```
Sort-and-Count(L)
  If η λίστα έχει ένα στοιχείο then
     δεν υπάρχουν αντιστροφές
  Else
     Διαίρεσε τη λίστα σε δύο τμήματα:
        το A περιέχει τα πρώτα ⌈n/2⌉ στοιχεία
        το B περιέχει τα υπόλοιπα ⌊n/2⌋ στοιχεία
     (r_A, A) = Sort-and-Count(A)
     (r_B, B) = Sort-and-Count(B)
     (r, L) = Merge-and-Count(A, B)
  Endif
  Return το r = r_A + r_B + r, και την ταξινομημένη λίστα L
```

Επειδή η διαδικασία Merge-and-Count χρειάζεται χρόνο $O(n)$, ο χρόνος εκτέλεσης $T(n)$ για ολόκληρη τη διαδικασία Sort-and-Count ικανοποιεί την αναδρομή (5.1). Λόγω της (5.2) έχουμε:

> **(5.7)** *Ο αλγόριθμος Sort-and-Count ταξινομεί σωστά τη λίστα εισόδου και μετρά τον αριθμό των αντιστροφών· ο αλγόριθμος αυτός εκτελείται σε χρόνο $O(n \log n)$ για μια λίστα με n στοιχεία.*

5.4 Εύρεση του πλησιέστερου ζεύγους σημείων

Θα περιγράψουμε τώρα ένα άλλο πρόβλημα που μπορεί να λυθεί με έναν αλγόριθμο της μορφής που έχουμε εξετάσει· όμως η εύρεση του σωστού τρόπου "συγχώνευσης" των λύσεων στα δύο υποπροβλήματα που παράγει ο αλγόριθμος χρειάζεται αρκετή επινοητικότητα.

Το πρόβλημα

Το πρόβλημα που εξετάζουμε είναι πολύ απλό στη διατύπωσή του: Αν δοθούν n σημεία στο επίπεδο, να βρεθεί το ζεύγος που είναι πιο κοντινό.

Το πρόβλημα αυτό εξετάστηκε από τους M. I. Shamos και D. Hoey στις αρχές της δεκαετίας του 1970, ως τμήμα της εργασίας τους για τη μελέτη αποδοτικών αλγορίθμων για βασικούς θεμελιώδεις υπολογισμούς στη γεωμετρία. Αυτοί οι αλγόριθμοι σχημάτισαν τις θεμελιώσεις του τότε αναπτυσσόμενου πεδίου της *υπολογιστικής γεωμετρίας* (computational geometry), και βρήκαν εφαρμογές σε τομείς όπως τα γραφικά, η όραση μέσω υπολογιστή, τα συστήματα γεωγραφικών πληροφοριών, και η μοντελοποίηση μορίων. Αν και το πρόβλημα του πλησιέστερου ζεύγους είναι από τα πιο φυσικά αλγοριθμικά προβλήματα της γεωμετρίας, είναι εντυπωσιακά δύσκολο να βρούμε έναν αποδοτικό αλγόριθμο για αυτό. Είναι αμέσως προφανές ότι υπάρχει μια λύση $O(n^2)$ — υπολογισμός της απόστασης ανάμεσα σε κάθε ζεύγος σημείων και λήψη της ελάχιστης τιμής — και έτσι οι Shamos και Hoey αναρωτήθηκαν αν μπορεί να βρεθεί αλγόριθμος που είναι ασυμπτωτικά ταχύτερος από τον τετραγωνικό. Χρειάστηκε πολύς χρόνος πριν βρουν την απάντηση σε αυτό το ερώτημα, και ο αλγόριθμος χρόνου $O(n \log n)$ που δίνουμε στη συνέχεια είναι ουσιαστικά αυτός που ανακάλυψαν. Στην πραγματικότητα, όταν επιστρέψουμε σε αυτό το πρόβλημα στο Κεφάλαιο 13, θα δούμε ότι είναι δυνατή περαιτέρω βελτίωση του χρόνου εκτέλεσης σε $O(n)$, με χρήση τυχαιοποιημένων μεθόδων.

