بسم الله الرحمن الرحیم

طراحی و تحلیل الگوریتم‌ها

مبحث سیزدهم

مطالعه‌ی موضوعی الگوریتم‌ها

هندسه‌ی محاسباتی

Computational Geometry

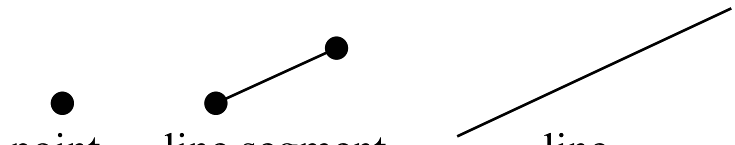
کاظم فولادی
دانشکده مهندسی برق و کامپیوتر
دانشگاه تهران

http://courses.fouladi.ir/algorithm

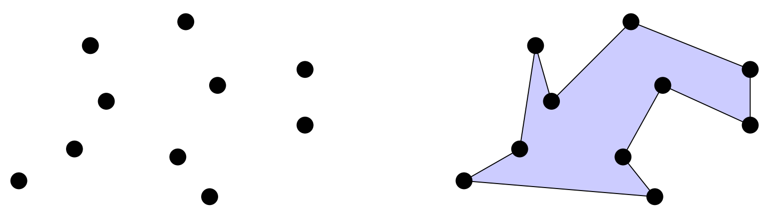
هندسه‌ی محاسباتی

[COMPUTATIONAL GEOMETRY]

الگوریتم‌هایی برای حل «مسائل هندسی» در دو بعد و بالاتر



اشیای مبنا
Fundamental Objects

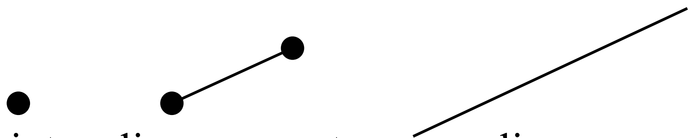


ساختارهای پایه
Basic Structures

هندسه‌ی محاسباتی

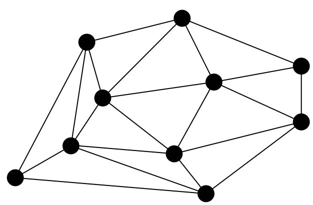
COMPUTATIONAL GEOMETRY

الگوریتم‌هایی برای حل «مسائل هندسی» در دو بعد و بالاتر

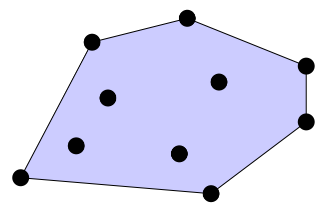


point — نقطه
line segment — پاره‌خط
line — خط

اشیای مبنا
Fundamental Objects



triangulation
مثلث‌بندی



convex hull
پوش محدب

ساختارهای پایه
Basic Structures

مسائل معروف
Well-known Problems

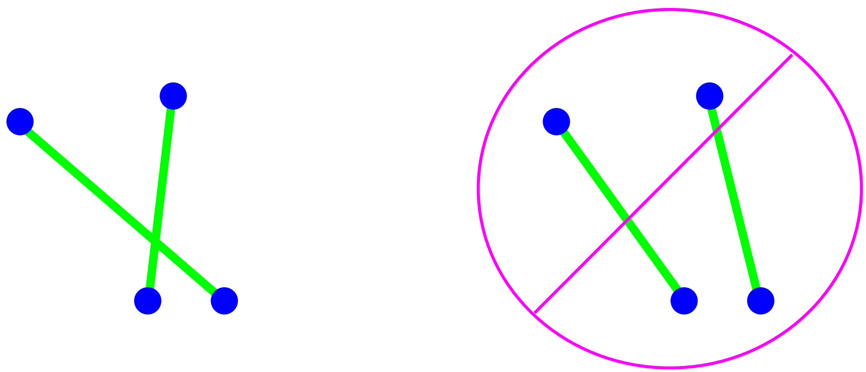
هندسه‌ی محاسباتی

مسئله‌ی نمونه (۱)

COMPUTATIONAL GEOMETRY

تقاطع دو پاره‌خط
Segment Intersection

دو پاره‌خط داده شده است: آیا با هم تقاطع دارند؟



هندسه‌ی محاسباتی

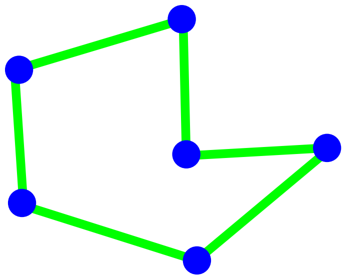
مسئله‌ی نمونه (۲)

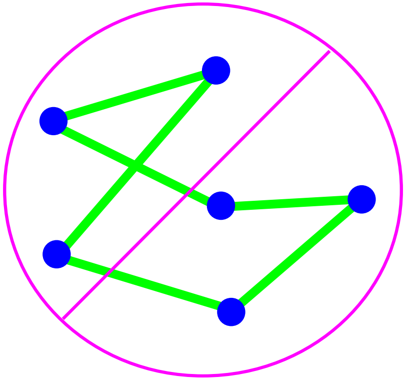
COMPUTATIONAL GEOMETRY

مسیر بسته‌ی ساده

Simple Closed Path

یک مجموعه از نقاط داده شده است: یک چندضلعی غیرمتقاطع که رئوس آن همین نقاط باشند، بیابید.





