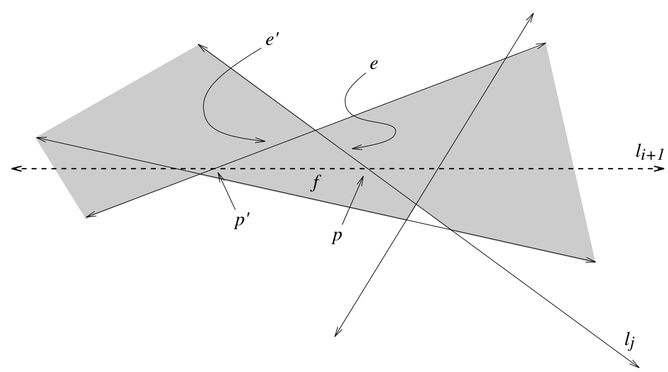
the arrangement of the objects added so far. This approach yields an optimal-time algorithm for arrangements of hyperplanes. The analysis of the running time is based on the zone result [ESS93] (Section 28.2).

We present the algorithm for a collection $\mathcal{L} = \{l_1, \ldots, l_n\}$ of n lines in the plane, assuming that the arrangement $\mathcal{A}(\mathcal{L})$ is simple. Let $\mathcal{L}_i$ denote the set $\{l_1, \ldots, l_i\}$. At stage $i+1$ we add l_{i+1} to the arrangement $\mathcal{A}(\mathcal{L}_i)$. We maintain the DCEL representation [BCKO08] for $\mathcal{A}(\mathcal{L}_i)$, so that in addition to the incidence information, we also have the order of edges along the boundary of each face. The addition of l_{i+1} is carried out in two steps: (i) we find a point p of intersection between l_{i+1} and an edge of $\mathcal{A}(\mathcal{L}_i)$ and split that edge into two, and (ii) we walk along l_{i+1} from p to the left (assuming l_{i+1} is not vertical) updating $\mathcal{A}(\mathcal{L}_i)$ as we go; we then walk along l_{i+1} from p to the right completing the construction of $\mathcal{A}(\mathcal{L}_{i+1})$. See Figure 28.4.1.

FIGURE 28.4.1
Adding the line l_{i+1} to the arrangement $\mathcal{A}(\mathcal{L}_i)$.
The shaded region is the zone of l_{i+1} in the arrangement of the other four lines.



Finding an edge of $\mathcal{A}(\mathcal{L}_i)$ that l_{i+1} intersects can be done in $O(i)$ time by choosing one line l_j from $\mathcal{L}_i$ and checking all the edges of $\mathcal{A}(\mathcal{L}_i)$ that lie on l_j for intersection with l_{i+1}. This intersection point p lies on an edge e that borders two faces of $\mathcal{A}(\mathcal{L}_i)$. We split e into two edges at p. Next, consider the face f intersected by the part of l_{i+1} to the left of p. Using the order information, we walk along the edges of f away from p and we check for another intersection p' of l_{i+1} with an edge e' on the boundary of f. At the intersection we split e' into two edges, we add an edge to the arrangement for the portion $\overline{pp'}$ of l_{i+1}, and we move to the face on the other (left) side of e'. Once we are done with the faces of $\mathcal{A}(\mathcal{L}_i)$ crossed by l_{i+1} to the left of p, we go back to p and walk to the other side. This way we visit all the faces of the zone of l_{i+1} in $\mathcal{A}(\mathcal{L}_i)$, as well as some of its edges. Updating the DCEL structure due to the splitting or addition of edges is straightforward. The amount of time spent is proportional to the number of edges we visit, and hence bounded by the complexity of the zone of l_{i+1} in $\mathcal{A}(\mathcal{L}_i)$, which is $O(i)$. The total time, over all insertions steps, is thus $O(n^2)$. The space required for the algorithm is the space to maintain the DCEL structure. The same approach extends to higher dimensions; for details see [Ede87, Chapter 7].