AN OPTIMAL ALGORITHM FOR INTERSECTING THREE-DIMENSIONAL CONVEX POLYHEDRA

Bernard Chazelle

CS-TR-205-89

February 1989

An Optimal Algorithm for Intersecting Three-Dimensional Convex Polyhedra

BERNARD CHAZELLE
Department of Computer Science
Princeton University
Princeton, NJ 08544

Abstract: We describe a linear-time algorithm for computing the intersection of two convex polyhedra in 3-space. Applications of this result to computing intersections, convex hulls, and Voronoi diagrams are given.

The author wishes to acknowledge the National Science Foundation for supporting this research in part under Grant CCR–8700917.

1. Introduction

Given two convex polyhedra in 3-space, how fast can we compute their intersection? Over a decade ago, Muller and Preparata [22] gave the first efficient solution to this problem by reducing it to a combination of intersection detection and convex hull computation. Another route was followed by Hertel et al. in 1985, who solved the problem by using space sweep [16]. In both cases, a running time of $\Theta(n \log n)$ was achieved, where n is the combined number of vertices in the two polyhedra. Resolving the true complexity of the problem, however, remained elusive.

The different but related problem of *detecting* whether two convex polyhedra intersect, using preprocessing, was studied in (Chazelle and Dobkin [5], Dobkin and Munro [9], Dobkin and Kirkpatrick [6]). More germane to our concerns here is the off-line version of the detection problem. Dobkin and Kirkpatrick [7] have shown that detecting whether two convex polyhedra intersect can be done in a linear number of operations. By stating the problem as a linear program over three variables, other linear-time algorithms fall out of the works of Megiddo [19] and Dyer [10]. Previous results also include an efficient algorithm for intersecting two polyhedra, one of which is convex (Mehlhorn and Simon [21]). Optimal solutions for intersecting convex polygons are given in (Shamos and Hoey [27], O'Rourke et al. [23]). For additional background material on polyhedral intersections, the reader should consult (Edelsbrunner [11], Mehlhorn [20], Preparata and Shamos [25]).

Our main result is an algorithm for constructing the intersection between two convex polyhedra in linear time. The algorithm does not use any complicated data structure and is a good candidate for practical implementation. As is customary, our result assumes that the input conforms with any one of the standard (linear-time equivalent) polyhedral representations given in the literature [3,15,22]. This is not a minor point, because nonstandard representations can easily make the problem more difficult. (Think of giving the vertices but hiding all other facial information, for example.) From the algorithm for pairwise intersections we immediately derive an efficient method for intersecting k convex polyhedra. The complexity of the algorithm is $O(n \log k)$, where n is the total number of vertices, which is provably optimal. Other immediate applications include merging Voronoi diagrams in two dimensions and computing convex hulls in 3-space.

2. Polyhedra and Shields

At the heart of Dobkin and Kirkpatrick's detection [6] and separation algorithms [7] is an ingenious hierarchical representation of a convex polyhedron. Further applications of that versatile data structure have been given in [12,21]. The representation can be seen as a specialization of Kirkpatrick's point location structure [18]. A convex polyhedron P of n vertices is made the first element of a descending chain of $O(\log n)$ nested convex polyhedra, such that the last one has a constant number of vertices and the others differ from their immediate predecessors by shelling off small, disjoint polyhedral cones. We will want to go further and modify this representation in three ways. First, we represent the set of approximating polyhedra as a single geometric object, namely, a simplicial cell complex, so we can walk freely through it. Second, we discard approximating polyhedra that are too distant from P. Last but not least, we supplement what is essentially an approximation scheme

from the inside by one emanating *from the outside* as well. All of this results into a so-called *shield* of P.

The main idea behind the algorithm is to compute the intersection and the convex hull of the two input polyhedra *simultaneously*, with either one of the two tasks *driving* the other in an alternate fashion. The convex hull part of the algorithm is Jarvis-like, and thus best performed in dual space. Shields are used to guide the exploration of the polyhedra, an operation called *broadcasting*. Deciding which polyhedron gets to direct the broadcasting at what time is a critical aspect of the algorithm. The directing polyhedron is called the *anchor*. Costs are balanced by shifting between anchors through a recursive form of dovetailing. Proving the correctness of the algorithm entails a thorough geometrical and topological investigation of the union of two convex polyhedra.

A. Background. We begin with some useful terminology. Given $X \subseteq \Re^d$, the closure (resp. interior) of X is denoted $cl\,X$ (resp. $int\,X$). The frontier of X is defined as $cl\,X \cap cl\,(\Re^d \setminus X)$, or as $cl\,X \cap cl\,(A \setminus X)$ if the relative topology of some $A \supseteq X$ is understood. The unit d-sphere and the open unit d-ball are denoted S^d and D^d, respectively. A disjoint union of k-faces (subsets of $\Re^d$ homeomorphic to D^k) is called a d-dimensional *cell complex*, if given any two faces f and g, the intersection of $cl\,f$ and $cl\,g$ is either a union of faces or the empty set. A cell complex is called *simplicial* (or a triangulation) if each face is the interior of a simplex (in the relative topology of its affine closure).

We take a rather general view of a polyhedron as any subset of 3-space that is locally a cone with a compact base [26]. Given a point $p \in \Re^3$ and a subset $C \subset \Re^3$, we say that the points $\alpha p + (1-\alpha)q$, for all $q \in C$ and $0 \leq \alpha \leq 1$, form a *cone* pC, if for each point of pC distinct from p, the choice of q is unique. A subset P of $\Re^3$ is called a *polyhedron* if each point $p \in P$ has a cone neighborhood pC, whose *base* C is compact. The *boundary* of a polyhedron P, denoted ∂P, is just another name for its frontier. It is not hard to see that a polyhedron is a locally finite union of simplices, and hence, is piecewise linear. It need not be a manifold, however. An example of a valid polyhedron is shown in figure 2.1. An open halfplane π is a polyhedron but, because of the compactness condition, ceases to be one as soon as we include one point on its frontier. Adding the whole frontier is fine, however.

We need this level of generality because we will sometimes be dealing with rather convoluted shapes. As it turns out, most of our time will be spent with convex polyhedra, for which a more global (but slightly restrictive) definition is preferable [11]. A *convex polyhedron* is a nonempty intersection of a finite number of closed halfspaces. It is called a *convex polytope* if it is bounded. All our algorithms can be easily extended to projective space where unboundedness has no effect on the topology, so we will restrict our discussion to convex polytopes. We assume that the boundary of a convex polytope P is facially structured as a two-dimensional cell complex with a minimal set of vertices. This last requirement means that a vertex (0-face) must be the intersection of three or more bounding planes (i.e., planes that delimit the defining halfspaces), but an edge (1-face) need not lie in the intersection of two bounding planes. This assumption allows us to triangulate ∂P and still have a convex polytope; on the other hand, it forbids the facets (2-faces) incident upon any given vertex from being all coplanar. For storing two- and three-dimensional cell complexes we shall

assume the representations of (Baumgart [3], Muller and Preparata [22], or Guibas and Stolfi [15]) and (Dobkin and Laszlo [8]), respectively, or any other data structures which allow us to navigate at ease between adjacent cells. Such representations will be called *standard*.

To conclude this laundry list of assumptions and definitions, we introduce a well-known duality between points and planes, namely, the polarity δ which maps any point $p = (\alpha, \beta, \gamma)$ distinct from the origin O to the plane of equation $\alpha x + \beta y + \gamma z = 1$: $\delta(p)$ is the plane normal to Op that lies at a distance $1/|Op|$ from the origin on the same side as p. Given a convex polytope P, whose interior contains the origin, the dual of P is the set of planes whose dual points lie in P. Forming the union of all these planes and taking the closure of the complement defines a convex polytope, called the *dual polytope* of P and denoted P^δ. If the polytope P has no coplanar facets (e.g., no triangulation has been forced upon its boundary) then each vertex, edge, and facet of P corresponds respectively to a unique facet, edge, and vertex of its dual polytope, and the correspondence is involutory. Given a standard representation of a convex polytope P, it is elementary to compute a standard representation of P^δ in linear time, supplemented with pointers between each k-face of P and its dual $(2-k)$-face. Note that if the origin is not interior to P then, instead of a single polytope, we obtain one or two unbounded polyhedra in 3-space (only one in projective space).

Central to the Muller-Preparata method [22] is the clever use of the fact that convex hulls and intersections play dual roles. Indeed, if the origin lies in the interior of two convex polytopes P and Q, then the convex hull of P^δ and Q^δ is the dual polytope of $P \cap Q$. In other words, identifying the binary operation *intersection* in primal space with the binary operation *convex hull* in dual space yields the following commutative diagram:

$$\begin{array}{ccc} P, Q & \xrightarrow{\delta} & P^\delta, Q^\delta \\ \downarrow & & \downarrow \\ P \cap Q & \xrightarrow{\delta} & (P \cap Q)^\delta = \text{Hull}\,(P^\delta \cup Q^\delta) \end{array}$$

Unlike the Muller-Preparata approach, which commutes through the diagram only once, our algorithm will spend most of its time traveling back and forth between primal and dual spaces.

B. Shielding a convex polytope. We open this discussion with a variant of the Dobkin-Kirkpatrick construction. Let P be a convex polytope of n vertices with nonempty interior. We assume that its boundary has been triangulated, which is easy to ensure in linear time. Recall that the *degree* of a vertex refers to its number of incident edges. We select a maximal independent set of constant-degree vertices: (i) Pick any vertex of degree at most 8, and mark it along with all its adjacent vertices; (ii) iterate on this process, always making sure to pick unmarked vertices. Termination occurs when we run out of unmarked vertices of degree at most 8. Because the number of edges is at most $3n - 6$, we find that the sum of all vertex-degrees does not exceed $6n - 12$. Since every vertex has degree at least 3, the number m of vertices of degree at most 8 satisfies $9(n-m) \leq 6n-12-3m$, and hence $m \geq n/2$. Therefore, at least $n/18$ vertices will be selected in this process. As shown in (Edelsbrunner [11]), we can actually do better by considering vertices in order of nondecreasing degree. This allows us to find an independent set of at least $n/7$ vertices of degree at most 12. In both cases, the time for selecting the desired vertices is linear.

Around each selected vertex v, we perform some local surgery by removing v and its "umbrella" of incident faces and recomputing the convex hull of P (figure 2.2). Since v has degree at most 12, this shelling operation can be done in constant time. Note that because of the independence of the selected set of vertices, the order in which vertices are "popped out" does not matter. In $O(n)$ time we thus will have (i) removed all selected vertices and their incident faces, (ii) computed the new convex hull P_1, and (iii) triangulated its boundary. We easily verify that P_1 is a valid convex polytope; in particular, each of its vertices lies on at least three distinct bounding planes. We can now repeat this process with respect to P_1 and define a sequence of nested convex polytopes $P_0 \supset P_1 \supset \cdots \supset P_\alpha$, where (i) $P_0 = P$, (ii) P_α has constant size, and (iii) each $cl(P_i \setminus P_{i+1})$ is a collection of three-dimensional cones whose interiors are disjoint. If the interior of P_1 is empty then at most two vertices were popped out and P has at most $12+2$ vertices. To avoid any difficulty, we don't bother with P_1, and set $\alpha = 0$ whenever P has at most 14 vertices. We use the same criterion to terminate the iteration.