هندسه‌ی محاسباتی

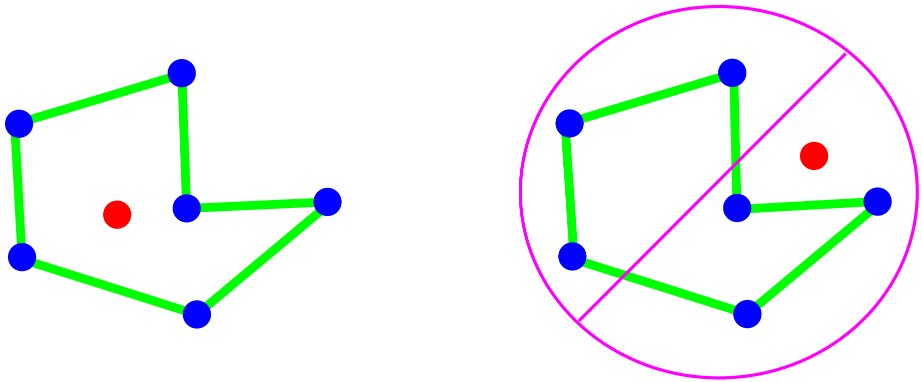
مسئله‌ی نمونه (۳)

COMPUTATIONAL GEOMETRY

شمول در چندضلعی

Inclusion in Polygon

آیا یک نقطه‌ی داده شده داخل چندضلعی است یا خارج آن؟

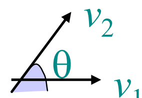


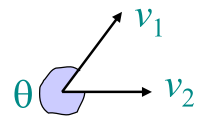
عملیات ابتدائی : ضرب خارجی

PRIMITIVE OPERATIONS: CROSSPRODUCT

Given two vectors $v_1 = (x_1, y_1)$ and $v_2 = (x_2, y_2)$, is their counterclockwise angle θ

- ***convex*** ($< 180^\circ$),
- ***reflex*** ($> 180^\circ$), or
- borderline (0 or 180°)?





Crossproduct $v_1 \times v_2 = x_1 y_2 - y_1 x_2$
$= |v_1|\,|v_2| \sin\theta$.

Thus, $\text{sign}(v_1 \times v_2) = \text{sign}(\sin\theta)$ > 0 if θ convex,
< 0 if θ reflex,
$= 0$ if θ borderline.

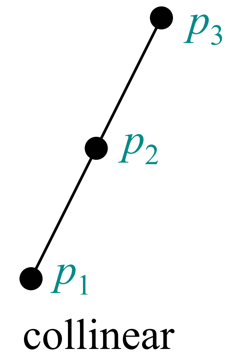
هندسه‌ی محاسباتی

عملیات ابتدائی : آزمون جهت

Primitive operations: Orientation test

Given three points p_1, p_2, p_3 are they

- in ***clockwise (cw) order***,
- in ***counterclockwise (ccw) order***, or
- ***collinear***?

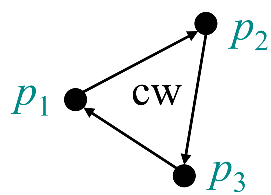


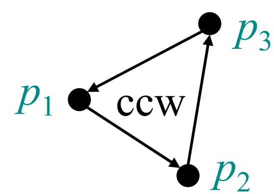
$(p_2 - p_1) \times (p_3 - p_1)$

> 0 if ccw

< 0 if cw

$= 0$ if collinear



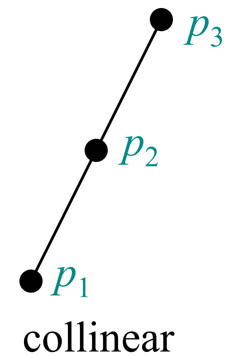


عملیات ابتدائی : آزمون سمت

PRIMITIVE OPERATIONS: SIDEDNESS TEST

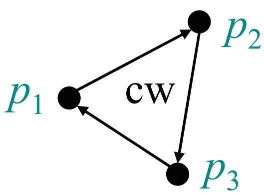
Given three points p_1, p_2, p_3 are they

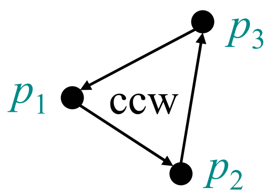
- in ***clockwise (cw) order***,
- in ***counterclockwise (ccw) order***, or
- ***collinear***?



Let L be the oriented line from p_1 to p_2.
Equivalently, is the point p_3

- ***right*** of L,
- ***left*** of L, or
- ***on*** L?