Σχεδιασμός του αλγορίθμου

Ξεκινάμε με λίγο συμβολισμό. Ας συμβολίσουμε το σύνολο των σημείων με $P = \{p_1, \ldots, p_n\}$, όπου το p_i έχει συντεταγμένες (x_i, y_i)· για δύο σημεία $p_i, p_j \in P$ χρησιμοποιούμε το $d(p_i, p_j)$ για να συμβολίσουμε την τυπική Ευκλείδεια απόστασή τους. Σκοπός μας είναι να βρούμε ένα ζεύγος σημείων p_i, p_j που να ελαχιστοποιεί την απόσταση $d(p_i, p_j)$.

Θα υποθέσουμε ότι δεν υπάρχουν δύο σημεία στο P που να έχουν την ίδια συνιστώσα x ή την ίδια συνιστώσα y. Αυτό θα κάνει πιο ξεκάθαρη την ανάλυση· και είναι εύκολο να εξαλείψουμε αυτή την υπόθεση, είτε με αρχική εφαρμογή περιστροφής στα σημεία έτσι ώστε να ισχύει η υπόθεση, ή με μικρή επέκταση του αλγορίθμου που αναπτύσσουμε εδώ.

? αν έχουν ⇓ απλά ελέγχω διαφορά στην άλλη

Είναι χρήσιμο να εξετάσουμε για λίγο τη μονοδιάστατη εκδοχή αυτού του προβλήματος, αφού είναι πολύ απλούστερο και οι αντιθέσεις είναι αποκαλυπτικές. Πώς θα μπορούσαμε να βρούμε το πλησιέστερο ζεύγος σημείων που ανήκουν σε μία γραμμή; Πρώτα θα τα ταξινομούσαμε, σε χρόνο $O(n \log n)$, και μετά θα διασχίζαμε την ταξινομημένη λίστα για να υπολογίσουμε την απόσταση κάθε σημείου από το επόμενό του. Είναι εύκολο να δούμε ότι μία από αυτές οι αποστάσεις πρέπει να είναι η ελάχιστη απόσταση που ζητάμε.

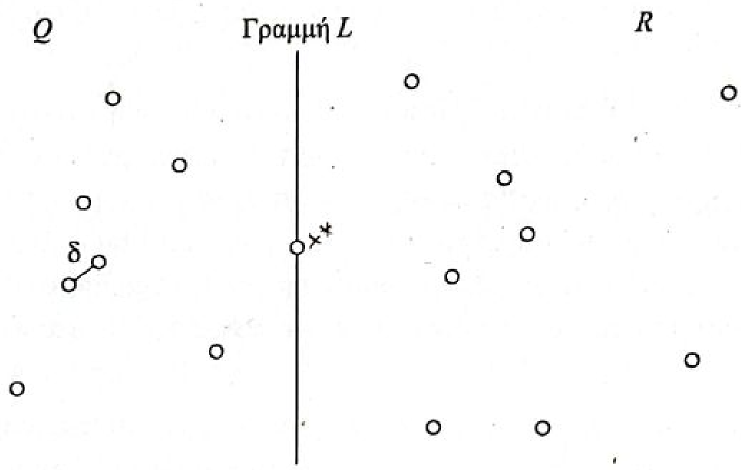
Στις δύο διαστάσεις θα μπορούσαμε να δοκιμάσουμε την ταξινόμηση των σημείων κατά τη συντεταγμένη y (ή τη συντεταγμένη x) και να ελπίζουμε ότι τα δύο πλησιέστερα σημεία θα ήταν το ένα κοντά στο άλλο σε αυτή την ταξινομημένη λίστα. Όμως είναι εύκολο να κατασκευάσουμε παραδείγματα στα οποία τα σημεία είναι πολύ μακριά, και αυτό μας εμποδίζει να προσαρμόσουμε την προσέγγιση που ακολουθήσαμε στην περίπτωση της μίας διάστασης.

Έτσι το σχέδιό μας είναι να εφαρμόσουμε το στυλ "διαίρει και βασίλευε" που χρησιμοποιήθηκε στον αλγόριθμο Mergesort: βρίσκουμε το πλησιέστερο ζευγάρι μεταξύ των σημείων στο "αριστερό μισό" του P και το πλησιέστερο ζευγάρι μεταξύ των σημείων στο "δεξιό μισό" του P· και μετά χρησιμοποιούμε αυτές τις πληροφορίες για να πάρουμε τη συνολική λύση σε γραμμικό χρόνο. Αν αναπτύξουμε έναν αλγόριθμο με αυτή τη δομή, τότε η λύση της βασικής μας αναδρομής, με βάση την (5.1), θα μας δώσει χρόνο εκτέλεσης $O(n \log n)$.

