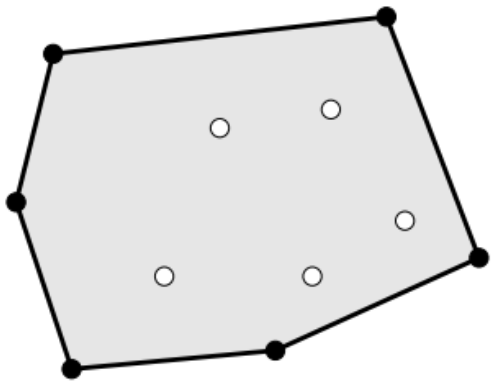
1. Hisoblash geometriyasi algoritmlari

Hisoblash geometriyasi - geometrik masalalarni yechish algoritmlari bilan shug‘ullanadigan informatika bo‘limi. Bu uchburchak, qavariq sirtlarni qurish, bitta obyektning boshqasiga tegishliligini aniqlash, ularning kesishishini topish va boshqalar kabi vazifalar bilan shug‘ullanadi, ular geometrik obyektlar bilan ishlaydi: nuqta, segment, ko‘pburchak, aylana va hokazolar.Hisoblash geometriyasi kompyuter grafikalarida, muhandislik dizaynida va boshqa ko‘plab geometriya sohalarida qo‘llaniladi.

Bizga tekislikdagi n nuqtalardan p to‘plami berilgan. Endi biz shunday eng kichik qavariq qobig’ini tortishimiz kerakki barcha nuqtalar shu shiziq ichkarisida joylashishi kerak. Bu masalan xuddi ko‘plab mix qoqilgan taxtaga o‘xshaydi. Endi biz arqon yordamida shunday mixlarni bog’lab chiqimiz kerakki minimum mixlardan o‘tilsin va barcha mixlar arqon ichida joylashsin. (67-rasm)



Rasm- 1 Qavariq qobiq

Nuqtalar va uning qavariq qobig’i. Qavariq qobiqning tepalari qora; ichki nuqtalari oq rangda berilgan.

Faqat aniqlashtirish uchun biz p dagi nuqtalarni ularning Dekart koordinatalari bo‘yicha ikkita X[1.. n] va Y [1 .. n] massivida taqdim etamiz. Biz qavariq qobig’ini soat miliga teskari tartibda aylana tarizda bog’langan tepaliklar ro‘yxati sifatida taqdim etamiz. Agar i-nuqta qavariq qobiqning tepasi bo‘lsa, keyingi[i] - keyingi

tepalikning soat miliga teskari ko‘rsatkichi va prev[i] - keyingi tepalikning soat yo‘nalishi bo‘yicha ko‘rsatkichi; aks holda keyingi[i] = prev[i] = 0. Ro‘yxatning "boshi" sifatida qaysi cho‘qqini tanlashimiz muhim emas.

Qavariq qobiq algoritmlarini taqdim etishni osonlashtirish uchun men nuqtalar umumiy holatda deb taxmin qilamiz, ya’ni (bu kontekstda) umumiy chiziqda uchta nuqta yotmaydi.

1.1 Oddiy holatlar

Bir nuqtaning qavariq qobig’ini hisoblash ahamiyatsiz, shunchaki bu nuqtani qaytaramiz. Ikki nuqtaning qavariq qobig’ini hisoblash ham ahamiyatsiz.

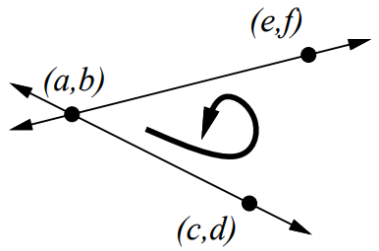
Uch nuqta uchun bizda ikki xil imkoniyat bor — yoki nuqtalar massivda soat yo‘nalishi bo‘yicha yoki soat miliga teskari tartibda keltirilgan. Faraz qilaylik, bizning uchta nuqtamiz (a, b), (c, d) va (e, f), shu tartibda berilgan va hozircha, birinchi nuqta chap tomonda, shuning uchun a < c va a < f . Keyin uch nuqta soat miliga teskari tartibda bo‘ladi, faqat chiziq bo‘lsa. (a, b)(c, d) chiziq (a, b)(e, f) dan burchak koeffitsiyentidan kam bo‘lsa.

