بسم الله الرحمن الرحیم

دانشکده فنی و مهندسی
گروه علمی و مهندسی کامپیوتر و فناوری اطلاعات

الگوریتم Convex Hull
گروه G-A03

استاد راهنما : دکتر سید علی رضوی ابراهیمی

تدوین : پروین حاجت پور بیرگانی

CONVEX HULL

هندسه محاسباتی

هندسه ی محاسباتی شاخه ای از علوم کامپیوتر است که به بررسی الگوریتم هایی برای حل مسائل هندسی می پردازد . در مهندسی و ریاضیات مدرن ، هندسه محاسباتی کاربردهای زیادی در بخش های مختلف مانند گرافیک کامپیوتری ، روبوتیک ، طراحی کامپیوتری ، مدل سازی ملکولی ، صنعت ، متالورژی و فلیدهای مختلف دیگر دارد . معمولاً ورودی یک مسئله هندسه ی محاسباتی ، توصیف مجوعه ای از اشیاء هندسی است ، مانند مجموعه ای از نقاط ، پاره خط ها ، و یا رأس های یک چند ضلعی به ترتیب در جهت چرخش عقربه های ساعت . و خروجی پاسخی است به یک پرسش در مورد اشیاء . مثلاً اینکه آیا هیچ یک از خطوط با هم برخورد دارند و یا یک شیء هندسی جدید مثل پوسته ی محدب (کوچک ترین چند ضلعی محاطی محدب) برای مجموعه ای از نقاط .

- **Computational Geometry : data structures and algorithms for geometric problems**

APPLICATIONS :

- **Robotics**
- **Computer-Aided Design**
- **GIS**
- **Computer Graphics**
- **Molecular Biology**
- **Data bases**

شاخه‌های اصلی هندسه محاسباتی عبارتند از:

- هندسه محاسباتی ترکیبی (هندسه الگوریتمی): این هندسه محاسباتی اشیای هندسی را به عنوان موجودات گسسته در نظر می‌گیرد. براساس کتابی که توسط پرپاراتا و شاموس نوشته شده‌است، لفظ هندسه محاسباتی با این مفهوم، نخستین بار در سال ۱۹۷۵ بیان شده است.
- هدف اصلی از پژوهش در زمینه هندسه محاسباتی ترکیبیاتی این است که، برای حل مسائلی که در زمینه اشیای پایه هندسی (نقاط، خطها، چند ضلعی‌ها، چند وجهی‌ها و...) مطرح می‌شوند، الگوریتم‌ها و ساختارهای داده ی مناسبی تولید شود.
- برخی از این مسائل به قدری آسان به نظر می‌رسند که تا زمان پیدایش رایانه‌ها مشکل به حساب نمی‌آمدند. برای مثال به مسئله نزدیک‌ترین زوج توجه کنید:
- n نقطه در صفحه داریم. دو نقطه‌ای که کمترین فاصله را از یکدیگر دارند، پیدا کنید :
- می‌توان فاصله بین جفت نقطه‌ها، که تعدادشان معلوم هست را پیدا کرد و بعد کوچک‌ترین عدد را انتخاب کرد. این الگوریتم از مرتبه $n2$ است. منظور این است که زمان اجرایش به مربع تعداد نقاط بستگی دارد. یک نتیجه کلاسیک در هندسه محاسباتی ایجاد الگوریتمی بود که از مرتبه $n \log n$ زمان بگیرد .الگوریتم‌های تصادفی که از مرتبه n زمان می‌برند، علاوه بر الگوریتم‌های تعیین‌کننده‌ای که از مرتبه $n \log n$ زمان می‌برند نیز کشف شده‌اند.
- برای *GIS* جدید، گرافیک کامپیوتری و سیستم‌های طراحی مدارهای مجتمع که روزانه ده‌ها و صدها میلیون نقطه را به کار می‌گیرند، تفاوت بین مرتبه $n^۲$ و مرتبه $n \log n$ می‌تواند تفاوت بین روزها و ثانیه‌ها محاسبه، باشد. به همین دلیل است که در هندسه محاسباتی تأکید زیادی روی پیچیدگی محاسباتی شده‌است.

شاخه‌های اصلی هندسه محاسباتی عبارتند از:

- هندسه محاسباتی عددی (هندسه ماشینی، طراحی هندسی با کمک رایانه یا مدل‌سازی هندسی) : اساس کار این هندسه محاسباتی به این صورت است که اشیای دنیای واقعی را به صورت مناسبی برای محاسبات رایانه‌ای در سیستم‌های کد/کم در می‌آورد. این شاخه ممکن است به عنوان هندسه توصیفی پیشرفته در نظر گرفته شود و اغلب یکی از شاخه‌های گرافیک کامپیوتری یا کَد به حساب می‌آید. هندسه محاسباتی با این معنا، از سال ۱۹۷۱ مورد استفاده قرار گرفت.
- در مسائل جستجوی هندسی ورودی از دو قسمت تشکیل شده‌است: فضای جستجو و قسمت جستجو
- برخی مسائل اساسی جستجوی هندسی عبارتند از:
- جستجوی محدوده :مجموعه‌ای از نقاط را پیش پردازش می‌کند برای اینکه در داخل محدوده مطلوب، تعداد نقاط را بشمارد.
- محل یابی نقطه :با دریافت فضایی که تقسیم بندی شده، یک ساختار داده تولید می‌کند که به ما می‌گوید نقطه مورد نظر، در کدام قسمت قرار دارد.
- نزدیک‌ترین همسایه :مجموعه‌ای از نقاط را به این منظور پیش پردازش می‌کند که تعیین کند کدام نقطه، به نقطه مورد نظر نزدیک‌تر است.
- ردیابی پرتو :با دریافت مجموعه‌ای از اجسام در فضا، یک ساختار داده تولید می‌کند تا تعیین کند که پرتوی جستجو ی مورد نظر، نخستین بار با کدام جسم برخورد می‌کند.
- اگر فضای جستجو ثابت باشد، پیچیدگی محاسباتی برای این دسته از مسائل بر اساس مطالب زیر تخمین زده می‌شود:
- زمان و حافظه لازم برای ساختن ساختار داده‌ای که باید در داخل آن جستجو شود.
- زمان (و برخی مواقع یک حافظه اضافی) برای پاسخ به جستجوها.

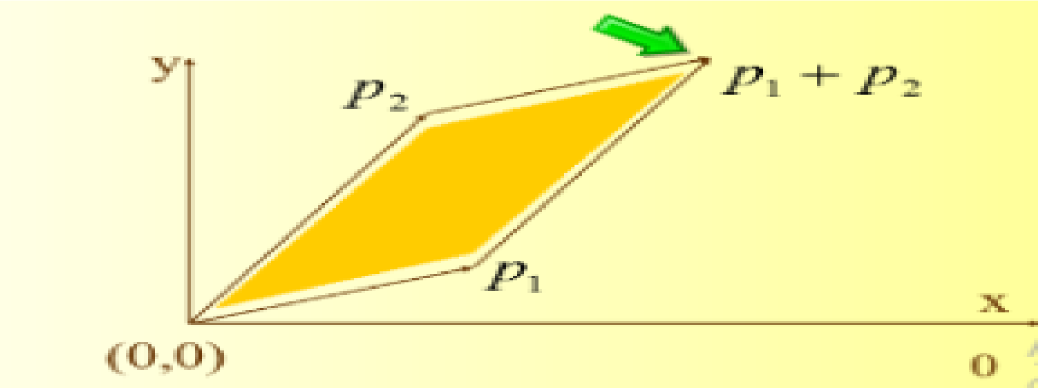
- برای حالتی که فضای جستجو تغییر می‌کند، **مسائل پویا مطرح است**
- یکی دیگر از گروه‌های اصلی مسائل، مسائل پویا هستند که در آن‌ها هدف، یافتن الگوریتمی مناسب برای یافتن راه حلی است که بعد از هر تغییر داده ورودی (اضافه یا حذف کردن عناصر هندسی) تکرار شود. الگوریتم‌های این نوع مسائل اغلب شامل ساختارهای داده پویا است . هر کدام از مسائل هندسه محاسباتی را می‌توان به مسئله پویا تبدیل کرد. برای مثال، مسئله جستجوی محدوده را می‌توان با اضافه کردن امکان اضافه یا حذف کردن نقطه‌ها به مسئله جستجوی پویای محدوده تبدیل کرد. مسئله پوسته محدب پویا همان کار مسئله پوسته محدب را برای مجموعه نقاطی که به‌طور پویا تغییر می‌کنند انجام می‌دهد.
- پیچیدگی محاسباتی این دسته از مسائل با توجه به عوامل زیر تخمین زده می‌شود:
- زمان و حافظه لازم برای ساختن ساختار داده‌ای که باید در داخل آن جستجو شود.
- زمان و حافظه لازم برای تغییر دادن ساختار داده مورد جستجو، بعد از یک تغییر در فضای جستجو.
- زمان (و برخی مواقع یک حافظه اضافی) برای پاسخ به جستجوها.

