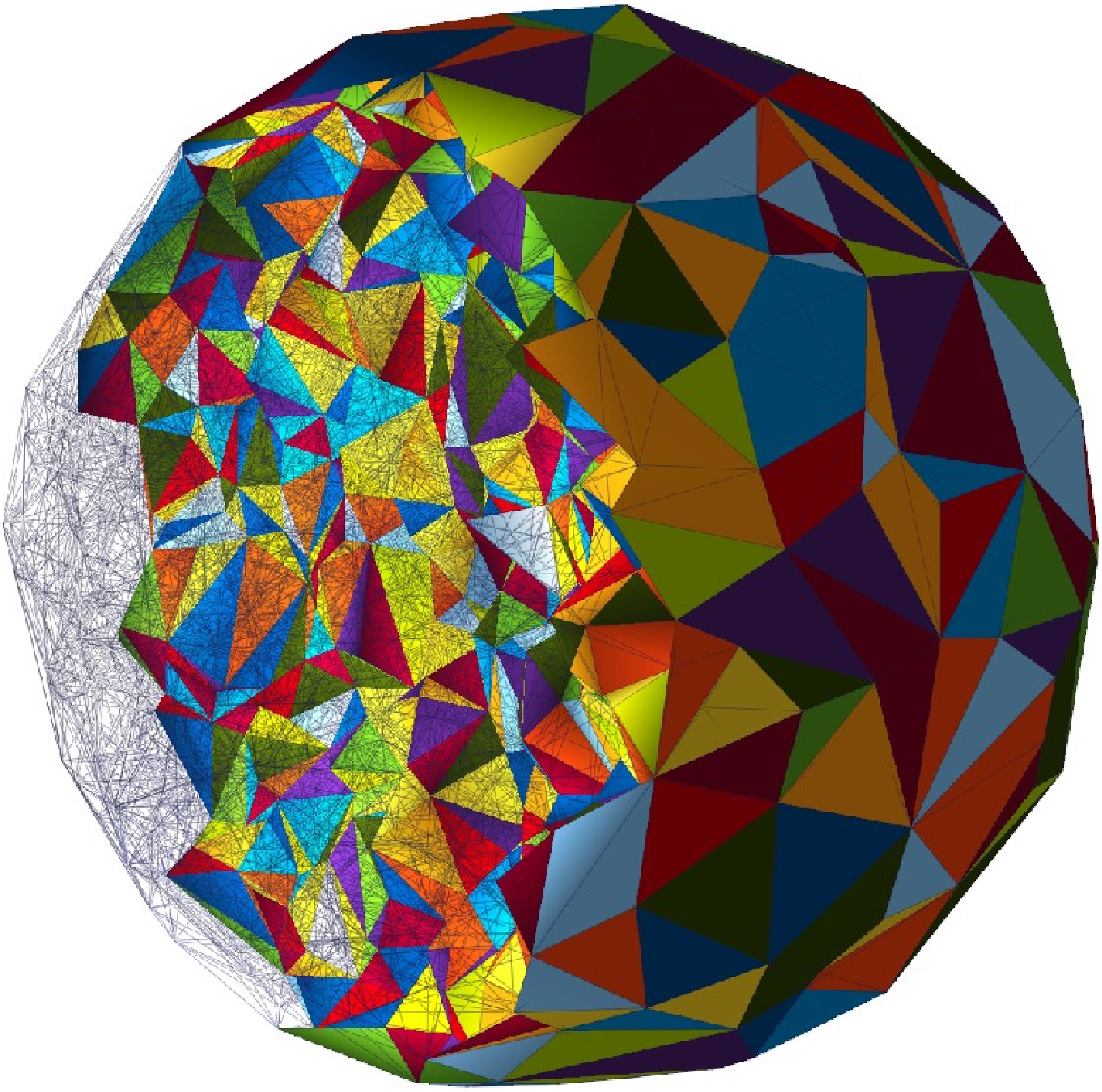
Υπολογιστική Γεωμετρία

Εργασία 2 (Delaunay)

ΒΛΑΣΣΗΣ ΠΑΝΑΓΙΩΤΗΣ
1115201400022

Εαρινό εξάμηνο
Ακαδ. Έτος: 2018-19



ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
Εθνικό και Καποδιστριακό
Πανεπιστήμιο Αθηνών

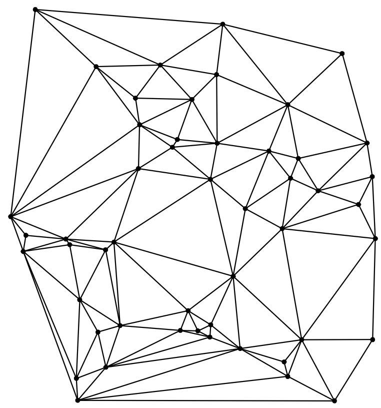
ΤΜΗΜΑ
ΠΛΗΡΟΦΟΡΙΚΗΣ & ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