$$soat\ miliga\ teskari = \frac{d-b}{c-a} - \frac{f-b}{e-a}$$

Ikkala maxraj ham musbat bo‘lgani uchun bu tengsizlikni quyidagicha yozishimiz mumkin:

$$soat\ miliga\ teskari = (f-b)(c-a) > (d-b)(e-a)$$

Bu oxirgi tengsizlik (a, b) o‘ta chap nuqta bo‘lmasa ham to‘g’ri. Agar tengsizlik teskari bo‘lsa, unda nuqtalar soat yo‘nalishi bo‘yicha tartibda bo‘ladi. Agar uchta nuqta kollinear bo‘lsa (esda tutingki, biz hech qachon bunday bo‘lmaydi deb o‘ylaymiz), unda ikkita ifoda tengdir.



Soat miliga teskari tartibda uchta nuqta.

Ushbu soat miliga teskari haqida fikr yuritishning yana bir usuli shundan iboratki, biz ikkita vektor (c, d) - (a, b) va (e, f) - (a, b) ning o‘zaro ko‘paytmasini hisoblayapmiz, bu 2x2 determinant sifatida aniqlaymiz:

$$soat\ miliga\ teskari \Leftrightarrow \begin{vmatrix} c-a & d-b \\ e-a & f-b \end{vmatrix} > 0$$

Biz 3x3 ta determinanti sifatida ham yozishimiz mumkin:

$$soat\ miliga\ teskari \Leftrightarrow \begin{vmatrix} 1 & a & b \\ 1 & c & d \\ 1 & e & f \end{vmatrix} > 0$$

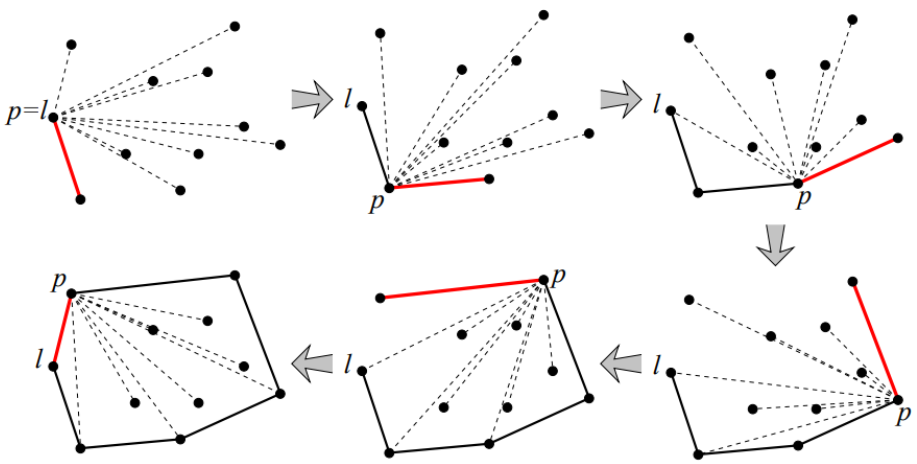
Berilgan uchta ibora ham algebraik jihatdan bir xil.

Ushbu soat miliga teskari harkat qavariq qobiq algoritmlarida saralash algoritmlarida taqqoslashlar bilan bir xil rol o‘ynaydi.

1.2 Jarvis algoritmi (sovg’ani o‘rash)

Ehtimol, qavariq qobiqlarni hisoblashning eng oddiy algoritmi shunchaki nuqta atrofida ipni o‘rash jarayoniga taqlid qiladi. Ushbu algoritm odatda Jarvis Marshi algoritmi yoki, u sovg’alarni o‘rash algoritmi deb ham ataladi.