خصوصیات پاره خط ها :

- پاره‌خط در هندسه به جزئی از خط گفته می‌شود که به دو نقطه انتهایی محدود شده، و تمامی نقاط مابین آن دو را در بر بگیرد.
- در مورد چندضلعی‌ها، پاره‌خط را ضلع می‌نامند هرگاه که دو نقطه انتهایی آن در حکم دو رأس مجاور چندضلعی باشد، و در غیر این صورت، به آن قطر گفته می‌شود.
- اگر A و Bدو نقطه انتهایی پاره خطی باشند این پاره خط را با نماد ABنشان می‌دهیم.

تعریف هندسی پاره‌خط

ضرب خارجی : CROSS PRODUCT



بردارهای P1 و P2 شکل را در نظر بگیرید که cross product یا همان ضرب خارجی P1×P2 را میتوان به صورت ناحیه ی علامت دار متواضی الاضلاع تشکیل شده از نقاط (0 و 0) ، P1 و P2 ، و P1+P2= (X1+X2,Y1+Y2) در نظر گرفت . ضرب خارجی در واقع یک مفهوم سه بعدی است و برداری است عمود بر P1 و P2 طبق " قانون دست راست " و اندازه آن برابر است با :

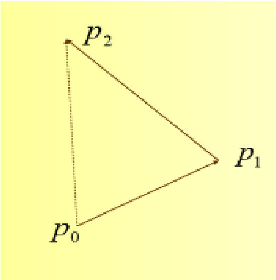
| X1Y2-X2Y1 |

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

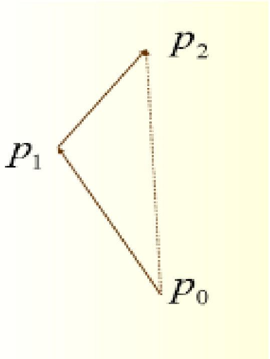
تعیین چرخش پاره خط های متوالی :

- استفاده از ضرب خارجی برای تعیین چرخش پاره خط های متوالی P0P1 و P1P2 در نقطه P1 . بررسی میکنیم که آیا پاره خط جهت دار P0P2نسبت به پاره خط جهت دار P0P1در جهت عقربه های ساعت است یا خلاف آن .
- شکل الف : اگر در جهت عکس حرکت عقربه های ساعت بود ، در P0 یک چرخش چپ خواهیم داشت.
- شکل ب : در غیر این صورت ، در جهت حرکت عقربه های ساعت ، یک چرخش به راست است .
- یک ضرب خارجی مثبت نشان دهنده ی نسبت در جهت عقربه های ساعت ، و در نتیجه چرخش راست است . ضرب خارجی صفر (۰) بدین معنی است که نقاط P0 ، P1 و P2 بر روی یک خط قرار دارند .

شکل الف



شکل ب



پوش محدب :

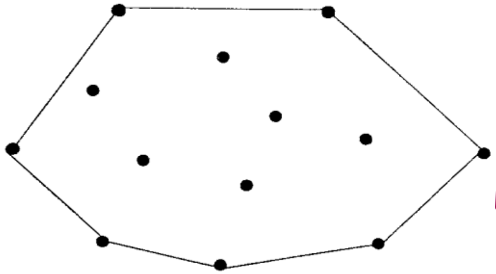
- الگوریتمی که پوش محدب اشیای مختلف را به دست می آورد کاربردهای وسیعی در ریاضیات و علوم کامپیوتر دارد.
- در هندسه محاسباتی، الگوریتم‌های متعددی با پیچیدگی‌های محاسباتی گوناگون برای محاسبه پوش محدب مجموعه‌ای محدود از نقاط مطرح شده‌است. محاسبه پوش محدب به معنی ارائه نمایشی نامبهم و کارا از شکل مطلوب می‌باشد. پیچیدگیهای الگوریتم‌های مربوطه معمولاً بر حسب n ، تعداد نقاط ورودی، وh ، تعداد نقاط درون پوش محدب، سنجیده می‌شوند.

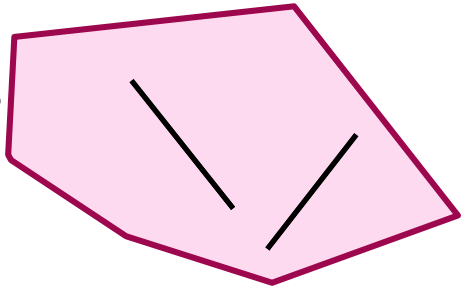
CONVEX HULL

A SET P IS CONVEX IF FOR ALL P,Q€ P, LINE SEGMENT PQ IS COMPLETELY CONTAINED IN P. THE CONVEX HULL OF P , DENOTED BU CH (P) , IS THE SMALLEST CONVEX SET CONTAINING P.

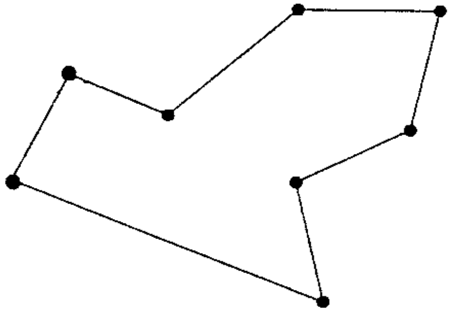
- در ریاضیات، پوشش محدب (Convex hull)یا لفاف محدب مجموعه از نقاط در صفحه اقلیدسی یا فضای اقلیدسی، کوچکترین مجموعه محدبی است که شامل این مجموعه می‌باشد. به عنوان مثال، هنگامی که X یک زیر مجموعه محدود از نقاط در صفحه است، پوشش محدب ممکن است به شکل نواری نشان داده شود که در اطراف X کشیده شده است. برای این که تصور بهتری از پوش محدب به دست آورید، نقاط صفحه را مانند میخ‌هایی در نظر بگیرید که به دیوار کوبیده شده‌اند. حال کش تنگی را در نظر بگیرید که همه میخ‌ها را احاطه کرده است. در این صورت پوش محدب نقاط شکلی خواهد بود که کش به خود می‌گیرد. مسئله یافتن پوشش محدب مجموعه نامحدود از نقاط در صفحه یا دیگر فضاهای اقلیدسی یکی از مسائل اساسی در هندسه محاسباتی است .