Περιεχόμενα

1. Γενικά – Τριγωνοποίηση

Ορισμός: <u>*Τριγωνοποίηση*</u> (triangulation) συνόλου σημείων ή εστιών στο επίπεδο καλείται ένα σύνολο τριγώνων με τις εξής ιδιότητες:

1. το σύνολο των κορυφών των τριγώνων ισούται με το σύνολο των δεδομένων εστιών,
2. η ένωση όλων των τριγώνων ισούται με το κυρτό περίβλημα των εστιών και
3. η τομή δύο τριγώνων αντιστοιχεί σε ακριβώς μία εκ των εξής περιπτώσεων: είναι κενή, ισούται με την κοινή κορυφή των δύο τριγώνων, ή ισούται με την κοινή τους ακμή.



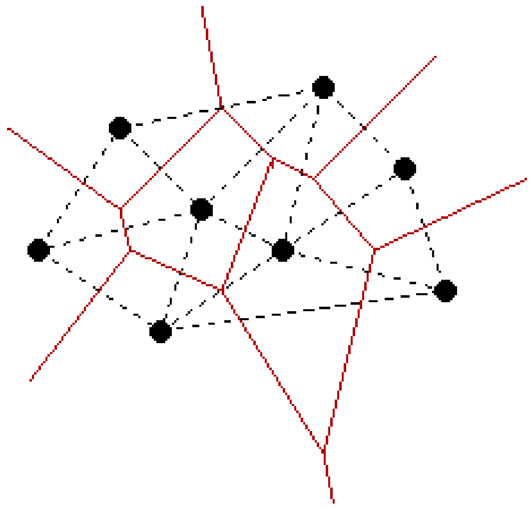
Εικόνα 1.1
Τριγωνοποίηση

Οι ιδιότητες 2 και 3 ορίζουν μια υποδιαίρεση (ή διαμέριση) του κυρτού τους περιβλήματος σε τρίγωνα. Η ιδιότητα 1 δηλώνει πως η τριγωνοποιήση αποτελεί ένα πολυγωνικό σύμπλοκο. Αφού τα τρίγωνα είναι άπλοκα, πρόκειται για πολυγωνικό απλοειδές σύμπλοκο. Η τριγωνοποίηση γενικεύεται αν αγνοήσουμε την πρώτη συνθήκη.

2. Τριγωνοποίηση Delaunay

Ορισμός: Η *τριγωνοποίηση Delaunay* δεδομένου συνόλου εστιών στο επίπεδο είναι ο δυϊκός γράφος του διαγράμματος Voronoi των εστιών:

1. Τα κελιά Voronoi αντιστοιχούν σε εστίες, δηλαδή κορυφές τριγώνων Delaunay.
2. Κάθε ζεύγος γειτονικών κελιών (δηλαδή ακμή Voronoi) αντιστοιχεί σε μία ακμή Delaunay, που ορίζεται από τις 2 εστίες: οι ευθείες των δύο ακμών είναι κάθετες μεταξύ τους.
3. Κάθε κορυφή Voronoi αντιστοιχεί σε ένα τρίγωνο Delaunay (ή κυρτό πολύγωνο)



Εικόνα 2.1
Δυϊκό διάγραμμα Voronoi - Delaunay

- *Θεώρημα*: Στην τριγωνοποίηση Delaunay ενός συνόλου εστιών:
    1. Τρεις εστίες ορίζουν τρίγωνο ανν ο περιγεγραμμένος ανοικτός κύκλος του είναι κενός, δηλαδή δεν περιέχει εστίες στο εσωτερικό του.
    2. Δύο εστίες ορίζουν ακμή ανν υπάρχει κενός κύκλος με τις εστίες στην περιφέρειά του, δηλαδή υπάρχει κλειστός δίσκος με χορδή η οποία ορίζεται από τις δύο εστίες και δεν περιέχει καμία άλλη εστία.

- *Λήμμα 1*: Η τριγωνοποίηση Delaunay είναι επίπεδος γράφος, με κόμβους τις κορυφές των τριγώνων→Γενικότερα, οποιαδήποτε τριγωνοποίηση είναι επίπεδος γράφος.
- *Λήμμα 2*: Έστω n μη συνευθειακές εστίες στο επίπεδο και k το πλήθος κορυφών στο κυρτό πολύγωνο τους. Τότε υπάρχουν 2n-k-2 τρίγωνα και 3n-k-3 ακμές σε κάθε τριγωνοποίηση των εστιών.
- *Θεώρημα*: Μία τριγωνοποίηση είναι έγκυρη ανν είναι Delaunay.
- *Πόρισμα*: Η τριγωνοποίηση Delaunay αντιστοιχεί στο λεξικογραφικά βέλτιστο διάνυσμα γωνιών.

