Γεωμετρικοί Αλγόριθμοι

(CLR, κεφάλαιο 35)

Στην ενότητα αυτή θα μελετηθούν τα εξής θέματα:

Γινόμενα σημείων, τομή ευθυγράμμων τμημάτων

Εύρεση κυρτών περιβλημάτων: Ο αλγόριθμος του Graham και ο αλγόριθμος του Jarvis

Το πρόβλημα "Closest Pair"

Σημεία και ευθύγραμμα τμήματα

- Έστω σημεία

$$p_1 = (x_1, y_1), \qquad p_2 = (x_2, y_2)$$

- Συμβολίζουμε με

$$\overline{p_1 p_2}$$

το *ευθύγραμμο τμήμα* που ενώνει τα δύο σημεία. Βλέπουμε ότι αυτό περιέχει όλα τα σημεία

$$p_3 = (ax_1 + (1-a)x_2, ay_1 + (1-a)y_2)$$

$0 \leq a \leq 1$.

- Συχνά η *κατεύθυνση* ενός ευθύγραμμου τμήματος έχει σημασία, οπότε γράφουμε

$$\overrightarrow{p_1 p_2} \qquad \overrightarrow{p_2 p_1}$$

Γινόμενα σημείων

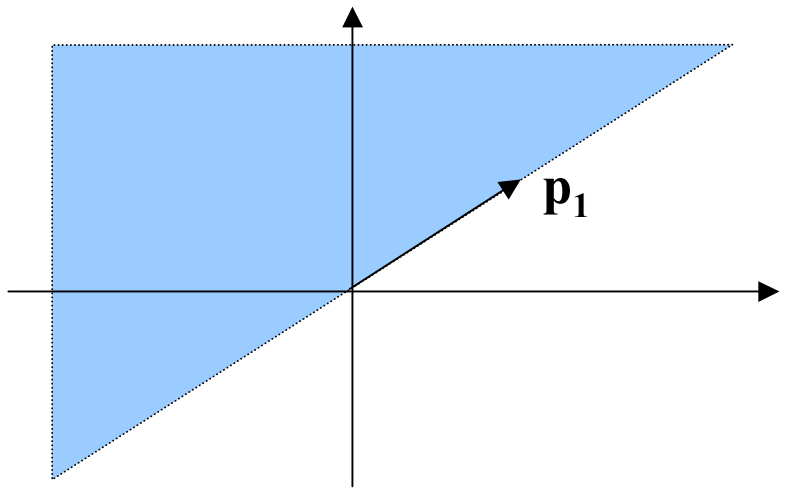
ΟΡΙΣΜΟΣ

Έστω σημεία $p_1 =(x_1, y_1)$ και $p_2=(x_2, y_2)$. Το *γινόμενο* των δύο σημείων $p_1 \times p_2$ ορίζεται ως η ορίζουσα ενός πίνακα:

$$p_1 \times p_2 = \det\begin{pmatrix} x_1 & x_2 \\ y_1 & y_2 \end{pmatrix} = x_1 y_2 - x_2 y_1$$

Έχουμε ότι:

1. αν $p_1 \times p_2 > 0$, τότε το σημείο p_1 βρίσκεται σε δεξιόστροφη κατεύθυνση από το σημείο p_2, ως προς το σημείο (0,0)
2. αν $p_1 \times p_2 < 0$, τότε το σημείο p_1 βρίσκεται σε αριστερόστροφη κατεύθυνση από το σημείο p_2, ως προς το σημείο (0,0)
3. αν $p_1 \times p_2 = 0$, τότε το σημείο p_1 βρίσκεται πάνω στη γραμμή που περιέχει τα σημεία p_2 και (0,0).



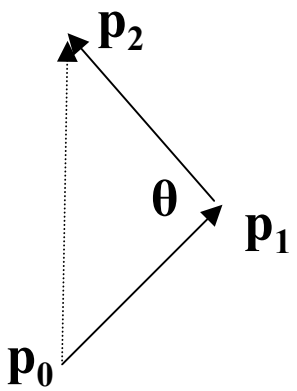
Γινόμενα σημείων

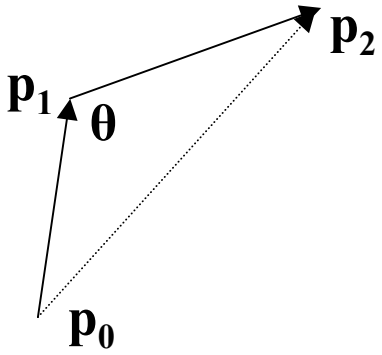
- Για να αποφασίσουμε σε ποια κατεύθυνση του ευθύγραμμου τμήματος p_0p_1 βρίσκεται το ευθύγραμμο τμήμα p_0p_2 μεταφέρουμε το σημείο p_0 στο (0,0) και υπολογίζουμε το γινόμενο

$$(p_1 - p_0)\times(p_2 - p_0)$$

Πρόβλημα

- Έστω δύο ευθύγραμμα τμήματα p_0p_1 και p_1p_2. Η γωνία θ που σχηματίζεται στο σημείο p_1 στρίβει προς τα δεξιά ή προς τα αριστερά;





Γινόμενα σημείων

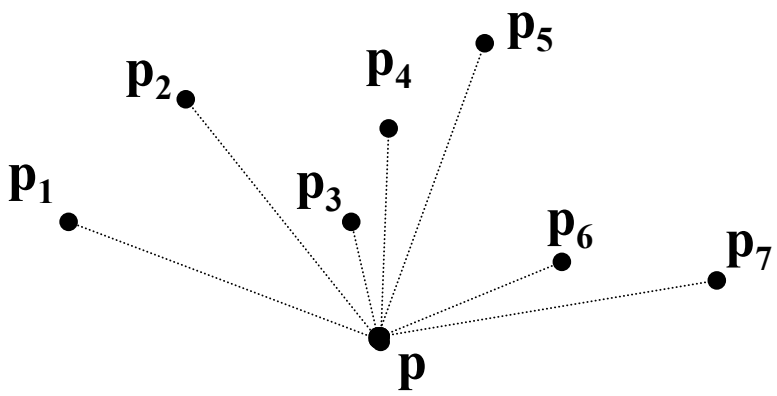
- Το ερώτημα αυτό είναι ισοδύναμο με το ερώτημα: σε ποια κατεύθυνση του p_0p_1 βρίσκεται το p_0p_2.