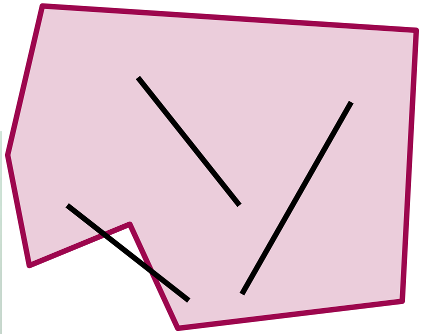
- **شکل Convex**

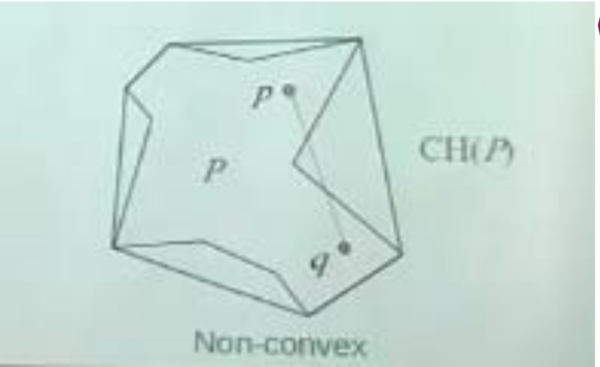




- **شکل Non-Convex**

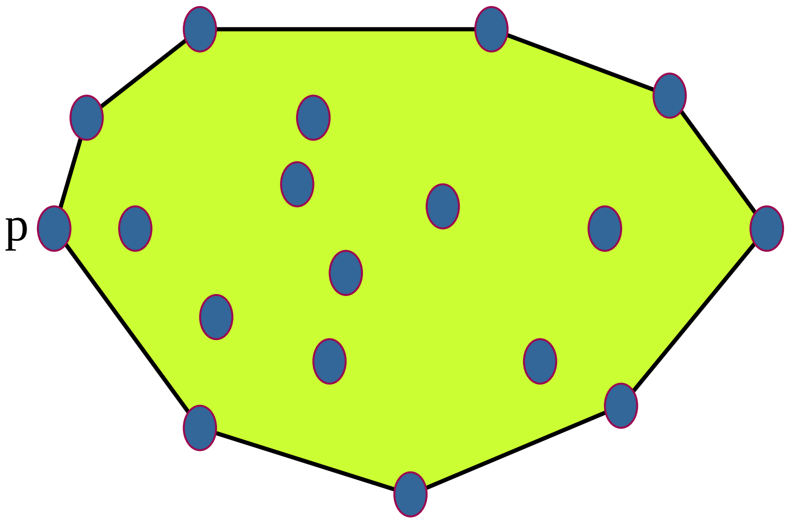




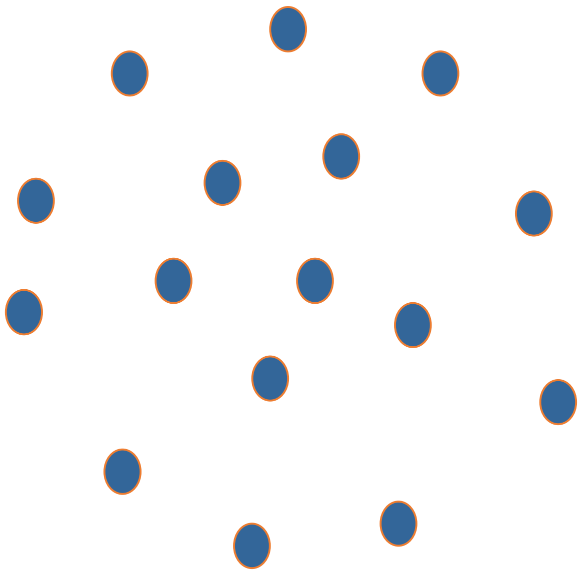


CONVEX HULL PROBLEM

GIVEN A SET P OF N POINTS , FIND ITS CONVEX HULL CH (P) .



CH(P)



- در خصوص مسئله پوسته ی محدب یا همان **convex Hull** :
- یک سری نقاط به ما داده شده است (**N** تا نقطه و میخواهیم **convex hull** آن را حساب کنیم :

2-D CONVEX HULLS

CHARACTERIZATIONS OF CH(P) :

The rubber-band analogy:

Place a rubber-band around p and let it tighten.

Smallest area convex polygon containing p .

▪ Example: If $CH(P_1) \cap CH(P_2) = \emptyset$, then objects P_1 and P_2 do not intersect.

Given $p \subseteq \Re^d$ $(d = 1,2,3, \ldots)$

Convex-Hull of p :

CH(p) = the set of all convex combinations of points in p

= the smallest convex set that contains p

= the intersection of all convex sets that contain p

Convex Hull و خصوصیات آن :

فرض کنید تعدادی نقاط داریم و میخواهیم کوچکترین مجموعه ای از این نقاط را پیدا کنیم که همه نقاط دیگر را در بر می گیرد که به این نقاط **convex hull** می گویند .

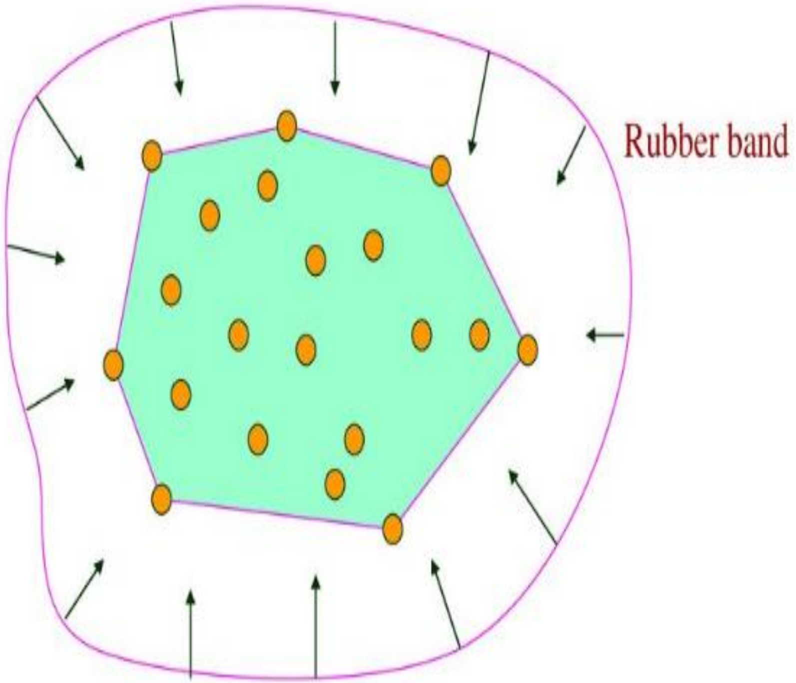
– **convex hull** مجموعه **p** کوچکترین (هم از نظر مساحت و هم از نظر محیط) **convex polygon** است که مجموعه **p** در نظر می گیرد .

تشبیه Convex Hull :

اگر بخواهیم تعریف پوسته محدب یا همان **convex hull** را تشبیه کنیم که این به چه معناست :

میتوان هر نقطه **p** را به صورت میخی در نظر گرفت که سر آن از یک تخته بیرون آمده است **convex hull** شکلی است که اگر یک کش لاستیکی دور همه نقاط یا همان میخ های بیرون آمده بیندازید ، تشکیل می شود.

The *convex hull* CH(Q) of a set Q is the *smallest* convex region that contains Q.



When Q is finite, its convex hull is the unique *convex polygon* whose vertices are from Q and that contains all points of Q.

نقاطی که روی پوسته محدب قرار می گیرند را نقاط extreme points می گویند .

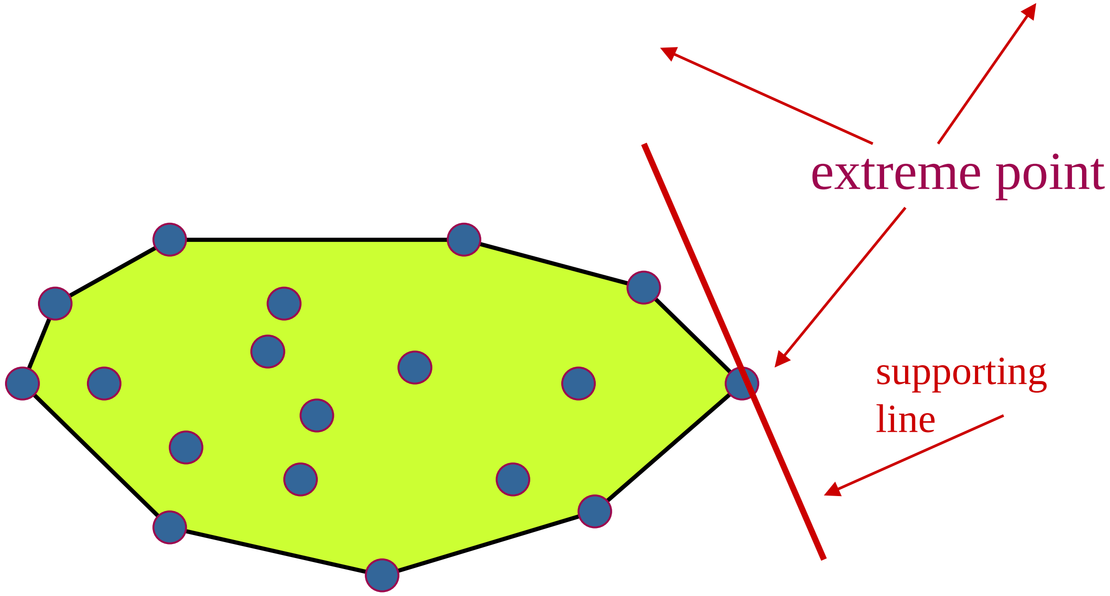
پیدا کردن این نقاط یکی از مسائل خیلی مهم است . یک مجموع نقطه به ما میدهند و می گویند convex hull آنها را حساب کنید.