Our next step is to compute a triangulation of P that is compatible with all the approximating polytopes. This might be awkward to do during the shelling phase, because we may inherit the "wrong" triangulation from outer polytopes and create tetrahedra with empty interiors (figure 2.3). The difficulty is that a facet incident upon a popped-out vertex v of P_i might still contribute a portion of a facet of P_{i+1}. A simple fix is to retriangulate inside out. Assume that P_i has been given a triangulation compatible with $P_{i+1}, \ldots, P_\alpha$. Each cone of $cl(P_{i-1} \setminus P_i)$ can be triangulated directly by connecting its apex to the triangulation of its base provided by P_i. This will lead to a compatible triangulation of P in $O(n)$ time. Note that the resulting triangulation of ∂P might be different from the one we started with.

Nothing startingly novel so far. The only slight twist from the Dobkin-Kirkpatrick construction is to couch the approximation scheme as a three-dimensional triangulation. This will allow us discover P from various angles by traversing it along straight lines, shooting rays through it, and in general, exploring its geometry from within. Unfortunately, we also need to approximate P from the outside. What is unfortunate about this is that the complement of P is not convex, so we cannot play the same game over. Its dual polytope is convex, however, so it might just be the right time to jump into dual space.

After ensuring that no two facets are coplanar, by removing edges if necessary, we choose an origin O in the interior of P and we form its dual polytope P^δ. Then, we triangulate ∂P^δ and submit P^δ to the in-growing process described above. This results in a sequence of nested convex polytopes $P^\delta = \Pi_0 \supset \Pi_1 \supset \cdots \supset \Pi_\beta$, where Π_β has constant size. The triangulation of $P_0 \setminus int\, P_\alpha$ is the *primal shield* of P; its counterpart between Π_0 and Π_β is called the *dual shield*. Note that $P_0 \setminus int\, P_\alpha$ is *never* a 3-manifold (why?) Unless specified otherwise, the term "shield" refers to both its primal and dual parts. Given any integer k such that $0 \leq k \leq \min\{\alpha, \beta\}$, the triangulations of $P_0 \setminus int\, P_k$ and $\Pi_0 \setminus int\, \Pi_k$ form the *k-shield* of P. All logarithms below are to the base 2.

Lemma 2.1. *Let P be a convex polytope with nonempty interior, and let m be its added number of vertices and bounding planes. The shield of P can be constructed in $O(m)$ time. The added*

number of vertices and bounding planes in each P_k (and Π_k) is at most $3(1-1/7)^k m$. The total number of approximating polytopes is at most $13 \log n$.

Proof: Let v_k and f_k be respectively the number of vertices and bounding planes in P_k. The reason for dealing with $m_k = v_k + f_k$ is that this quantity is invariant under duality. Since the boundary of a convex polytope has Euler characteristic 2 and every facet has at least three incident edges, we derive $f_k \leq 2v_k - 4$. Each pass in the algorithm removes at least one-seventh of the vertices, therefore (i) the preprocessing is linear, and (ii) $m_k \leq 3(1-1/7)^k v_0 - 4$. ∎

C. Navigating through a shield. The usefulness of a shield owes to its being both an approximation scheme and a cell complex. Indeed, it gives us a "two-way" sequence of approximations for P, through which we can easily navigate and "discover" the boundary of P from any desired angle. This assumes that we use a proper representation like the Dobkin-Laszlo structure [8]. Without getting into the details of the data structure, let us just say that from each tetrahedron of a shield we can gain access to any of its four incident facets in constant time. Conversely, any facet leads us directly to its two incident tetrahedra. Also, the tetrahedra and facets incident upon an edge are accessible in cyclical order around the edge.

Let us consider a simple operation such as being given a ray $\vec{r}$ with a starting point in a tetrahedron of, say, the primal shield of P, and being asked to traverse the primal shield along $\vec{r}$. In general, the ray will cut through a sequence of cells alternating between tetrahedra and triangles. When this is the case, there is no difficulty in discovering the sequences of cuts on the fly, at a cost of $O(1)$ per cut. Let us call a facet of the primal shield *primitive* if it lies on the boundary of an approximating polytope. Since the popped-out cones are bounded by primitive triangles, the ray $\vec{r}$ cannot cut more than a constant number of nonprimitive triangles in a row. Consequently, the total size of the cutting sequence is proportional to the number of primitive triangles intersected by the ray. A minor difficulty arises when the ray cuts through an edge or a vertex of the shield. In the case of an edge (resp. vertex), we are faced with two (resp. three) candidate triangles to be visited next. We can break ties by making arbitrary navigational conventions. For example, we might agree always to choose the triangle that is (locally) highest (or leftmost if there are several highest ones), etc. We can also submit the ray $\vec{r}$ to a *symbolic* perturbation — see (Edelsbrunner and Mücke [13], Yap [28]). Note that we can easily generalize the mode of traversal to polygonal lines embedded in 3-space. The following summarizes our discussion.

Lemma 2.2. *The complexity of traversing the primal (resp. dual) shield along a polygonal line in three dimensions, knowing the starting cell, is proportional to the complexity of the polygonal line plus the number of approximating polytopes P_i (resp. Π_i) whose boundaries are cut during the traversal.*

3. Intersecting Two Convex Polytopes

We begin with a brief discussion of what makes the problem difficult (or, let us say more humbly, not completely trivial). We can assume that we have a point O inside both convex polytopes P

and Q, since such a witness (if any) can be discovered in linear time [7,10,19]. What remains to be done is in some sense merging P and Q. Imagine a sphere centered at O, on which ∂P and ∂Q are centrally projected. This gives us two spherical subdivisions S_P and S_Q. Merging the two subdivisions would do the job, but this might cause a quadratic blow-up. The next "smartest" move might appear to be locating each vertex of S_P in S_Q and vice-versa, which we can do in $O(n \log n)$ time, where n is the total number of vertices in P and Q [18]. Unfortunately, it is an easy exercise to prove that this complexity is optimal. Clearly, we are still seeking too much information and the subdivisions S_P and S_Q appear essentially worthless. So, let us bring in shields into the picture. How about locating each vertex of P (resp. Q) in the primal shield of Q (resp. P)? Such information might be a good start from which to launch our intersecting attack. But actually this is still asking too much: Indeed, it is not difficult to prove that locating the vertices of P in the primal shield of Q requires $\Omega(n \log\log n)$ time in the worst case. (We leave this rather useless piece of trivia as a homework.) All these attempts fail because we are working too far from the boundaries of P and Q, and thus are giving free rein to our adversary. Pairwise intersections of convex polytopes possess a rich geometric structure into which we have yet to tap seriously. The time has come for a closer look at the geometry of the intersection problem.

A. Getting the show on the road. Let us restate our assumptions: P and Q are two convex polytopes with a total of n vertices, and their interiors contain the origin O. To simplify our discussion, we shall assume that P and Q have no two coplanar facets and are in general position relative to each other: No facet (resp. edge) of one is coplanar (resp. collinear) with a facet (resp. edge) of the other, no vertex of one lies on the boundary of the other, etc. To use the cliché, relaxing these assumptions is tedious but not difficult.

Assume that the boundaries of P and Q have been triangulated by inheritance from their primal shields. Since the two polytopes are convex and contain the origin in their interiors, the boundary $\Sigma = \partial(P \cup Q)$ is the graph of $\max\{f, g\}$, where f and g are continuous functions $S^2 \mapsto \Re^+$. It follows that Σ is homeomorphic to S^2. Let us now color ∂P cyan and ∂Q magenta.† Points that are both cyan and magenta are said to be purple. The facets of Σ become monochromatic polygons. Beware: Some of them may be of nonconstant size, nonconvex, and even perforated. However, Σ can still be regarded as a two-dimensional cell complex. Of particular interest to us are the connected components of $\partial P \cap \partial Q$. These are disjoint, purple, simple cycles in the facial graph of Σ, which we call *laces*. Removing all the laces from Σ creates relatively open polyhedral surfaces, called *regions* (figure 3.1). Regions and laces partition Σ into maximal monochromatic connected subsets. Assume for the time being that Σ has at least one lace. Then, the closure of a region is a (topological) manifold with boundary, which is homeomorphic to S^2 perforated by $k \geq 1$ disjoint copies of D^2. Unlike some of the sets we will encounter later, this is a rather friendly one: it is an orientable bounded surface, its number of boundary components is k, its Euler characteristic is $2-k$, its one-dimensional Betti number is $k-1$, etc. The boundary of the manifold is also the frontier

† It occurred to us that these two primary colors have been given short shrift in the computational geometry chromatology and that a rehabilitation was long overdue.

of R in the relative topology of Σ: by extension, we call it the *boundary* of R. The k connected components of the boundary are called the *bounding* laces of the region. Note that each lace of Σ bounds exactly two regions (of opposite color). A region R, being a monochromatic component of the graph $\max\{f,g\}$, is paired with a homeomorphic component R^c of the graph $\min\{f,g\}$: this component has the opposite color of the region R and is called its *co-region* (figure 3.2 — boundaries are left untriangulated for clarity).

Note that our discussion so far has yet to pin down the convexity of P and Q. Polytopes derived from polyhedral terrains would work just the same, for example. As one might expect, our topological investigations are far from over. We will pause for a moment, however, and get on with the "primal" part of the intersection algorithm. The entire approach revolves around the existence of a *broadcaster*. This is an algorithm which, as input, takes a vertex v on a lace and a color c, and returns at least one vertex on each of the laces bounding the unique c-colored region incident upon v. Each vertex of a lace is determined by the intersection a facet of P (or Q) and an edge of Q (or P): It is understood that this correspondence should be provided in full by the broadcaster.

Lemma 3.1. *If a broadcaster is available, it is sufficient to know a vertex of one lace of Σ (or to know that there is no lace) in order to compute the entire intersection of P and Q while incurring only linear overhead on top of the broadcasting time.*

Proof: If Σ has no lace, just knowing this fact will be enough for us to compute $P \cap Q$ in linear time, by checking whether a vertex v of P lies inside or outside Q. Indeed, $P \cap Q = P$ (resp. $P \cap Q = Q$) if and only if $v \in Q$ (resp. $v \notin Q$). Suppose now that Σ has at least one lace. The key observation is that if the regions of Σ are to be made into the nodes of a graph, with arcs connecting nodes whose associated regions are bounded by a common lace, then the graph in question will be connected. Therefore, starting from the vertex given to us by the input, a standard graph traversal algorithm will allow us to discover a vertex for each lace of Σ. Since the facets of P and Q are triangles, it is elementary to compute an entire lace in time linear in its size, by tracing it from one of its vertices. Once we know all the laces in full, we can easily compute $P \cap Q$ in linear time by marking the laces in both ∂P and ∂Q and exploring the facets of the co-regions. ∎

B. The joy of co-anchored broadcasting. We begin with a form of broadcasting which is symmetric in P and Q: it is called *primal-broadcasting*. The idea is to stay glued to the boundary of $P \cap Q$ and confine our traversals to co-regions. In one broadcast, we are navigating across the primal shield of Q while exploring the boundary of P. The next time around, roles are reversed and we find ourselves traversing ∂Q. Alternation between P and Q might go on like this forever (well, not quite forever). This is the "*Good Morning America*" scenario: the broadcast is co-anchored between P and Q. This is fair, simple, and attractive, but ultimately doomed. (To know why, stay tuned.) So, instead, we shall set our sights on a "*Nightline*"-like format, where one polytope takes over the entire broadcast and performs primal as well as *dual*-broadcasting.

We begin with the co-anchoring routine, which relies exclusively on primal-broadcasting. Given a vertex v of a certain lace γ of Σ, let R be the magenta region of Σ incident upon v and let R^c be its co-region. Suppose now that the broadcaster is given the vertex v and the color magenta as input.