Η τριγωνοποίηση Delaunay δεν είναι μοναδική αν το το δυϊκό του γράφου Voronoi περιέχει κυρτά πολύγωνα τα οποία τριγωνοποιήθηκαν αυθαίρετα. Είναι δυνατό να αποδειχτεί πως όλες οι τριγωνοποιήσεις Delaunay οδηγούν στο ίδιο (βέλτιστο) διάνυσμα γωνιών.

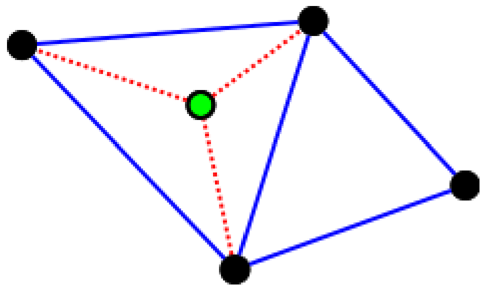
3. Αλγόριθμοι κατασκευής τριγωνοποίσης Delaunay

3.1 Αυξητικός αλγόριθμος (Incremental Algorithm) [1]

3.1.1 Περιγραφή αλγορίθμου

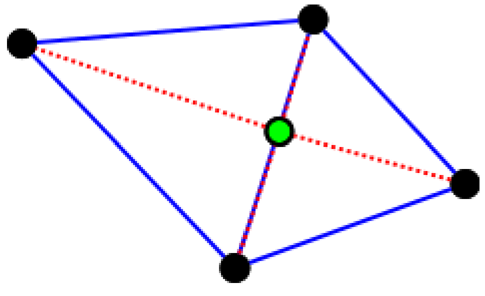
The incremental Delaunay triangulation computes the triangulation building an initial triangle with three points of the triangulation and then adding the remaining points. When adding a new point new triangles are created and it is necessary to check if those new triangles fulfill the Delaunay triangulation conditions; if they do, then it continues inserting the next point; if not, it is necessary to flip some edges of the triangulation and then continue with the next point.

For every new point to insert, the algorithm finds the triangle that encloses this new point, splits the triangle into 3 (or 4) new triangles and then check if any of the new edges must be flipped. The split triangle operation is depicted below: if a new point is inserted and it falls into any of the existing triangles then this triangle is splitted into three triangles. ***(Εικόνα 3.1.1)***



Εικόνα 3.1.1

There is a degenerate case: if the point inserted is collinear to any of the edges then two triangles are splitted into two new, so creating 4 new triangles. This split operation is illustrated as follows: ***(Εικόνα 3.1.2)***