چگونه متوجه شویم که دو نقطه convex hull هستند ؟

با الگوریتم ها و محاسبات ساده

۱- اول انتخاب دو نقطه از n نقطه است .

۲- همه نقاط در سمت راست یا چپ آنها باشند ، اگر همه نقاط یک سمت بودند اون دو نقطه روی Convex hull هستند . پس دو تا دو انتخاب می کنیم که به این حالت Brut Force می گویند .



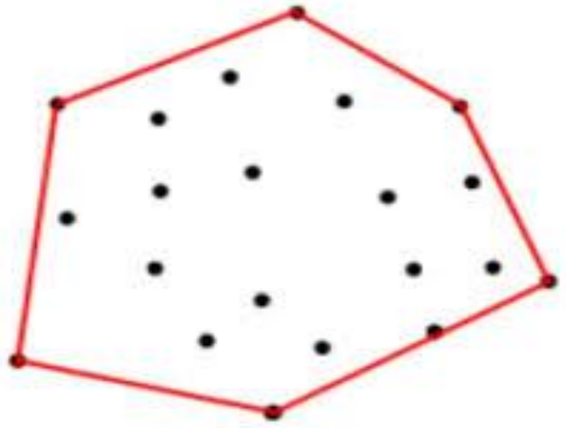
Convex Hull Algorithms:

How to find CH(p)?

- Identify extrem points
- Identify Hull Edges

OUT PUT :

- An ordered list of edges or vertices of CH(p)



CH(p)

There are many algorithms for computing the convex hull

a) Brute Force Algorithm
b) Quick Hull
c) Divide and Conquer
d) Grahams scan
e) Jarvis march(Gift wrapping)

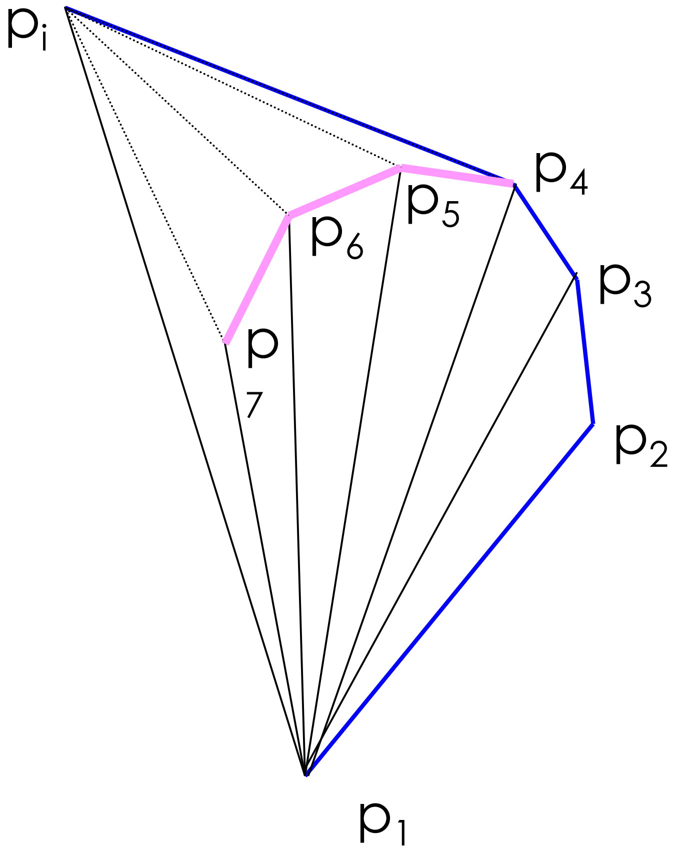
دو الگوریتم برای یافتن convex hull ، پوش محدب مجموعه ای از نقاط

الگوریتم پیمایش گراهام یا Graham Scan

روشی برای پیدا کردن convex hull است . این الگوریتم به افتخار رونالد گراهام که در سال ۱۹۷۲ الگوریتم اصلی را منتشر کرد Graham Scan نامیده شده است . مجموعه نقاط ورودی را Q در نظر بگیرید. الگوریتم پیمایش گراهام با در نظر گرفتن یک پشته از نقاط کاندید، پوش محدب را پیدا می‌کند (ما این پشته را s می نامیم). در این روش همه نقاط یک بار در پشته اضافه می‌شوند و نقاطی که بر روی محیط پوش محدب قرار ندارند در نهایت از پشته حذف می‌شوند و در نتیجه در پایان الگوریتم مجموعه نقاطی که در s قرار دارند همان رئوس پوش محدب است.

این الگوریتم در زمان (o (n lg n اجرا می شود .

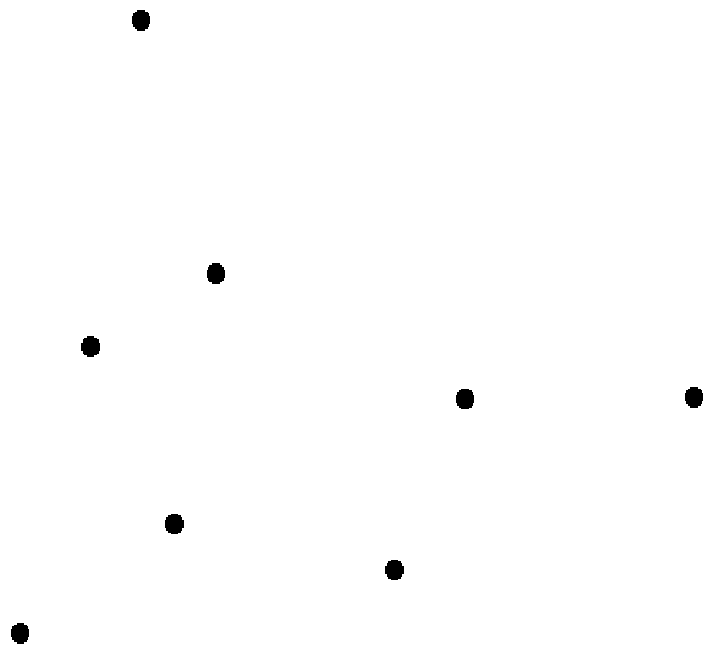
GRAHAM'S SCAN



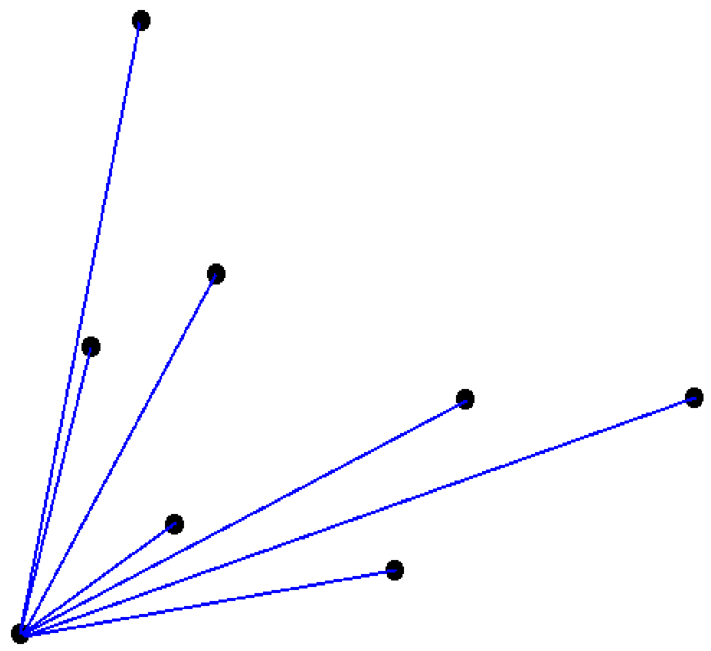
p_i
$\cancel{p_7}$
$\cancel{p_6}$
$\cancel{p_5}$
p_4
p_3
p_2
p_1