- Αν βρίσκεται σε δεξιόστροφη κατεύθυνση τότε η θ στρίβει προς τα δεξιά, και αν βρίσκεται σε αριστερόστροφη κατεύθυνση τότε η θ στρίβει προς τα αριστερά.

- Άρα μπορεί να λυθεί με τον υπολογισμό ενός γινομένου σημείων.

Άλλα προβλήματα

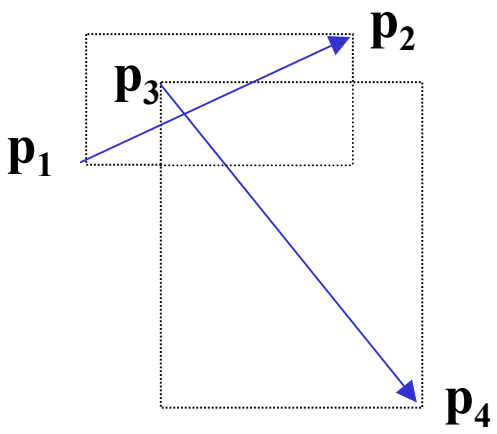
1. Εύρεση κατά πόσο δύο ευθύγραμμα τμήματα τέμνονται (σε χρόνο O(1))
2. Ταξινόμηση μιας ακολουθίας σημείων σε αύξουσα σειρά πολικής γωνίας ως προς κάποιο σημείο (σε χρόνο O(n lg n)).



- Η έννοια του γινομένου σημείων είναι σημαντική γιατί μας επιτρέπει να αποφύγουμε την πράξη της διαίρεσης (στην πρώτη εφαρμογή) και τριγωνομετρικές πράξεις/υπολογισμό γωνιών (στη δεύτερη εφαρμογή). Θα μελετήσουμε την πρώτη εφαρμογή:

Τομή ευθυγράμμων τμημάτων

- *Έστω δύο ευθύγραμμα τμήματα p_1p_2 και p_3p_4. Να αποφασίσετε αν τα δύο τμήματα τέμνονται.*



Πρώτο βήμα: βρες τα μικρότερα ορθογώνια που περικλύουν το κάθε ένα από τα δύο τμήματα. Αν τα κουτιά τέμνονται προχώρησε στο δεύτερο βήμα, διαφορετικά στάματα (τα ευθύγραμμα τμήματα οπωσδήποτε δεν τέμνονται).

Τομή ευθυγράμμων τμημάτων

Υλοποίηση:

- Έστω $p_1 = (x_1, y_1)$, $p_2 = (x_2, y_2)$, $p_3 = (x_3, y_3)$, $p_4 = (x_4, y_4)$.

- Το μικρότερο κουτί που περικλείει το p_1p_2 έχει την αριστερή και κάτω γωνιά του στο σημείο $q_1 = (\min(x_1, x_2), \min(y_1, y_2))$ και τη δεξιά και πάνω γωνιά του στο σημείο $q_2 = (\max(x_1, x_2), \max(y_1, y_2))$.

- Το μικρότερο κουτί που περικλείει το p_3p_4 έχει την αριστερή και κάτω γωνιά του στο σημείο $r_1 = (\min(x_3, x_4), \min(y_3, y_4))$ και τη δεξιά και πάνω γωνιά του στο σημείο $r_2 = (\max(x_3, x_4), \max(y_3, y_4))$.

- Τα κουτιά τέμνονται αν

$$\max(x_1, x_2) \geq \min(x_3, x_4) \quad \text{και} \quad \max(x_3, x_4) \geq \min(x_1, x_2)$$

$$\max(y_1, y_2) \geq \min(y_3, y_4) \quad \text{και} \quad \max(y_3, y_4) \geq \min(y_1, y_2)$$

Τομή ευθυγράμμων τμημάτων

Δεύτερο βήμα: Ελέγχουμε

(α) που βρίσκονται τα σημεία p_1 και p_2 σε σχέση με την p_3p_4 και

(β) που βρίσκονται τα σημεία p_3 και p_4 σε σχέση με την p_1p_2

(χρησιμοποιώντας γινόμενα σημείων).

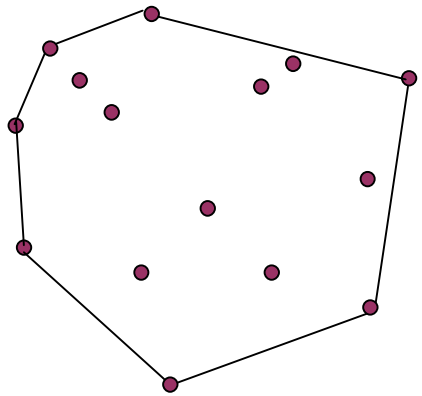
- Αν τα σημεία p_1 και p_2 βρίσκονται είτε σε αντίθετη πλευρά του ευθύγραμμου τμήματος p_3p_4 είτε πάνω στη γραμμή που ορίζεται από το ευθύγραμμο τμήμα, και το ίδιο ισχύει για το ζεύγος p_3 και p_4 και το p_1p_2, τότε οι ευθείες τέμνονται.

- Πόσες διαφορετικές περιπτώσεις υπάρχουν;

- Είναι αναγκαίο το βήμα 1;

Εύρεση Κυρτού Περιβλήματος

- *Κυρτό Περίβλημα* ενός συνόλου S από n σημεία είναι το ελάχιστο κυρτό πολύγωνο P τέτοιο ώστε κάθε ένα από τα n σημεία στο S βρίσκεται πάνω στο σύνορο του S ή στο εσωτερικό του S.
- Συμβολίζεται ως CH(S).