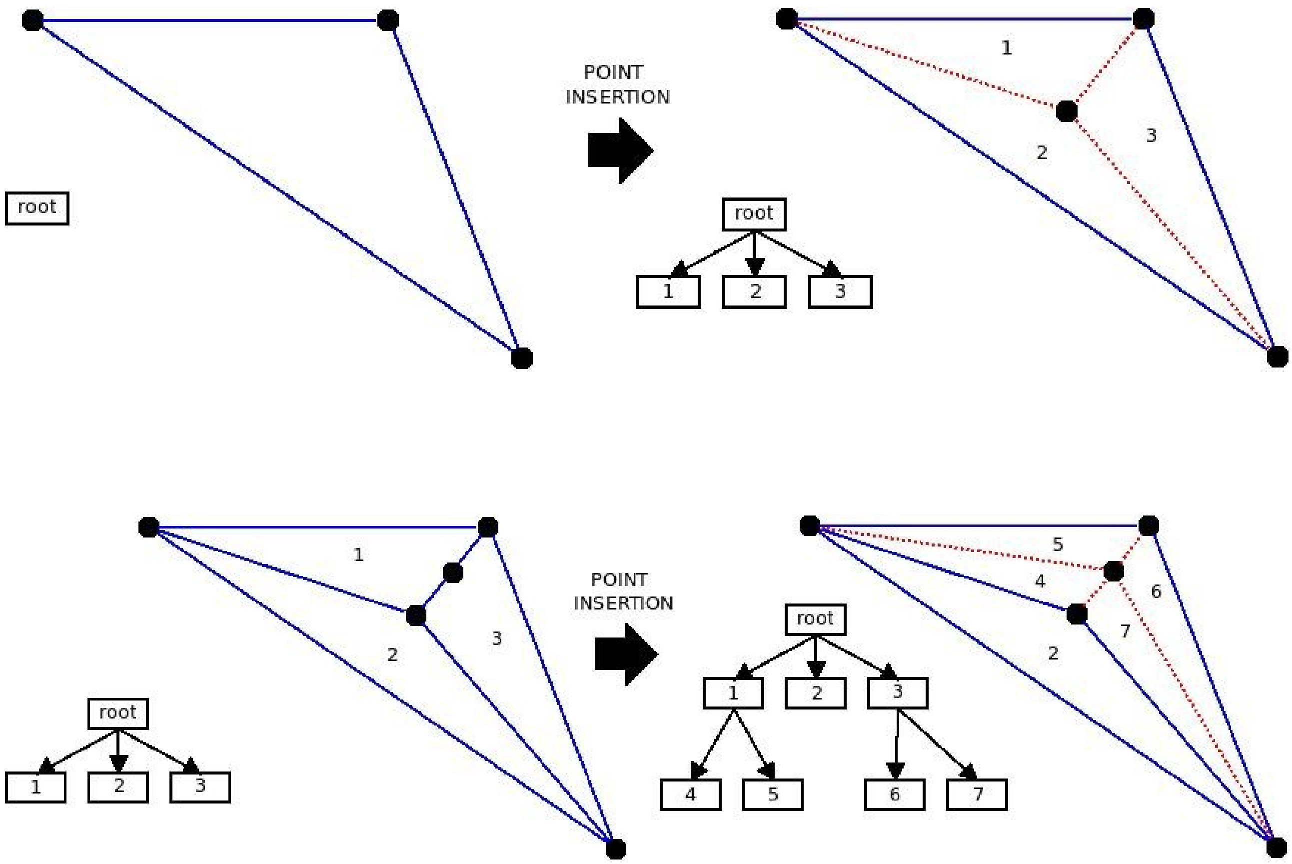
Εικόνα 3.1.2

3.1.2 Τρόπος υλοποίησης αλγορίθμου

The keys of the algorithm performance is ***(1)*** *<u>the location of the triangle that encloses the new point</u> and* ***(2)*** *<u>the number of edges to flip after new triangles are created</u>.*

To be able to efficiently solver first issue, find the triangle that encloses the point to add, a new data structure is used, a tree. When a new point is inserted, the triangle that encloses the point is found and the node in the tree that stores the triangle is updated holding new children nodes that stores the new triangles.

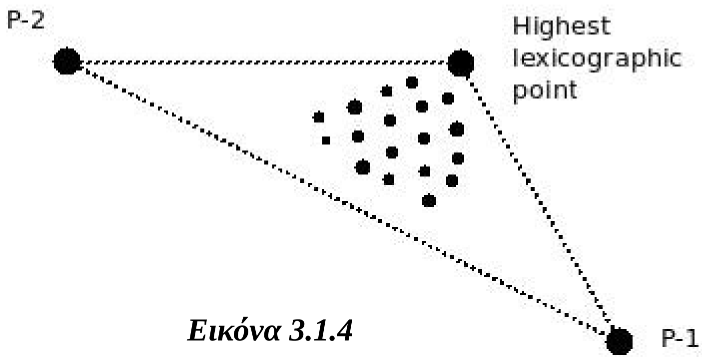
So, in this algorithm there are two data structures to maintain: the DCEL that stores the triangulation and the tree that stores the all triangles built during the process (and not only the final triangles of the triangulation).



Εικόνα 3.1.3

3.1.3 Βήματα υλοποίησης αλγορίθμου

*So, the **first step** of the algorithm is to create this imaginary triangle, inserting it as root in the tree and inserting it in the DCEL. This imaginary triangle idea is depicted below:*