Stack S

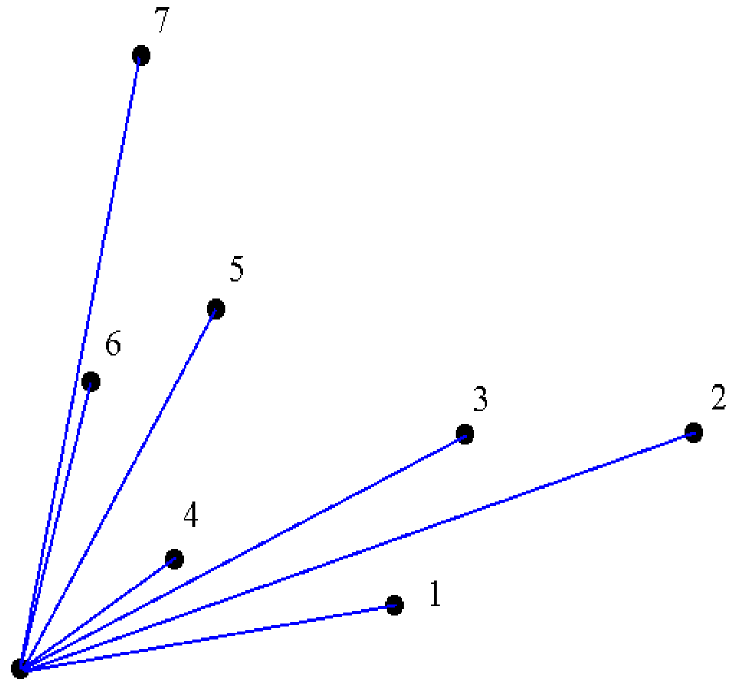
GRAHAM'S SCAN



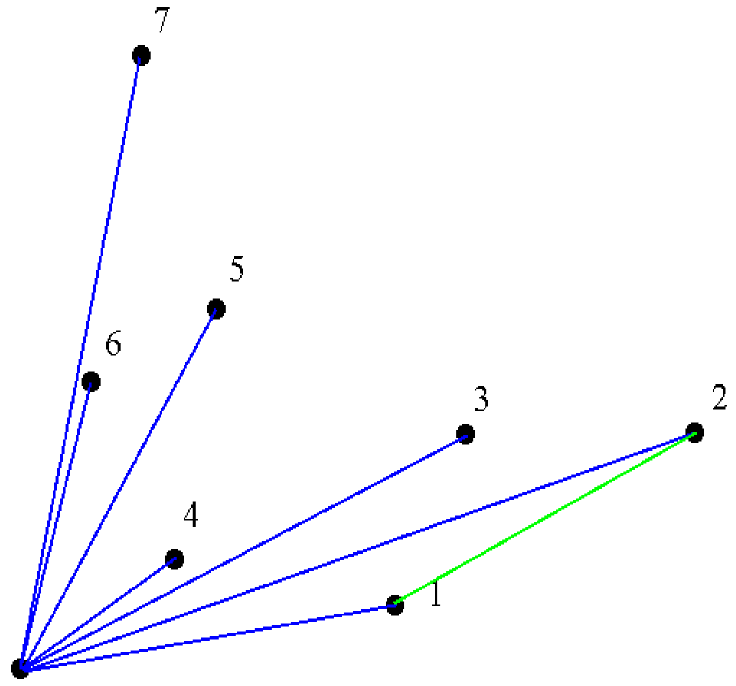
GRAHAM'S SCAN



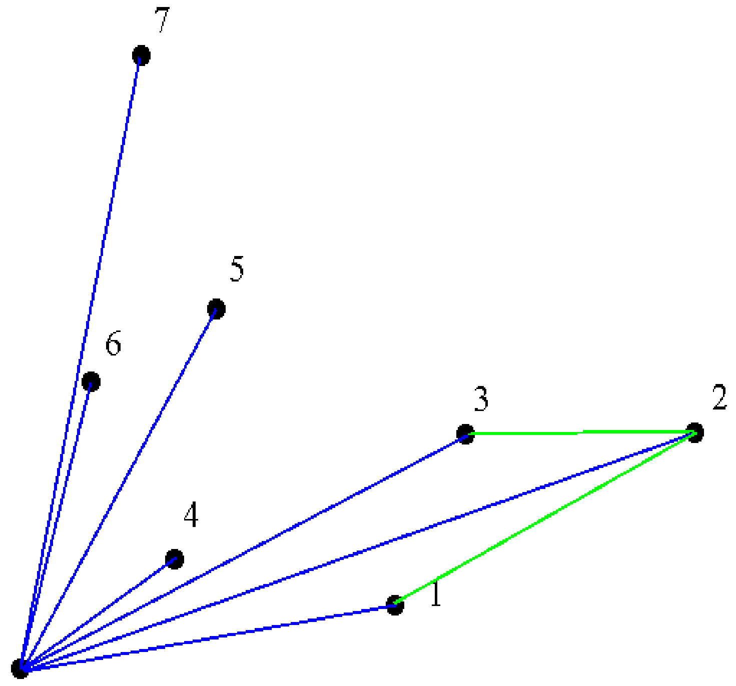
GRAHAM'S SCAN



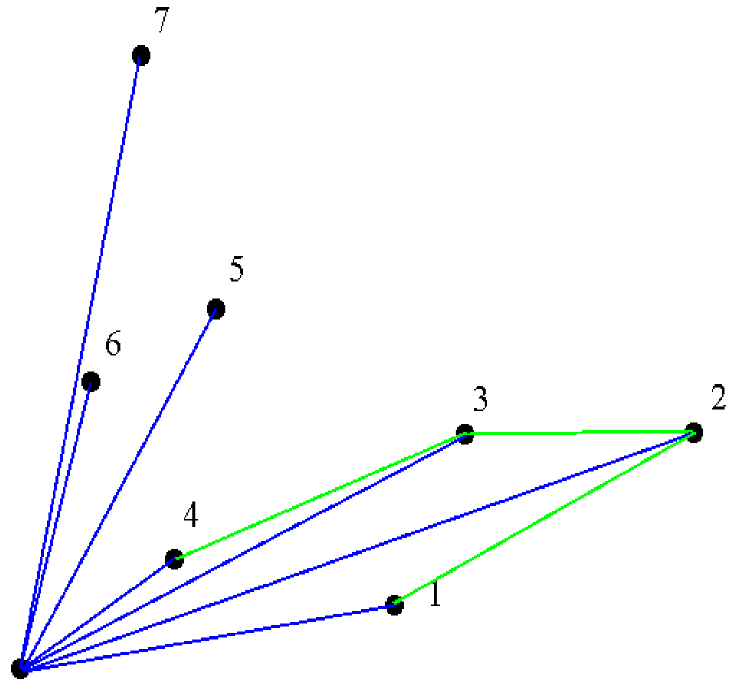
GRAHAM'S SCAN



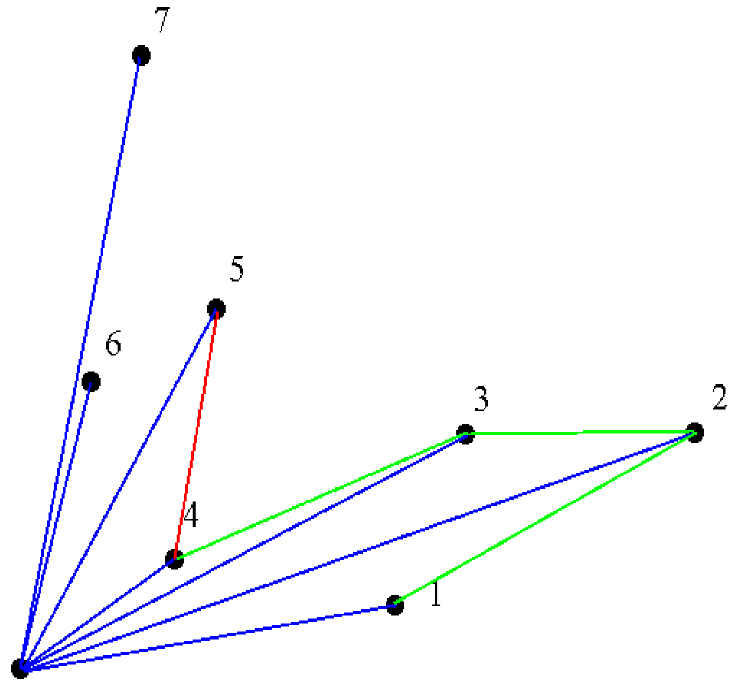
GRAHAM'S SCAN



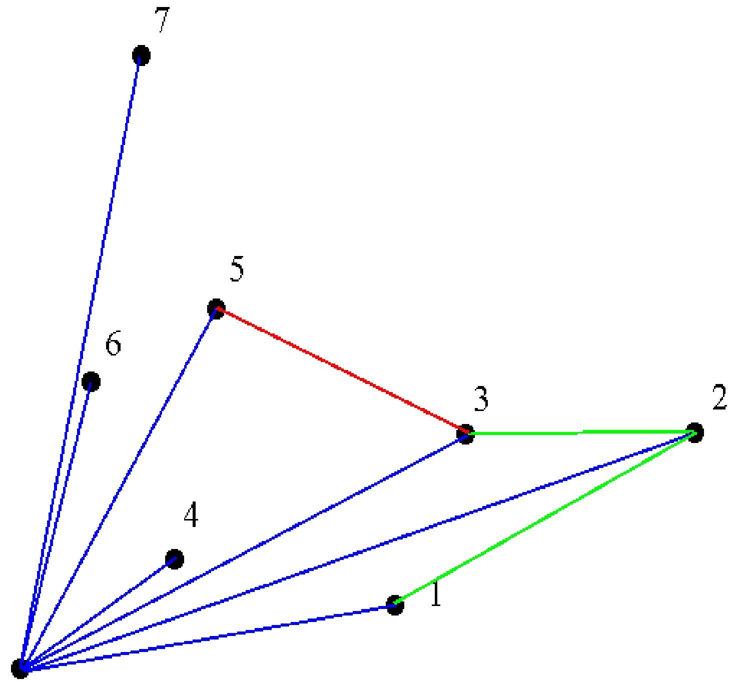
GRAHAM'S SCAN



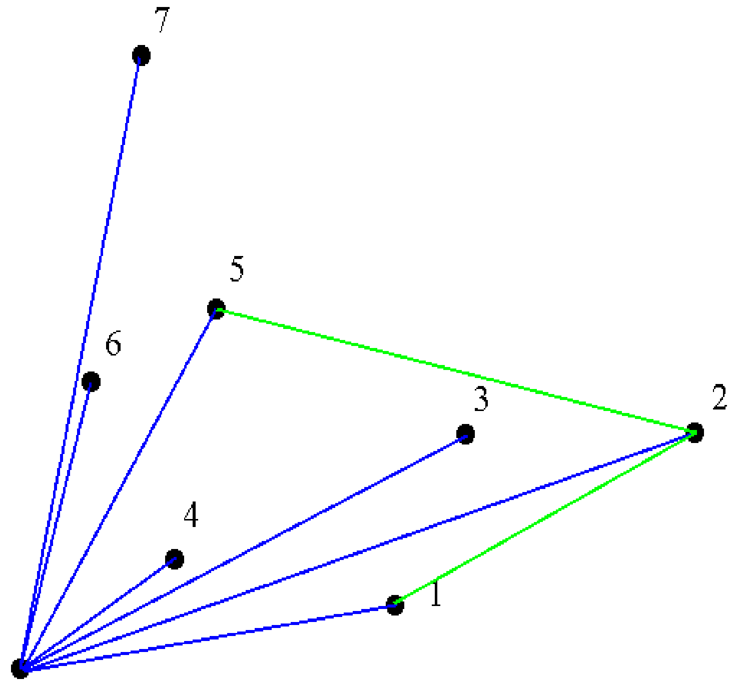
GRAHAM'S SCAN



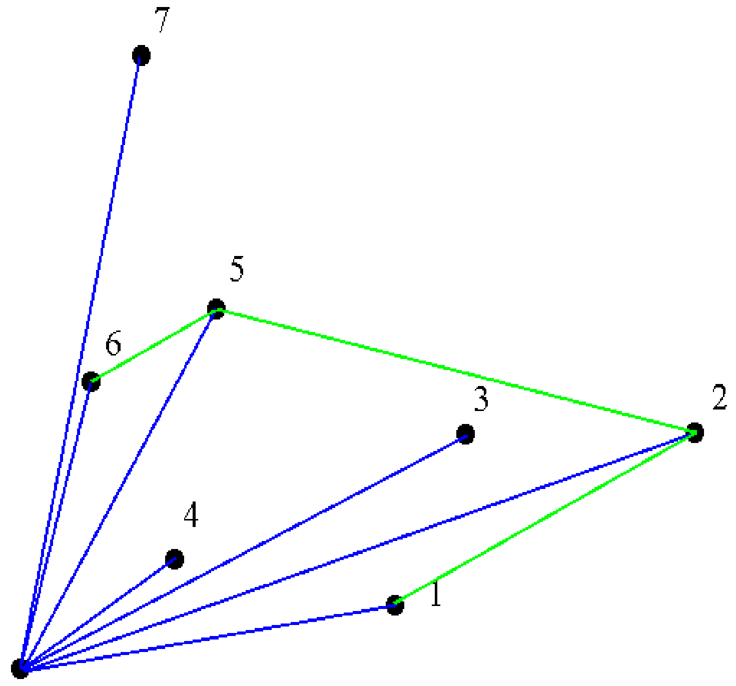