Jarvisning yurishi eng chap nuqtani l ni hisoblash bilan boshlanadi (ya’ni x koordinatasi eng kichik bo‘lgan nuqta), chunki biz bilamizki, eng chap nuqta qavariq qobiq tepasi bo‘lishi kerak. l ni topish chiziqli vaqt murakkabligini talab etadi.



Rasm- 2 Jarvis algoritimining yurishini ijro etish

Keyin algoritm qavariq qobig'ining har bir keyingi cho'qqisini topish uchun bir qator burilish bosqichlarini bajaradi, l dan boshlab va yana l ga yetguncha davom etadi.Keyin biz l nuqtandan eng o'ng tomonga qaragan nuqtani topamiz va uni p dep belgilaymiz. Bu ketma-ketlikni soat miliga teskari yo'nalishda bajarish orqali har bir keyingi nuqtani chiziqli vaqt ichida O (n) murakkablikta topishimiz mumkin.

```
JarvisMarch(X[1..n], Y[1..n]):
  ℓ ← 1
  for i ← 2 to n
      if X[i] < X[ℓ]
          ℓ ← i
  p ← ℓ
  repeat
      q ← p + 1        ⟨⟨Make sure p ≠ q⟩⟩
      for i ← 2 to n
          if CCW(p, i, q)
              q ← i
      next[p] ← q;  prev[q] ← p
      p ← q
  until p = ℓ
```

Algoritm qavariq qobiqning har bir tepasiga O(n) vaqt sarflaganligi sababli, eng yomon ish vaqti $O(n^2)$ ga teng. Biroq, bu sodda tahlil shuni yashiradiki, agar qavariq qobiqning nuqtalari juda kam bo'lsa, Jarvis marshi juda tez ishlaydi. Ish vaqtini yozishning eng yaxshi murakkabligi bu O(n*h), bu erda h - qavariq qobiqning nuqtalari soni. Eng yomon holatda, h = n va biz eski $O(n^2)$ vaqt murakkabligini

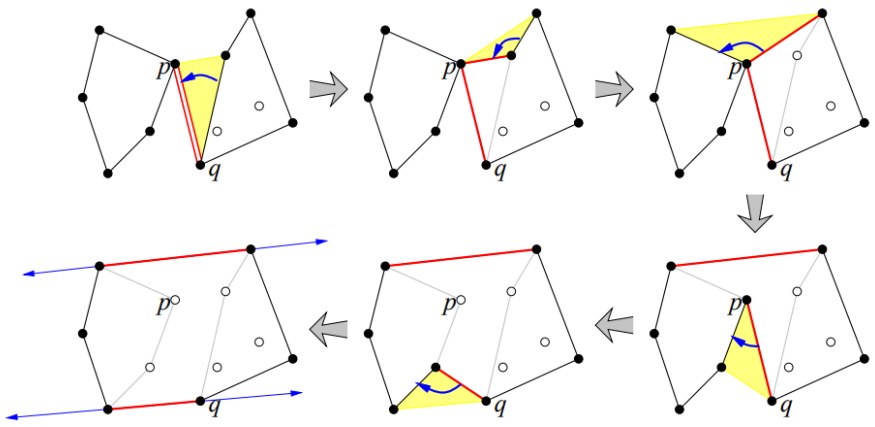
olamiz, lekin eng yaxshi holatda h = 3 va algoritm faqat O(n) vaqtni oladi. Hisoblash geometriyasi mutaxassislari buni chiqishga sezgir algoritm deb atashadi; chiqish signali qanchalik kichik bo‘lsa, algoritm tezroq ishlaydi.

1.3 Bo‘l va zabt eting (bo‘linish)

Jarvis Marsh algoritmining xatti-harakati tanlov bo‘yicha saralashga juda o‘xshaydi: keyingi nuqtaga o‘tadigan elementni qayta-qayta topadi. Aslida, ko‘pchilik qavariq qobiq algoritmlari qandaydir saralash algoritmiga o‘xshaydi.

Masalan, quyidagi qavariq qobiq algoritmi quicksortga o‘xshaydi. Biz markaz nuqtasini p.ni tanlash bilan boshlaymiz va kirish nuqtalarini ikkita to‘plamga ajratiladi $L \; va \; R$, nuqtalar x-koordinatalarini taqqoslash orqali chap va o‘ng tomonga jaratiladi.