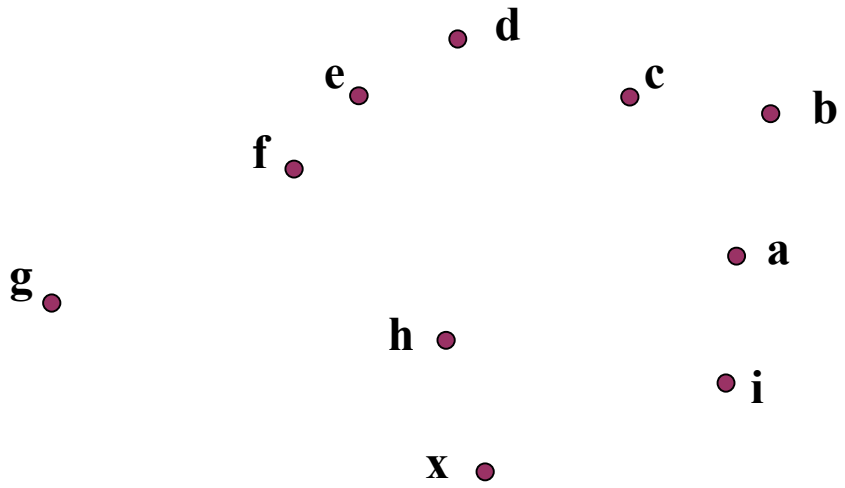


- Στον ορισμό, ελάχιστο κυρτό πολύγωνο P σημαίνει ότι δεν υπάρχει κυρτό πολύγωνο Q τέτοιο ώστε $S \subseteq Q \subset P$.

Ο Αλγόριθμος του Graham

- Διαλέγουμε το χαμηλότερο σημείο, x, του σημειοσυνόλου S.
- Ταξινομούμε τα σημεία ως προς την πολική γωνία που σχηματίζουν με το σημείο x (αντίστροφα προς τη φορά του ρολογιού).

- Παράδειγμα



- Με την ταξινόμηση παίρνουμε τη λίστα [i,a,b,c,d,e,h,f,g]

Ο Αλγόριθμος του Graham

- Στη συνέχεια επεξεργαζόμαστε τα σημεία καθ'αυτή τη σειρά και το κυρτό περίβλημα κατασκευάζεται βήμα προς βήμα.

- Έστω x[1], …, x[n] η ταξινομημένη λίστα.

- Σε κάθε βήμα το κυρτό περίβλημα θα είναι σωστό σύμφωνα με τα σημεία που έχουμε δει μέχρι στιγμής, όμως είναι δυνατό, σημεία που θα δούμε αργότερα να αλλάξουν προηγούμενες αποφάσεις μας.

- Το εκάστοτε μέρος του κυρτού περιβλήματος που έχει κατασκευαστεί μέχρι κάποιο βήμα διατηρείται σε μια σωρό.

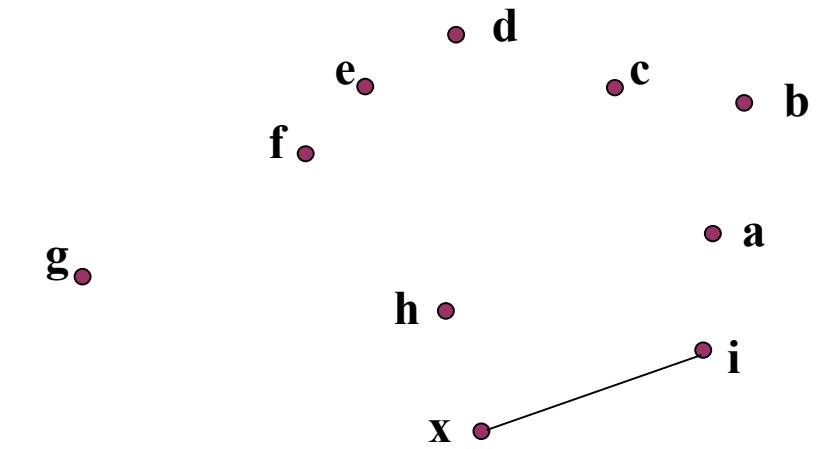
- Αρχικά η σωρός περιέχει τα τρία σημεία x, x[1], x[2].

Ο Αλγόριθμος του Graham

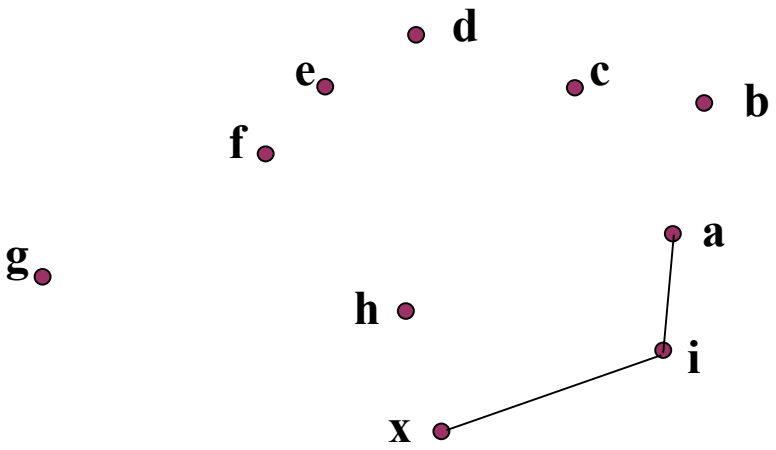
- Θεωρήστε το σημείο x[3]. Υπάρχουν δύο περιπτώσεις:
  1. αν η γωνία $\overset{\frown}{x[1]x[2]x[3]}$ στρίβει προς τα αριστερά, τότε το x[3] προστίθεται στη σωρό.
  2. αν η γωνία $\overset{\frown}{x[1]x[2]x[3]}$ στρίβει προς τα δεξιά ή είναι 180°, τότε η προηγούμενη απόφαση να εισάγουμε το x[2] στη σωρό αναιρείται: το x[2] αφαιρείται από τη σωρό και το x[3] προστίθεται στη σωρό.

- Θεωρώντας το σημείο x[i] οι περιπτώσεις είναι παρόμοιες. Έστω η σωρός είναι η s = s΄ab.
  1. αν η γωνία $\overset{\frown}{abx[i]}$ στρίβει προς τα αριστερά, τότε το x[i] προστίθεται στη σωρό,
  2. διαφορετικά, η προηγούμενη απόφαση να εισαγάγουμε το b στη σωρό αναιρείται: το b αφαιρείται από τη σωρό και επαναλαμβάνουμε το βήμα (προσπαθώντας να εισάγουμε το x[i] στη σωρό s΄a).