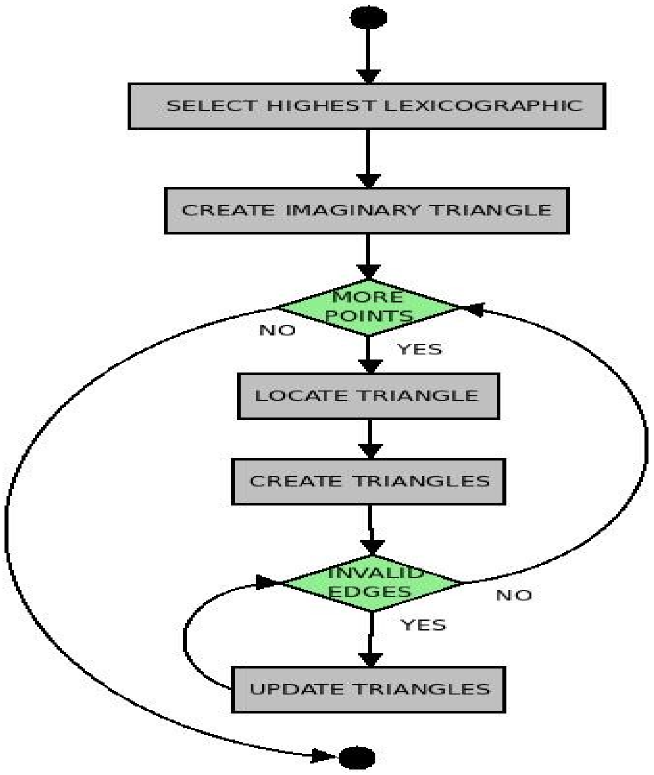


Εικόνα 3.1.4

__Second step__ is a loop executed for all the remaining points of the set. This step is executed in several stages:

- *Locate the triangle that encloses the point.*
- *Create three (or four) new triangles.*
- *Update the DCEL and the tree with the new triangles.*
- *Check if any edge is not a Delaunay edge and flip it if not.*

The last step can modify the current triangulation and it could require to check more edges.

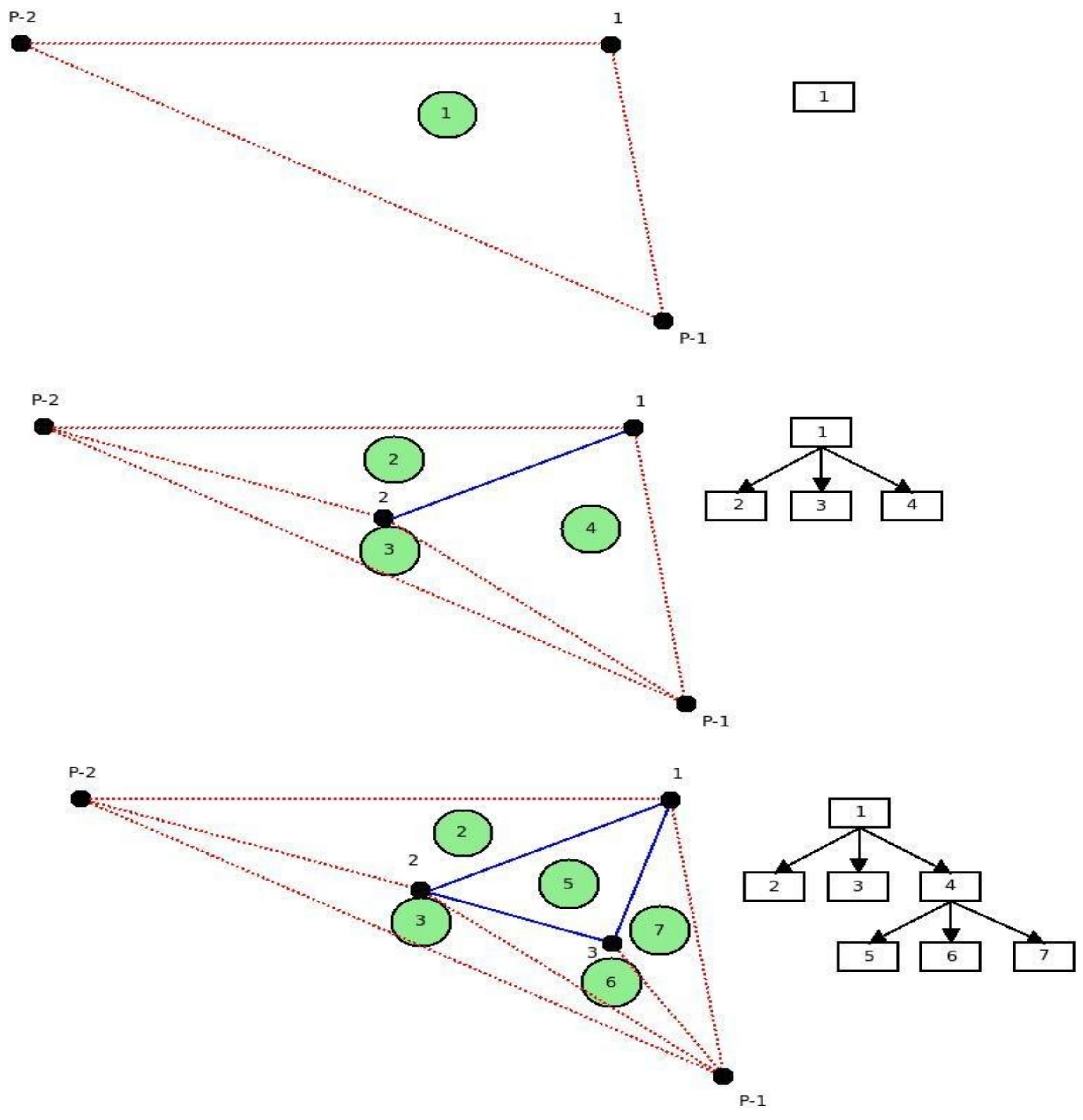


Πολυπλοκότητα: O(nlogn)