Its goal is to find vertices on all the laces $\gamma, \gamma_1, \ldots, \gamma_\ell$ of R. First of all, we (the broadcaster) can easily compute the entire lace γ by tracing the connected component of $\partial P \cap \partial Q$ passing through v. Since the boundaries are triangulated, the computation is linear in the size of γ.

Now we must reach out to all the other laces γ_i. Our strategy relies on the fact that the closure of a co-region is an edge-connected bounded surface. This seemingly obvious fact should not be taken for granted, as for example it does not necessarily hold for the closure of a region (figure 3.3). To support our claim, suppose that two vertices of Σ in the closure of a co-region R^c cannot be joined by a path of edges in $cl\,R^c$. Then, either one of these vertices can be separated from the other by a simple closed curve which lies entirely in $cl\,R^c$ but does not cut across any edge or vertex. It easily follows that this curve must lie in a single facet f of $cl\,R^c$ and that f is perforated. A local analysis of the perforation reveals that f contains two points p and q such that $pq \not\subseteq P \cap Q$. This contradicts the convexity of $P \cap Q$ and proves our claim. Observe that $cl\,R^c$ has vertices in $\partial P \cap \partial Q$ as well as possibly in $\partial P \setminus \partial Q$ (having previously assumed that R is magenta). The main difference is that the latter vertices are known ahead of time, while the former (the lace vertices) are discovered during the broadcast. Using standard graph traversal techniques, we can reach the vertices of all the γ_i's from v. The only problem is that whenever we visit an edge of P that crosses the boundary of Q, we need to know about it. The solution is to keep track — at all times during the broadcast — of where we are in the primal shield of Q. This means computing the intersections of the visited edges of P with the facets of the primal shield of Q. It follows trivially from Lemmas 2.1 and 2.2 that the broadcast will take $O(\rho \log n)$ time, where ρ denotes the number of edges in $cl\,R^c$. Obviously, we shall exchange the roles of P and Q if the color cyan is given as input to the broadcaster. To summarize, the cost of all the broadcasts will be proportional, up to a logarithmic factor, to the size of all the co-regions. This gives us a total broadcasting time of $O(n \log n)$.

We are now almost ready to apply Lemma 3.1. But how do we get started? We take an arbitrary *starting vertex* v of P and check whether it lies inside Q. If the answer is yes, we pursue the search and locate v in the primal shield of Q. From there, we start a regular broadcast-like routine, which involves traversing ∂P and keeping track of where we are in the primal shield of Q at all times. Either we will reach the boundary of Q, and hence, a vertex of a lace, or we won't. In the latter case, we know that $P \cap Q = P$. If v lies outside Q, on the other hand, we locate the point $w = Ov \cap \partial Q$ in the primal shield of P. Let z be one of the vertices incident upon the unique (simplicial) face of Q that contains w. Let us traverse the primal shield of P along the oriented segment wz. If we do not exit P (or if $w = z$) we are just back to the previous case, with the roles of P and Q reversed. If we do leave P, however, the exit point belongs to a lace and therefore is a valid starting vertex (or it lies on an edge incident upon one). In view of Lemmas 2.1, 2.2, 3.1, what we have now is an intersection algorithm with $O(n \log n)$ running time.

C. A topological excursion. Before we are able to describe dual-broadcasting, we must further explicate the relationship between polytopes and their duals. Everything we said of P and Q (e.g., regions, co-regions, laces) applies just the same to P^δ and Q^δ. In the following, Σ^δ will designate the analog of Σ vis-à-vis $P^\delta \cup Q^\delta$. We assume that the boundaries of all four polytopes P, P^δ, Q, Q^δ have been triangulated in accordance with their shields (and, hence, may have coplanar facets).

Correspondence between a polytope X and its dual is ensured by the usual pointers between a k-face and its dual $(2-k)$-face. This concerns the state of faces prior to boundary triangulation. With the introduction of simplicial faces, however, this representation must be slightly amended. Let us distinguish between the *old* faces of X (before triangulation of ∂X) and the *new* ones. Note that some of them, vertices in particular, are both old and new. Each vertex of X points to any one of the new facets to which it is dual, and each new facet points to its unique dual vertex. Each old edge of X points to the unique old edge of X^δ that is dual to it. Each new edge e points to the four old edges of X that are both adjacent to e and incident upon the old facet where e lies. It is a simple exercise to set up all these pointers in linear time. An attractive aspect of this representation is that, given a new facet abc of X, we can gain constant-time access not only to its dual vertex v, but also to three new facets of X^δ that are dual to a, b, and c, respectively, and incident upon v (figure 3.4).

Belts:

We will show that the laces of Σ can be individually "covered" by disjoint *belts* which dualize to laces of Σ^δ. Let us color yellow (so as to use up all subtractive primary colors) all the faces of the convex hull H of $P \cup Q$ that are faces of neither P nor Q. A maximal connected subset of yellow faces is a polyhedron (in our definition) which we call a *belt* of Σ. To be able to say more about belts we define the *envelope* of a polyhedron X (in a nonstandard manner) as the set of planes π such that (i) the affine closure of $X \cap \pi$ has dimension at least 1 and (ii) X lies entirely in either one of the closed halfspaces defined by π. It easily follows from the incidence-preserving properties of a polarity that the envelope of a belt B dualizes to a lace B^δ of Σ^δ. More specifically, as we walk along the lace B^δ, a certain plane $\pi(x)$ will be *rolling* around the belt in a continuous fashion as x goes around S^1. Therefore, B is a simple cycle of simplicial facets and edges (and no vertices), $t_0, e_0, \ldots, t_m, e_m$, where each facet t_i is incident upon the two edges e_i and $e_{i-1} \pmod{m+1}$. It follows that a belt is topologically equivalent to an open annulus $S^1 \times (0,1)$. Its frontier consists of two monochromatic connected components: one of them, denoted b_c, is cyan, while the other, b_m, is magenta. Let us look at these components more closely. We may restrict ourselves to one of them, say, b_c.

In general, b_c will be a simple closed polygonal curve, but unfortunately this might not always be the case.† Indeed, consider an old (i.e., untriangulated) facet f of P^δ that contributes at least one edge to the lace B^δ. It is certainly possible for B^δ to come in and out of f repeatedly (figure 3.5). To translate this into the language of belts, let $u_0, \ldots, u_k$ be the cycle of edges of b_c encountered while visiting $t_0, \ldots, t_m$ in that order. Identifications of the form $u_i = u_j$ might occur. The topological type of a belt enforces two important restrictions on the allowable identification patterns. First, three or more edges cannot be identified together. Second, in any sub-cycle u_i, u_j, u_k, u_ℓ, we cannot have both $u_i = u_k$ and $u_j = u_\ell$ (why?) Of course, we might have vertex identifications only and no edge identifications at all. It might even be the case that b_c consists of a single vertex, which

† A similar situation occurs in the merge step of Preparata and Hong's convex hull algorithm [24], which is also discussed in great detail in (Edelsbrunner [11]).

will happen if B^δ lies entirely in a single facet of P^δ (figure 3.6). The only reason why b_c is not always topologically equivalent to S^1 is that the boundaries of P and Q are not smooth. If they were, then b_c would actually be diffeomorphic to S^1. Thus, if we look at ∂P and ∂Q as limit sets of smooth 2-manifolds, it appears immediately that b_c is a simple closed curve which may have been pinched and collapsed along vertices and edges (figure 3.7). Observe that two distinct laces cannot share vertices or edges, but they can pass through the same (old) facet of P or Q. This implies that although two belts cannot share a common edge or facet, their frontiers might have vertices and edges in common. This fact will require special measures to be added to the intersection algorithm.

Bracelets:

A useful perspective on belts comes from looking at Σ and ∂H as the graphs of two continuous functions, respectively, φ and $\psi : S^2 \mapsto \Re^+$. Removing the kernel of $\psi - \varphi$ from S^2 leaves relatively open connected components $S_1, S_2, \ldots$, and each belt B is the graph of ψ's restriction to some distinct domain S_i. By analogy, the graph of φ's restriction to S_i is called the *co-belt* B^c of B. Belts and co-belts are homeomorphic and have the same frontiers. Therefore, the closure of $B \cup B^c$ is the frontier of a compact polyhedron $\mathcal{B}$, called a *bracelet* of Σ, whose interior is homeomorphic to an open filled torus $D^2 \times S^1$. A bracelet is the closure of a connected component of $H \setminus (P \cup Q)$ and its belt is the relative interior of the intersection of that component with ∂H. One should not hastily conclude that a bracelet is always topologically equivalent to a filled torus. It can actually assume rather contrived shapes, as the frontiers b_c and b_m might cause (homeomorphically) multiple point identifications along non-null homologous paths on the torus. This might give us a filled torus pinched at various places, or as in figure 3.6, a closed 3-ball pinched at a pair of antipodal points. Note that general position, alone, cannot prevent this type of pathology.

Again, let b_c and b_m be the frontier components of B. Since Σ is locally cyan (resp. magenta) around b_c (resp. b_m), we can define the maximal cyan (resp. magenta) connected subset B_c (resp. B_m) of B^c whose closure contains b_c (resp. b_m). We claim that B_c and B_m are joined together along a lace. To prove this claim, let $\gamma = B^c \setminus (B_c \cup B_m)$ and assume that γ contains a point p of color, say, magenta (figure 3.8). Let C be the maximal connected magenta subset of B^c that contains p. The closure of C does not intersect b_c or b_m, therefore C is a region of Σ. But, like every region, C contains a point in ∂H. To see why, consider a plane π supporting a facet of its co-region and let π^+ be the open halfplane bounded by π that does not contain the origin. Since P lies entirely outside π^+, we have $\pi^+ \cap \partial Q \subseteq C$, therefore the point of $\pi^+ \cap \partial Q$ furthest away from π is a vertex of Q in $C \cap \partial H$. But this contradicts our previous observation that the portion of the bracelet $\mathcal{B}$ that lies in ∂H is confined to the closure of its belt. Consequently, γ is a purple curve whose removal from B^c creates two monochromatic surfaces of opposite color, each homeomorphic to an open annulus. It follows that γ is homeomorphic to S^1 and therefore is a lace of Σ. Thus, we have shown that $\partial\mathcal{B}$ is the disjoint union of B, b_c, B_c, γ, B_m, and b_m. In general, the removal of any one of these six sets makes $\partial\mathcal{B}$ homeomorphic to an annulus (open for the lower-case sets, closed for the upper-case ones). But one should not count on it. Removing the belt or the co-belt makes $\partial\mathcal{B}$ equivalent to a closed 2-ball with usually one open perforation, but with possibly zero (figure 3.6) or an arbitrarily large number of them.

Dual bracelets and co-dual regions:

Each belt of Σ is associated with a unique lace of Σ. Since this is true in dual space as well, this association is bijective. Therefore, the envelope of a belt B of any bracelet $\mathcal{B}$ of Σ dualizes to the lace γ^* of a bracelet $\mathcal{B}^\delta$ of Σ^δ, whose belt B^* has for envelope the dual of the lace γ of $\mathcal{B}$. If λ and β map a bracelet to its lace and belt, respectively, we can extend our commutative diagram as follows:

$$\begin{array}{ccc} \gamma & \xrightarrow{\delta} & B^* \\ \downarrow{\scriptstyle \lambda^{-1}} & & \downarrow{\scriptstyle \beta^{-1}} \\ \mathcal{B} & \xrightarrow{\delta} & \mathcal{B}^\delta \\ \downarrow{\scriptstyle \beta} & & \downarrow{\scriptstyle \lambda} \\ B & \xrightarrow{\delta} & \gamma^* \end{array}$$

By carefully rolling a plane around the boundary of $\mathcal{B}$ in the appropriate manner, we will trace the entire boundary of $\mathcal{B}^\delta$ in dual space. (The rolling has to be "appropriate" in the sense that the passage from a belt to the co-belt and the passage across the lace must cut through the bracelet instead of tracing its envelope.) Obviously, the association between $\mathcal{B}$ and $\mathcal{B}^\delta$ is involutory. For all these good reasons, we call $\mathcal{B}^\delta$ the *dual bracelet* of $\mathcal{B}$. Note that a lace, alone, does not provide sufficient information to reconstruct its associated belt in dual space. One needs to add the planes supporting the facets incident to it. Another subtle point is that although the envelope of a belt in primal space is dual to a lace in dual space, the facial structures of these objects may not be in bijection. The reason is that all facets have been triangulated. As a result, a belt in primal (resp. dual) space might end up being facially less "refined" than its corresponding lace in dual (resp. primal) space. This fact will have significance on the algorithm.