Παράδειγμα (συνέχεια)

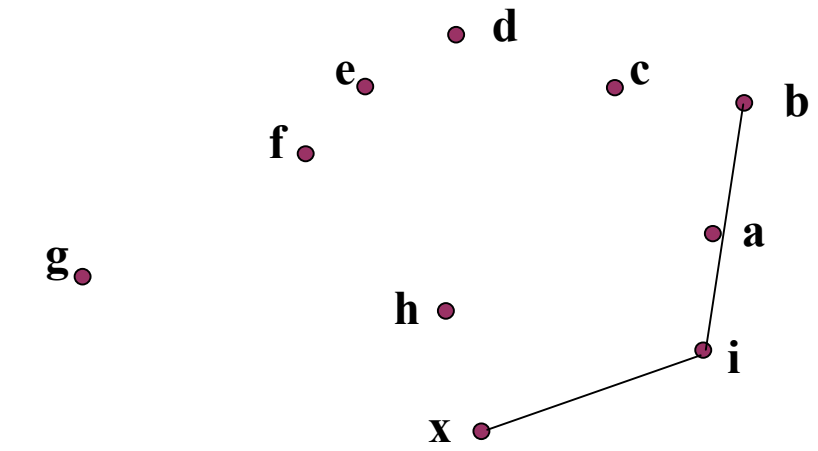


s={x,i}

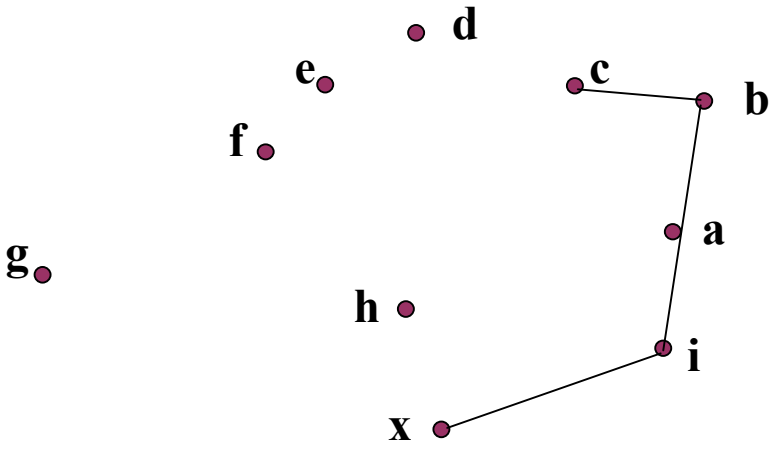


s={x,i,a}

Παράδειγμα (συνέχεια)

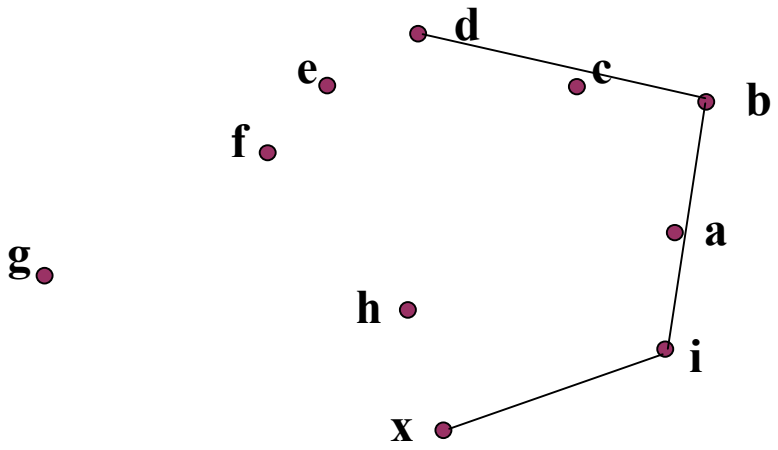


s={x,i,b}

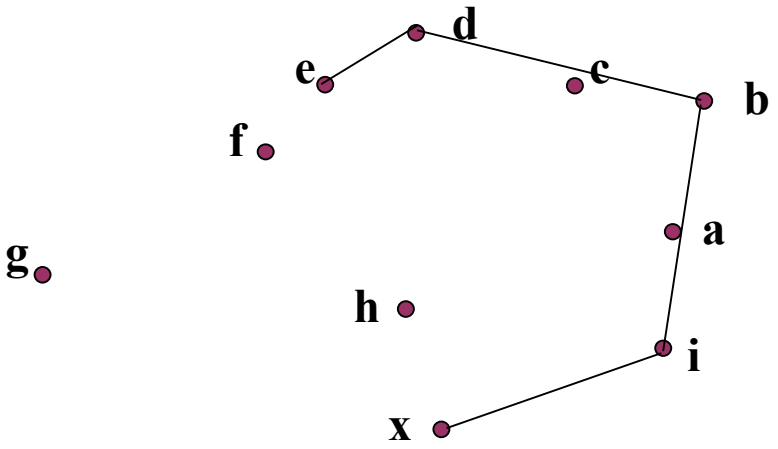


s={x,i,b,c}

Παράδειγμα (συνέχεια)