Εικόνα 3.1.5
Διάγραμμα ροής αυξητικού αλγορίθμου

3.1.4 Παράδειγμα εκτέλεσης αλγορίθμου

The image included below illustrates the three first insertions in the triangulation. As can be show, before there are three real points in the triangulation there are several triangles, but only one of them is a real triangle (the update of the tree structure is also included). The only real faces is face number 5, and the remaining faces are all imaginary faces.



Εικόνα 3.1.6

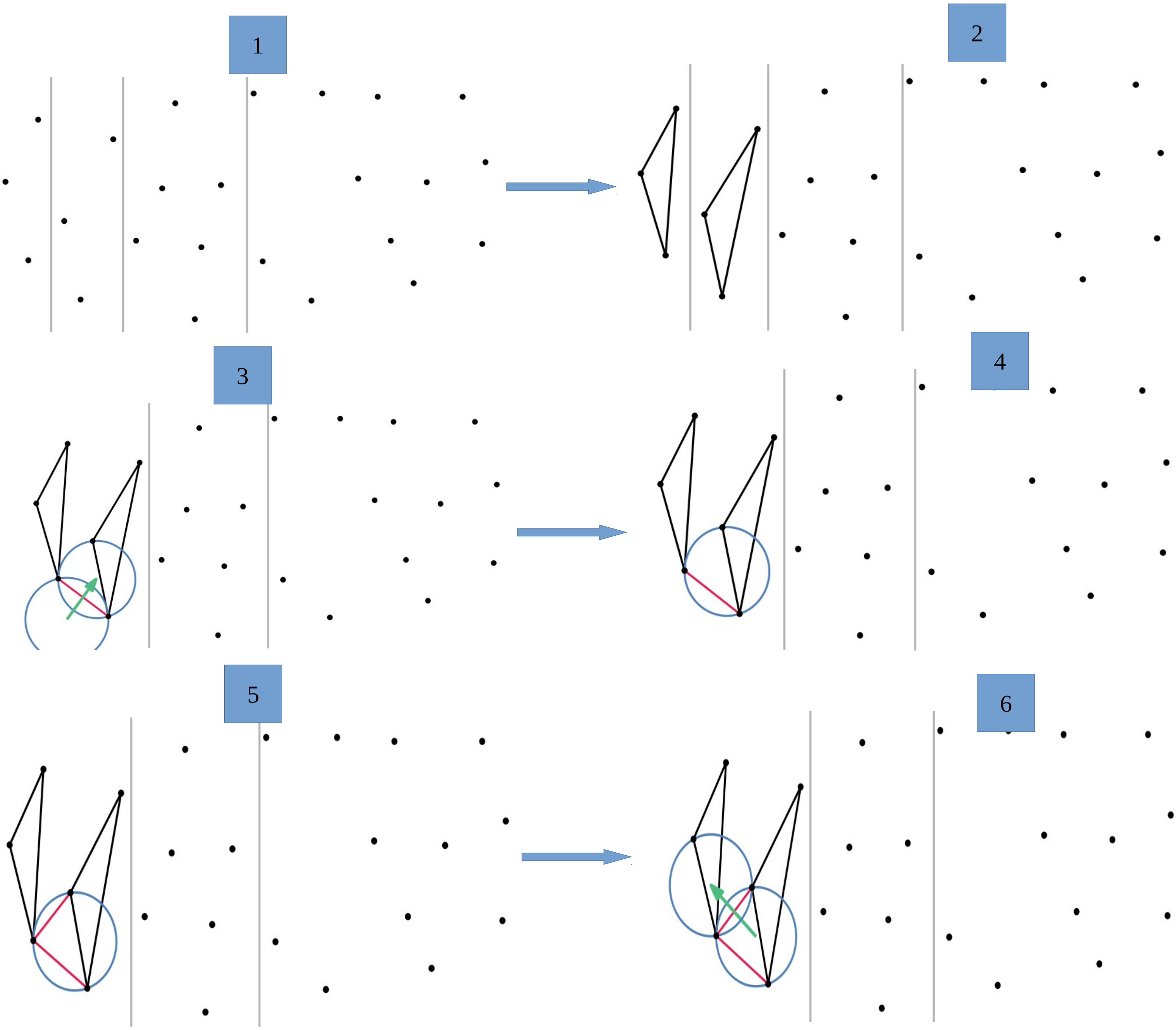
Παράδειγμα εκτέλεσης αυξητικού αλγορίθμου