Η τελευταία "συνδυαστική" φάση του αλγορίθμου είναι εκείνη που δημιουργεί περιπλοκές: οι αποστάσεις που δεν έχουν εξεταστεί από καμία από τις αναδρομικές μας κλήσεις είναι ακριβώς εκείνες που αφορούν ένα σημείο στο αριστερό μισό και ένα σημείο στο δεξιό μισό· υπάρχουν $\Omega(n^2)$ τέτοιες αποστάσεις, όμως πρέπει να βρούμε τη μικρότερη από αυτές σε χρόνο $O(n)$ μετά την επιστροφή των αναδρομικών κλήσεων. Αν μπορέσουμε να το κάνουμε αυτό, η λύση μας θα είναι πλήρης: θα είναι η μικρότερη από τις τιμές που υπολογίστηκαν στις αναδρομικές κλήσεις και την ελάχιστη απόσταση σημείων "από αριστερά προς τα δεξιά".

Διαμόρφωση της αναδρομής Καταρχήν, ας βγάλουμε από τη μέση κάποια εύκολα πράγματα. Θα ήταν πολύ χρήσιμο αν κάθε αναδρομική κλήση, για ένα σύνολο $P' \subseteq P$, ξεκινά με δύο λίστες: μία λίστα P_x στην οποία όλα τα σημεία του P' έχουν ταξινομηθεί κατά αύξουσα τιμή της συντεταγμένης x, και μία λίστα P_y στην οποία όλα τα σημεία P' έχουν ταξινομηθεί κατά αύξουσα τιμή της συντεταγμένης y. Μπορούμε να εξασφαλίσουμε ότι αυτό θα παραμένει αληθές σε όλη τη διάρκεια του αλγορίθμου με τον ακόλουθο τρόπο.

Πρώτον, πριν ξεκινήσει οποιαδήποτε αναδρομή, ταξινομούμε όλα τα σημεία του P κατά τη συντεταγμένη x και μετά κατά τη συντεταγμένη y, παράγοντας τις λίστες P_x και P_y. Σε κάθε στοιχείο κάθε λίστας προσαρτάμε μία εγγραφή με τη θέση του σημείου και στις δύο λίστες.



Εικόνα 5.6 Το πρώτο επίπεδο της αναδρομής: Το σύνολο των σημείων P χωρίζεται εξίσου στα τμήματα Q και R από τη γραμμή L, και σε κάθε πλευρά βρίσκουμε αναδρομικά το πλησιέστερο ζευγάρι σημείων.

Στο πρώτο επίπεδο αναδρομής οι υπολογισμοί θα γίνουν ως εξής, και στα υπόλοιπα επίπεδα θα προχωρήσουμε με εντελώς ανάλογο τρόπο. Ορίζουμε το Q ως το σύνολο των σημείων στις πρώτες $\lceil n/2 \rceil$ θέσεις της λίστας P_x (το "αριστερό μισό") και το R ως το σύνολο των σημείων στις τελευταίες $\lfloor n/2 \rfloor$ θέσεις της λίστας P_x (το "δεξιό μισό"). Δείτε σχετικά την Εικόνα 5.6. Με μία μόνο διέλευση μέσω των P_x και P_y, σε χρόνο $O(n)$, μπορούμε να δημιουργήσουμε τις ακόλουθες τέσσερις λίστες: Q_x, αποτελούμενη από τα σημεία του Q ταξινομημένα κατά αύξουσα τιμή της συντεταγμένης x· Q_y, αποτελούμενη από τα σημεία του Q ταξινομημένα κατά αύξουσα τιμή της συνιστώσας y· και τις ανάλογες λίστες R_x και R_y. Για κάθε στοιχείο σε αυτές τις λίστες, όπως και προηγουμένως, καταγράφουμε τη θέση του σημείου και στις δύο λίστες όπου ανήκει.