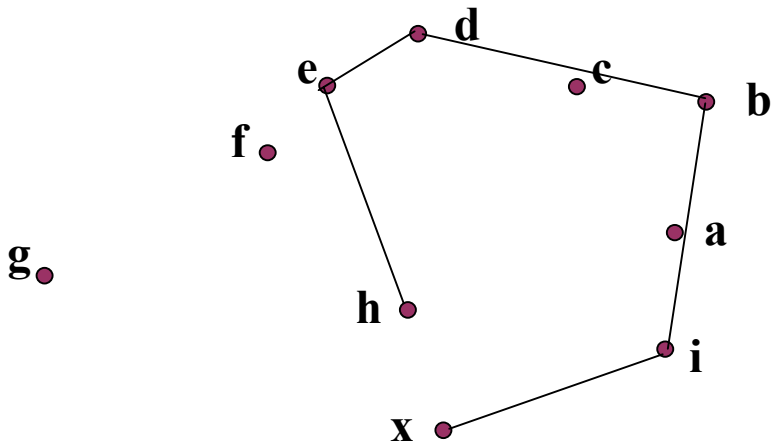


s={x,i,b,d}

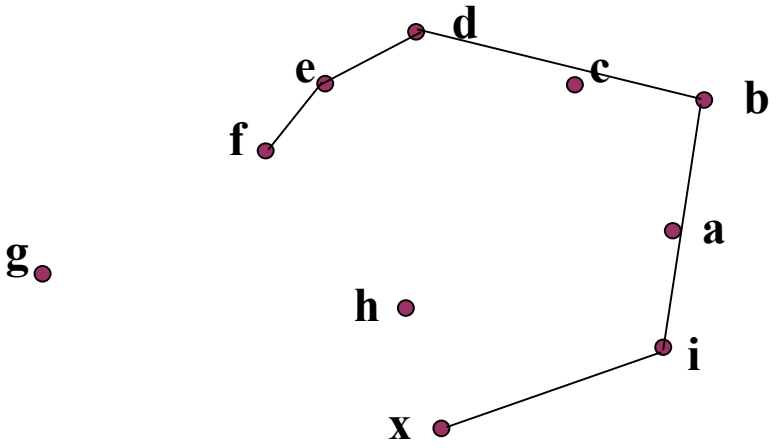


s={x,i,b,d,e}

Παράδειγμα (συνέχεια)

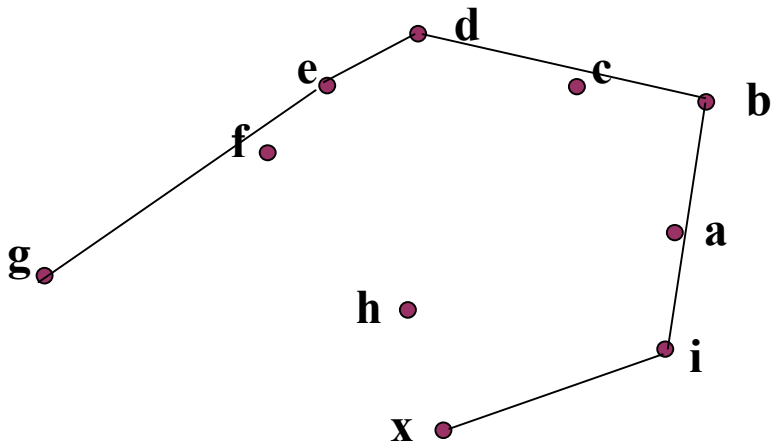


s={x,i,b,d,e,h}

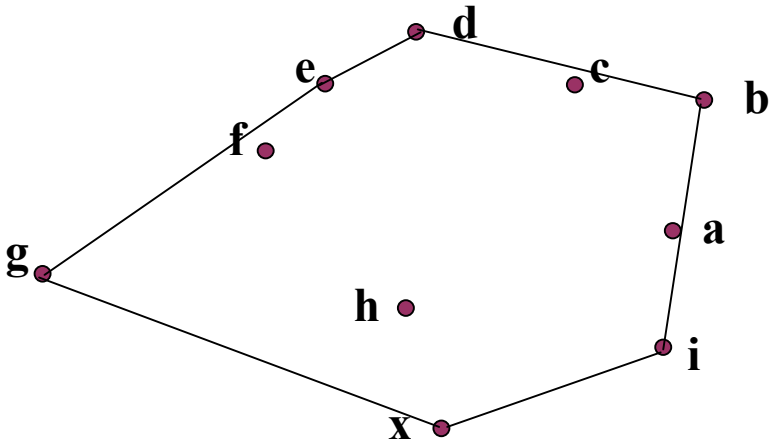


s={x,i,b,d,e,f}

Παράδειγμα (συνέχεια)



s={x,i,b,d,e,g}



s={x,i,b,d,e,g}

Ο Αλγόριθμος

```
CH(Q){

MakeEmptyStack(S);
let x be the lowest point of Q;
sort(Q-{x});
Push(x,S);
Push(Q[1],S);
Push(Q[2],S);
for (i=3;i≤n;i++)
   a=Top(S);
   b=Second(S);
   while baQ[i] does not turn left
       Pop(S);
   Push(Q[i],S);
   return S;
}
```

Χρονική Πολυπλοκότητα

Θ(n)	για την εύρεση του σημείου
Θ(n lg n)	για ταξινόμηση των σημείων ως προς τις πολικές γωνίες τους
Θ(n)	για την επεξεργασία της σωρού (κάθε σημείο μπορεί να μπει ακριβώς μια φορά στη σωρό και να βγει το πολύ μια φορά)

Συνολικά: Θ(n lg n)

Ο Αλγόριθμος του Jarvis

- Ξεκινούμε με το χαμηλότερο σημείο του συνόλου.

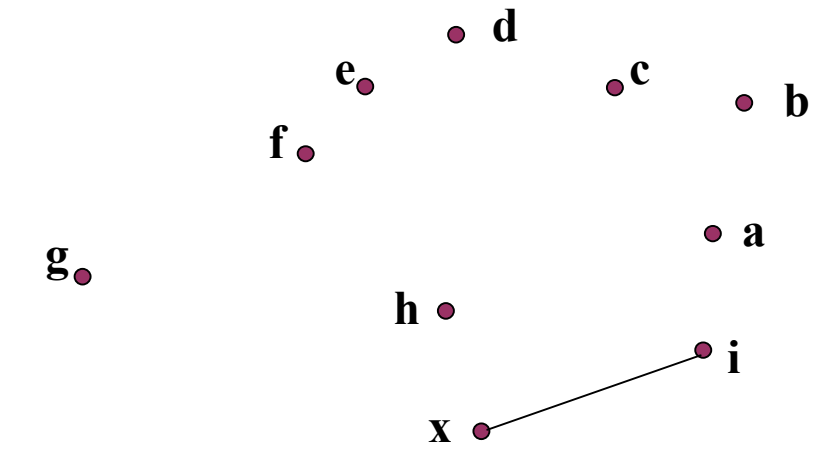
- Ο αλγόριθμος κτίζει μια λίστα p[1],…,p[k], με τα σημεία-κορυφές του κυρτού περιβλήματος ως εξής:

  1. αν p[1]=p[k] τότε σταμάτα, διαφορετικά,
  2. βρες το σημείο p, που σχηματίζει τη "μικρότερη" πολική γωνία με το p[k] και πρόσθεσε το στη λίστα του κυρτού περιβλήματος.

