line. (In fact, we could do this in $O(\log i)$ time by storing the slopes in an ordered dictionary, but this would not improve our overall running time. By our assumption that no two lines are parallel, there are no duplicate slopes.)

The newly inserted line cuts through a sequence of $i-1$ edges and i faces of the existing arrangement. In order to process the insertion, we need to determine which edges are cut by ℓ_i, and then we split each such edge and update the DCEL for the arrangement accordingly.

In order to determine which edges are cut by ℓ_i, we "walk" this line through the current arrangement, from one face to the next. Whenever we enter a face, we need to determine through which edge ℓ_i exits this face. We answer the question by a very simple strategy. We walk along the edges of the face, say in a counterclockwise direction until we find the exit edge, that is, the other edge that ℓ_i intersects. We then jump to the face on the other side of this edge and continue the trace with the neighboring face. This is illustrated in Fig. 2(a). The DCEL data structure supports such local traversals in time linear in the number of edges traversed. (You might wonder why we don't generalize the trapezoidal map algorithm. We could build a trapezoidal map of the arrangement and walk the new segment through a sequence of trapezoids. It turns out that this would be just as efficient.)

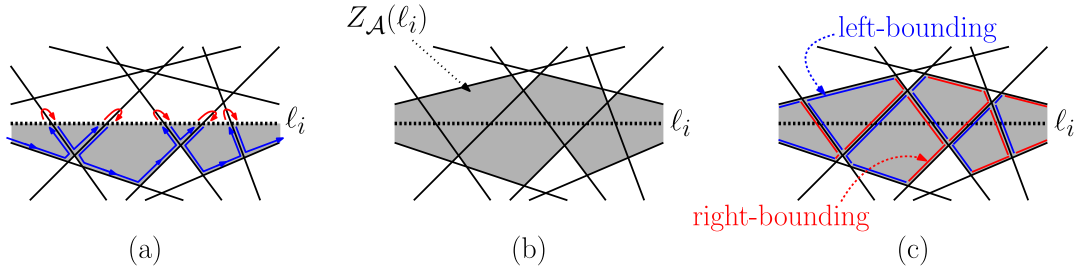

Fig. 2: Adding the line ℓ_i to the arrangement; (a) traversing the arrangement and (b) the zone of a line ℓ_i. (Note that only a portion of the zone is shown in the figure.)

Clearly, the time that it takes to perform the insertion is proportional to the total number of edges that have been traversed in this tracing process. A naive argument says that we encounter $i-1$ lines, and hence pass through i faces (assuming general position). Since each face is bounded by at most i lines, each facial traversal will take $O(i)$ time, and this gives a total $O(i^2)$, which is much higher than the $O(i)$ time that we promised earlier. Why is this wrong? It is based on bound of the total complexity of the faces traversed. To improve this, we need to delve more deeply into a concept of a *zone* of an arrangement.

Zone Theorem: The most important combinatorial property of arrangements (which is critical to their efficient construction) is a rather surprising result called the *zone theorem*. Given an arrangement $\mathcal{A}$ of a set L of n lines, and given a line ℓ that is not in L, the *zone* of ℓ in $\mathcal{A}(L)$, denoted $Z_{\mathcal{A}}(\ell)$, is the set of faces of the arrangement that are intersected by ℓ (shaded in Fig. 2(b)). For the purposes of the above construction, we are only interested in the edges of the zone that lie below ℓ_i, but if we bound the total complexity of the zone, then this will be an upper bound on the number of edges traversed in the above algorithm. The combinatorial complexity of a zone (as argued above) is at most $O(n^2)$. The Zone theorem states that the complexity is actually much smaller, only $O(n)$.