Σχετικά με την επιλογή της παραμέτρου για το αλγόριθμο αρχικοποίησης convex hull, η λογική λέει πως ανάλογα με το είδος της βελτιστοποίησης που θέλουμε, πρέπει να επιλέξουμε και την αντίστοιχη επιλογή στο edge selection. Λογικά με την τυχαία επιλογή αναμένουμε να παίρνουμε πολύγωνα χωρίς κάποια ιδιαίτερη ιδιότητα από άποψη εμβαδού, ενώ με edge selection min (και αντίστοιχα max) περιμένουμε να παίρνουμε πολύγωνα με σχετικά μικρά εμβαδά. Ας ρίξουμε μια ματιά και στα πειραματικά δεδομένα που παρουσιάζονται στους παρακάτω πίνακες:

PointSet	Avg.score (3 runs)	Avg.Time(3 runs)
euro-night-20	0.563	57 ms
euro-night-30	0.412	241 ms
euro-night-40	0.459	651 ms
euro-night-50	0.467	1598 ms
uniform-70-1	0.561	5479 ms
us-night-80	0.404	8800 ms
london-100	0.474	19321 ms

Πίνακας 1.Random Edge Selection Convex Hull

PointSet	Score	Time
euro-night-20	0.262	62 ms
euro-night-30	0.192	228 ms
euro-night-40	0.174	665 ms
euro-night-50	0.343	1604 ms
uniform-70-1	0.259	5637 ms
us-night-80	0.304	10316 ms
london-100	0.216	22726 ms

Πίνακας 2.Min Edge Selection Convex Hull

PointSet	Score	Time
euro-night-20	0.863	57 ms
euro-night-30	0.869	286 ms
euro-night-40	0.888	685 ms
euro-night-50	0.830	1885 ms
uniform-70-1	0.892	5257 ms
us-night-80	0.941	8427 ms
london-100	0.860	22724 ms

Πίνακας 3.Max Edge Selection Convex Hull

Πράγματι, η υπόθεση μας φαίνεται να επαληθεύεται αφού για random edge selection τα πολύγωνα που προκύπτουν δεν δείχνουν να έχουν

κάποια ιδιαίτερη προτίμηση ως προς το εμβαδό που παράγεται. Μάλιστα, αν έπρεπε να κάνουμε κάποιο σχόλιο, αυτό θα ήταν πως η τυχαία επιλογή παράγει μάλλον «κακά» πολύγωνα αφού το σκορ τους δεν είναι ούτε ιδιαίτερα μεγάλο (>0.70) ούτε ιδιαίτερα μικρό (<0.35) έτσι ώστε να επωφεληθούμε κάπως κατά την διαδικασία της βελτιστοποίησης. Από την άλλη η min edge selection (και αντίστοιχα και η max) παράγει αρκετά καλά πολύγωνα (το score τους είναι <0.35) κάτι που μας είναι χρήσιμο κατά την διαδικασία της βελτιστοποίησης. Το μόνο πρόβλημα που υπάρχει είναι πως το edge selection min και το edge selection max είναι πιο αργό από το random edge selection κάτι που γίνεται ιδιαίτερα αισθητό όσο μεγαλώνουν τα σημειοσύνολα. Σε γενικές γραμμές πάντως, για το σκοπό της παραγωγής και της βελτιστοποίησης ενός πολυγώνου προτιμούνται και επιλέγονται τα edge selection max και min (αντίστοιχα και με την βελτιστοποίηση) αφού τα πολύγωνα με χαμηλό αρχικό score πρέπει να περάσουν περισσότερο χρόνο στην διαδικασία της βελτιστοποίησης.

Για τον incremental, έχει βάση να κάνουμε τις ίδιες υποθέσεις με τον convex hull, σημειώνοντας πως η επιλογή initialization δεν επηρεάζει ,με κάποιο τρόπο που να μπορούμε να εκμεταλλευτούμε, το εμβαδό του πολυγώνου παρότι σίγουρα παράγονται διαφορετικά πολύγωνα. Αυτό που επηρεάζει το εμβαδό είναι το edge selection όπως και στον convex hull. Ας δούμε τα πειραματικά μας δεδομένα για τα ίδια σύνολα με πριν στους παρακάτω πίνακες:

	Avg.score (3 runs)	Avg.Time(3 runs)
euro-night-20	0.612	24 ms
euro-night-30	0.480	38 ms
euro-night-40	0.420	65 ms
euro-night-50	0.536	114 ms
uniform-70-1	0.509	250 ms
us-night-80	0.433	315 ms
london-100	0.451	643 ms

Πίνακας 4.Random Edge Selection Incremental

	Score	Time
euro-night-20	0.472	20 ms
euro-night-30	0.492	37 ms
euro-night-40	0.364	87 ms
euro-night-50	0.569	120 ms
uniform-70-1	0.463	218 ms
us-night-80	0.342	311 ms
london-100	0.378	387 ms

Πίνακας 5.Min Edge Selection Incremental

	Score	Time
euro-night-20	0.573	27 ms
euro-night-30	0.623	49 ms
euro-night-40	0.587	57 ms
euro-night-50	0.484	137 ms
uniform-70-1	0.543	261 ms
us-night-80	0.796	429 ms
london-100	0.630	480 ms

Πίνακας 6.Max Edge Selection Incremental

Όπως παρατηρούμε, τα αποτελέσματα ναι μεν ακολουθούν την λογική του convex hull αλλά όχι με τόσο μεγάλη συνέπεια. Τα εμβαδά που προκύπτουν από το random edge selection είναι οριακά χειρότερα από αυτά που ακολουθούν κάποιο άπληστο κριτήριο. Βέβαια, αυτό που έχει ενδιαφέρον είναι πως, ειδικά για το min edge selection, οι χρόνοι φαίνονται να είναι καλύτεροι από το random. Γενικά, στην περίπτωση μας που θέλουμε βελτιστοποίηση θα επιλέγαμε το edge selection με το άπληστο κριτήριο απλά και μόνο λόγω του καλύτερου εμβαδού.