GRAHAM'S SCAN



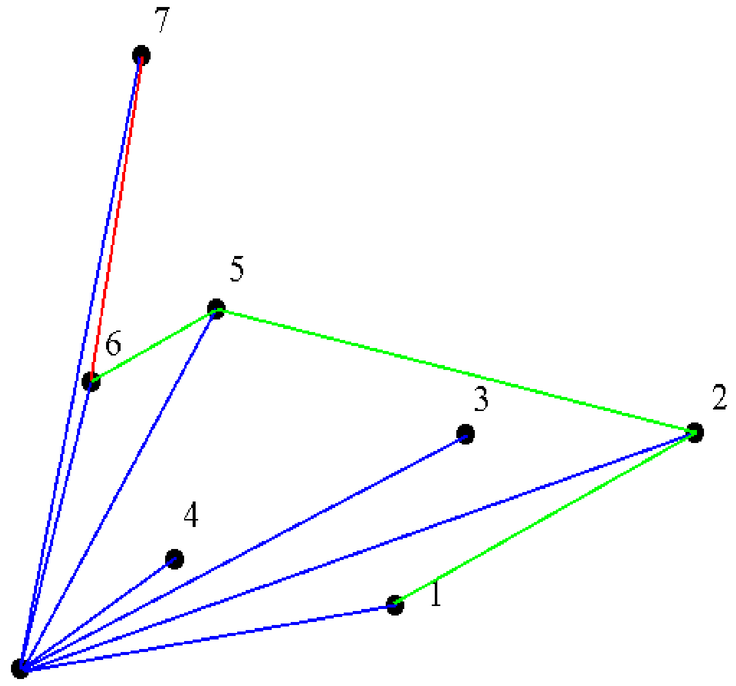
GRAHAM'S SCAN



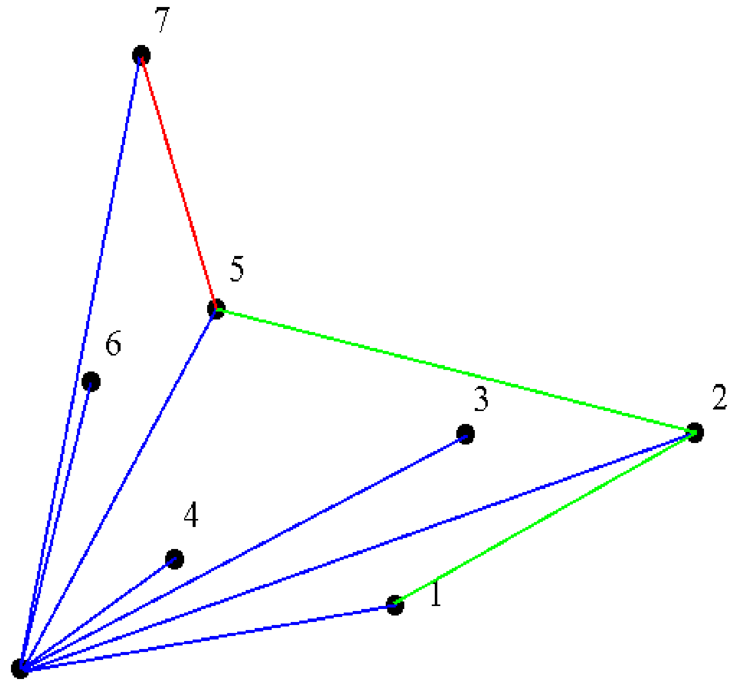
GRAHAM'S SCAN



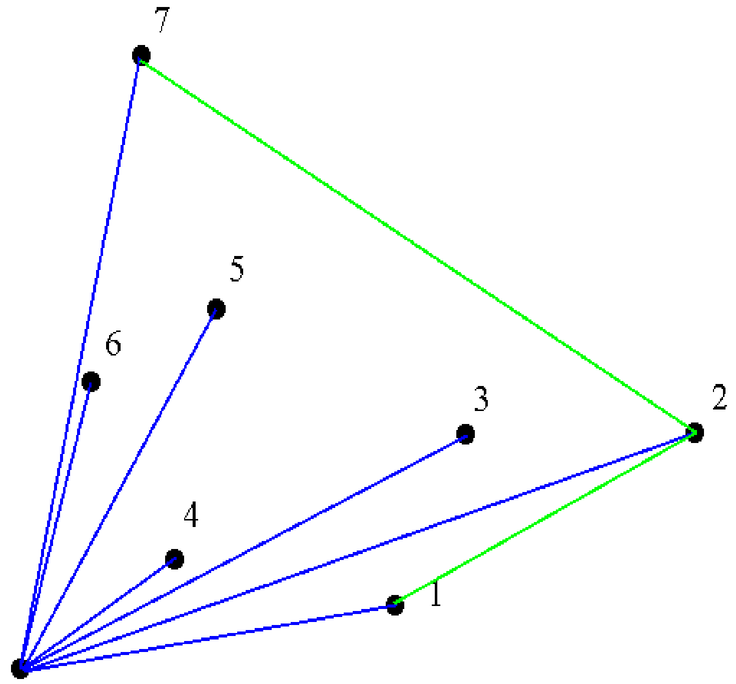
GRAHAM'S SCAN



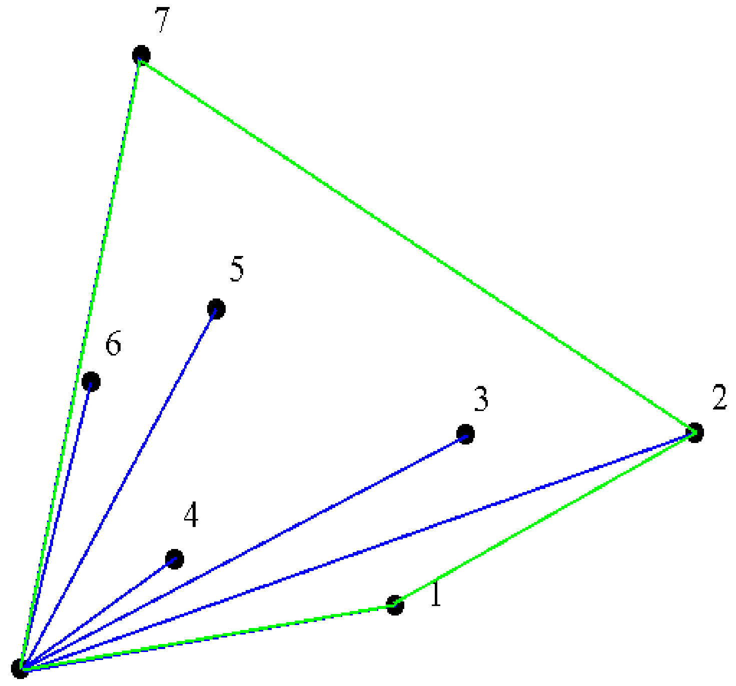
GRAHAM'S SCAN



GRAHAM'S SCAN



GRAHAM'S SCAN



شبه کد زیر الگوریتم Graham Scan را پیاده سازی می‌کند:

A more detailed algorithm

```
GRAHAM-SCAN(Q)
let p0 be the point in Q with the minimum y-coordinate,
        or the leftmost such point in case of a tie
let ⟨p1, p2, ..., pm⟩ be the remaining points in Q,
        sorted by polar angle in counterclockwise order around p0
        (if more than one point has the same angle, remove all but
        the one that is farthest from p0)
PUSH(p0, S)
PUSH(p1, S)
PUSH(p2, S)
for i ← 3 to m
    do while the angle formed by points NEXT-TO-TOP(S), TOP(S),
                and pi makes a nonleft turn
            do POP(S)
        PUSH(pi, S)
return S
```