Birlashtirish bosqichi biroz tushuntirishni talab qiladi. Biz ikkita qobiqni L qobig’ining eng o‘ng nuqtasi bilan R qobig’ining eng chap nuqtasi orasidagi chiziq segmenti bilan bog’lashdan boshlaymiz. (Ha, bu bir xil p.) aslida, $p \; va \; q$ segmentining ikkita nusxasini qo‘shamiz va ularni ko‘priklar deb ataymiz. $p \; va \; q$ bir-birlarini ko‘rishlari mumkin, bu ko‘priklarning so‘nggi nuqtalaridan tashqari qavariq shaklidagi ko‘pburchakni yaratadi.

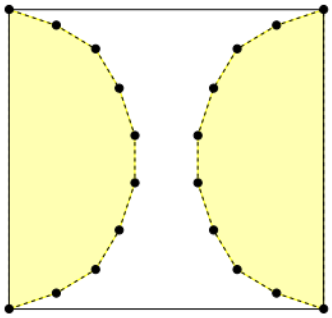


Rasm- 3 Chap va o‘ng pastki qobiqlarni birlashtirish

Ko‘prikning har ikkala so‘nggi nuqtasida soat yo‘nalishi bo‘yicha burilish mavjud ekan, biz bu nuqtani aylana nuqtalar ketma-ketligidan olib tashlaymiz va uning ikki qo‘shnisini bog’laymiz. Ikkala ko‘prikning ikkala chegaraviy nuqtasidagi burilishlar soat miliga teskari bo‘lganda, biz to‘xtashimiz mumkin. O‘sha paytda ko‘priklar ikkita pastki qobiqning yuqori va pastki umumiy tangens (shu nuqtada egri chiziqqa "tegadigan" to‘g’ri chiziq) chiziqlarida yotadi. Bu ikkala pastki qobiqqa tegadigan ikkita chiziq, chunki ikkala pastki qobiq ham yuqori umumiy teginish chizig’i ostida va pastki umumiy teginish chizig’i ustida yotadi.

Ikki pastki qobiqni birlashtirish eng yomon holatda O(n) vaqtni oladi. Shunday qilib, vaqt murakkabligi xuddi quicksort kabi $T(n) = O(n) + T(k) + T(n-k)$ ga teng, bu yerda k R daginuqtlara soni.Xuddi quicksort singari, agar biz p burilish nuqtasini tanlash uchun naokursve deterministik algoritmidan foydalansak, ushbu algoritmning eng yomon ish vaqti $O(n^2)$. Agar biz burilish nuqtasini tasodifiy tanlasak, kutilgan vaqti murakkabligi O(nlogn) ga teng.

Ushbu algoritm aniq isrofgar bo‘lgan holatlar mavjud. Agar biz haqiqatan ham omadsiz bo‘lsak, birlashish bosqichida biz qobiqlarni yig’ish uchun uzoq vaqt sarflaymiz (xxx-rasm).

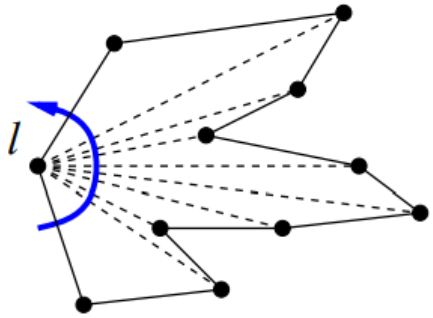


Rasm- 4 Bo‘l va zabt algoritimin qo‘llash tavsiya etilmaydigan nuqtalar to‘plami.

1.4 Graham Scan algoritmi (skanerlash)

Graham Scan deb nomlangan uchinchi qavariq qobiq algoritmimiz avval O(nlogn) dagi nuqtalarni saralaydi, so‘ngra qobiq qurilishini yakunlash uchun chiziqli vaqtni skanerlash algoritmini qo‘llaydi.