Τώρα προσδιορίζουμε αναδρομικά ένα πλησιέστερο ζεύγος σημείων στο Q (με προσπέλαση στις λίστες Q_x και Q_y). Υποθέστε ότι τα q_0^* και q_1^* έχουν επιστραφεί (σωστά) ως το πλησιέστερο ζεύγος σημείων στο Q. Παρομοίως, προσδιορίζουμε ένα πλησιέστερο ζεύγος σημείων στο R, που είναι τα σημεία r_0^* και r_1^*.

Συνδυασμός των λύσεων Ο γενικός μηχανισμός της τεχνικής "διαίρει και βασίλευε" μας έχει φέρει μέχρι εδώ, χωρίς να έχουμε πραγματικά εντρυφήσει στη δομή του προβλήματος εύρεσης του πλησιέστερου ζεύγους σημείων. Όμως παραμένει ακόμα το πρόβλημα που είδαμε να προβάλλεται από την αρχή: Πώς θα χρησιμοποιήσουμε τις λύσεις των δύο υποπροβλημάτων ως τμήμα μιας λειτουργίας "συνδυασμού" γραμμικού χρόνου;

Έστω δ η ελάχιστη τιμή των $d(q_0^*, q_1^*)$ και $d(r_0^*, r_1^*)$. Το πραγματικό ερώτημα είναι: Υπάρχουν σημεία $q \in Q$ και $r \in R$ για τα οποία $d(q, r) < \delta$; Αν δεν υπάρχουν, τότε έχουμε ήδη βρει το πλησιέστερο ζευγάρι σε μία από τις αναδρομικές μας κλήσεις. Αν όμως υπάρχουν, τότε τα πλησιέστερα αυτά σημεία q και r σχηματίζουν το πλησιέστερο ζεύγος στο P.

Έστω ότι το x^* συμβολίζει τη συντεταγμένη x του δεξιότερου σημείου στο Q, και έστω ότι το L συμβολίζει την κάθετη γραμμή που περιγράφεται από την εξίσωση $x = x^*$. Αυτή η γραμμή L "διαχωρίζει" το Q από το R. Ακολουθεί ένας απλός ισχυρισμός.

L → δεν χωρίζει στη μέση των συντεταγμένων, αλλά μισά μισά σημεία

(5.8) *Αν υπάρχει $q \in Q$ και $r \in R$ για το οποίο $d(q, r) < \delta$, τότε και το q και το r βρίσκονται σε απόσταση δ από τη γραμμή L.*

Απόδειξη. Υποθέστε ότι υπάρχουν τέτοια σημεία q και r· γράφουμε $q = (q_x, q_y)$ και $r = (r_x, r_y)$. Από τον ορισμό του x^*, γνωρίζουμε ότι $q_x \le x^* \le r_x$. Έτσι έχουμε

$$x^* - q_x \le r_x - q_x \le d(q, r) < \delta$$

και

$$r_x - x^* \le r_x - q_x \le d(q, r) < \delta,$$

οπότε και το q και το r έχουν μια συντεταγμένη x μέσα σε απόσταση δ από το x^*, και επομένως βρίσκονται σε απόσταση δ από τη γραμμή L. ■

Έτσι, αν θέλουμε να βρούμε ένα πλησιέστερο ζευγάρι q και r, μπορούμε να περιορίσουμε την αναζήτησή μας στη στενή ζώνη που αποτελείται μόνο από σημεία του P σε απόσταση δ από τη γραμμή L. Έστω ότι αυτό το σύνολο συμβολίζεται ως $S \subseteq P$, και έστω ότι S_y συμβολίζει τη λίστα που αποτελείται από τα σημεία του S ταξινομημένα κατά αύξουσα τιμή της συνιστώσας y. Με απλή διέλευση μέσω της λίστας P_y μπορούμε να κατασκευάσουμε τη λίστα S_y σε χρόνο $O(n)$.

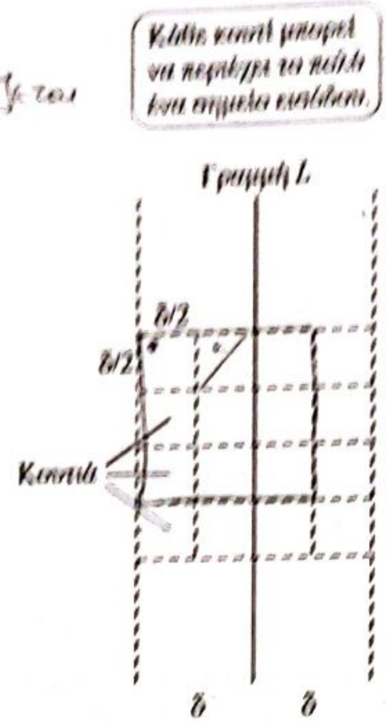
Μπορούμε να επαναδιατυπώσουμε την (5.8) σε συνάρτηση με το σύνολο S με τον ακόλουθο τρόπο.