3.2 Αλγόριθμος Διαίρει και Βασίλευε (Divide and Conquer Algorithm) [2]

Για την εκτέλεση του αλγορίθμου Διαίρει και Βασίλευε απαιτούνται 2 βασικά βήματα κατά τη φάση της συγχώνευσης (merging):

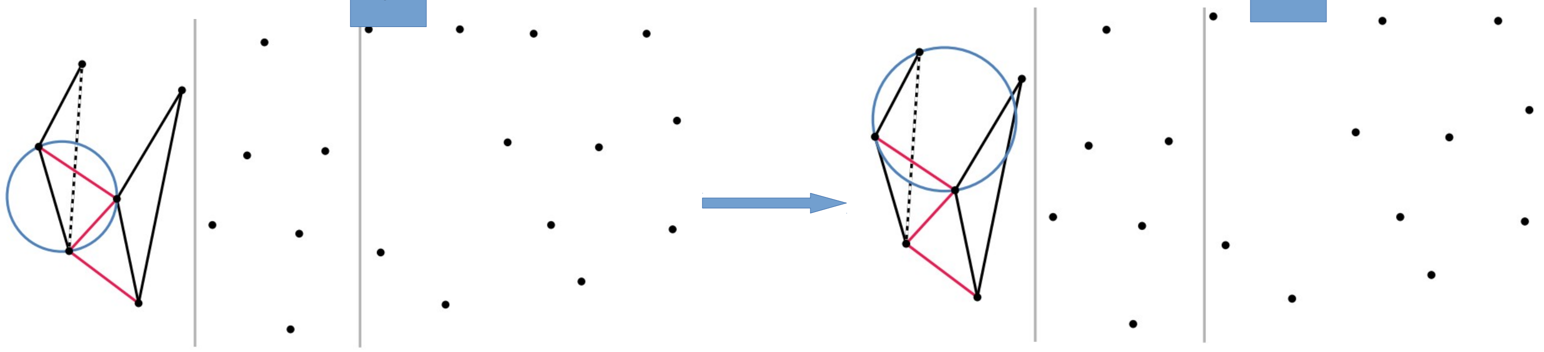
- Create triangles with empty circumcircles
- Remove conflicting edges

Πολυπλοκότητα: O(nlogn)