Our discussion so far has been building up to what will be the most useful extension of the commutative diagram: a bijection between the regions of Σ and those of Σ^δ. Every bounding lace of a region is covered by a band (a co-belt) whose frontier usually consists of two simple closed curves (figure 3.9). Unfortunately, as we saw earlier, things might not be nearly as simple. Let R be a cyan region of Σ and let $\mathcal{B}_1, \ldots, \mathcal{B}_\ell$ be the bracelets of its bounding laces. Since the corresponding co-belts $B_1^c, \ldots, B_\ell^c$ are pairwise disjoint, the set $K = R \setminus \bigcup_{1 \le i \le \ell} B_i^c$ is a connected subset of ∂H — dotted area in figure 3.9. If again we think of P and Q as limit sets of an infinite sequence of isotopic smooth manifolds, we can interpret K as a deformation retract of a copy of $cl\, D^2$ with zero, one, or several open perforations (figure 3.10). Thus, any two planes supporting H at points of K can be brought together by continuous rolling around H without ever leaving contact with K. This proves that among the laces of the dual bracelets $\mathcal{B}_1^\delta, \ldots, \mathcal{B}_\ell^\delta$, any two can be connected by a cyan path in a co-region of Σ^δ. An immediate consequence is that the laces of $\mathcal{B}_1^\delta, \ldots, \mathcal{B}_\ell^\delta$ bound a common magenta region of Σ^δ. Since the argument is involutory, the region in question must be entirely bounded by these laces. For this reason it is only natural to call it the *co-dual region* R^δ of R (the prefix "co" as a reminder that it is really its co-region that is dual to R). Again, this map is involutory, and of course, consistent with the bijection previously defined between laces in primal and dual spaces.

Unlike k-faces which become $(2-k)$-faces in dual space, or for that matter, laces and belts, the type "bracelet" (resp. "region") is invariant under duality (resp. co-duality). The best illustration of this comes from the self-dual case, where $Q = P^\delta$ (figure 3.6). There, we have the remarkable situation that dual bracelets and co-dual regions remain invariant; only their colors change! The following lemma summarizes most of our discussion so far.

Lemma 3.2. *The interior of a bracelet $\mathcal{B}$ of Σ is homeomorphic to an open filled torus. Its boundary contains exactly one belt and one lace of Σ, which are homeomorphic to $S^1 \times (0,1)$ and S^1, respectively. With $\mathcal{B}$ is associated a unique dual bracelet $\mathcal{B}^\delta$ of Σ^δ: The dual of the envelope of the belt of $\mathcal{B}$ is the lace of $\mathcal{B}^\delta$; conversely, the dual of the lace of $\mathcal{B}$ is the envelope of the belt of $\mathcal{B}^\delta$. Also, to each region R corresponds a unique co-dual region R^δ of Σ^δ of opposite color, and the bracelets of their bounding laces are dual to each other.*

D. On to dual-broadcasting. We shall deliberately break the symmetry between P and Q, and at the same time, obligate the broadcaster to alternate between two modes of operation: primal and dual. Unfortunately, this increase in complication will not be offset by a decrease in complexity — at least not immediately. The motivation for this devious move is that symmetry gets our hands tied. On the contrary, opting for a single anchor gives us the freedom to choose whichever is best equipped to fight the ubiquitous adversary at the right time. Suppose once and for all that P is the anchor. (How to choose the right anchor is to be discussed later.) Using the previous notation, what shall the broadcaster do if presented with the input pair (v, magenta)? Since we mean to let the boundary of P lead the search, this is the easy case where primal-broadcasting through Q can be used (section 3.B).

So, assume now that the input is of the form (v, cyan), where v is a vertex of a lace γ, and let R be the cyan region to be explored. Traversing R entails leaving the polytope Q altogether. As usual, if γ is the only lace of R and we know that for a fact, everything is easy. But what if we have other laces $\gamma_1, \ldots, \gamma_t$? The difficulty is not to traverse R per se, but to tell when we might be re-entering Q and hitting upon vertices of γ_i. Primal shields are useless at this point, and we must turn to the dual shield of Q for help. Dual-broadcasting from a lace of Σ to another one will be accomplished in three stages:

1. Starting from the belt of Σ^δ associated with the starting lace, navigate in dual space to the lace of its bracelet.
2. Primal-broadcast through Q^δ in dual space.
3. Starting from the belt of Σ associated with a newly discovered lace, navigate to the lace of its bracelet.

Either of the tasks performed in steps 1 and 3 is called a *mutation*: We are given a vertex on a lace of a bracelet $\mathcal{B}$ and we must find one vertex on the lace of the dual bracelet of $\mathcal{B}$. Note that from the algorithm description a mutation involves going from a belt to its associated lace and not the other way around. One might think that reversing the process is just dualizing it, and hence, computationally equivalent. That is not quite true. The subtlety here relates to our previous remark about belts being facially less refined than the laces of their dual bracelets. As a result, navigating toward belts can be more difficult than toward laces (albeit still doable).

Lemma 3.3. *Mutating from a lace of a bracelet can be performed in time proportional to the number of edges in its dual bracelet.*

Proof: To mutate from the lace of a bracelet $\mathcal{B}$ of Σ to the lace of its dual bracelet $\mathcal{B}^\delta$, our first action is to collect some relevant information from the input vertex v. As we indicated earlier, a vertex of a lace is never given by itself, but along with the new facet of P (or Q) and the new edge of Q (or P) upon which it is in contact. General position ensures that v is incident upon a constant number of faces, so we can easily get two (new) facets, one in ∂P and the other in ∂Q, whose intersection contributes one edge e of the lace of $\mathcal{B}$. By duality, these two facets specify an edge ab of the belt of $\mathcal{B}^\delta$. More interesting, each facet contributes at least one of its own vertices to the bracelet: which one can be determined in constant time by local examination. Dualizing these chosen vertices gives old facets f_a, f_b which, locally around a and b, lie on the boundary of $\mathcal{B}^\delta$. Note that these facets need not lie entirely in $\partial\mathcal{B}^\delta$ (figure 3.11). Instead of computing these old facets, which might be time-consuming, we retrieve one new facet within f_a (resp. f_b) that is incident upon a (resp. b). Recall that we added a special provision to the correspondence between a polytope and its dual to make this possible in constant time. This is an important consideration, because examining all the facets incident upon a and b at each mutation might eventually lead to a quadratic algorithm!

We are now ready to get truly started. We are in possession of an edge ab of the belt of $\mathcal{B}^\delta$ and a simplicial facet A (resp. B) incident upon a (resp. b) that contributes a facet to $\mathcal{B}^\delta$ (but might not be one itself). Let P^* (resp. Q^*) be the intersection of P^δ (resp. Q^δ) with the plane π passing through O, a, b, and let H^* be the convex hull of P^* and Q^*. Note that, in general, H^* is *not* the intersection of π with the convex hull of $P^\delta \cup Q^\delta$. Because the origin lies in the interior of both P^* and Q^*, and neither polygon contains the other, the curves ∂P^* and ∂Q^* must intersect. Furthermore, the closure $\mathcal{B}^*$ of the connected component of $H^* \setminus (P^* \cup Q^*)$ that contains the edge ab is the two-dimensional equivalent of a bracelet (figure 3.12): it is simple to analyze, so we will assume its basic properties. The edge ab is the "belt" of $\mathcal{B}^*$ (more appropriately called a *bridge*) and its "lace" is the point $p = \mathcal{B}^* \cap \partial\mathcal{P}^* \cap \partial\mathcal{Q}^*$. Using standard techniques, we can compute p by a simultaneous traversal of ∂P^* and ∂Q^* starting from a and b. With a bit of care, we can find p in time proportional to the number of vertices in $\mathcal{B}^*$.

Of course, this assumes that we have full knowledge of P^* and Q^*. But we do not, and do not wish to. Since the boundaries of P^δ and Q^δ have been triangulated, however, it is easy to go from one edge to an adjacent one in constant time, and thus achieve the same effect. To obtain the starting edge may require a little extra work, since A (resp. B) might not intersect π (recall that a facet does not contain its incident vertices). But we know that A (resp. B) lies in $\partial\mathcal{B}^\delta$ locally around a (resp. b). Therefore, beginning at A (resp. B), we can go around the cyclical order of new faces around a (resp. b) until we find one that intersects π. If we are careful to go in the right direction, this preliminary work should involve looking only at new faces of P^δ and Q^δ that contribute to the boundary of $\mathcal{B}^\delta$. To choose the right direction, it is helpful to realize that either of the two faces sought is dual to a vertex of the (old) dual facet of a or b, whose Euclidean distance to the line passing through the edge e is a local maximum among the vertices of that facet. This suggests how to find the direction to follow by constant-time local examination. Once we have p, we also know

two simplicial facets whose intersection contributes an edge to the desired lace, so the mutation is over. The total running time is at most proportional to the size of the dual bracelet. ∎

The lemma gives us ammunition for primal-broadcasting through Q^δ. This will reveal to us one vertex for each lace bounding the co-dual region of R. By virtue of Lemma 3.2, mutating back from each lace bounding R^δ will finally take us to the remaining laces of R and complete our dual-broadcasting routine. Thus, to summarize, dual-broadcasting is effected in a three-step sequence: (1) mutate to dual space, (2) primal-broadcast in dual space, and (3) mutate back to primal space. One should appreciate that dual-broadcasting is not synonimous to primal-broadcasting in dual space. Another basic observation is that the input to a primal-broadcast need not be a vertex of a lace: any non-lace vertex v in a co-region of Σ (resp. Σ^δ) will work just as well, as long as we know the location of v in the primal (resp. dual) shield of Q. This might be handy when looking for a starting vertex.

What is the cost of broadcasting as a whole? Let κ (resp. κ') be the maximum number of approximating polytopes in the primal (resp. dual) shield of Q whose boundaries are cut by a single edge of P (resp. P^δ). It follows from Lemma 2.2 that the broadcasting time will be $O\big((\kappa+\kappa'+1)n\big)$, not counting mutations. But from Lemma 3.3, the cost of all mutations is at most proportional to the number of edges in Σ and Σ^δ, which is $O(n)$. In light of Lemmas 2.1 and 3.1, we can draw the following conclusions:

Lemma 3.4. *Given a starting vertex, we can compute the intersection of P and Q in time $O(n + \kappa n + \kappa' n)$, where κ (resp. κ') is the maximum number of approximating polytopes in the primal (resp. dual) shield of Q whose boundaries are cut by a single edge of P (resp. P^δ).*

What is a valid starting vertex in the context of the lemma? Any vertex on a lace of Σ or Σ^δ will do, as will any vertex of P (resp. P^δ) that lies in the primal (resp. dual) shield of Q and has been located in it. It is not difficult to compute a starting vertex in linear time. Therefore, we could package our findings into yet another $O(n \log n)$ algorithm for intersecting two convex polyhedra of size n. But we can do better than that.