(5.9) *Υπάρχουν $q \in Q$ και $r \in R$ για τα οποία $d(q, r) < \delta$ αν και μόνο αν υπάρχουν $s, s' \in S$ για τα οποία $d(s, s') < \delta$.*

$(s, s') \neq (q_0^*, q_1^*)$
(r_0^*, r_1^*)
γιατί θέλω $< \delta$

Αξίζει να παρατηρήσουμε σε αυτό το σημείο ότι το S μπορεί στην πραγματικότητα να είναι ολόκληρο το σύνολο P, οπότε σε αυτή την περίπτωση οι (5.8) και (5.9) δεν φαίνεται να μας προσφέρουν κάτι σημαντικό. Όμως αυτό απέχει πολύ από την πραγματικότητα, όπως δείχνει η ακόλουθη εκπληκτική πρόταση.

Επειδή το x φράσσεται → περιορίζεται ο αριθμός των προς έλεγχο σημείων



$a = \sqrt{\frac{\delta^2}{4} + \frac{\delta^2}{4}} = \frac{\delta\sqrt{2}}{2}$

$\boxed{2a > \delta}$

Εικόνα 5.7 Το τμήμα του επιπέδου κοντά στη διαχωριστική γραμμή L, όπως αναλύεται στην απόδειξη της (5.10).

κόστος $d(s,s')$?

(5.10) *Αν τα s, $s' \in S$ έχουν την ιδιότητα $d(s, s') < \delta$, τότε τα s και s' απέχουν το πολύ 15 θέσεις το ένα από το άλλο στην ταξινομημένη λίστα S'_y.*

Ουσιαστικά ... σε αυτά ... έχω ... που ξέρω ότι είναι από την ίδια πλευρά ↓ δεν είναι κοντά αυτοί έχουν ...

Απόδειξη. Θεωρήστε το υποσύνολο Z του επιπέδου που αποτελείται από όλα τα σημεία τα οποία απέχουν το πολύ απόσταση δ από τη γραμμή L. Χωρίζουμε το Z σε κουτιά: τετράγωνα με οριζόντιες και κατακόρυφες πλευρές μήκους δ/2. Μία σειρά του Z θα αποτελείται από τέσσερα κουτιά των οποίων οι οριζόντιες πλευρές έχουν τις ίδιες συντεταγμένες y. Αυτή η συλλογή κουτιών φαίνεται στην Εικόνα 5.7.

Υποθέστε ότι δύο σημεία του S βρίσκονται στο ίδιο κουτί. Επειδή όλα τα σημεία του κουτιού βρίσκονται στην ίδια πλευρά της L, αυτά τα δύο σημεία είτε θα ανήκουν στο Q, είτε θα ανήκουν στο R. Όμως δύο τυχαία σημεία του ίδιου κουτιού βρίσκονται σε απόσταση $\delta \cdot \sqrt{2}/2 < \delta$, κάτι που αντιφάσκει με τον ορισμό του δ ως ελάχιστης απόστασης ανάμεσα σε οποιοδήποτε ζευγάρι σημείων του Q ή του R. Κατά συνέπεια, κάθε κουτί περιέχει το πολύ ένα σημείο του S.

Υποθέστε τώρα ότι τα σημεία s, $s' \in S$ έχουν την ιδιότητα $d(s, s') < \delta$ και απέχουν τουλάχιστον 16 θέσεις στο S_y. Υποθέστε, χωρίς απώλεια της γενικότητας, ότι το s έχει τη μικρότερη συντεταγμένη y. Τότε, επειδή μπορεί να υπάρχει μόνο ένα σημείο σε κάθε κουτί, υπάρχουν τουλάχιστον τρεις σειρές του Z που βρίσκονται μεταξύ s και s'. Όμως δύο τυχαία σημεία του Z που χωρίζονται από τρεις τουλάχιστον σειρές θα πρέπει να έχουν απόσταση τουλάχιστον 3δ/2 — άτοπο. ■

Σημειώστε ότι η τιμή 15 μπορεί να ελαττωθεί· όμως για τους σκοπούς μας αυτή τη στιγμή, το σημαντικό είναι ότι πρόκειται για μια απόλυτη σταθερά.

