2.5 (a) True (b) True (c) False (d) True

2.11 First split each circle into an upper and lower arc where the left endpoint goes to the lower arc and the right endpoint goes to the upper arc. So the upper and lower arc of a given circle do not intersect.

Now, we modify the line segment intersection algorithm to work with arcs. The endpoint events are handled in the same manner. However the intersection events need to be modified. Two arcs can intersect zero, one or two times, and when they intersect they need not change position. So we now have events for `CrossIntersection` and `TangentIntersection`. For a `CrossIntersection` we proceed as with line segments, and for a `TangentIntersection` we record the intersection but do not change the position of the arc in the data structure. Like with line segments we can find the intersections algebraically in closed form. So the algorithm for circles has the same asymptotic runtime as the line segment algorithm, i.e., $O((n+k)\log n)$ where k is the number of intersections.

8.3 Let F, E and V be the number of faces, edges and vertices in the original arrangement. To use Euler's formula we first need to remove the lines going to infinity. One way to do this is to add large circle containing every intersection in the arrangement and clipping the infinite lines. This produces a planar graph with $F+1$ faces $E+2n$ edges and $V+2n$ vertices. We plug this into Euler's formula and get

$$F + 1 = 2 + (E + 2n) - (V + 2n) = 2 + n^2 - n(n-1)/2 = n^2/2 + n/2 + 2,$$

i.e., $F = n^2/2 + n/2 + 1$, which was to be shown.

A second approach is to add a "point at infinity" and connect all the infinite lines to this point. This produces a planar graph with F faces, E edges and $V + 1$ vertices. Doing the algebra yields the same result.

8.14 Consider the dual problem of finding the points with the most lines going through it. This can be solved by computing the arrangement and walking through the DCEL to find the max degree vertex. This take $O(n^2)$ time using the algorithm in the book.

2.3 A solutions is given on page 29.

8.4 It suffices to show how to find the left most intersection. Consider a vertical line ℓ to the far left of the intersections. The ordering of the lines with respect to their intersection point on ℓ is the same as their slope order. Furthermore, the two lines participating in the left most intersection must be adjacent in this order. So to find the left most intersection we sort the lines by their slope, compute all intersections of neighboring lines, and find the one with smallest x-coordinate. The sort step is the slowest at $O(n \log n)$.