هندسه‌ی محاسباتی

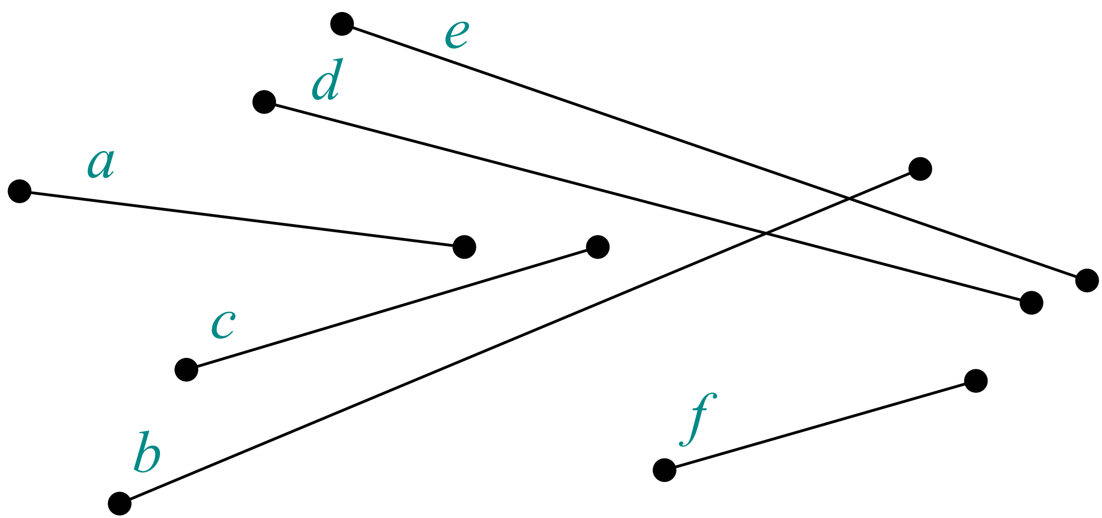
مسئله‌ی تقاطع پاره‌خط‌ها

LINE-SEGMENT INTERSECTION

تقاطع پاره‌خط‌ها

Line-Segment Intersection

n پاره‌خط داده شده است. آیا هیچ دو پاره‌خطی وجود دارند که یکدیگر را قطع کنند؟

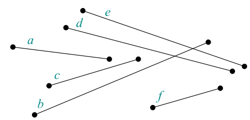


الگوریتم ناشیانه از مرتبه‌ی $O(n^2)$ است.

هندسه‌ی محاسباتی

مسئله‌ی تقاطع پاره‌خطها: الگوریتم خط روبش

LINE-SEGMENT INTERSECTION



تقاطع پاره‌خطها
Line-Segment Intersection

n پاره‌خط داده شده است. آیا هیچ دو پاره‌خطی وجود دارند که یکدیگر را قطع کنند؟

الگوریتم خط روبش
Sweep-Line Algorithm

۱) یافتن **نقاط رویداد**

- یک خط عمودی را از سمت چپ به راست حرکت دهید (روبش)
  (به لحاظ مفهومی، مختصات محور x را با زمان جایگزین می‌کنیم.)
- یک مجموعه‌ی پویای S از پاره‌خطهایی که با خط روبش تقاطع می‌کنند ایجاد می‌کنیم و عناصر را به مرور با مختصات y محل تقاطع مرتب می‌کنیم.
- ترتیب‌ها تغییر می‌کنند هرگاه:
  - به یک پاره‌خط جدید برخورد کنیم، یا
  - پاره‌خط موجود تمام شود، یا
  - دو پاره‌خط تقاطع کنند.