Λόγω της (5.10), μπορούμε να ολοκληρώσουμε τον αλγόριθμο ως εξής. Πραγματοποιούμε μία διέλευση μέσω του S, και για κάθε $s \in S$ υπολογίζουμε την απόστασή του από τα επόμενα 15 σημεία του S. Η πρόταση (5.10) μας δείχνει ότι έτσι θα έχουμε υπολογίσει την απόσταση από κάθε ζευγάρι σημείων του S που βρίσκονται σε απόσταση μικρότερη από δ το ένα από το άλλο (αν υπάρχουν τέτοια σημεία). Μετά από αυτό μπορούμε να συγκρίνουμε τη μικρότερη από αυτές τις αποστάσεις με το δ, και μπορούμε να αναφέρουμε ένα από τα εξής δύο πράγματα: (i) το πλησιέστερο ζευγάρι σημείων στο S, αν η απόστασή τους είναι μικρότερη από δ· ή (ii) το (σωστό) συμπέρασμα ότι δεν υπάρχουν ζευγάρια σημείων στο S που να απέχουν απόσταση δ το ένα από το άλλο. Στην περίπτωση (i), αυτό το ζεύγος είναι το πλησιέστερο ζεύγος του P· στην περίπτωση (ii), το πλησιέστερο ζεύγος που βρέθηκε από τις αναδρομικές μας κλήσεις είναι το πλησιέστερο ζεύγος του P.

Παρατηρήστε την ομοιότητα ανάμεσα σε αυτή τη διαδικασία και τον αλγόριθμο που απορρίψαμε στην αρχή, ο οποίος προσπαθούσε να κάνει μία διέλευση μέσω του P με βάση τη συντεταγμένη y. Ο λόγος για τον οποίο αυτή η προσέγγιση αποδίδει τώρα οφείλεται στην πρόσθετη γνώση (την τιμή του δ) που έχουμε αποκτήσει από τις αναδρομικές κλήσεις, καθώς και στην ειδική δομή του συνόλου S.

Με αυτό ολοκληρώνεται η περιγραφή του "συνδυαστικού" μέρους του αλγορίθμου, αφού λόγω της (5.9) έχουμε τώρα προσδιορίσει αν η ελάχιστη απόσταση ανάμεσα σε ένα σημείο του Q και ένα σημείο του R είναι μικρότερη από δ, και αν συμβαίνει αυτό έχουμε βρει το πλησιέστερο ζεύγος σημείων.

Η πλήρης περιγραφή του αλγορίθμου και η απόδειξη της ορθότητάς του υπονοείται από την ανάλυση που κάναμε, όμως για λόγους πληρότητας θα τα συνοψίσουμε και τα δύο στη συνέχεια.

Σύνοψη του αλγορίθμου Ακολουθεί μία υψηλού επιπέδου περιγραφή του αλγορίθμου, όπου χρησιμοποιούμε το συμβολισμό τον οποίο αναπτύξαμε προηγουμένως.

```
Closest-Pair(P)
   Δημιούργησε τα Px και Py (χρόνος O(n logn))
   (p*0, p*1) = Closest-Pair-Rec(Px, Py)

Closest-Pair-Rec(Px, Py)
   If |P| ≤ 3 then
     Βρες το πλησιέστερο ζεύγος με μέτρηση όλων των αποστάσεων
         των ζευγαριών
    Endif

   Δημιούργησε τα Qx, Qy, Rx, Ry (χρόνος O(n))
   (q*0, q*1) = Closest-Pair-Rec(Qx, Qy) ← βρίσκει κοντινά αριστ.
   (r*0, r*1) = Closest-Pair-Rec(Rx, Ry) ← — // —  δεξ

   δ = min(d(q*0, q*1), d(r*0, r*1))
   x* = μέγιστη συντεταγμένη x ενός σημείου του συνόλου Q
   L = {(x, y): x = x*}
   S = σημεία του P με απόσταση μέχρι δ από τη γραμμή L.  διαφορές των x συντεταγμ. από x*
   Δημιούργησε το Sy (χρόνος O(n))  Διάταξη αυτών ως προς y
   For κάθε σημείο s ∈ Sy, υπολόγισε την απόσταση του s
```