E. Putting it all together. The only data structure we shall need is the k-shield of Q, where k is a fixed constant to be determined later. Let $Q = Q_0 \supset Q_1 \supset \cdots \supset Q_k$ and $Q^\delta = Q'_0 \supset Q'_1 \supset \cdots \supset Q'_k$ be the sequences of approximating polytopes provided by, respectively, the primal and dual parts of the k-shield. Suppose that the intersections $P \cap Q_k$ and $P^\delta \cap Q'_k$ are fully available. We will show that we can emulate primal-broadcasting through Q and Q^δ, even though we only have a portion of the shield at our disposal. Let us discuss the case of P and Q_k, with the understanding that the same applies to P^δ and Q'_k. Assume that both the boundaries and the interiors of P and Q_k intersect. Then, the whole theory of regions, co-regions, laces, belts, and bracelets applies verbatim to the surface $\partial(P \cup Q_k)$. Since the intersection of P and Q_k is available, we can *precompute* all primal-broadcasting through Q_k anchored by P. We do this by marking the (new) facets of P and Q_k that contribute an edge to a lace of $\partial(P \cup Q_k)$. Also, for each region of $\partial(P \cup Q_k)$, we link into a circular list representative vertices of its bounding laces. In this way, we are able to primal-broadcast

from any vertex of a lace in $\partial(P \cup Q_k)$ by tracing the lace in question until we hit a representative vertex. From there, we jump to all the other laces bounding the desired region in time proportional to their number. This gives us the capability to primal-broadcast through the polytope Q with P as anchor, even though we might only know the outer layers of its primal shield. Obviously, the same trick can be used for primal-broadcasting in dual space. Note that mutations are not affected by this change. The advantage of this new scenario is to place an upper bound of k on the values of κ and κ' in Lemma 3.4.

What now qualifies as a starting vertex? As usual, any vertex on a lace of Σ or Σ^δ will do. Another valid situation is a vertex of P (resp. P^δ) that lies in the (possibly disconnected) set $Q \setminus Q_k$ (resp. $Q^\delta \setminus Q'_k$), along with its location in the primal (resp. dual) part of the k-shield of Q. Finally, any intersection between ∂Q_k (resp. $\partial Q'_k$) and an edge of P (resp. P^δ) and its location in the k-shield would form an appropriate starting vertex. Note that we do not extend this qualification to just any point in $P \cap \partial Q_k$ or $P^\delta \cap \partial Q'_k$ because these points might not be reached by any broadcast. Indeed, primal-broadcasting anchored at P involves traversing the edges of P and *not* its facets. Thus, there could be a nonempty intersection between ∂P and ∂Q_k, even though no edge of P intersects Q_k. In that case, the primal-broadcast would never make contact with Q_k, so obviously, no point of ∂Q_k should serve as a valid starting vertex.

The intersection algorithm

1. Check whether the interiors of P and Q intersect and conclude immediately if they do not, using the information provided by the Dobkin-Kirkpatrick algorithm. Else, pick a point O in the interior of both P and Q, and compute their dual polytopes P^δ and Q^δ.
2. Unless the anchor has already been chosen, declare P the anchor and compute the k-shield of Q. Let Q_k (resp. Q'_k) be the innermost polytope in the primal (resp. dual) part of the k-shield.
3. Compute $P \cap Q_k$ and $P^\delta \cap Q'_k$ recursively (the boundary case where any of the polytopes involved has constant size can be handled directly in linear time). *Crucial point*: Make Q_k and Q'_k the anchors in the recursive calls.
4. If $P \cap Q_k = P$, return P as the intersection of P and Q, and stop. If $P^\delta \cap Q'_k = P^\delta$, return Q as the intersection of P and Q, and stop.
5. If the interiors and boundaries of P (resp. P^δ) and Q_k (resp. Q'_k) intersect, then precompute all primal-broadcasting through Q_k (resp. Q'_k) anchored by P (resp. P^δ).
6. Compute a starting vertex (see below).
7. Launch a broadcast from the starting vertex and pursue it until all the laces of $\partial(P \cup Q)$ have been found.
8. Use the laces to compute $P \cap Q$ explicitly, and stop.

A few words of comments about the algorithm. Step 1 uses the linear-time algorithm of (Dobkin and Kirkpatrick [7]). If there is no intersection, the algorithm will say so and report the two closest points in P and Q. If P and Q intersect only at their boundaries (which, incidentally, is against our general-position assumption), the Dobkin-Kirkpatrick method will still allow us to compute the full

intersection in linear time. If we have a full-fledged intersection, however, the method will return a point interior to both polytopes. The dual polytopes of P and Q are easily computed in linear time. In step 2, we declare either one of the two polytopes, say, P, the anchor, unless we are responding to a recursive call in which case the choice of the anchor is forced upon us. From Lemma 2.1, the k-shield of Q can be computed in linear time. Step 3 consists of two recursive calls. As the analysis will show, switching anchors is a crucial feature of the algorithm. Failure to do so would jeopardize the linearity of the algorithm. Step 4 takes care of two trivial terminating cases. In step 5, we build the shortcuts, if any, provided by $P \cap Q_k$ and $P^\delta \cap Q'_k$.

Step 6 determines a starting vertex. If the boundaries of P and Q_k intersect, we pick a starting vertex among the vertices of $\partial P \cap \partial Q_k$ that emanate from an edge of P (if any). Then, we locate the vertex in question in the primal part of the k-shield of Q. If this does not work, we try the same operation in dual space. Namely, if the boundaries of P^δ and Q'_k intersect, we pick a starting vertex among the vertices of $\partial P^\delta \cap \partial Q'_k$ that emanate from an edge of P^δ (if any). Then, we locate the vertex in question in the dual part of the k-shield of Q. If this also fails, we then know that the skeleton (i.e., set of vertices and edges) of P (resp. P^δ) lies entirely outside Q_k (resp. Q'_k). So, we pick a vertex v of P and check whether it lies in Q. If it does then it must be sandwiched between Q and Q_k, so we can locate v in the primal part of the k-shield and make it the starting vertex. Otherwise, let f be the (old) facet of P^δ that is dual to v and let π be its supporting plane. We compute the intersection I of the two convex polygons f and $\pi \cap Q^\delta$, using a linear-time algorithm [23,27]. Note that because v lies outside Q, the polygon $\pi \cap Q^\delta$ is nonempty. If now the polygon I is nonempty, any of its vertices either belongs to a lace of Σ^δ or is a vertex of P in $Q^\delta \setminus Q'_k$, and therefore qualifies as a starting vertex. If I is empty then we compute a line ℓ of π that separates the two polygons f and $\pi \cap Q^\delta$ (in the standard sense of the term). Next, we compute the orthogonal projections of P^δ and Q^δ on a plane normal to ℓ. Because (i) f lies outside Q^δ, (ii) π intersects Q^δ, and (iii) $P^\delta \cap Q^\delta \neq \emptyset$, it immediately follows that the boundaries of the two projections intersect. Again, we can compute a point of the intersection in linear time. Any such point is the projection of an edge of a lace of Σ^δ. This gives us a starting vertex and concludes step 6. With such a vertex in hand, Lemma 3.4 tells us how to compute the intersection of P and Q. Figure 3.13 is a feeble attempt at illustrating the main phases of the algorithm in two dimensions. The polytope P is the nonconvex (sorry about that) blob wiggling across the primal part of the k-shield of Q (which itself, obviously, should not be made of *disjoint* rings...)

Put $m = p + q$, where p (resp. q) is the added number of vertices and bounding planes in P (resp. Q), and let $T(p, q)$ be the worst-case running time of the algorithm. If either $p = O(1)$ or $q = O(1)$ then, trivially, $T(p, q) = O(p + q)$. From Lemmas 2.1 and 3.4, we derive the general relation

$$T(p, q) = 2T\left(p, 3(1 - 1/7)^k q\right) + O(p + q).$$

This recurrence is headed straight toward $O(n \log n)$, which is not exactly what we had in mind. However, the trick of switching anchors at each recursive call will now pay off. Indeed, the recurrence can be more accurately expressed as

$$T(p, q) = 2T'\left(p, 3(1 - 1/7)^k q\right) + O(p + q),$$

and

$$T'(p,q) = 2T\left(3(1-1/7)^k p, q\right) + O(p+q),$$

which, after substitution, yields

$$T(p,q) = 4T\left(3(1-1/7)^k p, 3(1-1/7)^k q\right) + O(p+q),$$

or more simply,

$$T(m) = 4T(m/5) + O(m),$$

for $k = 18$. Thus, $T(m) = O(m)$ and our quest for a linear-time algorithm for intersecting two convex polyhedra is now satisfied.

Reflecting back on the algorithm, it is interesting to observe that switching anchors at each recursive call makes all the difference between an $O(n)$ and $O(n \log n)$ algorithm. The process can be regarded as a form of *branching dovetailing*. Notice that alternative rules are possible, of course, such as always choosing the smallest polytope as the anchor.

A second observation is that, in the end, all the laces in both Σ and Σ^δ will have been fully computed. (Or at least this can be easily ensured at no extra cost.) Since the laces of Σ^δ dualize to the belts of Σ, we get the convex hull of P and Q as a bonus. Of course, another method is to compute the intersection of the dual polytopes and dualize back. If P and Q are disjoint, then we can use the Preparata-Hong linear-time wrapping routine. In all cases, therefore, we are able to compute the convex hull of two convex polytopes in linear time.

Theorem 3.5. *It is possible to compute the intersection (and the convex hull) of two three-dimensional convex polyhedra in linear time. It is assumed that the polyhedra are given in standard representation.*

4. Miscellaneous Applications

The intersection algorithm can be put to use for improving or simplifying the solutions to a number of geometric problems. These applications are all very simple, so we limit our discussion to a minimum.

A. Intersecting several convex polyhedra. Consider the problem of computing the common intersection of k convex polyhedra $P_1, \ldots, P_k$, given in standard representation. We can do this in optimal $O(n \log k)$ time, where n is the total number of vertices among the k polytopes. We use a straightforward scheme, borrowed from multi-way merging: Put the polyhedra in bijection with the leaves of a complete binary tree, and compute intersections in an order consistent with the tree. The $O(n \log k)$ running time of this algorithm is worst-case optimal even in two dimensions, because we can reduce any k-way merge to polygon intersection. To see this, consider a collection of k sorted lists $L_1, \ldots, L_k$ with distinct elements. Form the polygons $P_1, \ldots, P_k$, where P_i be the unbounded polygon defined by the intersection of the halfplanes $y \geq x_j(2x - x_j)$: P_i is bounded by the tangents to the parabola $y = x^2$ at the points (x_j, x_j^2), for $j = 1, \ldots, \ell$, where $L_i = (x_1, \ldots, x_\ell)$. Now, observe that

the boundary of the polygon $P = \bigcap_{1 \le i \le k} P_i$ contains all the points in $\{ (x, x^2) \mid x \in \bigcup_{1 \le i \le k} L_i \}$. Therefore the merged sequence of all the k lists can be read off by going around the boundary of P. To obtain the desired lower bound, we form k lists of size $m = \lfloor n/k \rfloor$, and observe that they can be merged in $M = \binom{n}{m_1, \ldots, m_k}$ ways, where $m_i = m$. The lower bound follows from the fact that $\log M = \Omega(n \log k)$.