Biz Jarvis yurishida bo‘lgani kabi, l ning eng chap nuqtasini topib, Graham Scan skanerlashni boshlaymiz. Biz buni har qanday taqqoslash asosida saralash algoritmi (quicksort, mergesort, heapsort va boshqalar) yordamida O(nlogn) vaqtida bajarishimiz mumkin. P va q ikkita nuqtani solishtirish uchun biz uchlik l, p, q soat yo‘nalishi bo‘yicha yoki teskari yo‘nalishda ekanligini tekshiramiz. Nuqtalar saralangandan so‘ng, biz ularni l nuqtadan boshlab va tugaydigan soat miliga teskari tartibda bog’ladik.



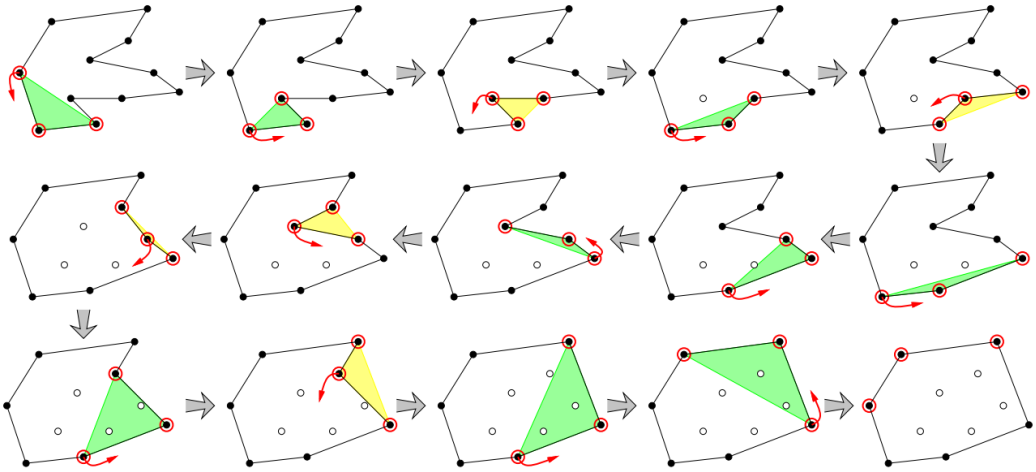
Rasm- 5 Graham skanerlash algoritmi

Graham skanerlash algoritmida saralash bosqichida hosil bo‘lgan oddiy ko‘pburchak.

Ushbu ko‘pburchakni qavariq qobiqqa aylantirish uchun biz quyidagi "uch nuqta algoritmi" ni qo‘llaymiz. Bizda p, q, r ko‘pburchakning ketma-ket uchta tepasida joylashgan uchta nuqta bor; dastlab bu l va l dan keyin ikkita nuqta. endi biz nuqta l oldinga siljiguncha quyidagi ikkita qoidani qayta-qayta qo‘llaymiz:

- Agar p, q, r soat miliga teskari tartibda bo‘lsa, orqa nuqtani r vorisiga oldinga siljiting.

- Agar p, q, r soat yo‘nalishi bo‘yicha tartibda bo‘lsa, q ni olib tashlang ko‘pburchakdan, p r chetini qo‘shing, va p.o‘rta nuqtani oldingisiga orqaga siljiting.



Rasm- 6 Graham scan 'uch nuqtalik' skanerlash bosqichi

Har doim bir nuqta oldinga siljiganida, u ilgari bir nuqtani ko‘rmagan cho‘qqiga chiqadi (oxirgi silzish bundan mustasno), shuning uchun birinchi qoida n−2 marta qo‘llaniladi. Bir nuqta orqaga qarab harakat qilganda, tepalik ko‘pburchakdan chiqariladi, shuning uchun ikkinchi qoida aniq qo‘llaniladi n − h marta, bu erda h, odatdagidek, qavariq qobig’ining tepalari soni. Har bir sinov soat yo‘nalishiga farqli o‘laroq doimiy vaqtni talab qilganligi sababli, skanerlash bosqichi odatda O(n) vaqtni oladi.