دو الگوریتم برای یافتن convex hull ، پوش محدب مجموعه ای از نقاط

الگوریتم Jarvis's march

Jarvis's marchاز روشی به نام بسته‌بندی بسته که

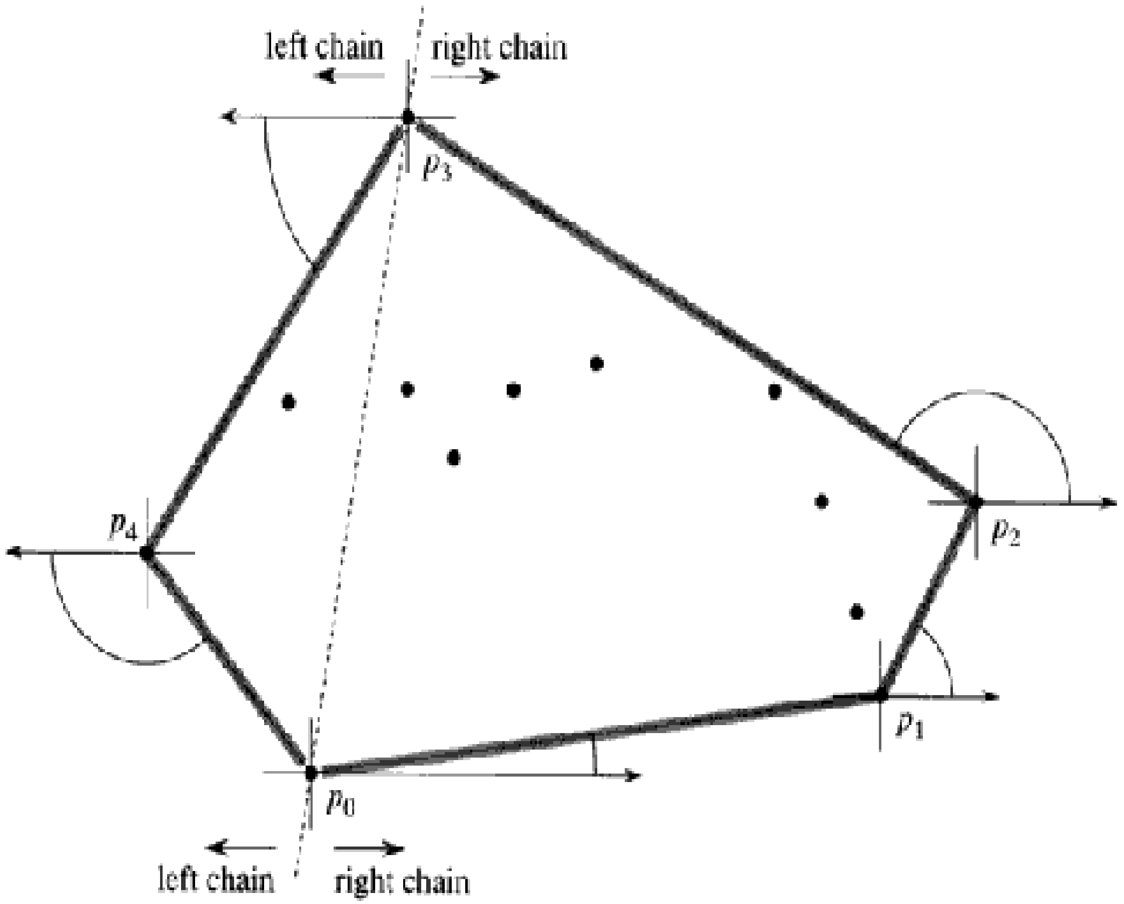
به انگلیسی: package wrapping یا Gift Wrapping

نامیده می شود برای یافتن پوش محدب مجموعه Q از نقاط صفحه

استفاده می‌کند.

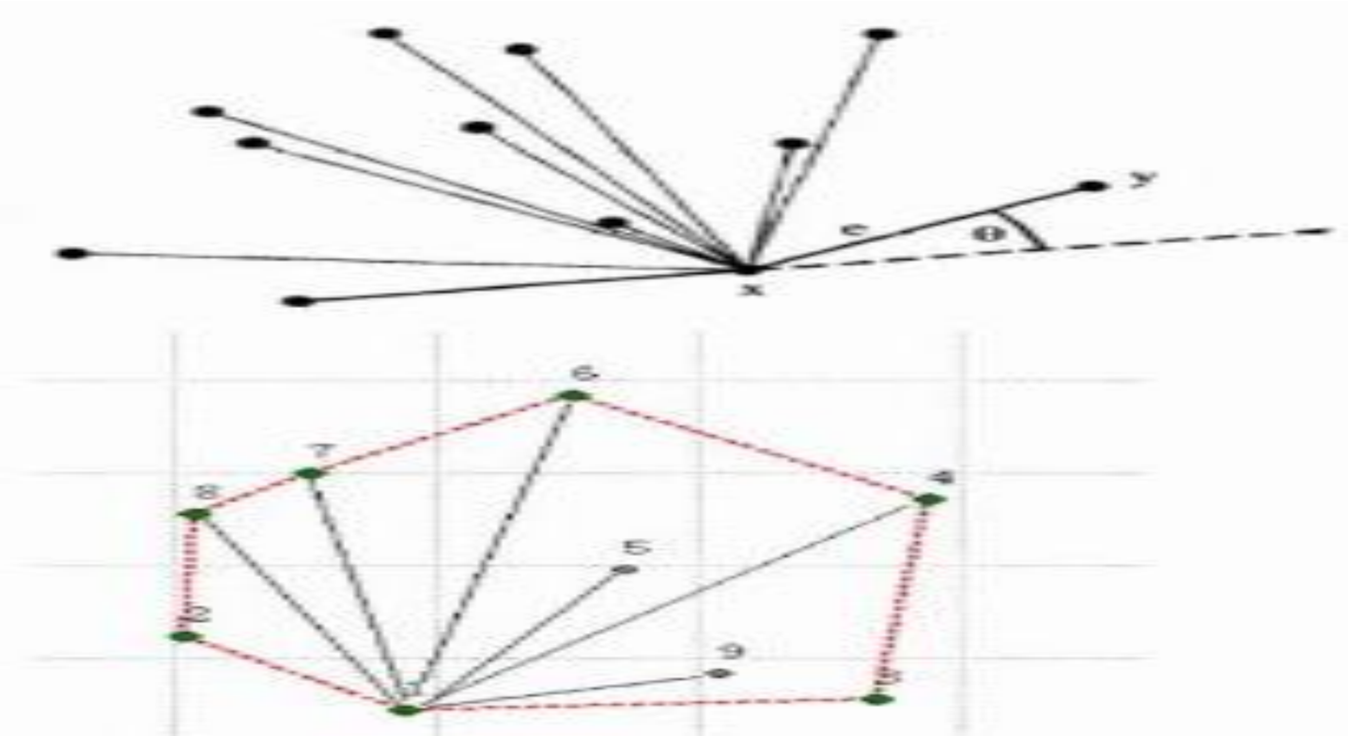
نمونه ای از فرآیند اجرای الگوریتم Jarvis's march به صورت

شکل مقابل می باشد :



Jarvis's march

جارویس در سال ۱۹۷۳، در مقاله‌ای این الگوریتم را به چاپ رساند. علت انتخاب نام بسته‌بندی هدیه برای این الگوریتم به این دلیل است که در فضای دوبُعدی نحوه عملکرد این روش مانند پیچاندان کاغذ کادو به دور مجموعه نقاط و احاطه کردن آن‌ها توسط اضلاعی است که به ترتیب به عنوان خروجی الگوریتم حاصل می‌شوند.



Jarvis's march

این الگوریتم در زمان o (nh) اجرا می شود که h تعداد رأس های پوسته محدب است . زیرا به ازای هر کدام از رئوس پوش محدب یک بار هر یک از نقاط را با عملی از O(1) چک می کنیم .

اگر h از مرتبه o (lg n) باشد، راهپیمایی Jarvis به صورت حدی سریع تر از الگوریتم Graham-scan است .

شبه کد این الگوریتم بصورت روبرو می باشد :

```
Jarvis(s)
    pointOnHull = leftmost point in S
    i = 0
    repeat
          P[i] = pointOnHull
          endpoint = S[0]          // initial endpoint for a candidate edge on the hull
          for j from 1 to |S|
              if (endpoint == pointOnHull) or (S[j] is on left of line from P[i] to endpoint)
                  endpoint = S[j]   // found greater left turn, update endpoint
          i = i+1
          pointOnHull = endpoint
    until endpoint == P[0]      // wrapped around to first hull point
```

Jarvis's march

پیچیدگی :

اگر تعداد نقاط مجموعه برابر با n و تعداد نقاط روی پوش محدب برابر با h باشد، پیچیدگی زمانی این الگوریتم از O(nh) است؛ زیرا برای هر نقطه p که عضو پوش محدب باشد باید زاویه قطبی O(n) رأس دیگر نسبت به p ، با هم مقایسه شوند که هر هر عملیات مقایسه از O(1) است. چون h نقطه روی پوش محدب قرار دارند بنابراین زمان اجرا از O(nh) است. این الگوریتم در صورتی از لحاظ زمانی کارا خواهد بود که n عددی بزرگ نباشد یا اینکه h نسبت به n کوچک باشد. به عبارت دقیق‌تر اگر $h \in o(logn)$ ، این الگوریتم از الگوریتم گراهام سریع تر است. در غیر این‌صورت این الگوریتم در مقایسه با سایر روش‌ها به‌طور مجانبی کُندتر خواهد بود که در نتیجه از الگوریتم‌های مشابه که زمان اجرای کمتری دارند استفاده می‌شود؛ مانند الگوریتم چان که زمان اجرای آن از O(nlogh) است.

تقدیم به استاد عزیزم ، جناب آقای

دکتر سید علی رضوی ابراهیمی که

تجربیات بسیار باارزشی را به من

آموخته اند.