B. Convex hulls. Bentley and Shamos [4] have shown how to take advantage of certain point distributions to obtain linear expected-time algorithms for computing convex hulls. The idea is to use divide-and-conquer by splitting the input set in a fixed manner (independent of the point-set itself) and build the convex hull bottom up. For their method to work efficiently, the merge step must be capable of computing the convex hull of two (possibly intersecting) convex polytopes reasonably fast. As observed by Seidel [11], we can use the Preparata-Hong algorithm for that purpose and get linear expected complexity for a wide class of point distributions. Using Theorem 3.5 widens that class. Specifically, any distribution for which the average size of the convex hull of a random set of n points is $O(n/\log^{1+\epsilon} n)$ will trivially yield a linear expected-time complexity.

C. Merging Voronoi diagrams. Kirkpatrick [17] has shown that two planar Voronoi diagrams can be "merged" in linear time. His algorithm is very ingenious but somewhat complicated. Using standard reductions, the same result falls straight out of Theorem 3.5. The problem is this: Given two sets of n points in 2-space and their respective Voronoi diagrams, compute the diagram of their union. Using a reduction due to Edelsbrunner and Seidel [11,14], we compute a Voronoi diagram of n points by intersecting n halfspaces. Let $h(p)$ denote the closed halfspace bounded below by the tangent to the paraboloid $z = x^2 + y^2$ at the point whose xy-projection is p. The Voronoi diagram of $p_1, \ldots, p_n$ is the xy-projection of the two-dimensional cell complex formed by the boundary of the convex polyhedron $\bigcap_{1 \le i \le n} h(p_i)$. Thus, merging Voronoi diagrams becomes a special case of intersecting two convex polyhedra. Applications include computing the Voronoi diagram of a polygon (Kirkpatrick [17]) and of the vertices of a convex polygon (Aggarwal et al. [1]).

5. Conclusions

Our main result is a linear-time algorithm for intersecting two convex polyhedra in 3-space. Whether the algorithm lends itself to efficient and robust implementations remains to be seen. Going for it is its relative simplicity and the fact that its stays away from any complicated data structure. But let's face it, nothing is simple in three dimensions. Intersecting two tetrahedra in a truly robust fashion could be someone's idea of a crual punishment, and dealing with more complex objects has rarely been known to make things easier. An outstanding open problem is that of intersecting two nonconvex polyhedra efficiently. The problem of intersecting arbitrarily placed triangles in 3-space has been investigated in (Aronov and Sharir [2]). How much we can gain by having collections of faces structured into the boundaries of simple polyhedra is an intriguing open question.

Acknowledgments: I wish to thank Herbert Edelsbrunner and David Kirkpatrick for helpful conversations.

REFERENCES

1. Aggarwal, A., Guibas, L.J., Saxe, J., Shor, P. *A linear time algorithm for computing the Voronoi diagram of a convex polygon*, Proc. 19th Ann. ACM Sympos. Theory Comput. (1987), 39–45.
2. Aronov, B., Sharir, M. *Triangles in space, or building and analyzing castles in the air*, Proc. 4th Ann. ACM Sympos. Comput. Geom. (1988), 381-391.
3. Baumgart, B.G. *A polyhedron representation for computer vision*, 1975 National Computer Conference, AFIPS Conf. Proceedings, 44, AFIPS Press (1975), 589–596.
4. Bentley, J.L., Shamos, M.I. *Divide and conquer for linear expected time*, Inform. Process. Lett. 7 (1978), 87–91.
5. Chazelle, B., Dobkin, D.P. *Intersection of convex objects in two and three dimensions*, J. ACM 34 (1987), 1–27.
6. Dobkin, D.P., Kirkpatrick, D.G. *Fast detection of polyhedral intersection*, Theoret. Comput. Sci. 27 (1983), 241–253.
7. Dobkin, D.P., Kirkpatrick, D.G. *A linear algorithm for determining the separation of convex polyhedra*, J. Algorithms 6 (1985), 381–392.
8. Dobkin, D.P., Laszlo, M.J. *Primitives for the manipulation of three-dimensional subdivisions*, Proc. 3rd Ann. ACM Sympos. Comput. Geom. (1987), 86–99.
9. Dobkin, D.P., Munro, J.I. *Efficient uses of the past*, J. Algorithms 6 (1985), 455–465.
10. Dyer, M.E. *Linear time algorithms for two and three-variable linear programs*, SIAM J. Comput. 13 (1984), 31–45.
11. Edelsbrunner, H. *Algorithms in Combinatorial Geometry*, Springer-Verlag, Heidelberg, Germany, 1987.
12. Edelsbrunner, H., Maurer, H. *Finding extreme points in three dimensions and solving the post-office problem in the plane*, Inform. Process. Lett. 21 (1985), 39–47.
13. Edelsbrunner, H., Mücke, E.P. *Simulation of simplicity: a technique to cope with degenerate cases in geometric algorithms*, Proc. 4th Ann. ACM Sympos. Comput. Geom. (1988), 118–133.
14. Edelsbrunner, H., Seidel, R. *Voronoi diagrams and arrangements*, Discrete Comput. Geom. 1 (1986), 25–44.
15. Guibas, L.J., Stolfi, J. *Primitives for the manipulation of general subdivisions and the computation of Voronoi diagrams*, ACM Trans. Graphics 4 (1985), 75–123.
16. Hertel, S., Mäntylä, M., Mehlhorn, K., Nievergelt, J. *Space sweep solves intersection of convex polyhedra*, Acta Informatica 21 (1984), 501–519.
17. Kirkpatrick, D.G. *Efficient computation of continuous skeletons*, Proc. 20th Annu. IEEE Symp. on Foundat. of Comput. Sci. (1979), 18–27.
18. Kirkpatrick, D.G. *Optimal search in planar subdivisions*, SIAM J. Comput. 12 (1983), 28–35.
19. Megiddo, N. *Linear-time algorithms for linear programming in $\Re^3$ and related problems*, SIAM J. Comput. 12 (1983), 759–776.

20. Mehlhorn, K. *Data Structures and Algorithms 3: Multidimensional Searching and Computational Geometry*, Springer-Verlag, Heidelberg, Germany, 1984.
21. Mehlhorn, K., Simon, K. *Intersecting two polyhedra one of which is convex*, Univ. Saarland, Tech. Report, Saarbrücken, West Germany, 1986.
22. Muller D.E., Preparata, F.P. *Finding the intersection of two convex polyhedra*, Theoret. Comput. Sci. 7 (1978), 217–236.
23. O'Rourke, J., Chien, C.B., Olson, T., Naddor, D. *A new linear algorithm for intersecting convex polygons*, Comput. Graphics and Image Process. 19 (1982), 384–391.
24. Preparata, F.P., Hong, S.J. *Convex hulls of finite sets of points in two and three dimensions*, Comm. ACM 20 (1977), 87–93.
25. Preparata, F.P., Shamos, M.I. *Computational Geometry*, Springer-Verlag, New York, 1985.
26. Rourke, C.P., Sanderson, B.J. *Introduction to piecewise-linear topology*, Springer-Verlag, Heidelberg, Germany, 1982.
27. Shamos, M.I., Hoey, D. *Geometric intersection problems*, Proc. 17th Annu. IEEE Symp. on Foundat. of Comput. Sci. (1976), 208–215.
28. Yap, C.K. *A geometric consistency theorem for a symbolic perturbation scheme*, Proc. 4th Ann. ACM Sympos. Comput. Geom. (1988), 134–142.