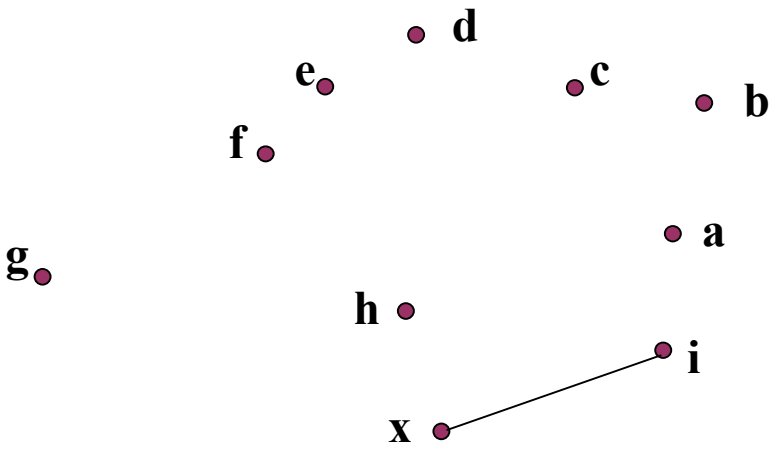
     (αν δύο σημεία σχηματίζουν την ίδια μικρότερη πολική γωνία, τότε διάλεξε το πιο μακρινό.)

- Προσοχή: κατά τη διάρκεια της εκτέλεσης του αλγόριθμου η έννοια της γωνίας πρέπει να μετριέται σε δεξιόστροφη κατεύθυνση.

Παράδειγμα

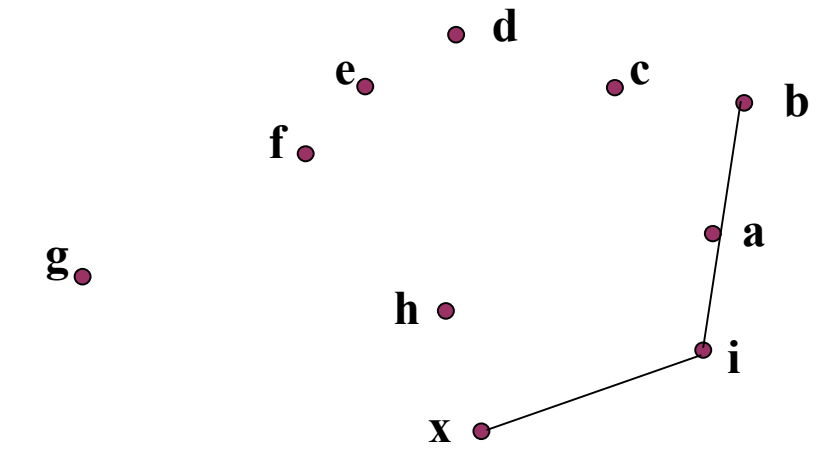


Λίστα=[x]

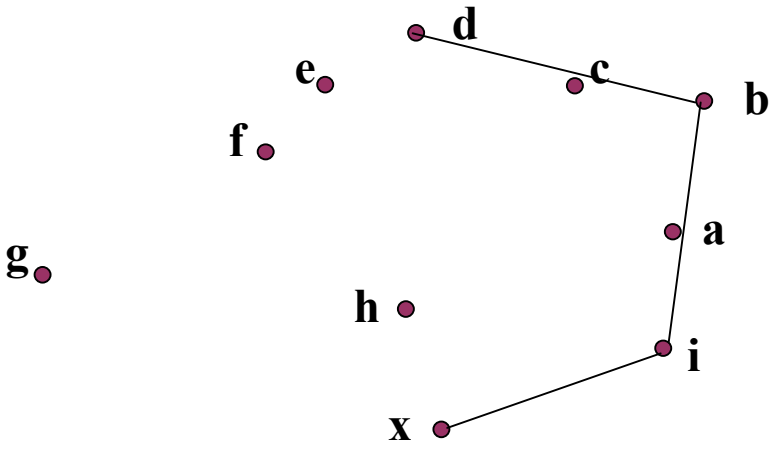


Λίστα=[x,i]

Παράδειγμα

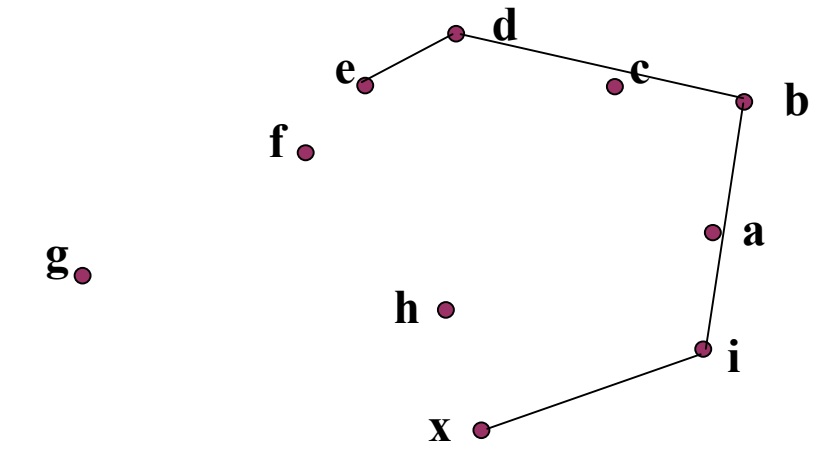


Λίστα=[x,i,b]

