CSL 852, Computational Geometry: Practice Problems

Range searching

1. For the 2d range search tree, we would like to modify the construction in the following way to save space.
   Instead of storing the points in every level, we will store the points in every t-th level, thereby reducing the storage to $O(n/t \log n)$. Consequently, we cannot do the search for the y interval in every node of every level but only in every t-th node of the search path (the canonical subintervals of the x interval). How would you modify the range search reporting and obtain an exact expression for the trade-off between search time and space in terms of t.

   Solution: While reporting points present in the 2D range given, search in the primary x-tree till level t. Atmost 2^t nodes are present in the range required. Search within each of them to get the relevant nodes corresponding to y range. Since only 2 of such nodes at level t can have partial overlaps, so at level $2t$ again, atmost 2^t are selected. Thus overall time complexity of reporting points is $\frac{2^t}{t}(\log n)^2 + k$ when the space is reduced by a factor of t.

2. Given a set S of n points in the plane, design a data structure that returns *number of* points that lie within distance D (for some fixed D) from a given point q. The query should be answered in $O(\text{polylog}(n))$ time and the size of the data structure should be polynomial in n.

   Solution: Construct the arrangement of n circles of radius D centered at the given points. For each cell of the arrangement, the answer to the query q is fixed (exactly those points whose circles contain q. You must show how to preprocess the circle arrangement to do point location in polylog time.

   Otherwise, you can argue on the basis of the lifting transform, that the inside and outside of the disks are mapped to below and above a hyperplane. So the number of intersecting disks at any given point is the number of planes (in 3d) above the vertical projection.

   For this you must show how to preprocess a set of planes in 3d to do point location in polylog time. *This was not done explicitly in the lectures.*