Βλέποντας και τους χρόνους, το random edge selection φαντάζει όλο και λιγότερο ελκυστικό. Σαν γενικό σχόλιο, αυτό που φανερώνουν τα δεδομένα είναι πως ο incremental παράγει αισθητά χειρότερα εμβαδά σε σχέση με τον convex hull αλλά είναι πολύ πιο γρήγορος, κάτι που γίνεται αισθητό στα μεγάλα σημειοσύνολα. Οπότε, από ένα σημείο και μετά, ο incremental προτιμάται από τον convex hull απλά και μόνο λόγω του χρόνου.

Βέβαια, εδώ πρέπει να σημειωθεί πως και ο incremental έχει τα όρια του. Έτσι, σαν τελική επιλογή για σημειοσύνολα αρκετά μεγάλα (500+) επιλέγεται ο αλγόριθμος onion. Η επιλογή της αρχική κορυφής m, αν και προφανώς διαφοροποιεί τα παραγόμενα πολύγωνα, δεν είναι κάτι το οποίο μπορούμε να εκμεταλλευτούμε άμεσα και αυτό γιατί δεν συσχετίζεται με κάποιο άπληστο κριτήριο για το εμβαδόν. Παρακάτω παρουσιάζονται τα πειραματικά δεδομένα για τα ίδια σημειοσύνολα με πριν αλλά για τον αλγόριθμο onion:

	score	Time
euro-night-20	0.649	1 ms
euro-night-30	0.492	1 ms
euro-night-40	0.525	2 ms
euro-night-50	0.640	4 ms
uniform-70-1	0.530	8 ms
us-night-80	0.587	11 ms
london-100	0.636	16 ms

Πίνακας 7.Onion Algorithm

Όπως είναι πασιφανές, ο onion αλγόριθμος είναι τρομακτικά πιο γρήγορος σε σχέση με τους 2 προηγούμενους. Αυτό βέβαια οφείλεται στο γεγονός ότι δεν εξετάζει ποτέ του τα εμβαδά που αφαιρούνται κατά την σύνδεση των επάλληλων ΚΠ. Σαν συνέπεια, τα πολύγωνα που προκύπτουν έχουν score το οποίο δεν τα καθιστά καλά ούτε για μεγιστοποίηση αλλά ούτε και για ελαχιστοποίηση. Αυτό όμως είναι κάτι το οποίο θα πρέπει να ανεχτούμε, αφού για σημειοσύνολα με 1000+ σημεία ο αλγόριθμος αυτός αποτελεί μονόδρομο.

Εν κατακλείδι, όσο αναφορά την αρχικοποίηση, καταλήγουμε σε κάποια συμπεράσματα. Αρχικά, η random edge selection, όπου προσφέρεται,

δεν έχει ιδιαίτερο νόημα. Οι αλγόριθμοι που την υποστηρίζουν έχουν την δυνατότητα για πολύ καλύτερη αποτελέσματα (ανάλογα πάντα με την βελτιστοποίηση που θέλουμε) αν στην θέση της επιλεγεί min ή max. Έτσι, τόσο ο incremental όσο και ο convex hull, θα επιλέγουν το αντίστοιχο άπληστο κριτήριο ανάλογα και με την βελτιστοποίηση που θέλουμε. Έπειτα, ο convex hull είναι αυτός ο οποίος , χρησιμοποιώντας πάντα και min/max στο edge selection, επιστρέφει τα καλύτερα εμβαδά, κάτι που επηρεάζει και την βελτιστοποίηση αφού όσο καλύτερο είναι το αρχικό πολύγωνο που έχουμε τόσο λιγότερο χρόνο θα ξοδέψουμε στο να το βελτιστοποιήσουμε. Παρόλα αυτά, είναι και ο πιο χρονοβόρος αλγόριθμος, γεγονός που περιορίζει την εφαρμογή του σε σημειοσύνολα σχετικά μικρά (<=100 σημεία). Επόμενος αλγόριθμος που προτιμάται, είναι ο incremental ο οποίος επιστρέφει τα 2^{α} καλύτερα αποτελέσματα, είναι πιο γρήγορος από τον convex hull αλλά και αυτός έχει περιορισμένη εφαρμογή, με τα όρια του να δοκιμάζονται στα 500 σημεία. Τέλος, όσο αυξάνεται το μέγεθος των σημειοσυνόλων τόσο περισσότερο μοιάζει με μονόδρομο η χρήση του onion. Μπορεί τα πολύγωνα που επιστρέφει να μην είναι ιδιαίτερα καλά αλλά η ταχύτητά του είναι ασυναγώνιστη, για αυτό άλλωστε και προτιμάτε για τα πραγματικά μεγάλα σημειοσύνολα (1000+ σημεία).