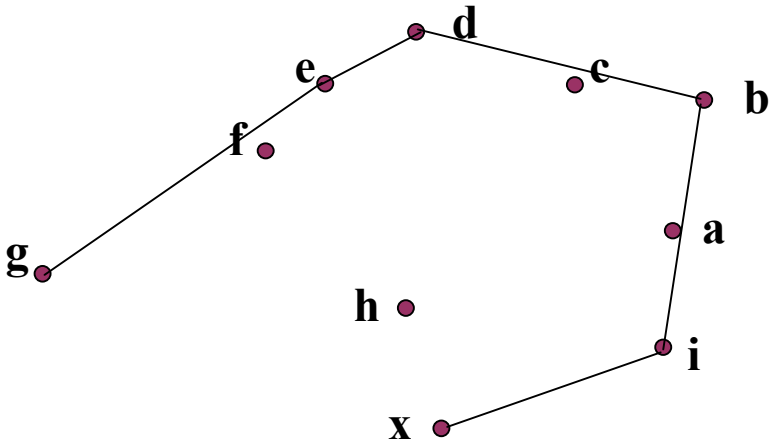


Λίστα=[x,i,b,d]

Παράδειγμα (συνέχεια)

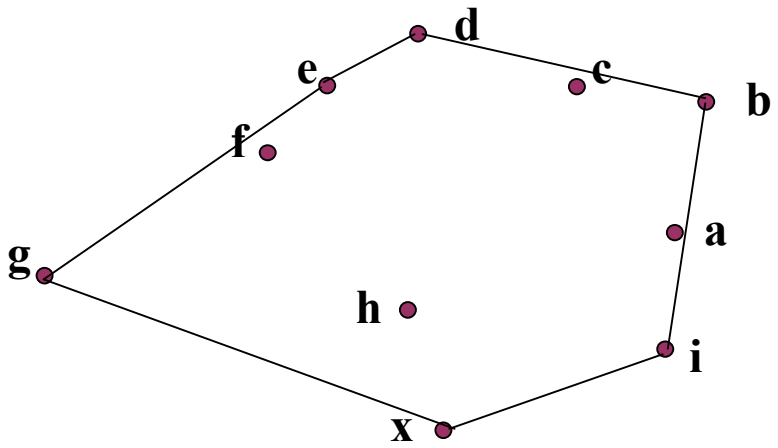


Λίστα=[x,i,b,d,e]



Λίστα={x,i,b,d,e,g}

Παράδειγμα (συνέχεια)



Λίστα=[x,i,b,d,e,g]

Χρονική Πολυπλοκότητα: O(hn)

όπου h είναι ο αριθμός των κορυφών του κυρτού περιβλήματος.

Πρόβλημα: Closest-Pair

- Δεδομένο Εισόδου: σημειοσύνολο Q με |Q|= n.
- Η *απόσταση* μεταξύ δύο σημείων a=(x1, y1) και b=(x2,y2) είναι

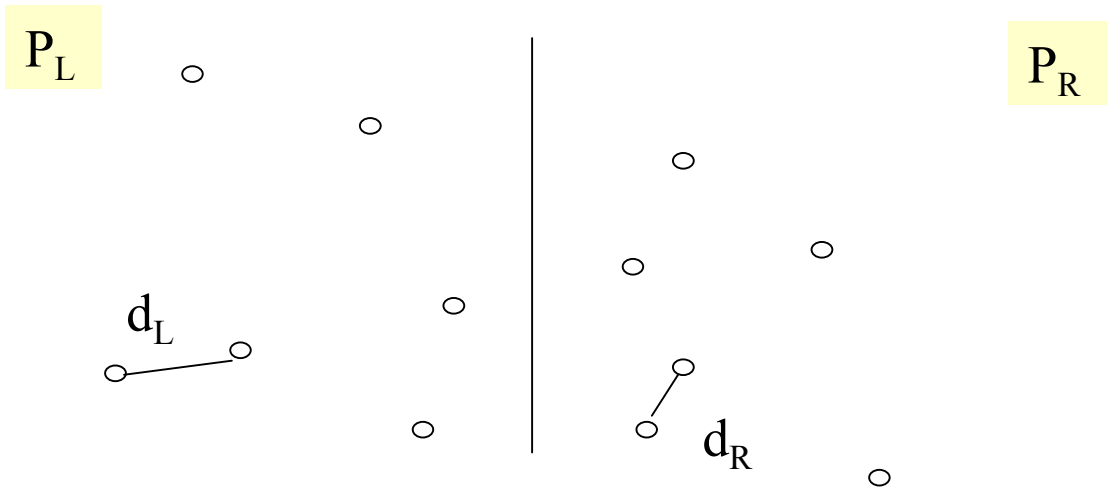
$$\text{distance } (a,b) = \sqrt{(x_1-x_2)^2+(y_1-y_2)^2}$$

- Αν δύο σημεία συμπίπτουν, η μεταξύ τους απόσταση είναι 0.
- Ζητούμενο: να βρεθεί το ζεύγος των σημείων με τη μικρότερη μεταξύ τους απόσταση.
- 'Φανερός΄ αλγόριθμος: υπολόγισε όλες τις αποστάσεις μεταξύ σημείων και διάλεξε τη μικρότερη.
- Χρόνος εκτέλεσης:

Αλγόριθμος Διαίρει και Βασίλευε

Φάση 'Διαίρει':

- Δοθέντος συνόλου σημείων P, αν $|P| \leq 3$ τότε βρες τη μικρότερη απόσταση μεταξύ των σημείων, υπολογίζοντας τις 3 διαφορετικές αποστάσεις, διαφορετικά...
- Δημιουργησε δύο μισά P_L και P_R, τέτοια ώστε τα σημεία του P_L να έχουν μικρότερες x-συνιστώσες από τα σημεία του P_R.
- Αν υπάρχει διατηρημένη ταξινομημένη λίστα του P ως προς την x-συνιστώσα, τότε το πιο πάνω βήμα απαιτεί χρόνο O(n).