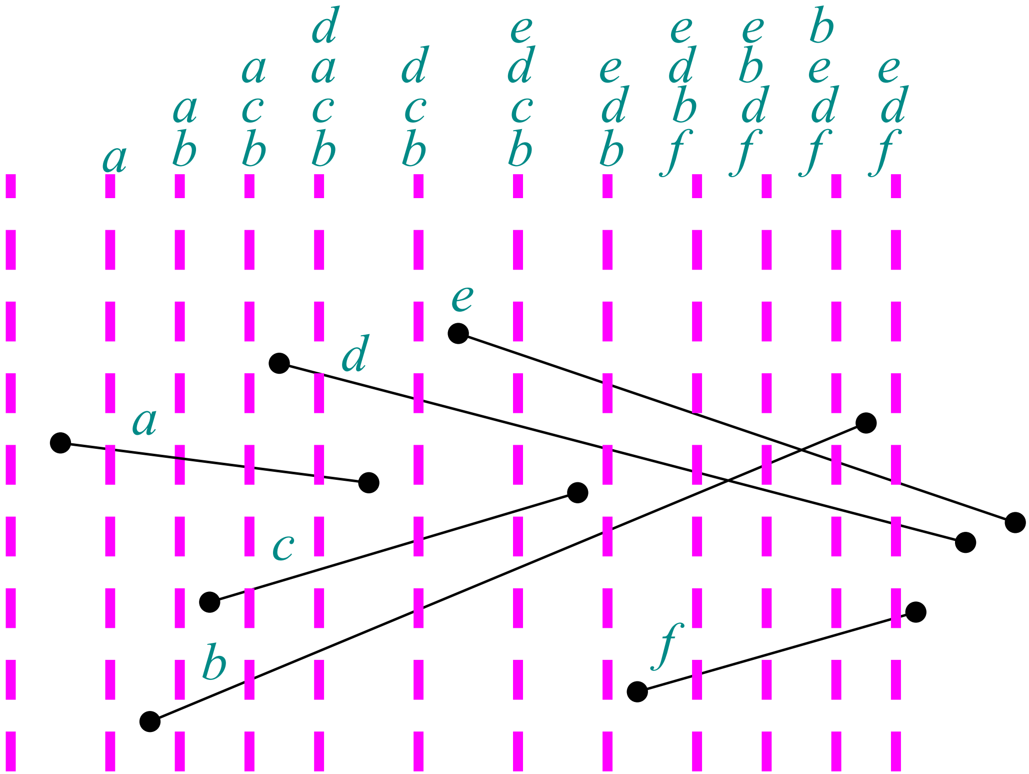
نقاط انتهایی پاره‌خط
Segment Endpoints

- بنابراین، **نقاط رویداد** کلیدی، نقاط انتهایی پاره‌خط هستند.

هندسه‌ی محاسباتی

مسئله‌ی تقاطع پاره‌خط‌ها : الگوریتم خط روبش : مثال

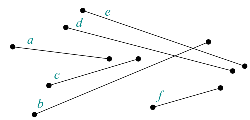
LINE-SEGMENT INTERSECTION



هندسه‌ی محاسباتی

مسئله‌ی تقاطع پاره‌خط‌ها: الگوریتم خط روبش

LINE-SEGMENT INTERSECTION



تقاطع پاره‌خط‌ها

Line-Segment Intersection

n پاره‌خط داده شده است. آیا هیچ دو پاره‌خطی وجود دارند که یکدیگر را قطع کنند؟

الگوریتم خط روبش

Sweep-Line Algorithm

۲) یافتن **نقاط تقاطع**

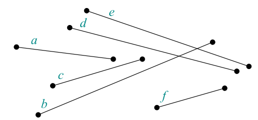
نقاط رویداد را به‌ترتیب با مرتب‌سازی نقاط انتهایی پاره‌خط بر اساس مختصات x و حلقه‌ی زیر پردازش می‌کنیم:

- برای یک نقطه‌ی انتهایی چپ پاره‌خط s:
  - پاره‌خط s را به مجموعه‌ی پویای S اضافه می‌کنیم.
  - تقاطع میان s و همسایه‌های آن در S را بررسی می‌کنیم.
- برای یک نقطه‌ی انتهایی راست پاره‌خط s:
  - پاره‌خط s را از مجموعه‌ی پویای S حذف می‌کنیم.
  - تقاطع میان s و همسایه‌های آن در S را بررسی می‌کنیم.

هندسه‌ی محاسباتی

مسئله‌ی تقاطع پاره‌خط‌ها: الگوریتم خط روبش: تحلیل زمان اجرا

LINE-SEGMENT INTERSECTION



تقاطع پاره‌خط‌ها

Line-Segment Intersection

n پاره‌خط داده شده است. آیا هیچ دو پاره‌خطی وجود دارند که یکدیگر را قطع کنند؟

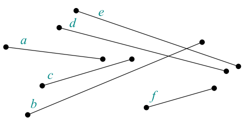
با استفاده از **درخت قرمز–سیاه** برای ذخیره‌سازی مجموعه‌ی پویای S:

کل زمان اجرا: $O(n \log n)$

هندسه‌ی محاسباتی

مسئله‌ی تقاطع پاره‌خط‌ها : الگوریتم خط روبش : اثبات درستی الگوریتم خط روبش

LINE-SEGMENT INTERSECTION



تقاطع پاره‌خط‌ها

Line-Segment Intersection

n پاره‌خط داده شده است. آیا هیچ دو پاره‌خطی وجود دارند که یکدیگر را قطع کنند؟

قضیه اگر تقاطعی وجود داشته باشد، الگوریتم خط روبش، آن را می‌یابد.

Proof: Let X be the leftmost intersection point.
Assume for simplicity that

- only two segments s_1, s_2 pass through X, and
- no two points have the same x-coordinate.

At some point before we reach X,
s_1 and s_2 become consecutive in the order of S.
Either initially consecutive when s_1 or s_2 inserted,
or became consecutive when another deleted. □

Prepared by Kazim Fouladi | Fall 2016 | 2nd Edition

کاظم فولادی

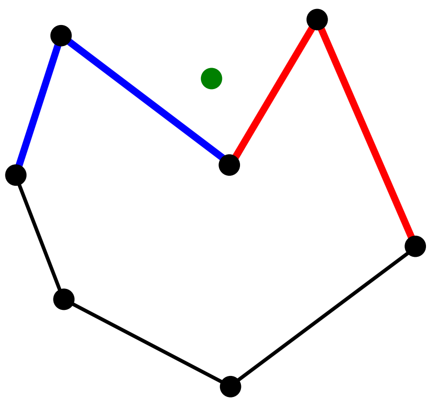
هندسه‌ی محاسباتی

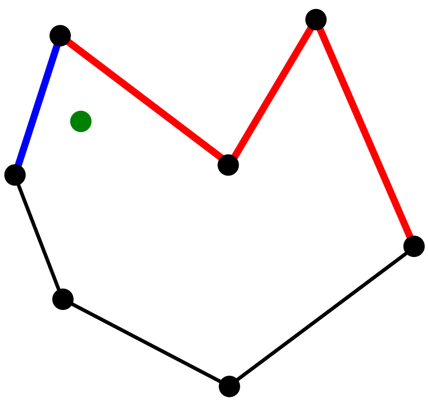
مسئله‌ی شمول نقطه

POINT INCLUSION

شمول نقطه
Point Inclusion

یک چندضلعی و یک نقطه داده شده است. آیا این نقطه داخل چندضلعی است یا خارج آن؟





مفهوم «جهت» می‌تواند به حل این مسئله در زمان خطی کمک کند.

هندسه‌ی محاسباتی

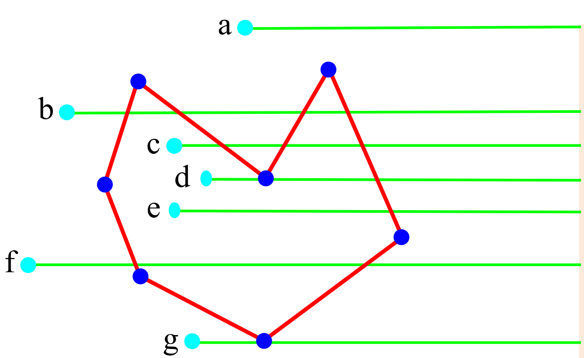
مسئله‌ی شمول نقطه

POINT INCLUSION

شمول نقطه
Point Inclusion

یک چندضلعی و یک نقطه داده شده است. آیا این نقطه داخل چندضلعی است یا خارج آن؟

الگوریتم شمارش
Counting Algorithm



- یک خط افقی از هر نقطه به سمت راست آن می‌کشیم و آن را تا بی‌نهایت امتداد می‌دهیم.
- تعداد دفعاتی که خط، چندضلعی را قطع می‌کند، می‌شماریم:
  - اگر **زوج** بود ⇐ نقطه **خارج** چندضلعی است.
  - اگر **فرد** بود ⇐ نقطه **داخل** چندضلعی است.

حالت‌های تکین: تقاطع با رئوس چندضلعی (مثل d و g)
راه حل؟: دو بار شمردن تقاطع با رئوس چندضلعی؟؟

Prepared by Kazim Fouladi | Fall 2016 | 2nd Edition

کاظم فولادی

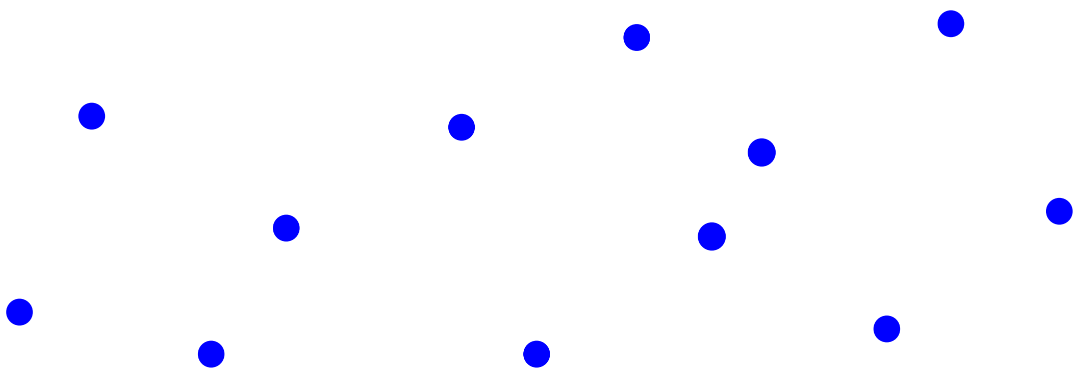
هندسه‌ی محاسباتی

مسئله‌ی نزدیک‌ترین جفت نقاط

Closest Pair of Points

نزدیک‌ترین جفت نقاط
Closest Pair of Points

یک مجموعه‌ی P از n نقطه داده شده است.
دو نقطه‌ی p و q متعلق به P را بیابید که فاصله‌ی $d(p,q)$ آنها می‌نیمم باشد.

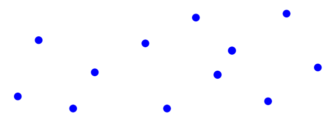


الگوریتم ناشیانه از مرتبه‌ی $O(n^2)$ است.

هندسه‌ی محاسباتی

مسئله‌ی نزدیک‌ترین جفت نقاط

CLOSEST PAIR OF POINTS



نزدیک‌ترین جفت نقاط

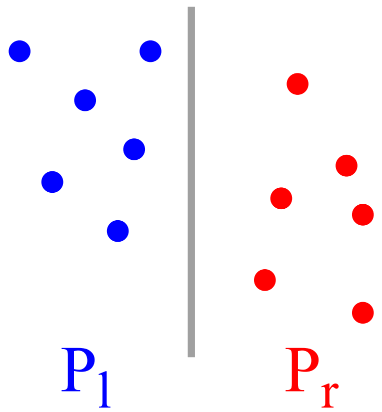
Closest Pair of Points

یک مجموعه‌ی P از n نقطه داده شده است.

دو نقطه‌ی p و q متعلق به P را بیابید که فاصله‌ی $d(p,q)$ آنها می‌نیمم باشد.

الگوریتم تقسیم و غلبه

Divide & Conquer Algorithm



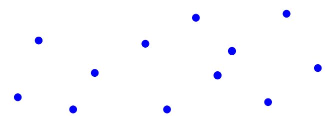
- **مرحله ۱)** نقاط را برحسب مختصات x آنها مرتب می‌کنیم و بر این اساس آنها را به دو نیمه تقسیم می‌کنیم.
- **مرحله ۲)** نزدیک‌ترین دو نقطه یکی از سه حالت زیر را دارند:
  - هر دو در نیمه‌ی اول هستند.
  - هر دو در نیمه‌ی دوم هستند.
  - یکی در نیمه‌ی اول و دیگری در نیمه‌ی دوم قرار دارد.

  که می‌نیمم آنها پاسخ مسئله است.

هندسه‌ی محاسباتی

مسئله‌ی نزدیک‌ترین جفت نقاط

CLOSEST PAIR OF POINTS

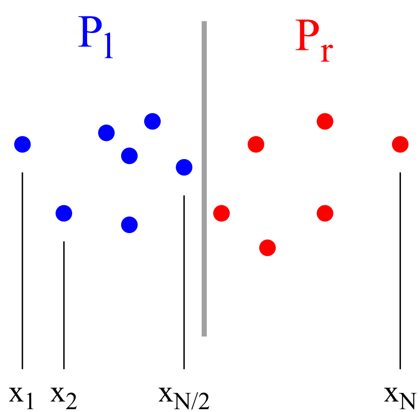


نزدیک‌ترین جفت نقاط
Closest Pair of Points

یک مجموعه‌ی P از n نقطه داده شده است.
دو نقطه‌ی p و q متعلق به P را بیابید که فاصله‌ی $d(p,q)$ آنها می‌نیمم باشد.

$p_1\ p_2\ \dots\ p_{N/2}\ \dots\ p_{N/2+1}\ \dots\ p_N$

الگوریتم تقسیم و غلبه
Divide & Conquer Algorithm



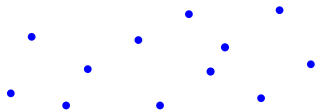
- **مرحله ۱)** نقاط را برحسب مختصات x آنها مرتب می‌کنیم و بر این اساس آنها را به دو نیمه تقسیم می‌کنیم.

Prepared by Kazim Fouladi | Fall 2016 | 2nd Edition

هندسه‌ی محاسباتی

مسئله‌ی نزدیک‌ترین جفت نقاط

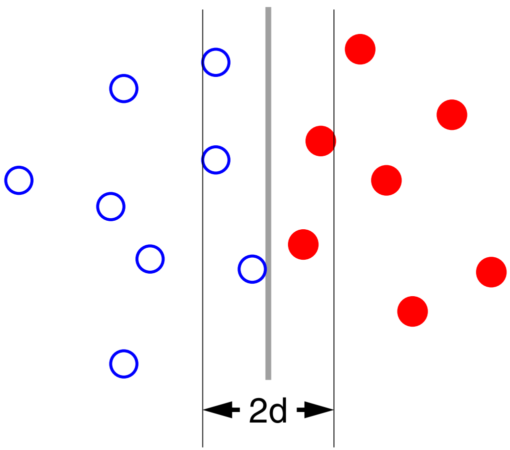
CLOSEST PAIR OF POINTS



نزدیک‌ترین جفت نقاط
Closest Pair of Points

یک مجموعه‌ی P از n نقطه داده شده است.
دو نقطه‌ی p و q متعلق به P را بیابید که فاصله‌ی $d(p,q)$ آنها می‌نیمم باشد.

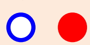
الگوریتم تقسیم و غلبه
Divide & Conquer Algorithm



- **مرحله ۲)** نزدیک‌ترین جفت نقاط را به‌صورت بازگشتی محاسبه می‌کنیم و می‌نیمم فاصله‌های d_l و d_r را در

$$P_l = \{p_1, p_2, \dots, p_{n/2}\}$$
$$P_r = \{p_{n/2+1}, \dots, p_n\}$$

می‌یابیم.
سپس، نزدیک‌ترین جفت و نزدیک‌ترین فاصله در نوار مرکزی به عرض $2d$ را می‌یابیم که در آن $d = \min(d_l, d_r)$

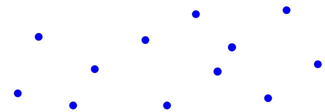


زیرا بدیهی است که جفت‌های زیر نمی‌توانند نزدیک‌تر از d باشند:

هندسه‌ی محاسباتی

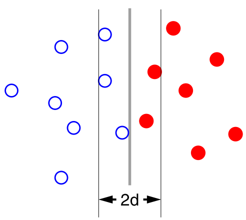
مسئله‌ی نزدیک‌ترین جفت نقاط

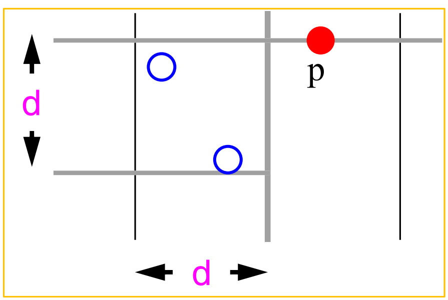
CLOSEST PAIR OF POINTS



نزدیک‌ترین جفت نقاط
Closest Pair of Points

یک مجموعه‌ی P از n نقطه داده شده است.
دو نقطه‌ی p و q متعلق به P را بیابید که فاصله‌ی $d(p,q)$ آنها می‌نیمم باشد.





الگوریتم تقسیم و غلبه
Divide & Conquer Algorithm

- **مرحله ۲) زیرمسئله:**
  برای هر نقطه‌ی p در این نوار، فاصله‌های $d(p,q)$ را محاسبه می‌کنیم که در آن p و q دارای رنگ‌های متفاوت هستند و

$$y(p) - d \le y(q) \le y(p)$$

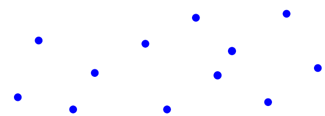
بنابراین، بیش از چهار مقایسه برای هر نقطه لازم نیست!

Prepared by Kazim Fouladi | Fall 2016 | 2nd Edition

هندسه‌ی محاسباتی

مسئله‌ی نزدیک‌ترین جفت نقاط: تحلیل زمان اجرای الگوریتم تقسیم و غلبه

CLOSEST PAIR OF POINTS



نزدیک‌ترین جفت نقاط

Closest Pair of Points

یک مجموعه‌ی P از n نقطه داده شده است.

دو نقطه‌ی p و q متعلق به P را بیابید که فاصله‌ی $d(p,q)$ آنها می‌نیمم باشد.

الگوریتم تقسیم و غلبه

Divide & Conquer Algorithm

با توجه به مرتب‌سازی n نقطه بر حسب مختصات y برای هر مرتبه:

$$\begin{cases} T(n) = 2T\left(\frac{n}{2}\right) + \Theta(n\log n) + \Theta(1) \\ T(1) = 1 \end{cases}$$

$$\Rightarrow T(n) = \Theta(n\log^2 n)$$

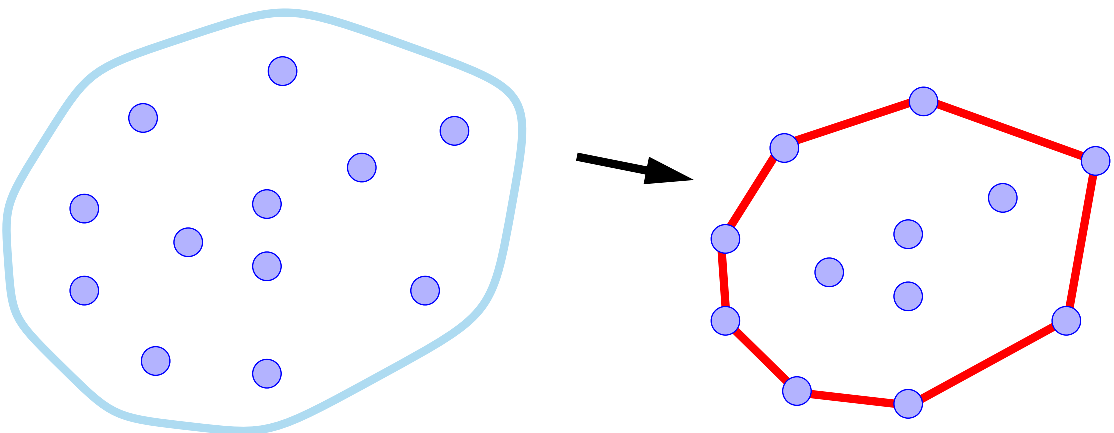
هندسه‌ی محاسباتی

مسئله‌ی پوش محدب

CONVEX HULL

پوش محدب
Convex Hull

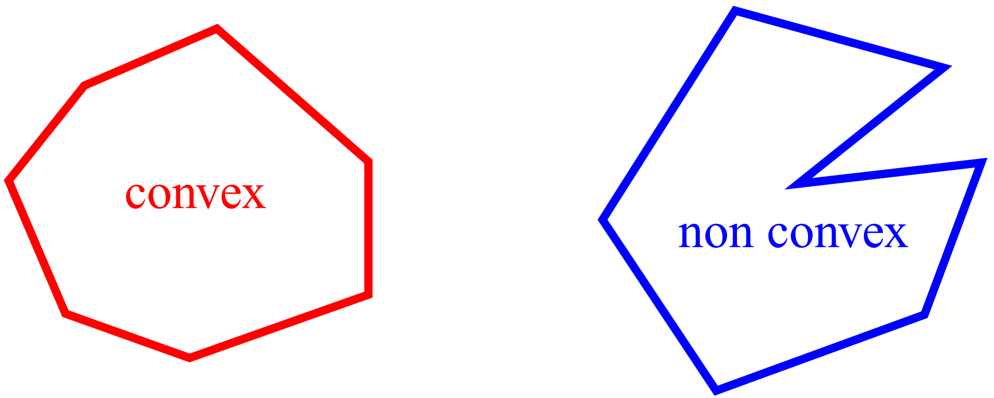
یک مجموعه‌ی P از n نقطه داده شده است.
پوش محدب، کوچک‌ترین چندضلعی محدب است که حاوی همه‌ی نقاط P می‌باشد.



هندسه‌ی محاسباتی

مسئله‌ی پوش محدب : چندضلعی محدب

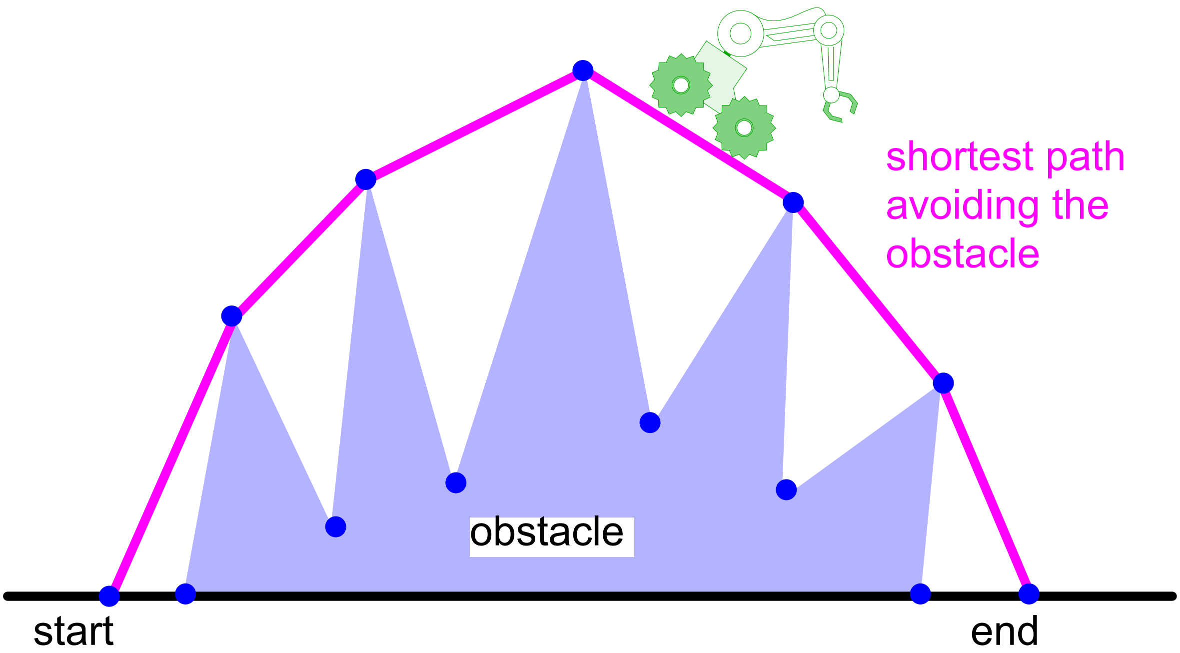
CONVEX HULL



هندسه‌ی محاسباتی

مسئله‌ی پوش محدب : کاربرد

CONVEX HULL



کوتاه‌ترین مسیر برای حرکت روبات به‌منظور اجتناب از موانع

هندسه‌ی محاسباتی

مسئله‌ی نمودار ورونوی

VORONOI DIAGRAM

نمودار ورونوی
Voronoi Diagram

یک مجموعه‌ی P از n نقطه داده شده است.
نواحی دارای فاصله‌ی نزدیک‌تر با هر یک از نقطه‌های داده‌شده را بیابید.