τα προηγούμενα τα έχω ελέγξει ; πώς τα βρίσκω
δεξιά → επόμενο x
αριστ. → επόμενο y
στη διάταξη ως προς y είναι κοντά αυτά που θέλω να ελέγξω γιατί

ως προς τη διάταξη

```
      από τα επόμενα 15 σημεία του S
    Έστω s, s' ένα ζευγάρι που επιτυγχάνει το ελάχιστο αυτών
        των αποστάσεων
    (χρόνος O(n))

  If d(s,s') < δ then
    Return (s, s')
  Else if d(q0*, q1*) < d(r0*, r1*) then
    Return (q0*, q1*)
  Else
    Return (r0*, r1*)
  Endif
```

τα ζητούμενα

αυτό είναι σαν να ταξινομεί τη διάταξη

κοντά τα x τους

Ανάλυση του αλγορίθμου

Θα αποδείξουμε καταρχήν ότι ο αλγόριθμος παράγει μια σωστή απάντηση, χρησιμοποιώντας τις προτάσεις που έχουμε αποδείξει κατά τη διαδικασία σχεδιασμού του.

(5.11) *Ο αλγόριθμος παράγει σωστά ως έξοδο το πλησιέστερο ζεύγος σημείων του P.*

ΟΡΘΟΤΗΤΑ

Απόδειξη. Όπως έχουμε σημειώσει, όλα τα συστατικά στοιχεία της απόδειξης έχουν ήδη μελετηθεί, οπότε εδώ θα συνοψίσουμε απλώς τον τρόπο συνδυασμού τους.

Η απόδειξη της ορθότητας του αλγορίθμου γίνεται με επαγωγή ως προς το μέγεθος του P, όπου η περίπτωση $|P| \leq 3$ είναι προφανής. Για ένα δεδομένο P, το πλησιέστερο σημείο στις αναδρομικές κλήσεις υπολογίζεται σωστά λόγω της επαγωγής. Λόγω των (5.9) και (5.10), το υπόλοιπο του αλγορίθμου προσδιορίζει σωστά αν κάποιο ζεύγος σημείων στο S βρίσκεται σε απόσταση μικρότερη από δ, και αν συμβαίνει κάτι τέτοιο επιστρέφει αυτό το πλησιέστερο ζεύγος. Τώρα το πλησιέστερο ζεύγος του P είτε έχει και τα δύο στοιχεία στο Q ή στο R, ή έχει ένα στοιχείο σε κάθε σύνολο. Στην πρώτη περίπτωση η αναδρομική κλήση βρίσκει σωστά το πλησιέστερο ζεύγος· στη δεύτερη περίπτωση αυτό το ζεύγος βρίσκεται σε απόσταση μικρότερη από δ, και το υπόλοιπο τμήμα του αλγορίθμου το βρίσκει σωστά. ■

Μπορούμε τώρα να βρούμε και το όριο ως προς το χρόνο εκτέλεσης, χρησιμοποιώντας την (5.2).

(5.12) *Ο χρόνος εκτέλεσης του αλγορίθμου είναι $O(n \log n)$.*

Απόδειξη. Η αρχική ταξινόμηση του P με βάση τις συντεταγμένες x και y απαιτεί χρόνο $O(n \log n)$. Ο χρόνος εκτέλεσης του υπόλοιπου τμήματος του αλγορίθμου ικανοποιεί την αναδρομή (5.1), επομένως λόγω της (5.2) έχουμε χρόνο $O(n \log n)$. ■

5.5 Πολλαπλασιασμός ακεραίων

Θα εξετάσουμε τώρα μια διαφορετική εφαρμογή τής τεχνικής "διαίρει και βασίλευε", στην οποία ο "προεπιλεγμένος" τετραγωνικός αλγόριθμος βελτιώνεται μέσω μιας διαφορετικής αναδρομής. Κατά την ανάλυση του ταχύτερου αλγορίθμου θα εκμεταλλευ-