2 Delaunay triangulation: definitions, motivations, properties, classical algorithms.

2.1 Drawing
Draw the Delaunay triangulation of the attached point set.

2.1 Correction:

2.2 Nearest neighbor graphs

\( S \) a set of \( n \) points. \( q_1 \in S \). Let \( q_1 \) denote the nearest neighbor of \( q_0 \) in \( S \setminus \{q_0\} \). Let \( q_2 \) denote the second nearest neighbor of \( q_0 \) in \( S \), i.e., the nearest neighbor in \( S \setminus \{q_0, q_1\} \). Similarly \( q_i \) the \( i^{th} \) nearest neighbor.

The directed nearest neighbor graph of \( S \) is the graph whose vertices are the points in \( S \) and \( pq \) is an edge of the graph if \( q \) is the nearest neighbor of \( p \).
Fact: The degree of the nearest neighbor graph is \( \leq 6 \). (proof optional).

2.2.1 Nearest neighbor
Prove that \( q_0q_1 \) is an edge of the Delaunay triangulation of \( S \).

2.2.2 Second nearest neighbor
Prove that \( q_0q_2 \) or \( q_1q_2 \) is an edge of the Delaunay triangulation of \( S \).

2.2.3 \( k^{th} \) nearest neighbor
Prove that \( \forall k \exists i < k \) such that \( q_kq_i \) is an edge of the Delaunay triangulation of \( S \).

2.2.4 Nearest neighbor graph
Write an algorithm that takes the Delaunay triangulation of \( S \) and output the directed nearest neighbor graph of \( S \).

You can write things like:

\begin{verbatim}
for v enumerating all vertices of DT(S),
  for w enumerating the neighbor of v in DT(S),
  or output edge(v, w),
  or v.color = red to add some information in a vertex (or edge or...)
\end{verbatim}

What is the complexity of this algorithm?

2.2.5 Nearest neighbor graph
Write an algorithm that takes the Delaunay triangulation of \( S \) and output the directed second nearest neighbor graph of \( S \).

What is the complexity of this algorithm?
2.2 Correction:

2.2.1 Nearest neighbor

The disk centered at \( q_0 \) passing through \( q_1 \) contains only \( q_0 \), thus the disk of diameter \( q_0 q_1 \), which is included in the previous one is empty. By the empty circle property, \( q_0 q_1 \) is a Delaunay edge.

2.2.2 Second nearest neighbor

The disk \( D_2 \) centered at \( q_0 \) passing through \( q_2 \) contains only \( q_0 \) and \( q_1 \), thus we consider the two disks \( Z_0 \) and \( Z_1 \) passing through \( q_2 \) tangent in \( q_0 \) to \( D_2 \) and respectively passing through \( q_0 \) and \( q_1 \). We have to cases:

- \( Z_0 \subset Z_1 \subset D_2 \) and \( Z_0 \) is empty, by the empty circle property, \( q_0 q_2 \) is a Delaunay edge.
- \( Z_1 \subset Z_0 \subset D_2 \) and \( Z_1 \) is empty, by the empty circle property, \( q_1 q_2 \) is a Delaunay edge.

2.2.3 \( k^{th} \) nearest neighbor

The disk of center \( q_0 \) through \( q_k \) verifies \( D_k \cap S = \{q_0,q_1 \ldots q_{k-1}\} \). Consider the pencil of circles through \( q_k \) tangent to \( D_k \). The biggest empty circle of that pencil inside \( D_k \) pass through a point inside \( D_k \) that is some \( q_i \) with \( i < k \) and by the empty circle property, \( q_i q_k \) is a Delaunay edge.

2.2.4 Nearest neighbor graph

for \( u \) enumerating all vertices of \( DT(S) \) {
  \( d = \infty \);
  for \( w \) enumerating the neighbor of \( u \) in \( DT(S) \) {
    if \( \|uw\| < d \) then \{ \( nn = w; \ \ d = \|uw\|; \} \)
  }
  output edge\((u,nn)\),
}

The inside loop costs \( d^i(u) \), thus the total cost of the algorithm is \( \sum_{u \in S} d^i(u) < 6n \).

2.2.5 Nearest neighbor graph

for \( u \) enumerating all vertices of \( DT(S) \) {
  \( u.d = \infty \);
  for \( w \) enumerating the neighbor of \( u \) in \( DT(S) \) {
    if \( \|uw\| < d \) then \{ \( nn = w; \ \ d = \|uw\|; \} \)
  }
}
for \( u \) enumerating all vertices of \( DT(S) \) {
  \( d = \infty \);
  for \( w \) enumerating the neighbor of \( u \) in \( DT(S) \) {
    if \( \|uw\| < d \) and \( w \neq u.nn \) then \{ \( sn = w; \ \ d = \|uw\|; \} \)
    for \( w \) enumerating the neighbor of \( u.nn \) in \( DT(S) \) {
      if \( \|uw\| < d \) and \( w \neq u \) then \{ \( sn = w; \ \ d = \|uw\|; \} \)
      output edge\((u,sn)\),
    }
  }
}

The cost is

\[
\sum_{u \in S} (d^i_{DT}(u) + d^i_{DT}(u.nn)) = \sum_{u \in S} d^i_{DT}(u) + \sum_{u \in S} \sum_{v \in \{u.nn\}} d^i_{DT}(v)
= \sum_{u \in S} d^i_{DT}(u) + \sum_{v \in S} \sum_{u \text{ such that } v = u.nn} d^i_{DT}(v)
= \sum_{u \in S} d^i_{DT}(u) + \sum_{v \in S} d^i_{NN}(v) \cdot d^i_{DT}(v)
= \sum_{u \in S} d^i_{DT}(u) + \sum_{v \in S} 6d^i_{DT}(v) \leq 42n
\]