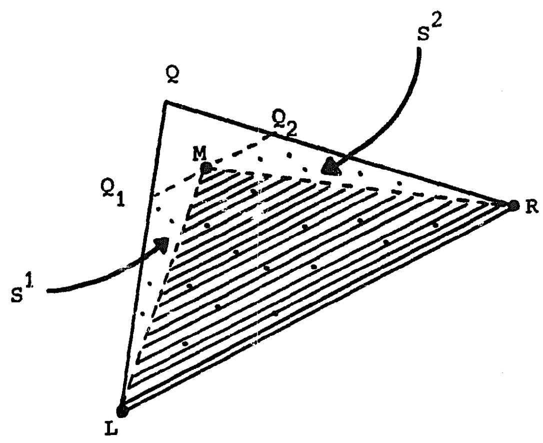

Fig. 4.

Note that all elements of S lie within $\overline{LQ_1Q_2R}$. Compute the subsets S^1 and S^2 of S contained in $\overline{LQ_1M}$ and $\overline{MQ_2R}$, respectively.

Step 2: If $S^1 = \emptyset$ then output(L) else call BE(S^1, L, Q_1, M). If $S^2 = \emptyset$ then output(M) else call BE(S^2, M, Q_2, R).

{One easily verifies that, by the time the recursive calls have 'returned', the desired part of the convex hull is presented with its points listed in clockwise order}

end

{End of BE}

After initializing, the Bykat–Eddy algorithm simply makes the appropriate calls of procedure BE to fill in the parts of the convex hull.

The expected run-time of a convex hull algorithm depends heavily on the probability distribution assumed for points in the plane. In fact one must begin by delineating a region of bounded Lebesgue measure to which the distribution is applied, in order that the problem be meaningful at all (see [3]). Our main result is the following:

Theorem 1. If n points are chosen independently at random from a uniform distribution in a bounded convex region F, then the expected time complexity of the Bykat–Eddy algorithm is O(n).

Proof. The initialization only takes O(n) steps, as one readily verifies. Note that LTRB $\subseteq$ F (see Fig. 3) by convexity. An elementary geometric argument shows that

$$\sum_1^4 \text{area}(S^i) = \text{area}(\overline{LTRB}) = \tfrac{1}{2}\,\text{area}(G),$$

where G was the enclosing parallellogram. While the assumed distribution only applies to F, we know that all points must lie within G and can conclude that the expected number of points in $S^1 \cup \cdots \cup S^4$ is (at most) $\frac{1}{2}$n.

Next consider the procedure BE, operating on each of the S^i. Imagine that the recursions are expanded in parallel. Step 1 of BE takes only linear time. The subtriangles that are split off at any level of recursion have a joint area less than $\frac{1}{2}$ the area of the triangles that are being split, as one can easily verify by inspecting each triangle separately (see Fig. 4). Note for each triangle that $\overline{LMR} \subseteq$ F by convexity and that the subtriangles split off actually have a joint area at most equal to the area of $\overline{LMR}$. Hence the expected number of points contained in the joint subtriangles considered at the next level of recursion is at most $\frac{1}{2}$ the number of points contained in the triangles at the current level. It follows that the expected number of steps BE performs, when summed over the levels of recursion, must be proportional to at most

$$\tfrac{1}{2}n + \tfrac{1}{4}n + \cdots$$

which is O(n).

Referring to the general expected time analysis of the Bykat–Eddy algorithm as given by Dévai and Szendrényi [6], the above argument shows that $\max(\alpha+\beta) \leqslant \frac{1}{2}$, i.e., the assumption that $\max(\alpha+\beta) < 1$ which Dévai and Szendrényi had to make in their analysis is indeed valid for a uniform distribution of points. It follows from [6] (also [5]) that the expected time complexity of the Bykat–Eddy algorithm can be estimated more precisely at $(c_0 + 4c_1)n$, where c_0 and c_1 are implementation dependent constants. It would be interesting to try and extend Theorem 1 to other probability distributions as well.

References

[1] A.M. Andrew, Another efficient algorithm for convex hulls in two dimensions, Information Processing Lett. 9 (1979) 216–219.

[2] J.L. Bentley, Notes on a taxonomy of planar convex hull algorithms, Department of Computer Science, Carnegie Mellon University, Pittsburgh (1980) in preparation.
[3] J.L. Bentley and M.I. Shamos, Divide and conquer for linear expected time, Information Processing Lett. 7 (1978) 87–91.
[4] A. Bykat, Convex hull of a finite set of points in two dimensions, Information Processing Lett. 7 (1978) 296–298.
[5] F. Dévai, private communication (1979).
[6] F. Dévai and T. Szendrényi, Comments on convex hull of a finite set of points in two dimensions, Information Processing Lett. 9 (1979) 141–142.
[7] W.F. Eddy, A new convex hull algorithm for planar sets, ACM Trans. Math. Software 3 (1977) 398–403.
[8] A. Fournier, Comments on convex hull of a finite set of points in two dimensions, Information Processing Lett. 8 (1979) 173.