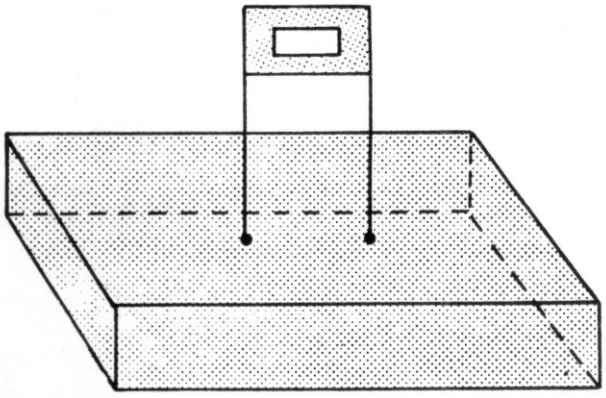

FIGURE 2.1

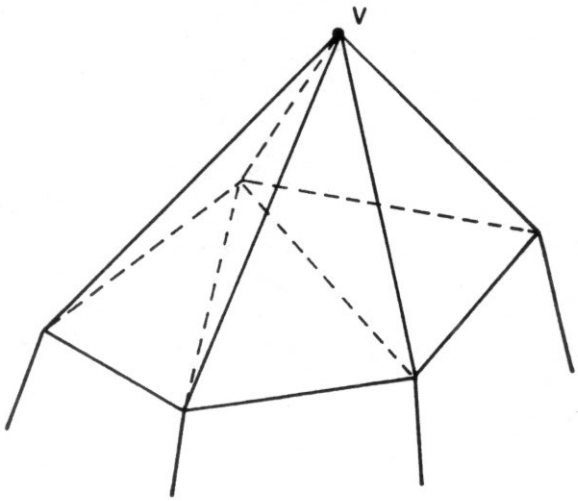

FIGURE 2.2

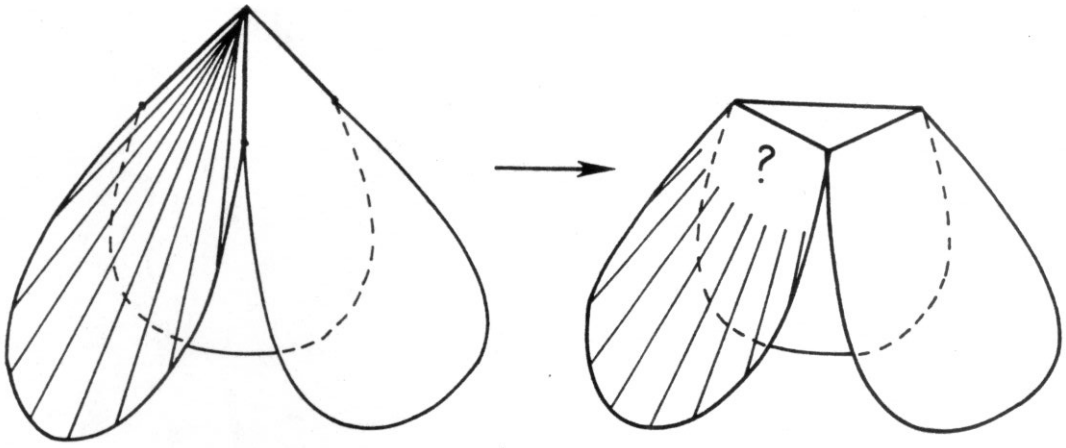

FIGURE 2.3

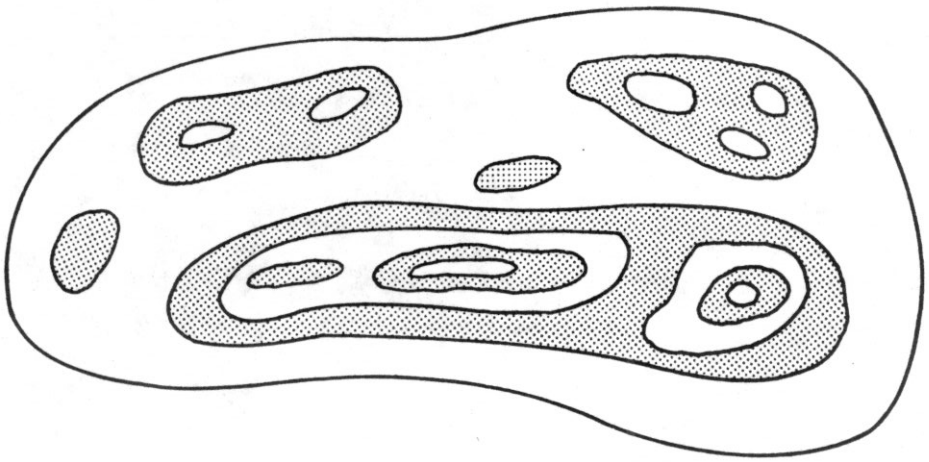

FIGURE 3.1

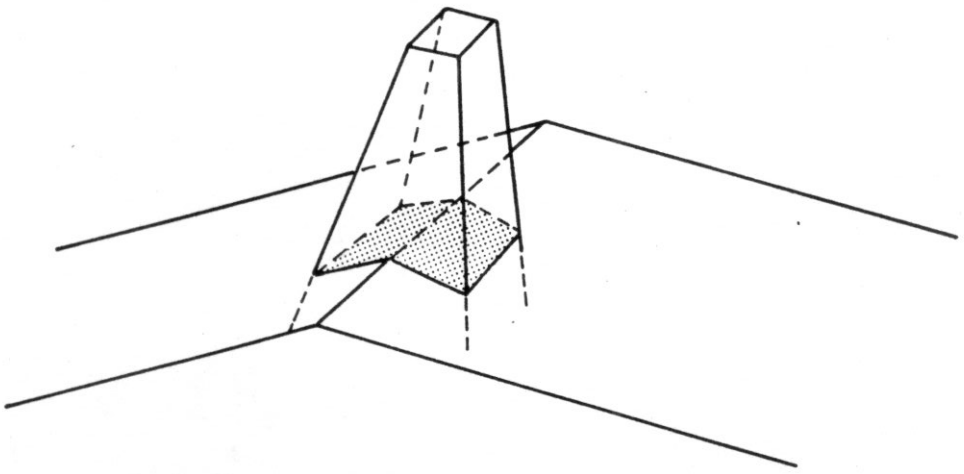

FIGURE 3.2

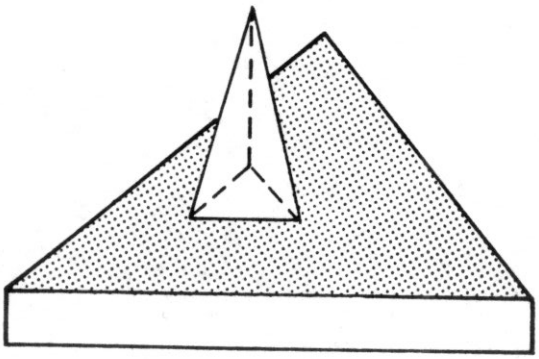

FIGURE 3.3

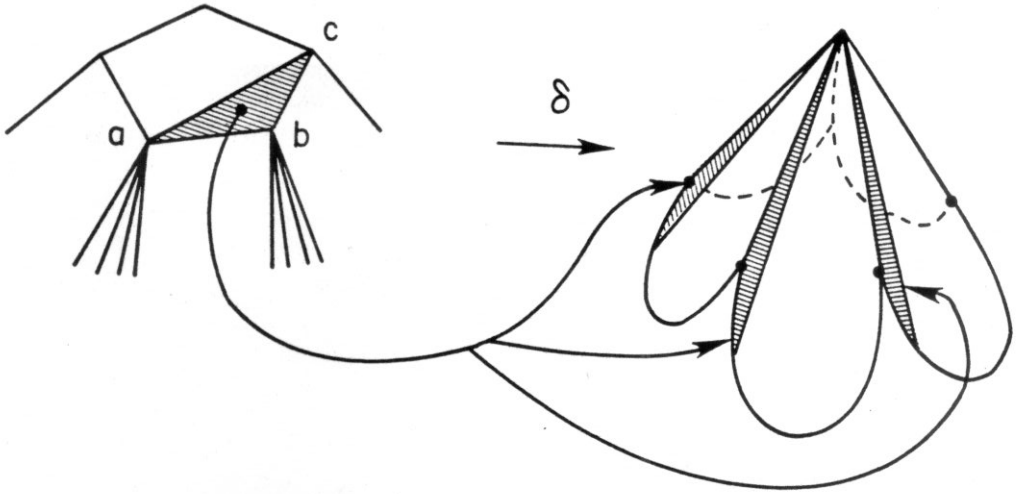

FIGURE 3.4

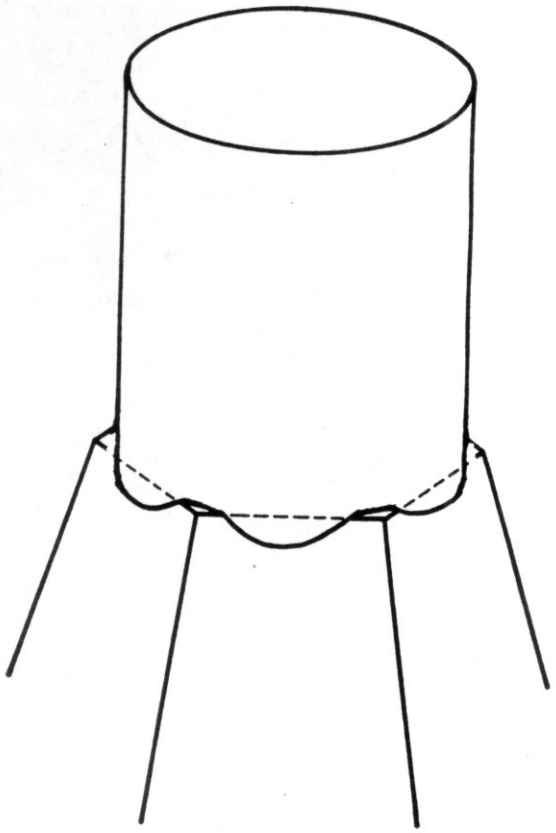

FIGURE 3.5

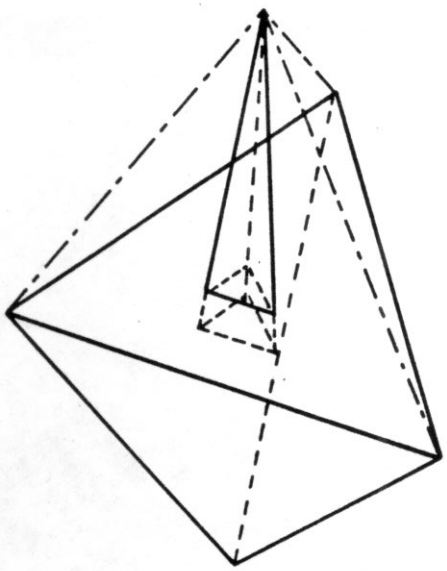

FIGURE 3.6

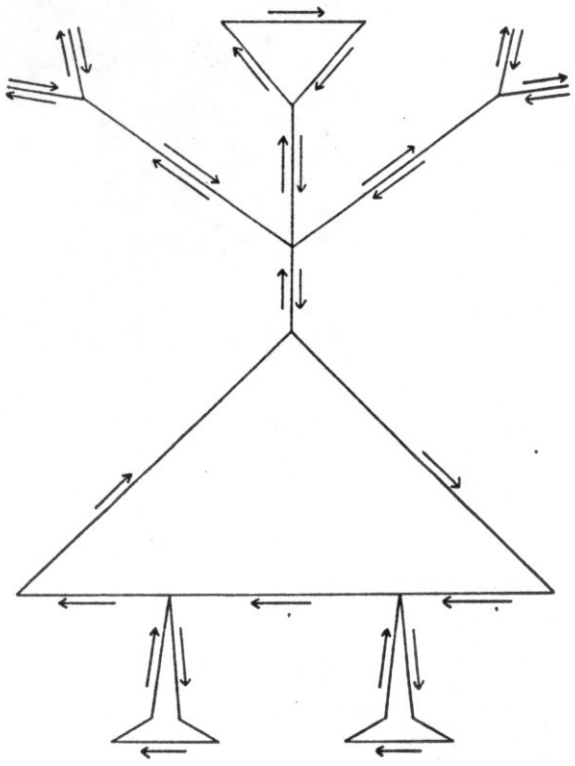

FIGURE 3.7

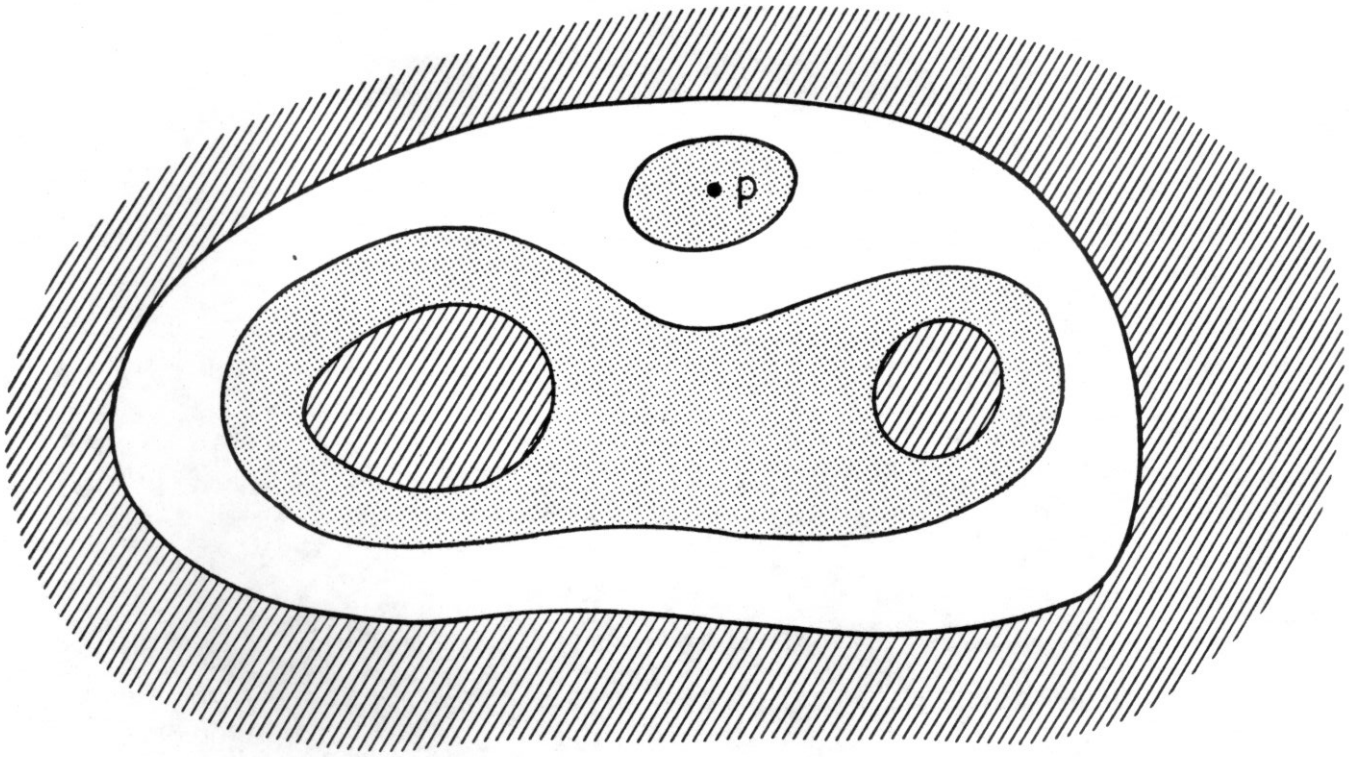

FIGURE 3.8

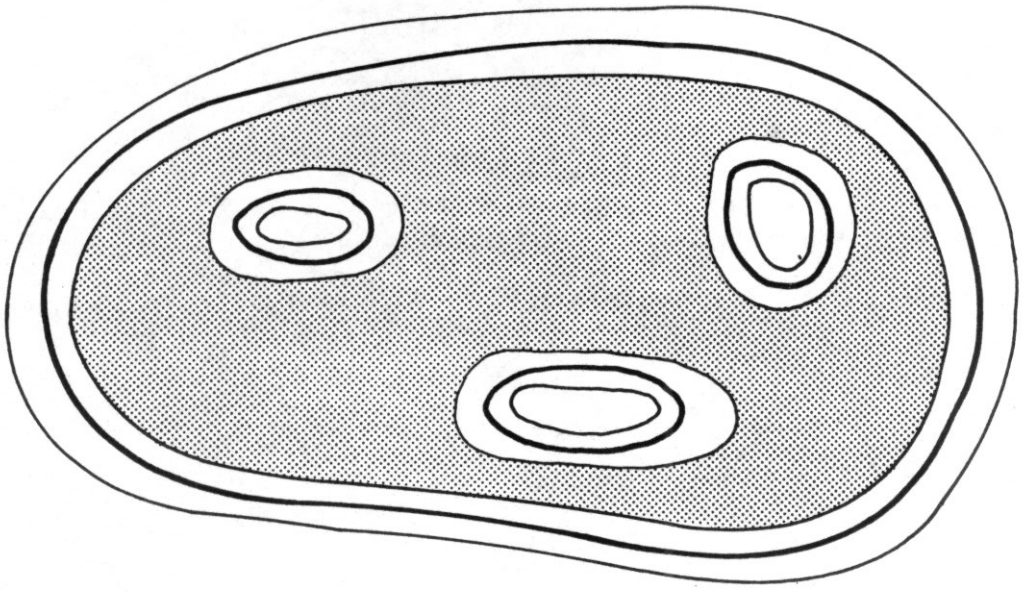

FIGURE 3.9

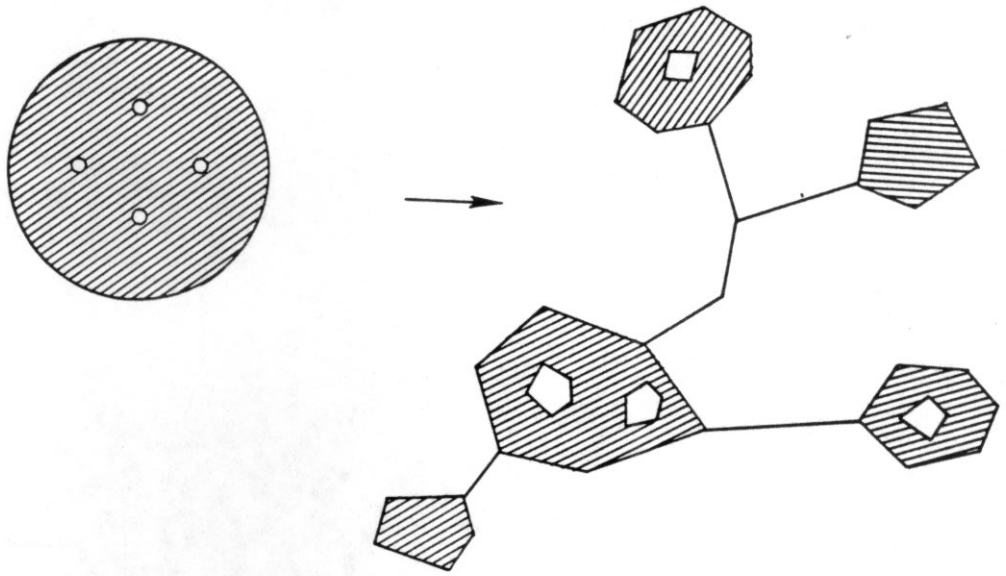

FIGURE 3.10

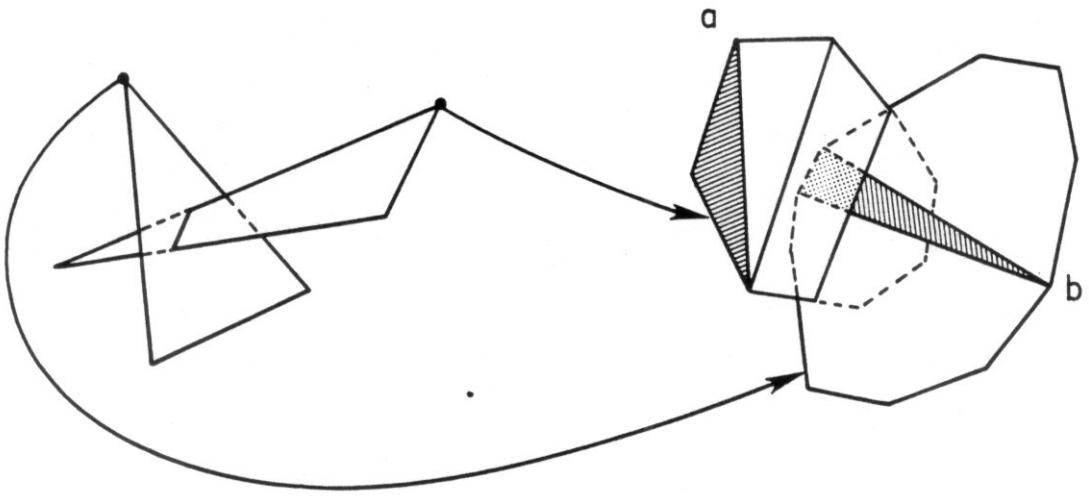

FIGURE 3.11

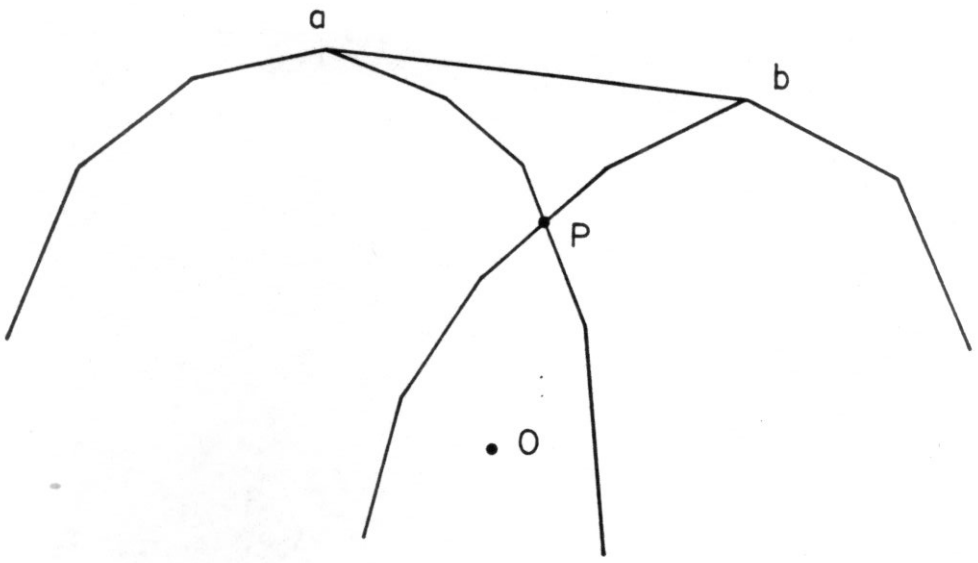

FIGURE 3.12

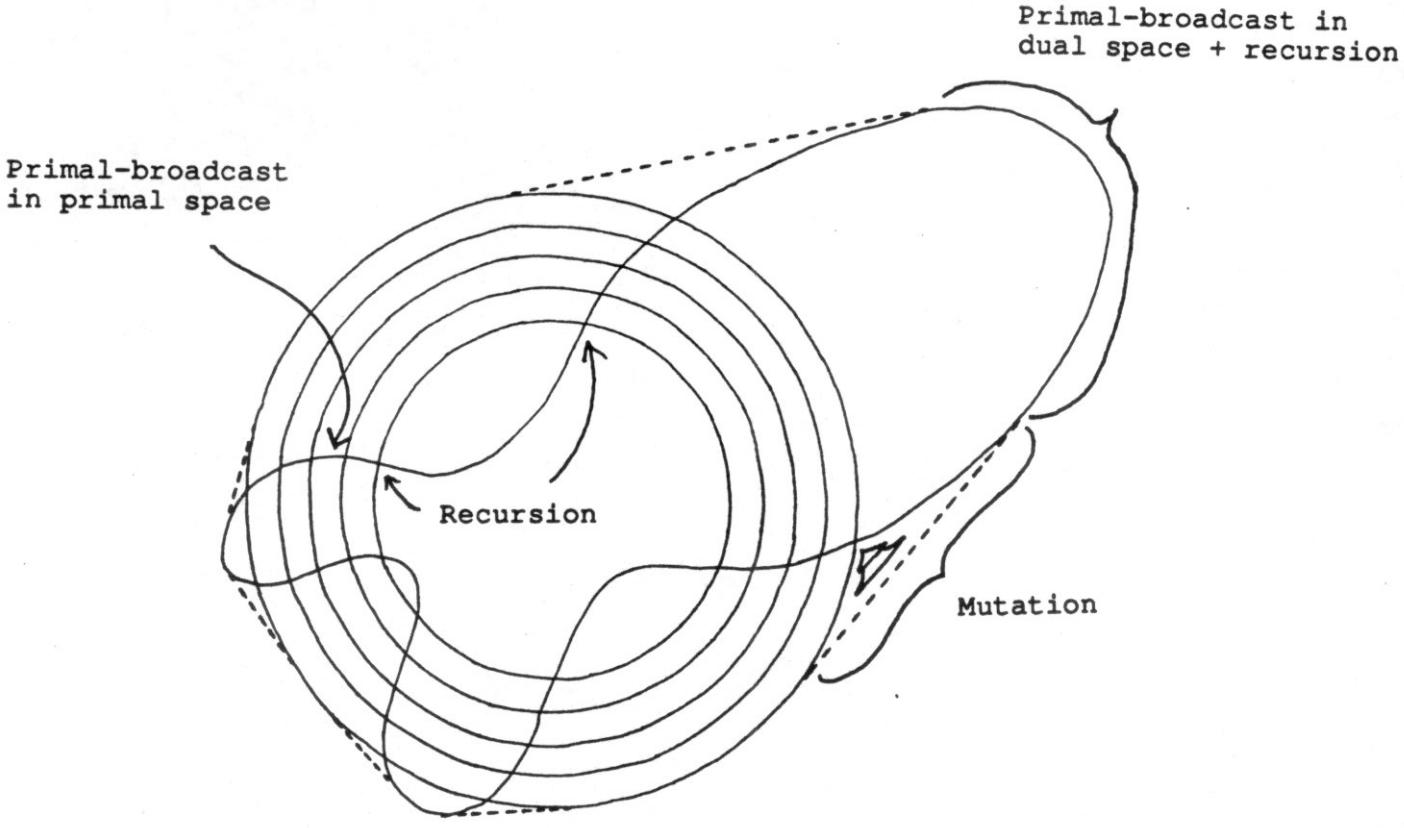

FIGURE 3.13