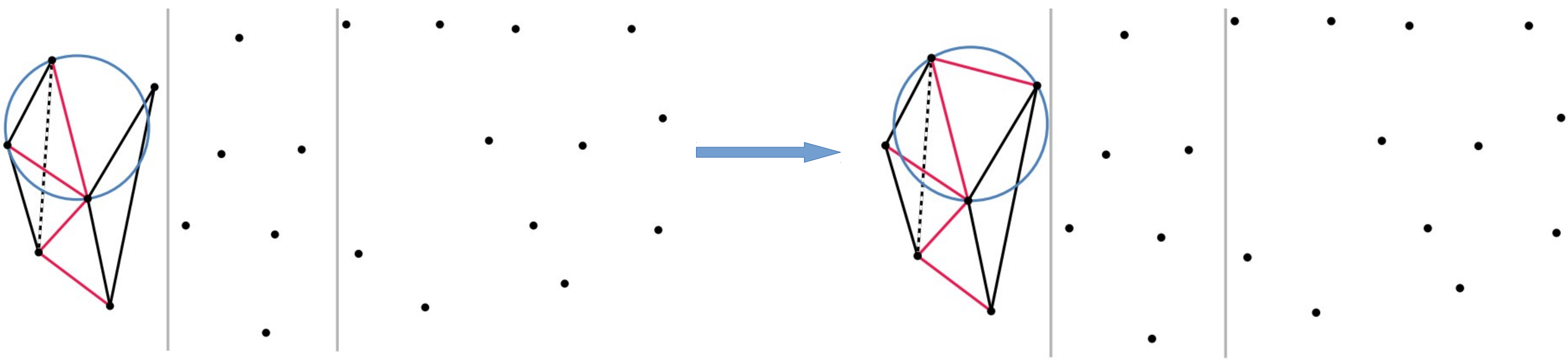
7

8

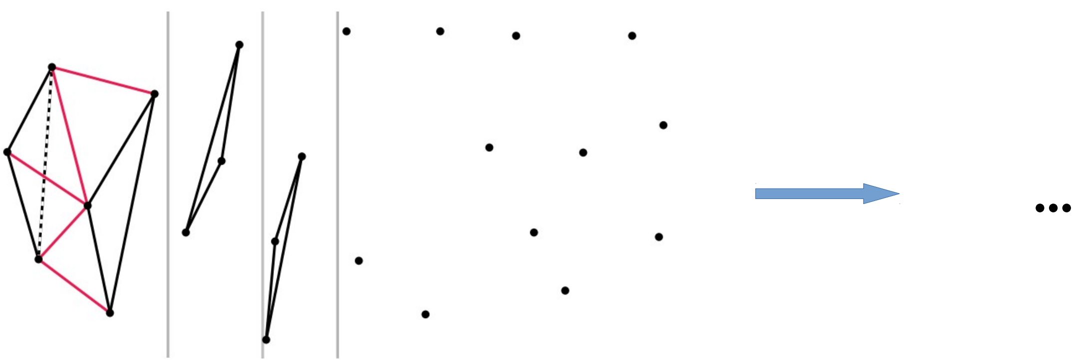


9

10

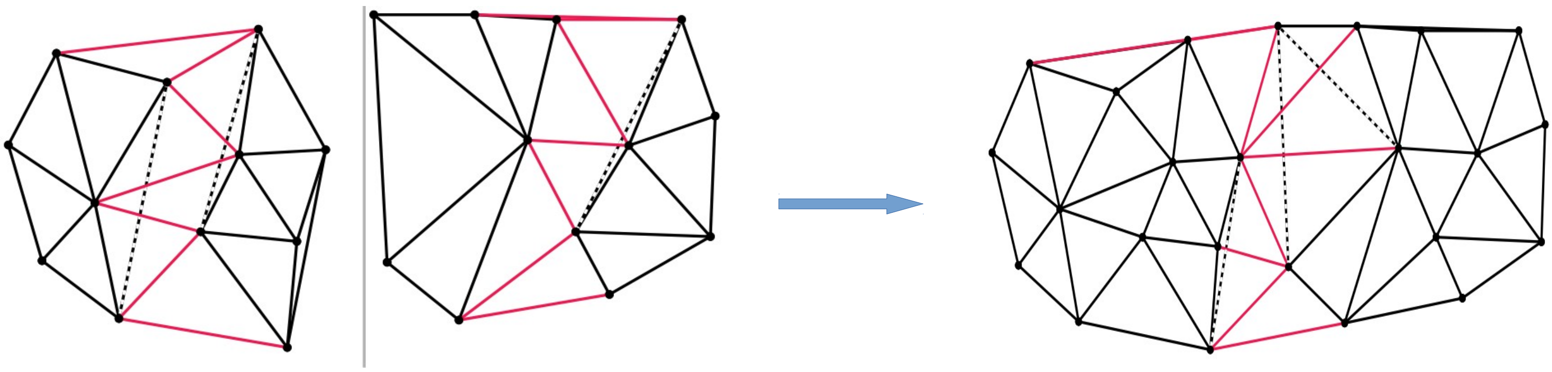


11

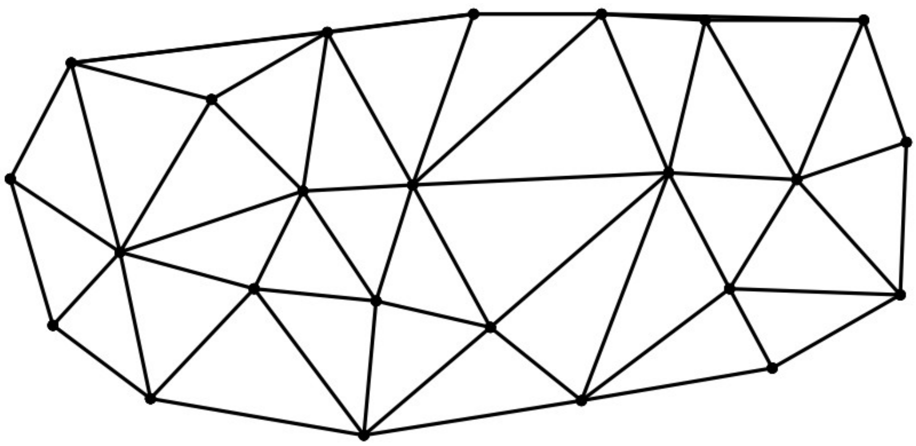


12

13



15



Τριγωνοποίηση Delaunay με Divide & Conquer

Βιβλιογραφία

[1] https://github.com/juannavascalvente/Delaunay/wiki/Incremental-Delaunay-triangulation

[2] http://www.ist.tugraz.at/_attach/Publish/Dcg/Delaunay_4.pdf

[3] https://www.comp.nus.edu.sg/~tants/gdel3d_files/3ddt.png

[4] *Γιάννης Ζ. Εμίρης, Υπολογιστική Γεωμετρία, σελ.200-211*