Αλγόριθμος Διαίρει και Βασίλευε

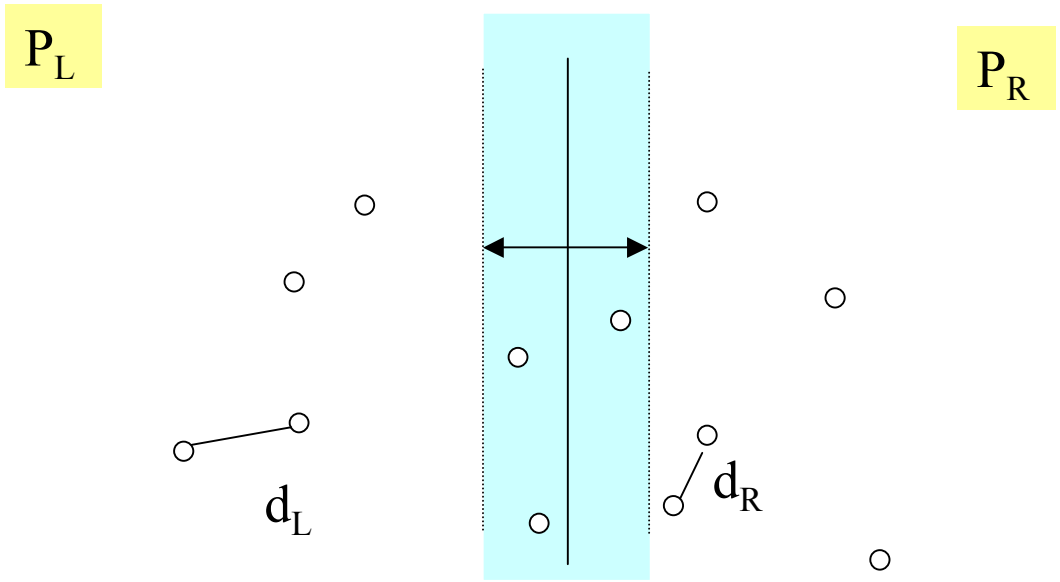
Φάση 'Κατάκτησε'

- Βρες τις μικρότερες αποστάσεις d_L και d_R με αναδρομικές κλήσεις στα σύνολα P_L και P_R.
- Θέσε $d=\min(d_L, d_R)$

Φάση 'Συνδύασε'

- Υπάρχει ζεύγος σημείων (a,b), $a \in P_L$ και $b \in P_R$ με distance(a,b)<d;
- Είναι αρκετό να κοιτάξουμε μόνο τα σημεία σε οριζόντια απόσταση < d από την κάθετη γραμμή που χωρίζει τα σύνολα P_L και P_R. Ονομάζουμε αυτό τον τομέα L.

Αλγόριθμος Διαίρει και Βασίλευε

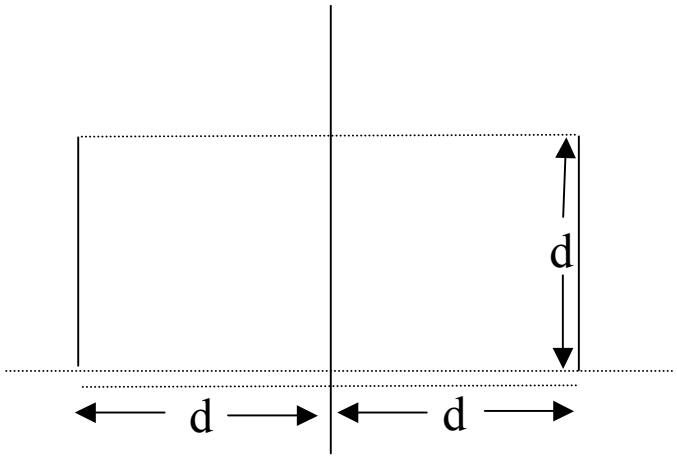


- Αν για κάθε σημείο στον τομέα L μετρήσουμε την απόσταση του από κάθε άλλο σημείο στο L, τότε η διαδικασία απαιτεί χρόνο O(n) στη μέση περίπτωση αλλά $O(n^2)$ στη χείριστη περίπτωση.

Αλγόριθμος Διαίρει και Βασίλευε

Φάση 'Συνδύασε' 2

- Έστω T το σύνολο των σημείων με x-συνιστώσες στον τομέα L.
- Είναι αρκετό να ελέγξουμε ζευγάρια (a,b) για τα οποία οι y-συνιστώσες διαφέρουν το πολύ κατά d.
- Ένα d×d τετράγωνο έχει το πολύ 4 τέτοια σημεία, άρα για κάθε σημείο a χρειάζεται να θεωρήσουμε σταθερό αριθμό σημείων b, συγκεκριμένα 7.



Χρόνος Εκτέλεσης

- Αν το σύνολο T είναι ταξινομημένο σε αύξουσα σειρά ως προς τη συνιστώσα y, τότε επεξεργασία κάθε σημείου p γίνεται σε χρόνο O(1). Απλά κοιτάζουμε τα 7 σημεία που ακολουθούν το p μέσα στη ταξινομημένη λίστα.
- Επίσης, αν το δεδομένο εισόδου είναι ταξινομημένο σε αύξουσα σειρά ως προς τη συνιστώσα x, τότε το σύνολο T μπορεί να κτιστεί σε χρόνο O(n).
- Επομένως, ο συνολικός χρόνος εκτέλεσης της φάσης 'συνδύασε' είναι O(n).
- Ο χρόνος εκτέλεσης του αλγόριθμου δίδεται από την εξίσωση
- $T(n) = 2 \cdot T(n/2) + O(n) \in O(n \lg n)$

Χρόνος Εκτέλεσης

- Πως μπορούμε κάθε φορά που καλούμε τη διαδικασία σε σημειοσύνολο P να έχουμε στη διάθεση μας δύο πίνακες με τα σημεία του P ταξινομημένα ως προς τις συνιστώσες x και y αντίστοιχα;
- Αν σε κάθε αναδρομική κλήση χρειαζόταν να εφαρμόσουμε ταξινόμηση θα είχαμε αλγόριθμο χρονικής πολυπλοκότητας $O(n^2 \lg n)$. Μπορούμε να αποφύγουμε αυτή τη συνεχή ταξινόμηση;
- Ναι, ταξινομώντας το αρχικό σύνολο και κτίζοντας από αυτό όλους τους άλλους χρήσιμους ταξινομημένους πίνακες (σε χρόνο $O(n)$).
- Χρονική Πολυπλοκότητα του αλγόριθμου: