Computational Geometry

Tutorial #6 — Algorithm design and Plane partitions

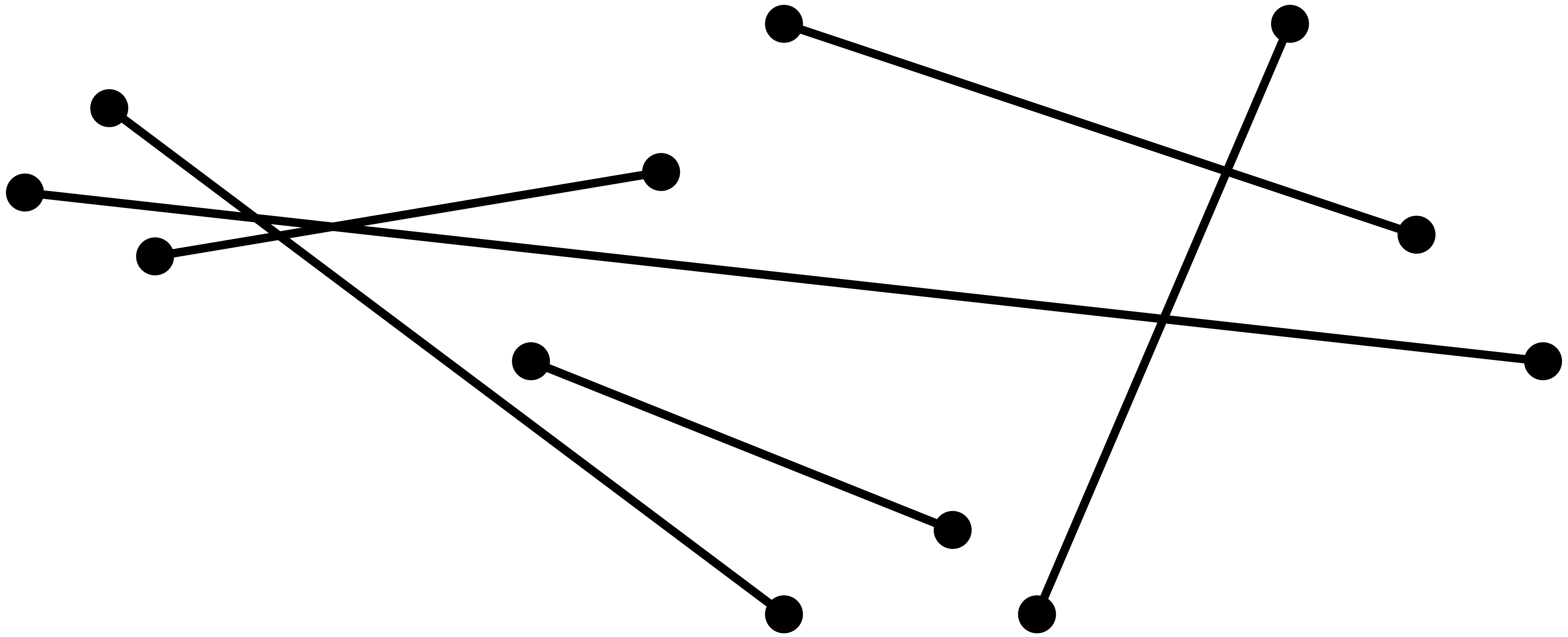
Institut für Betriebssysteme
und Rechnerverbund
Algorithmik

Peter Kramer | *January 23rd, 2025*

Algorithm Design

Peter Kramer | January 23rd, 2025

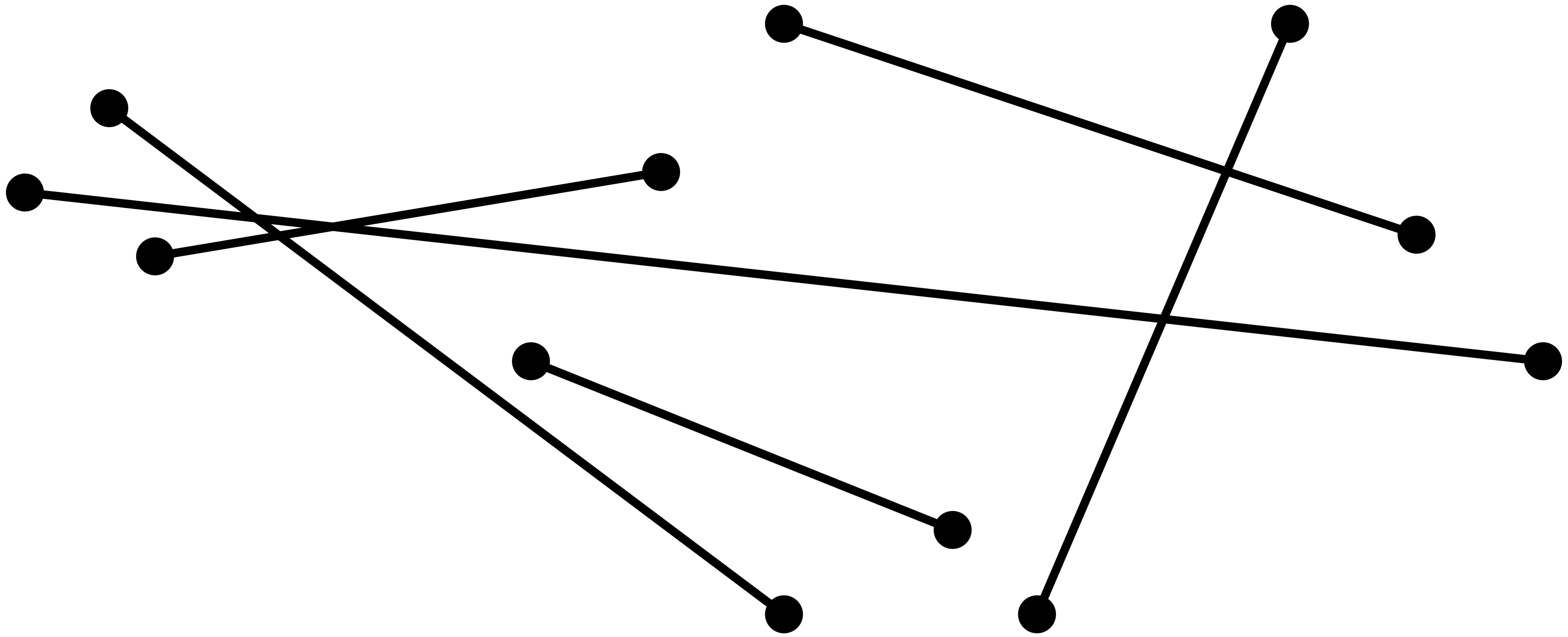
Intersections of line segments
Algorithm design



- **Given:** Endpoints $(p_1, q_1), \ldots, (p_n, q_n)$ of n line segments $\overline{p_i q_i}$ in the plane.
- **Wanted:** Intersections of segments, so …
    - … the number of intersections k, and
    - … the involved edges of each crossing.

Intersections of line segments

Algorithm design

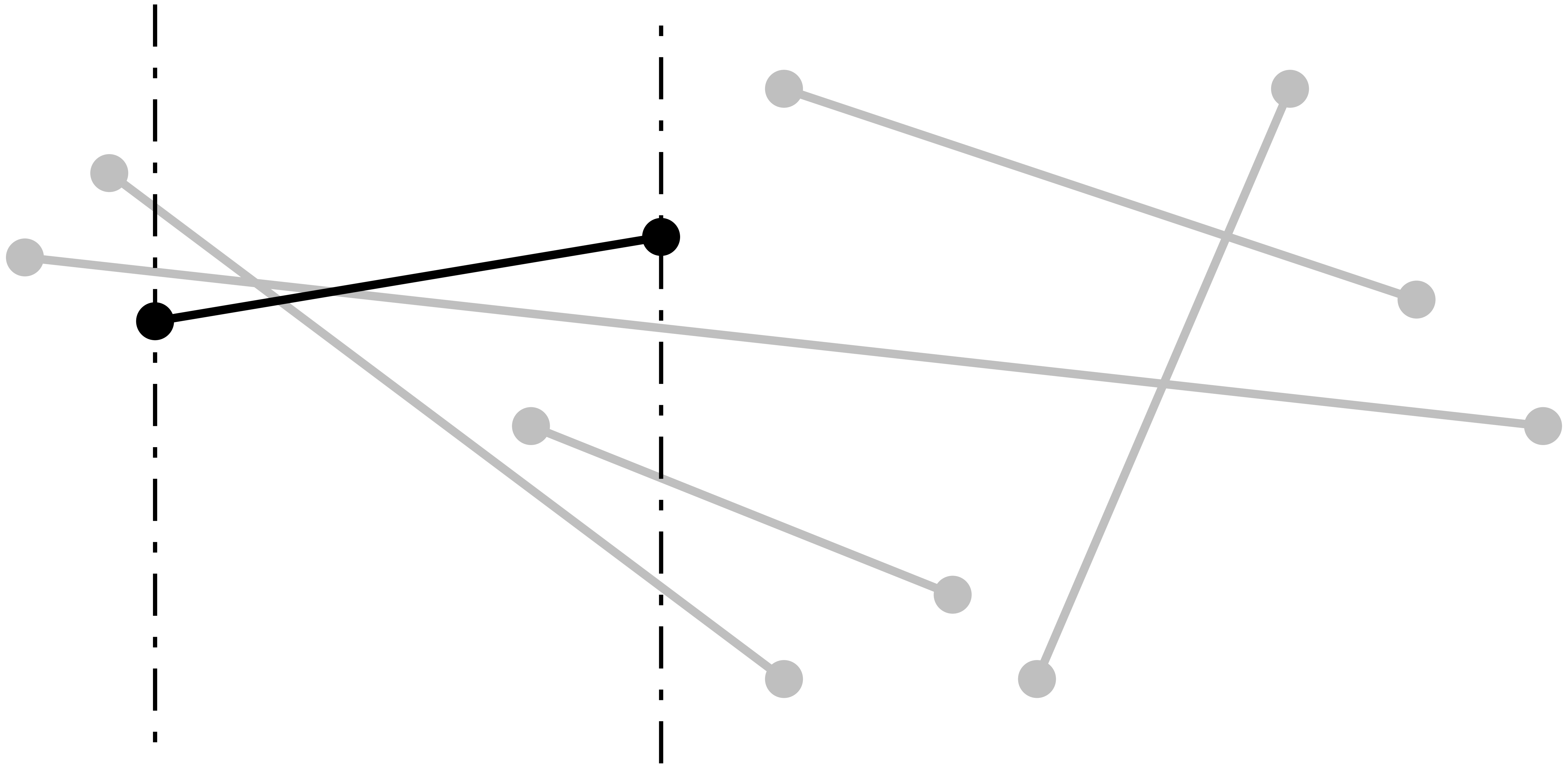


- **Given:** Endpoints $(p_1, q_1), \ldots, (p_n, q_n)$ of n line segments $\overline{p_i q_i}$ in the plane.
- **Wanted:** Intersections of segments, so …
  - … the number of intersections k, and
  - … the involved edges of each crossing.

- *When do two lines intersect (criteria)?*
  How many crossings can there be?
  Is there structure to this problem?

Intersections of line segments

Algorithm design

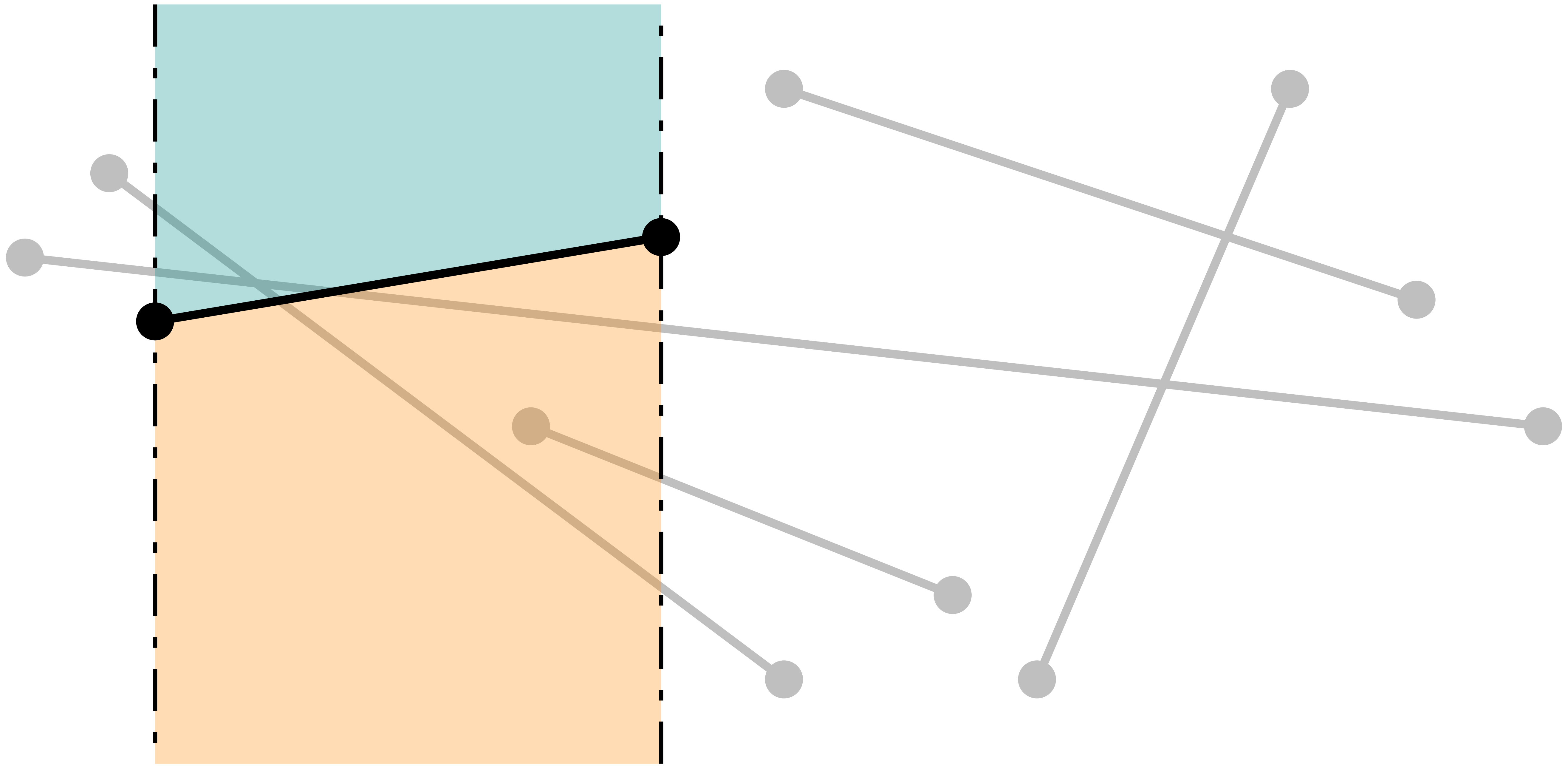


- **Given:** Endpoints $(p_1, q_1), \ldots, (p_n, q_n)$ of n line segments $\overline{p_i q_i}$ in the plane.
- **Wanted:** Intersections of segments, so …
  - … the number of intersections k, and
  - … the involved edges of each crossing.

- *When do two lines intersect (criteria)?*
  How many crossings can there be?
  Is there structure to this problem?

Intersections of line segments

Algorithm design

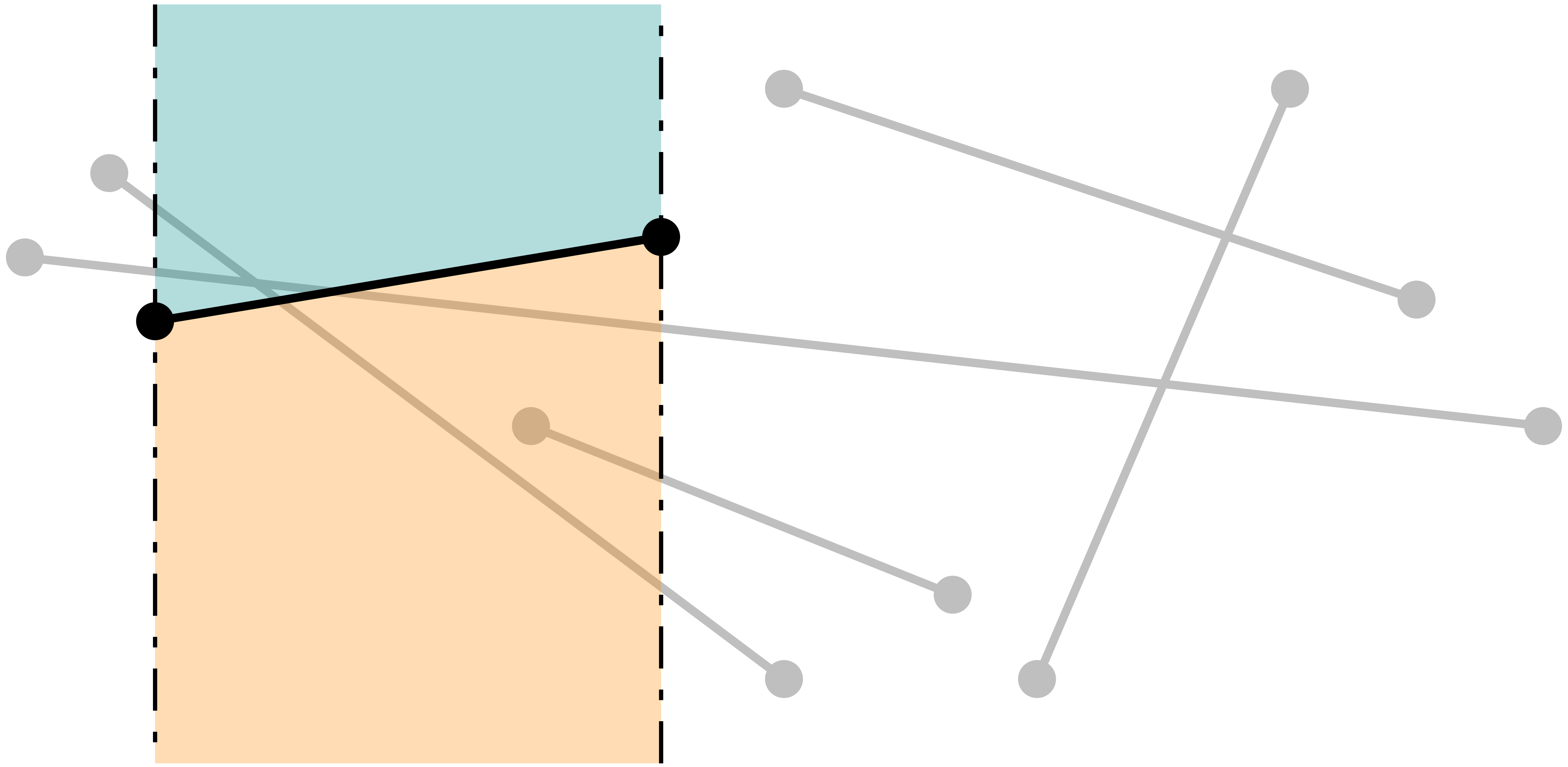


- **Given:** Endpoints $(p_1, q_1), \ldots, (p_n, q_n)$ of n line segments $\overline{p_i q_i}$ in the plane.
- **Wanted:** Intersections of segments, so …
  - … the number of intersections k, and
  - … the involved edges of each crossing.

- *When do two lines intersect (criteria)?*
  How many crossings can there be?
  Is there structure to this problem?

Intersections of line segments

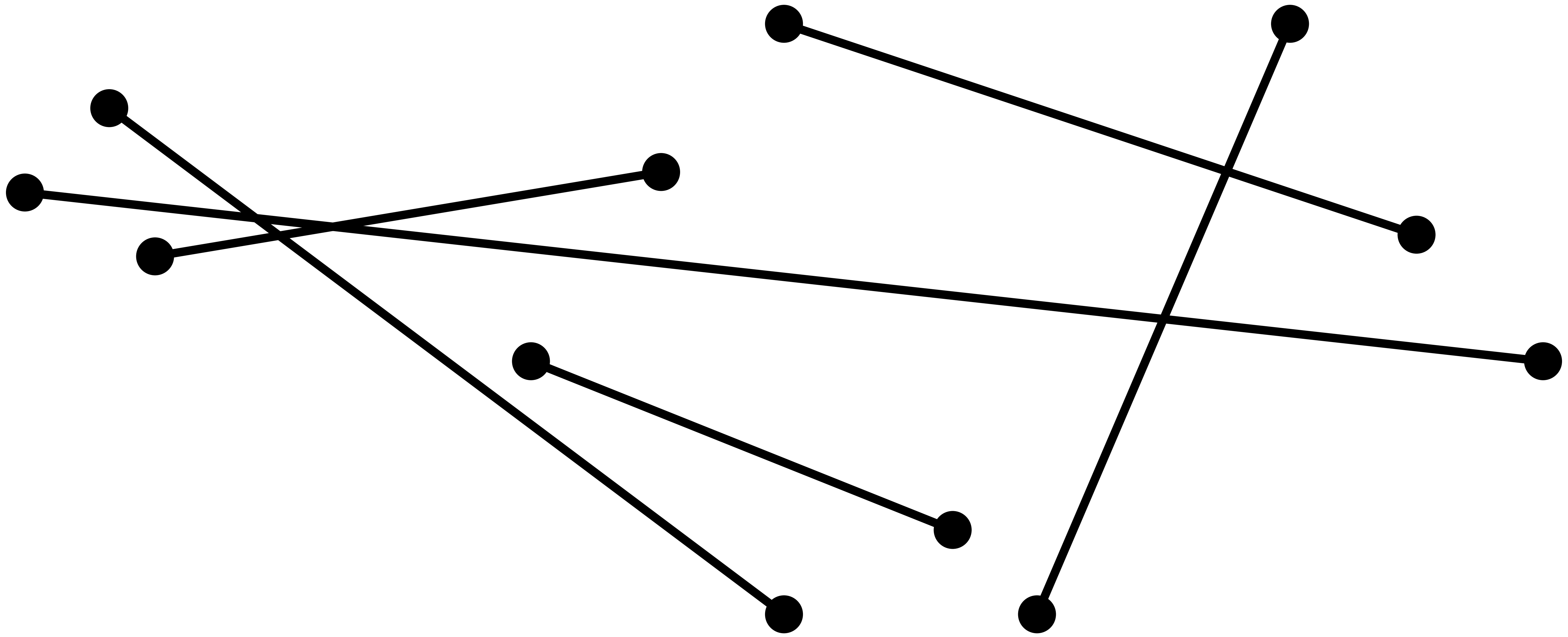
Algorithm design



- **Given:** Endpoints $(p_1, q_1), \ldots, (p_n, q_n)$ of n line segments $\overline{p_i q_i}$ in the plane.
- **Wanted:** Intersections of segments, so …
  - … the number of intersections k, and
  - … the involved edges of each crossing.
- *When do two lines intersect (criteria)?*
  How many crossings can there be?
  Is there structure to this problem?
- **Goal:** Construct a sweep-line algorithm that computes this in $\mathcal{O}((n+k)\log n)$ time.

Intersections of line segments

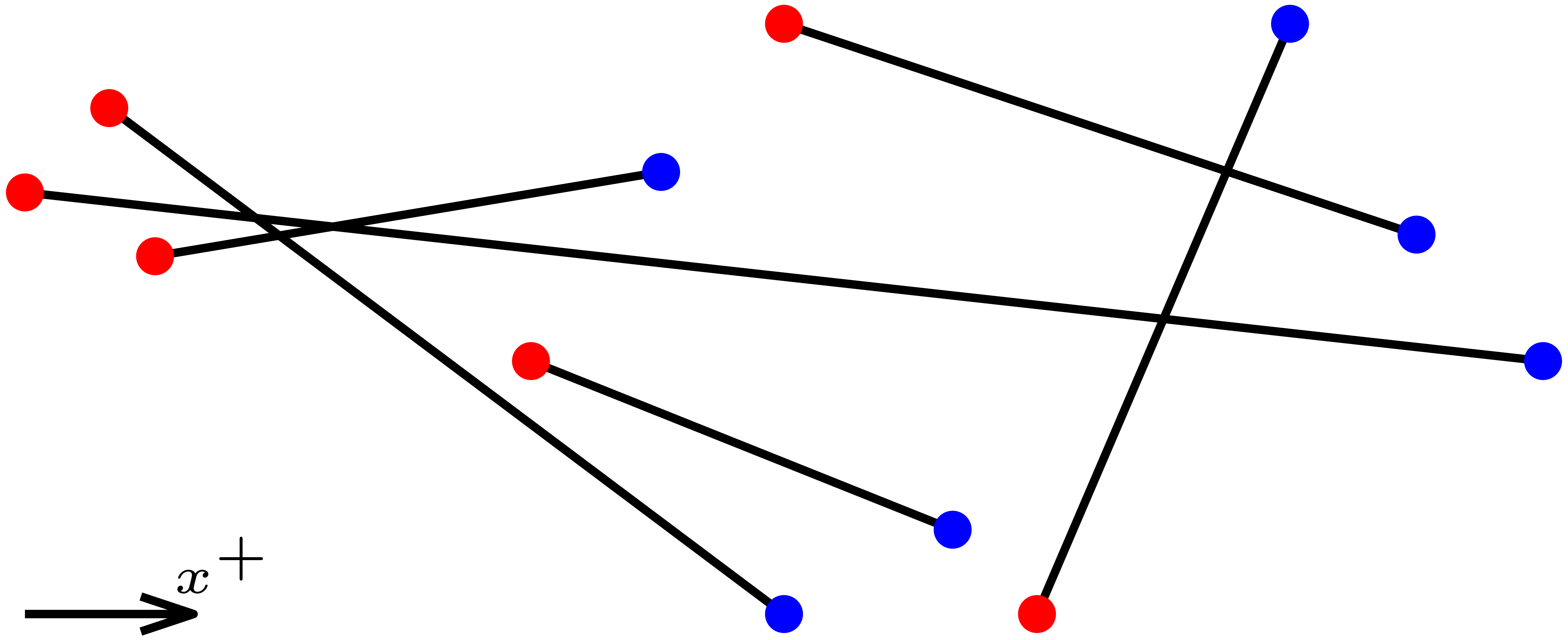
Sweep-Line Algorithm



Intersections of line segments

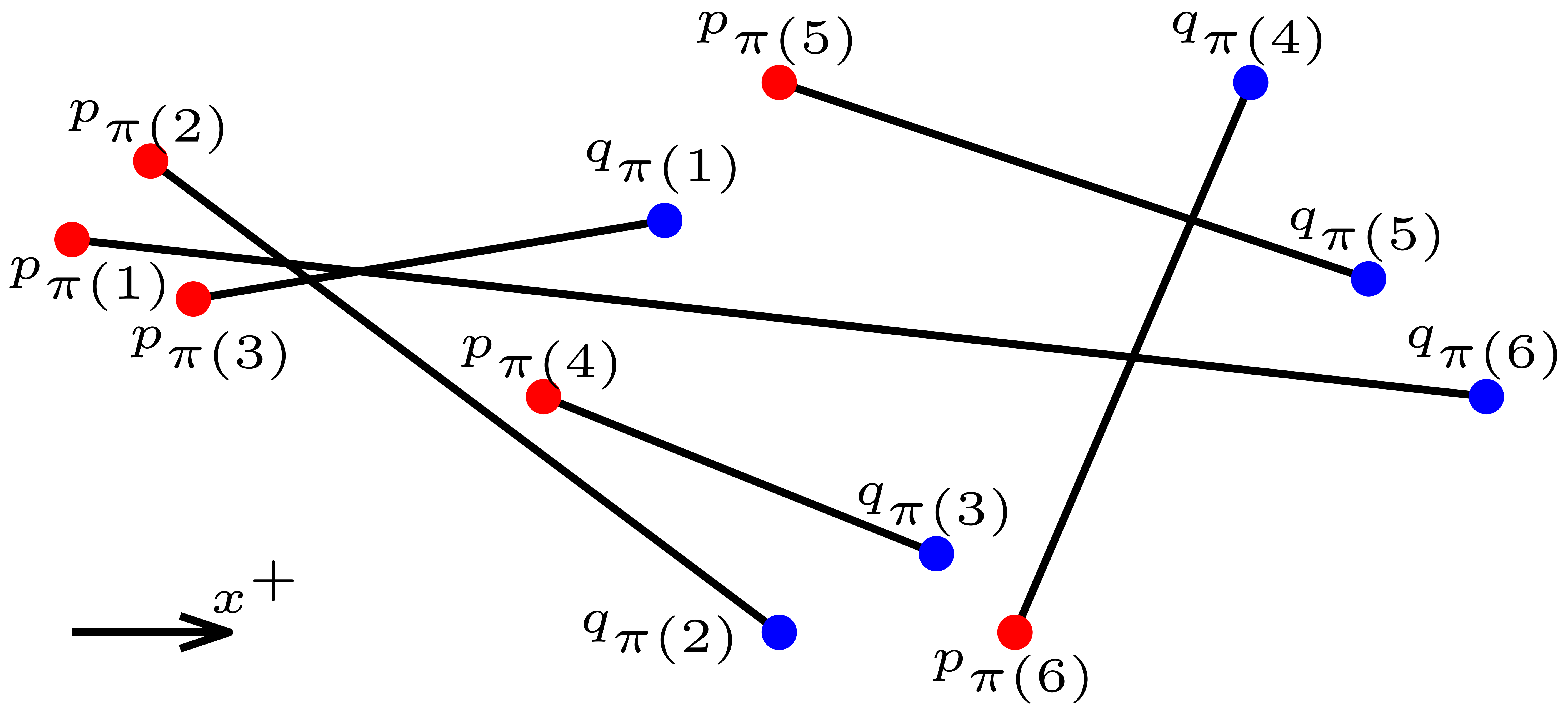
Sweep-Line Algorithm

(i) ‘Sort’ each segment such that $x(p_i) \leq x(q_i)$.



Intersections of line segments

Sweep-Line Algorithm

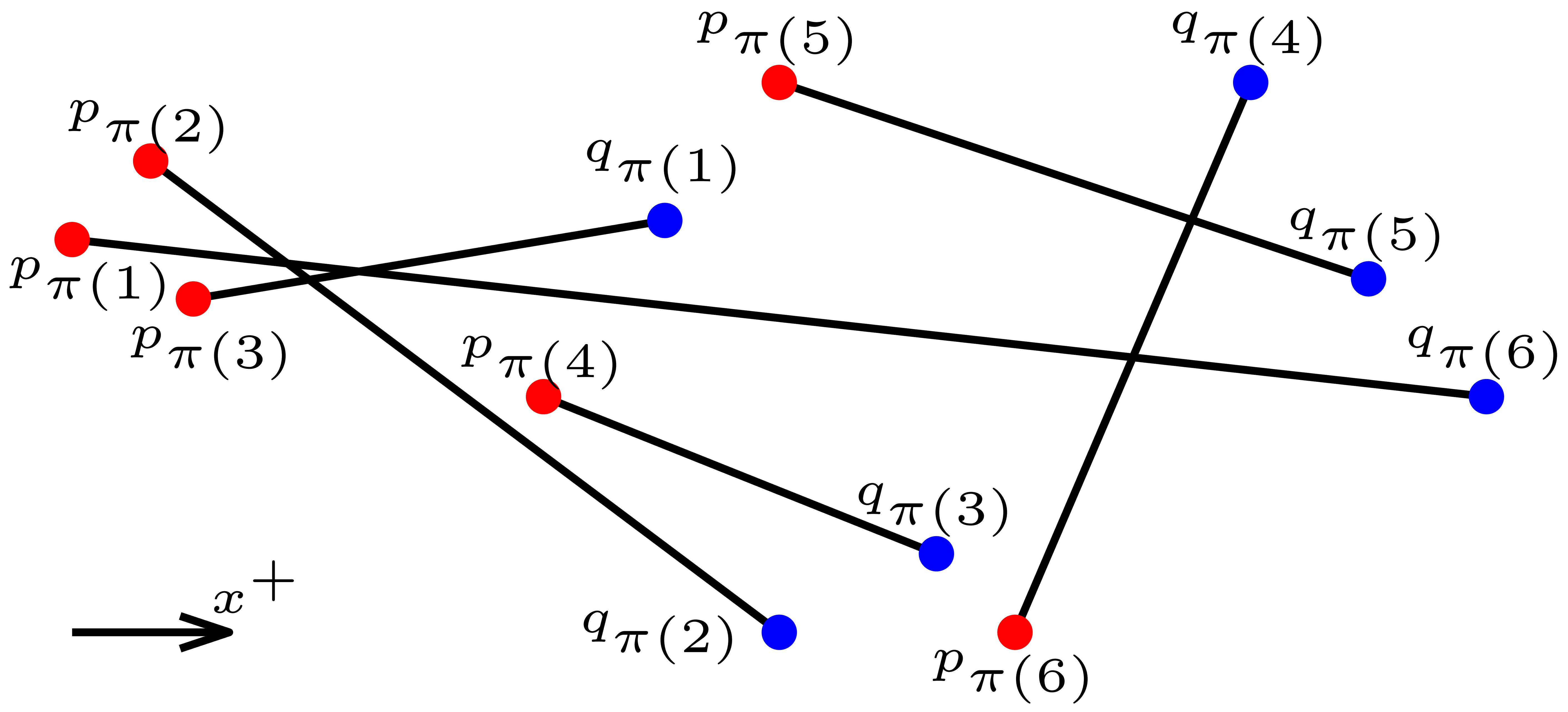


(i) ‘Sort’ each segment such that $x(p_i) \leq x(q_i)$.

(ii) Sort ps and qs by their x-coordinates.
Let $p_{\pi(i)}$ be the element rank i (ith lowest).

Intersections of line segments

Sweep-Line Algorithm



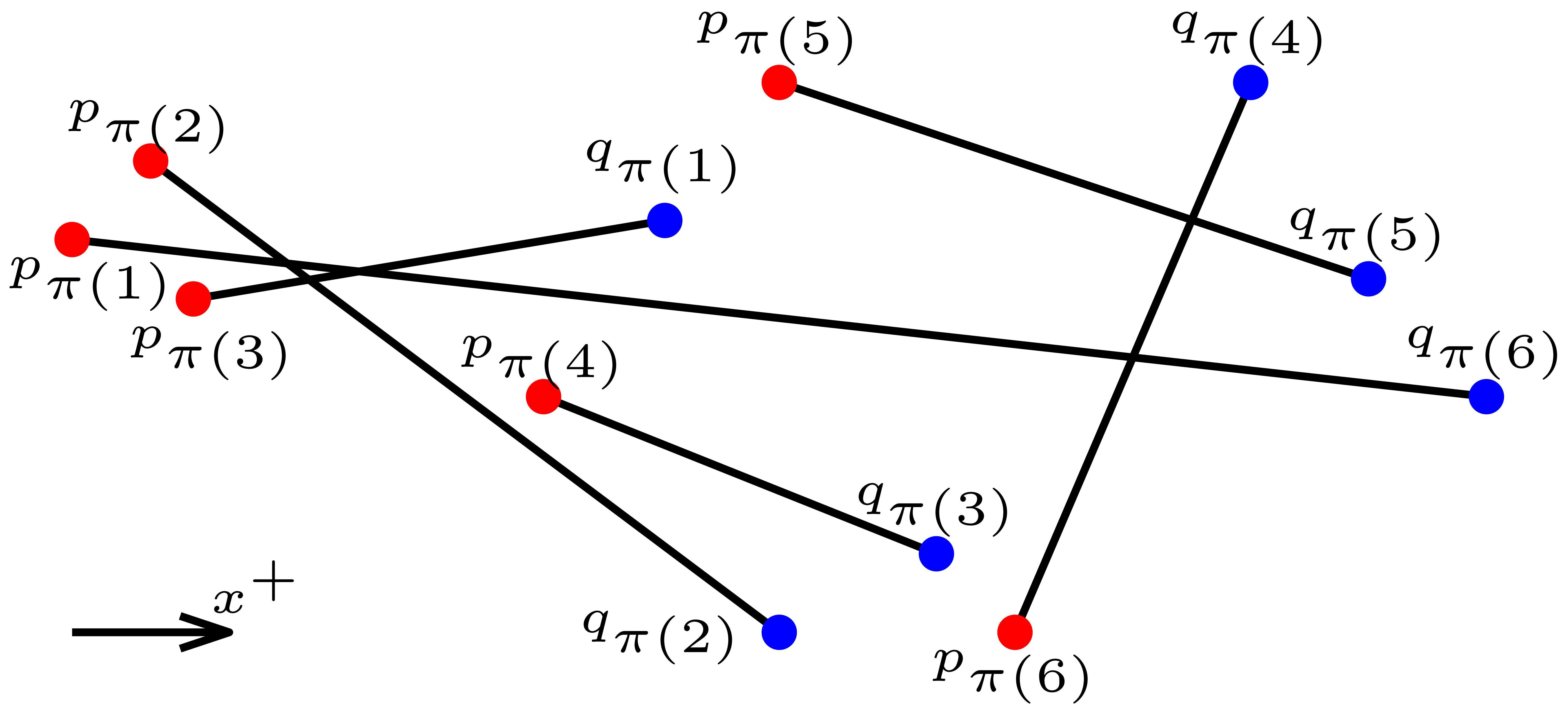
(i) ‘Sort’ each segment such that $x(p_i) \leq x(q_i)$.

(ii) Sort ps and qs by their x-coordinates. Let $p_{\pi(i)}$ be the element rank i (ith lowest).

(iii) Let $i, j = 1$, initialise a BST T (sweep state) and a priority queue C (crossings).

Intersections of line segments

Sweep-Line Algorithm



(i) 'Sort' each segment such that $x(p_i) \leq x(q_i)$.

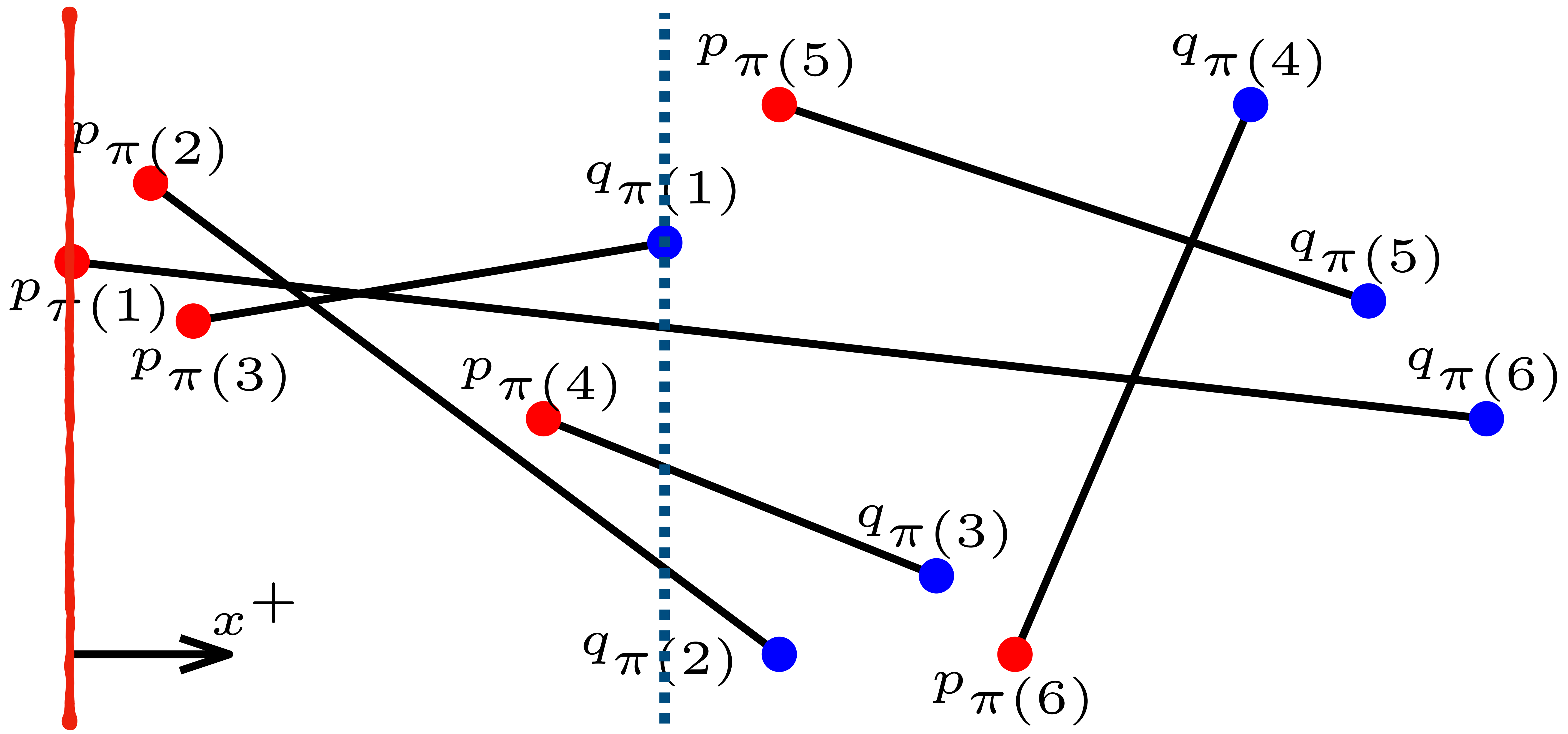
(ii) Sort ps and qs by their x-coordinates.
Let $p_{\pi(i)}$ be the element rank i (ith lowest).

(iii) Let $i, j = 1$, initialise a BST T (sweep state) and a priority queue C (crossings).

(iv) Sweep the vertical bisector at $\pi(i)$ along the x-axis, tracking intersecting segments in y-monotone order using the BST.
This order changes exactly at a crossings.

Intersections of line segments

Sweep-Line Algorithm



(i) 'Sort' each segment such that $x(p_i) \leq x(q_i)$.

(ii) Sort ps and qs by their x-coordinates. Let $p_{\pi(i)}$ be the element rank i (ith lowest).

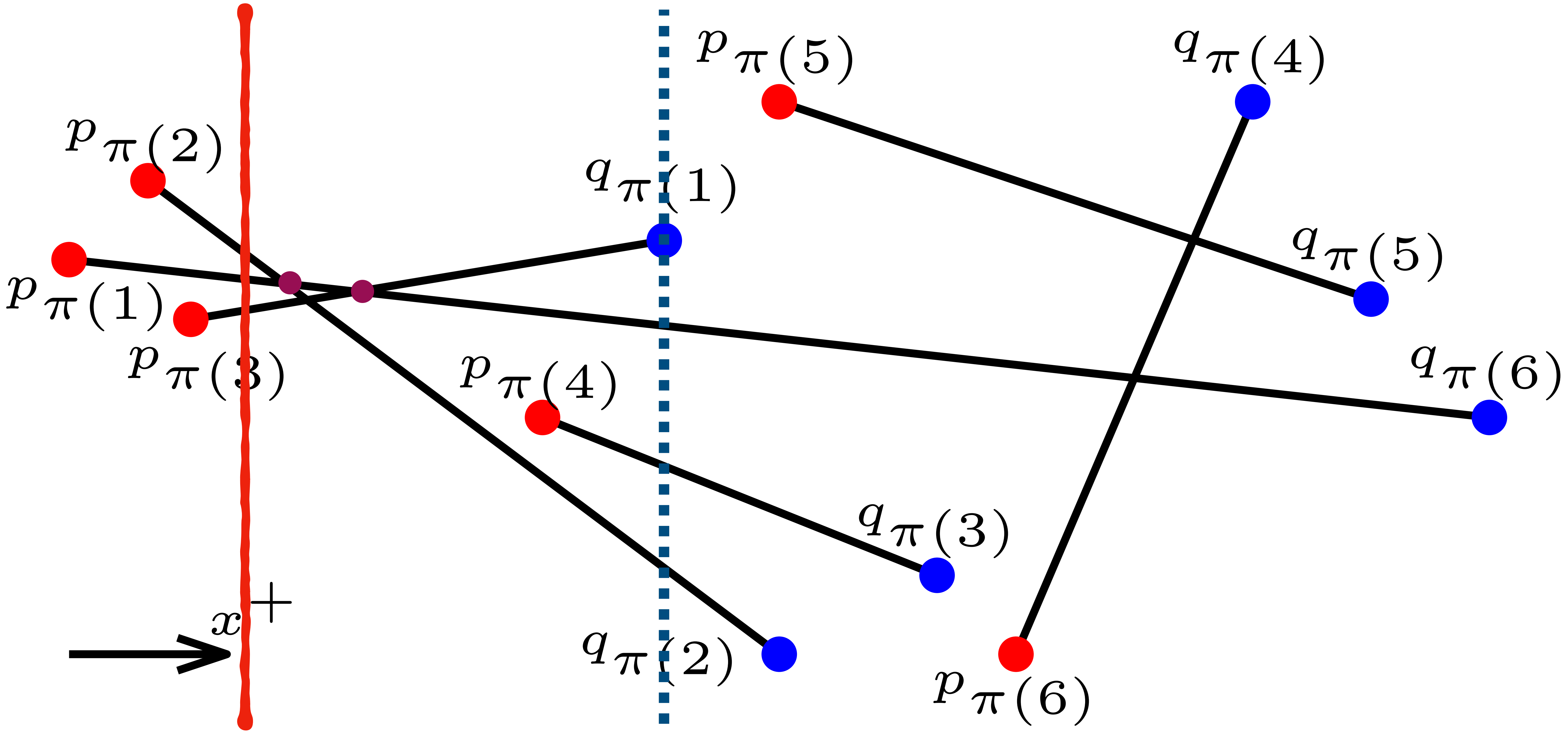
(iii) Let $i, j = 1$, initialise a BST T (sweep state) and a priority queue C (crossings).

(iv) Sweep the vertical bisector at $\pi(i)$ along the x-axis, tracking intersecting segments in y-monotone order using the BST. This order changes exactly at a crossings.

(board).

Intersections of line segments

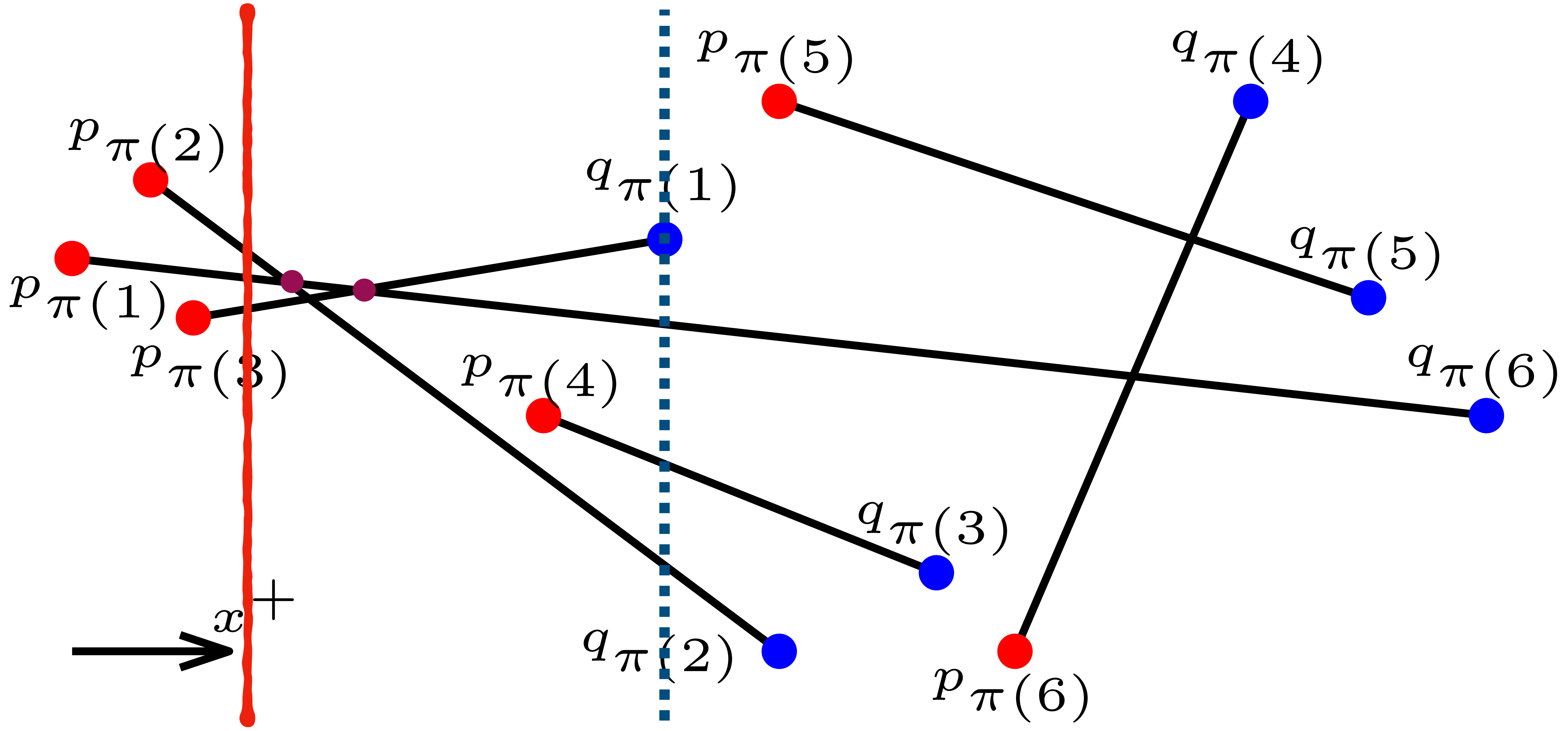
Sweep-Line Algorithm



Intersections of line segments

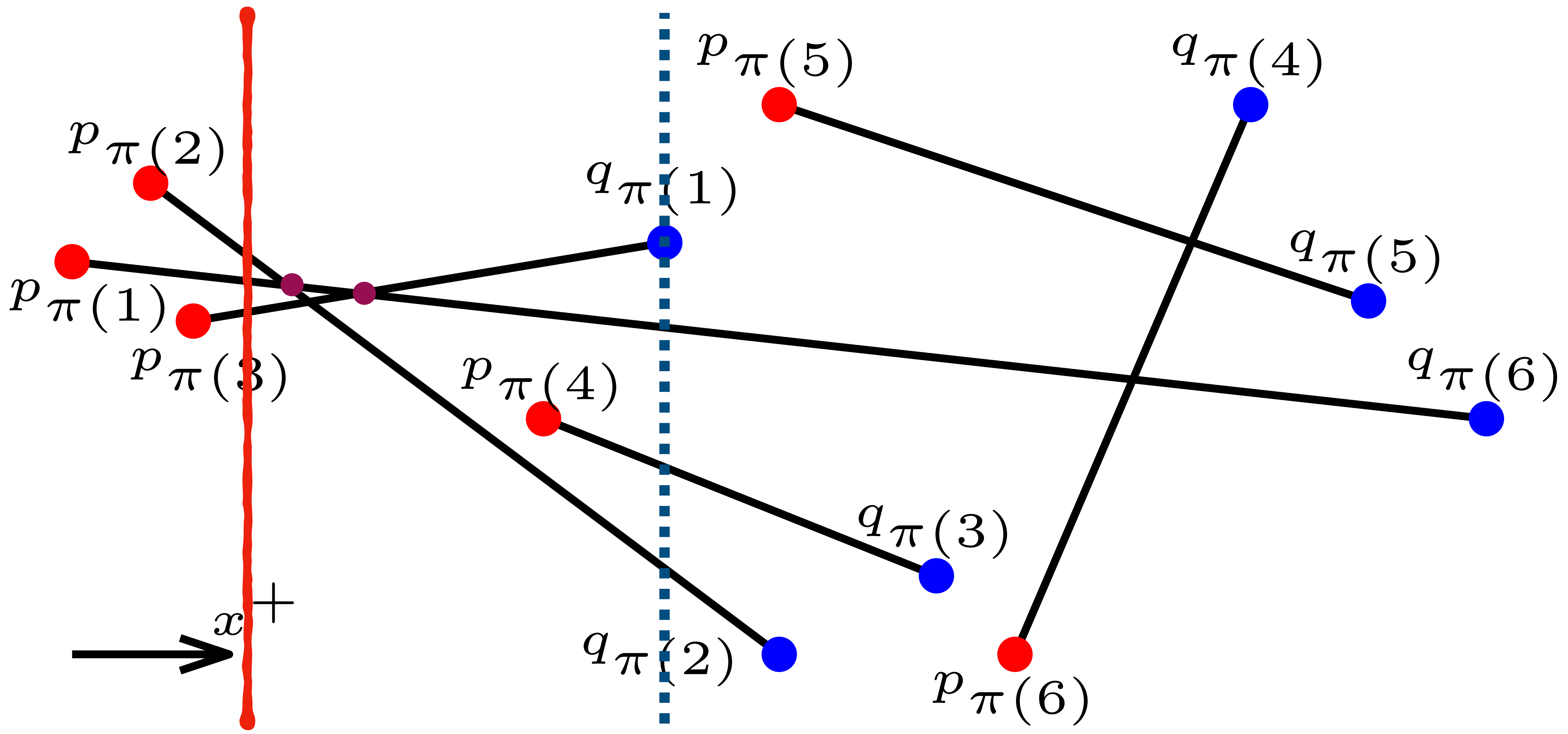
Sweep-Line Algorithm

(i) ‘Sort’ each segment such that $x(p_i) \leq x(q_i)$.



Intersections of line segments

Sweep-Line Algorithm

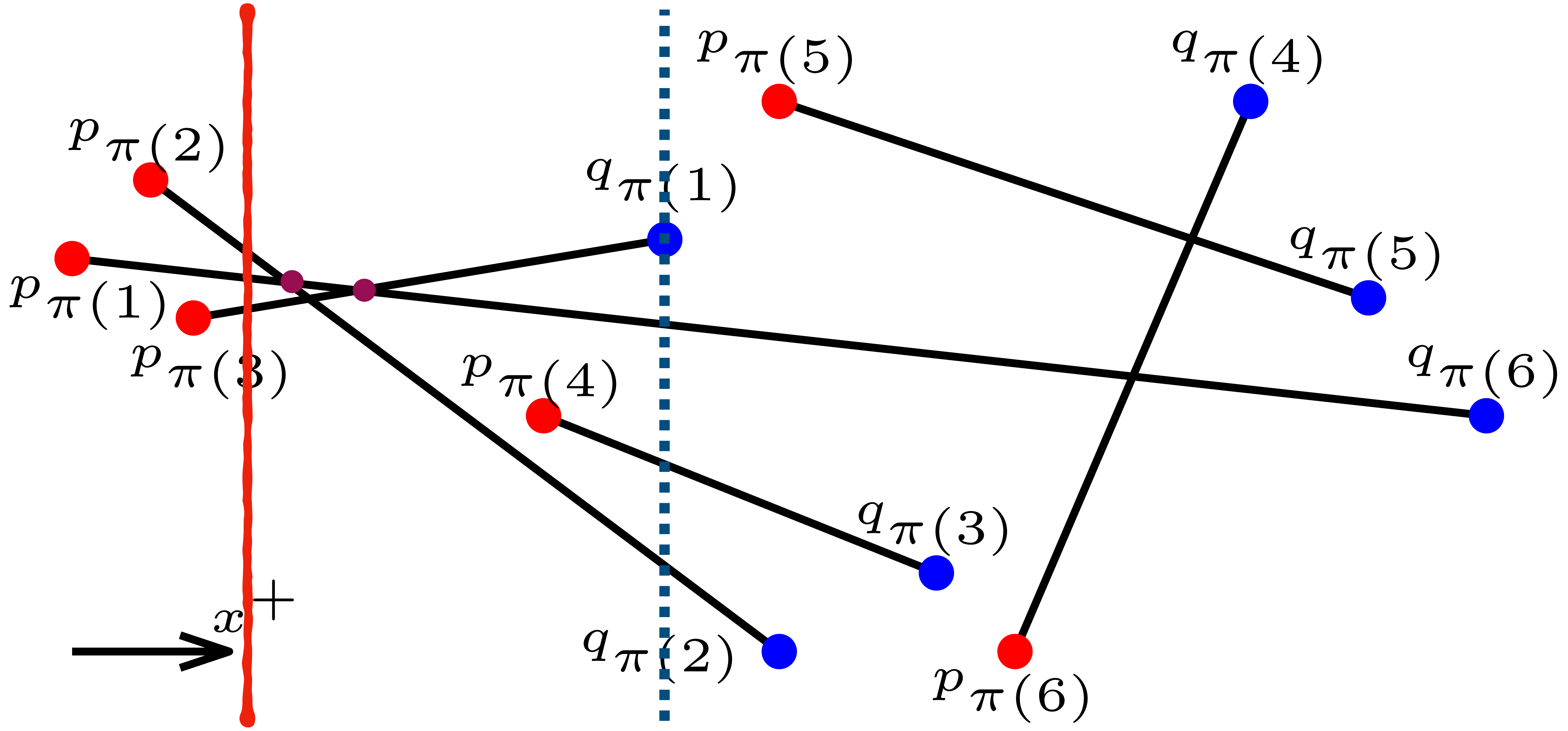


(i) 'Sort' each segment such that $x(p_i) \leq x(q_i)$.

(ii) Sort ps and qs by their x-coordinates.
Let $p_{\pi(i)}$ be the element rank i (ith lowest).

Intersections of line segments

Sweep-Line Algorithm



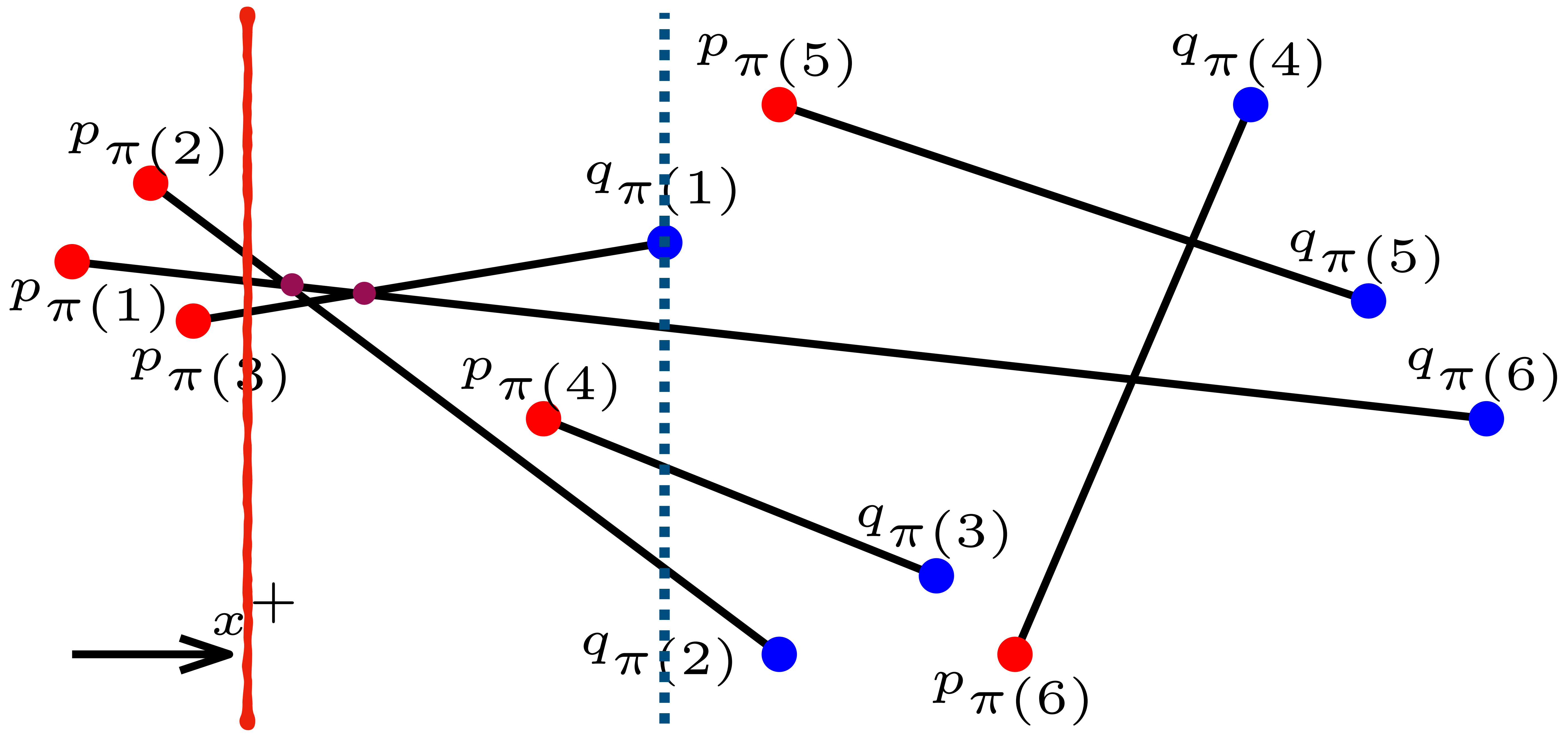
(i) 'Sort' each segment such that $x(p_i) \leq x(q_i)$.

(ii) Sort ps and qs by their x-coordinates. Let $p_{\pi(i)}$ be the element rank i (ith lowest).

(iii) Let $i, j = 1$, initialise a BST T (sweep state) and a priority queue C (crossings).

Intersections of line segments

Sweep-Line Algorithm



(i) 'Sort' each segment such that $x(p_i) \leq x(q_i)$.

(ii) Sort ps and qs by their x-coordinates. Let $p_{\pi(i)}$ be the element rank i (ith lowest).

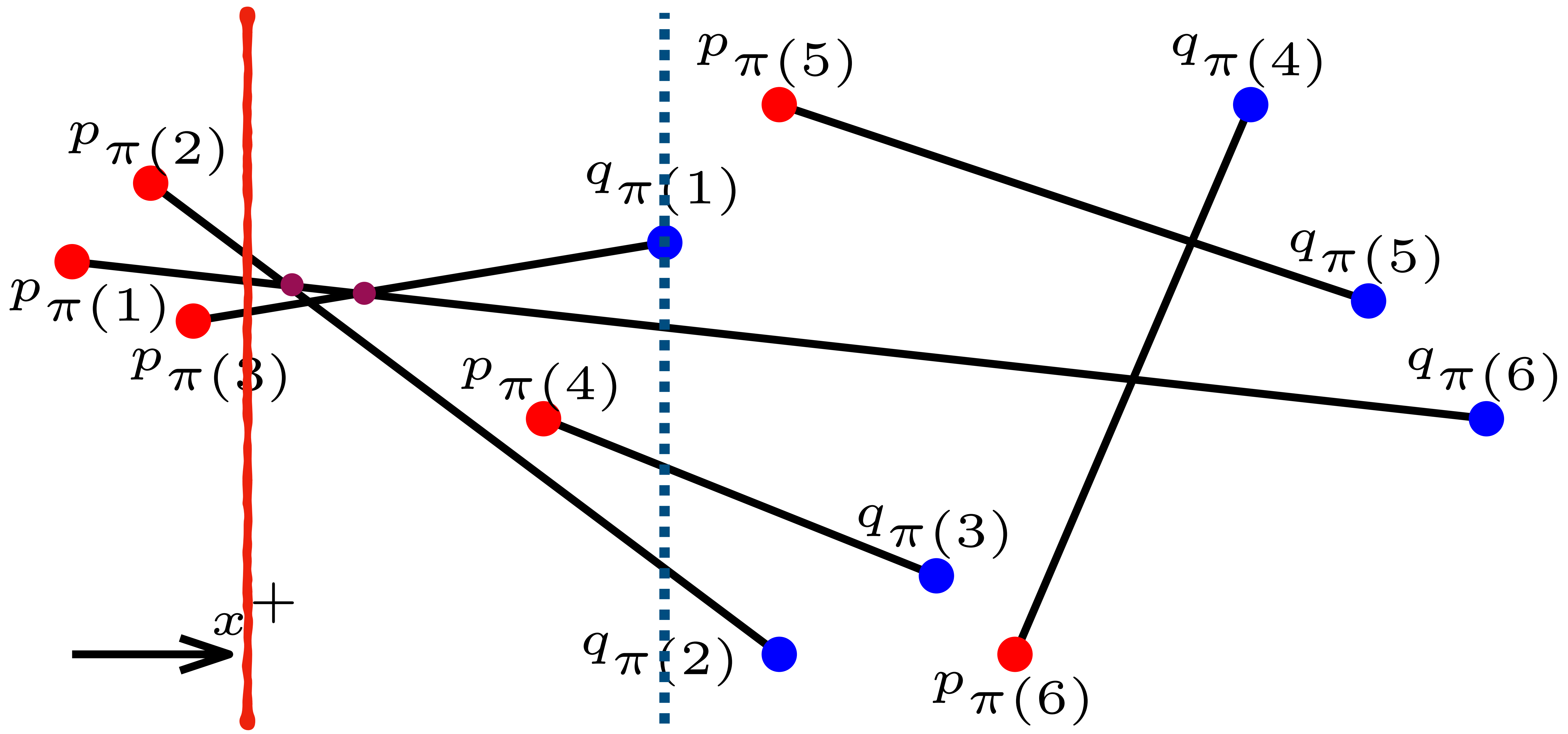
(iii) Let $i, j = 1$, initialise a BST T (sweep state) and a priority queue C (crossings).

Crossings:

Sweep state:

Intersections of line segments

Sweep-Line Algorithm



(i) ‘Sort’ each segment such that $x(p_i) \leq x(q_i)$.

(ii) Sort ps and qs by their x-coordinates. Let $p_{\pi(i)}$ be the element rank i (ith lowest).

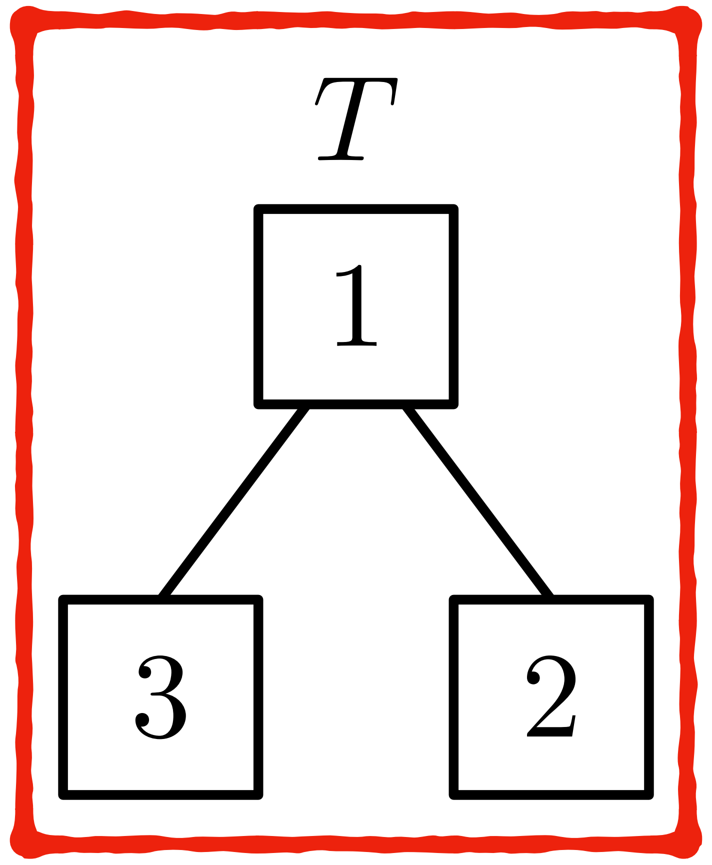
(iii) Let $i, j = 1$, initialise a BST T (sweep state) and a priority queue C (crossings).

Crossings:

Each $(c, a, b) \in C$ with

- $c \in \mathbb{R}^2$ position
- $a \in [1,n]$ segment A
- $b \in [1,n]$ segment B
- **Priority $x(c)$.**

Sweep state:



Geometric path planning

Peter Kramer | January 23rd, 2025

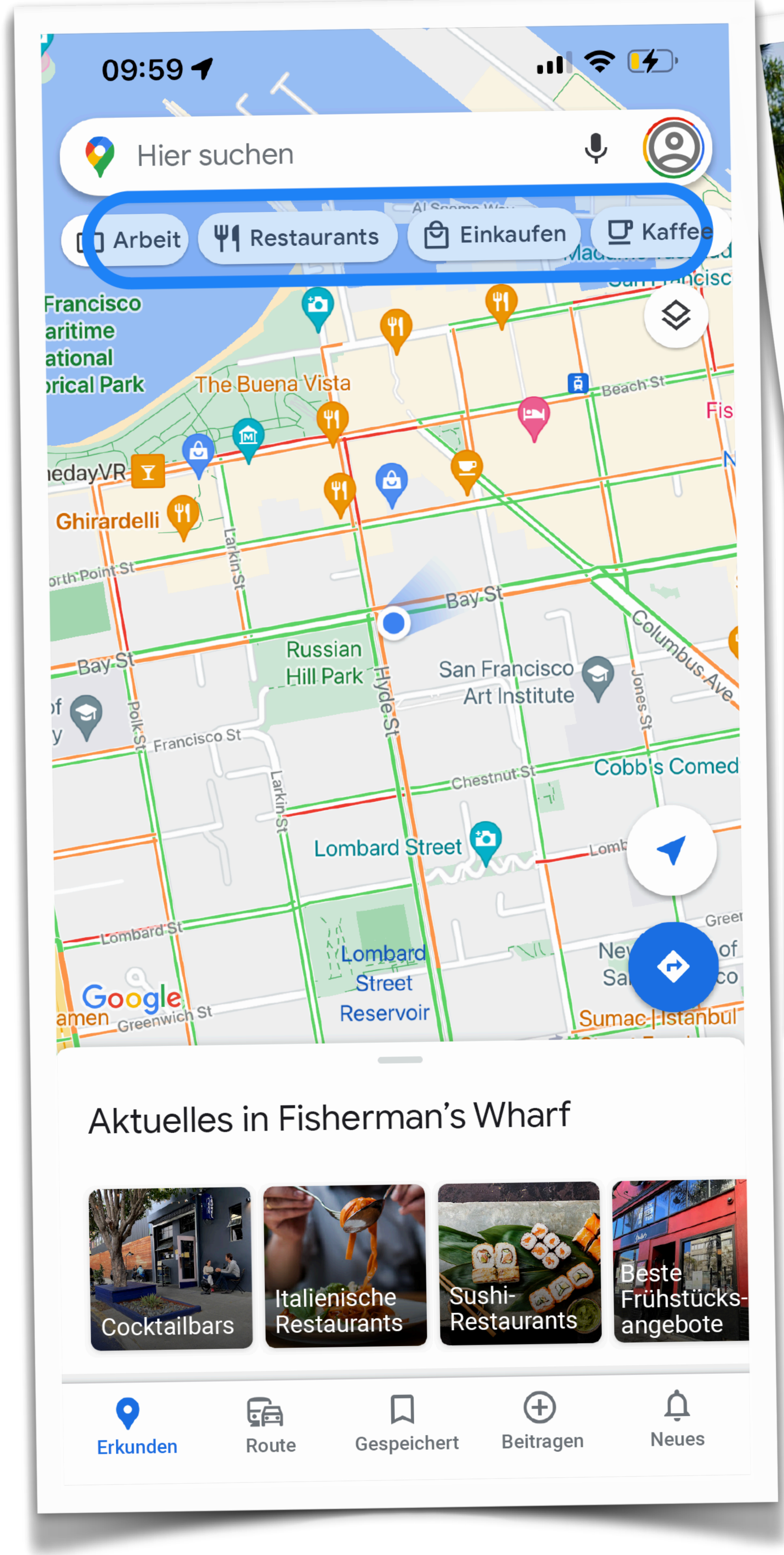
Point Location Problems

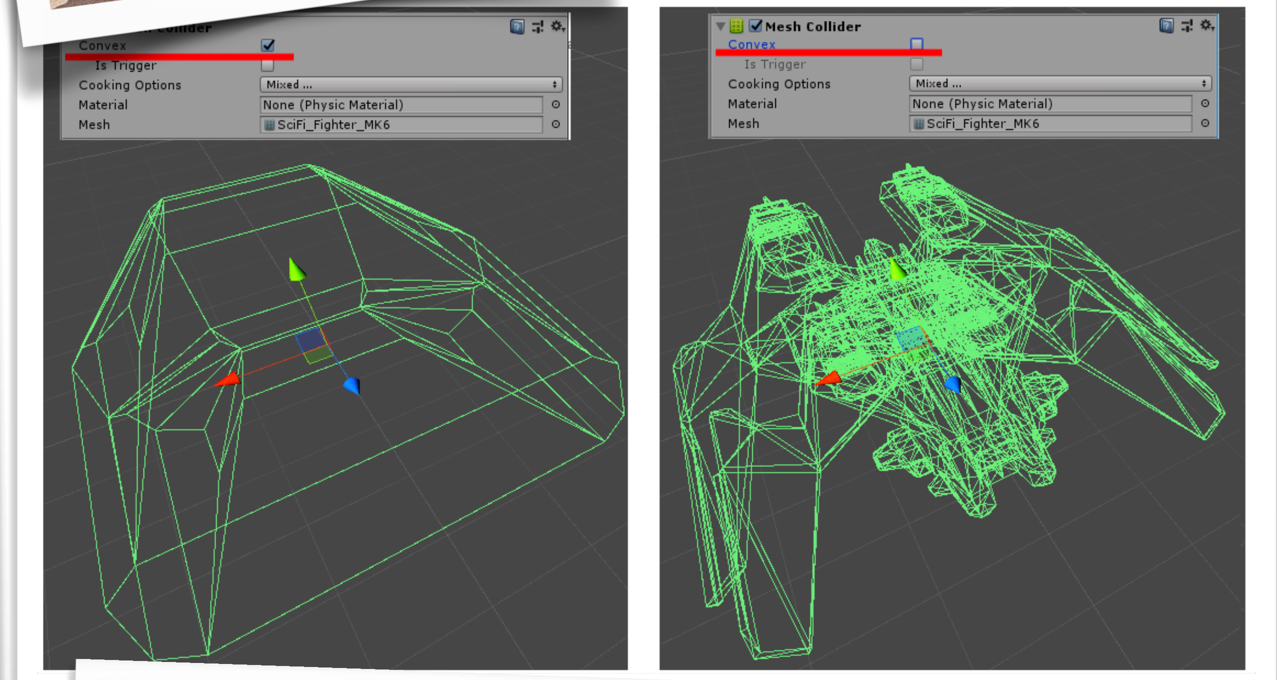
“Where am I?”

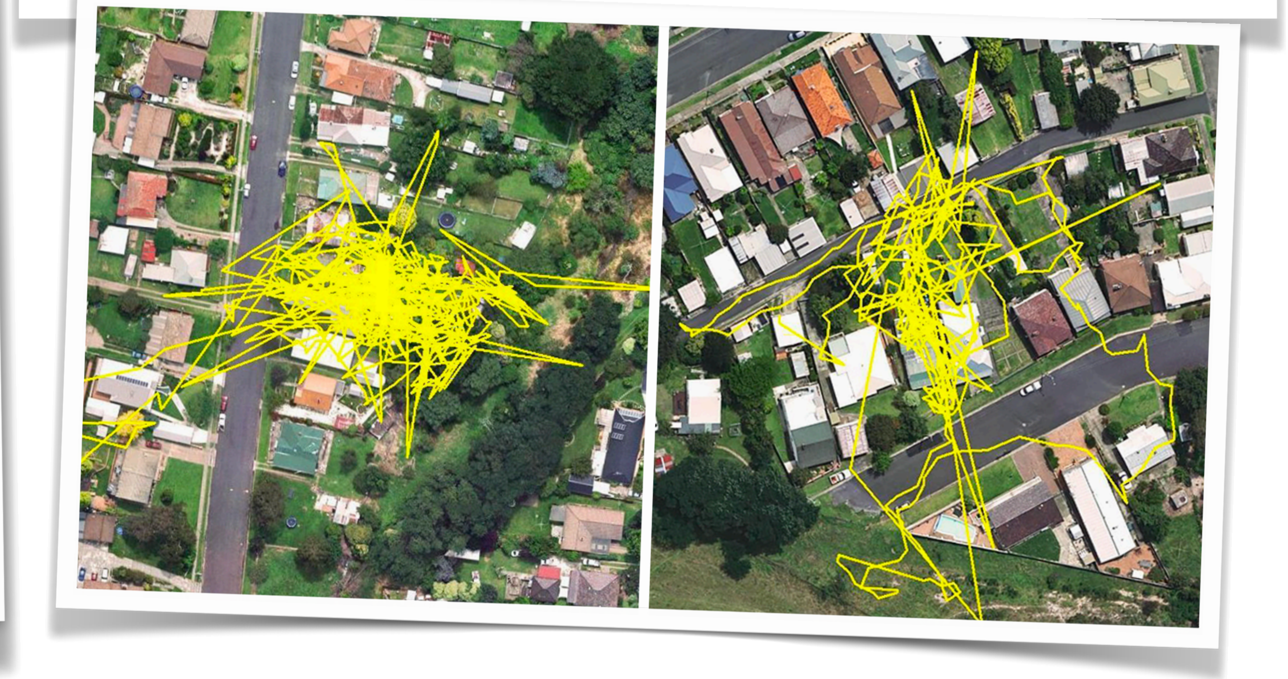
- Given geometric information such as a map in the plane, how can we decide **where** we are?

- *“In which country am I right now?”*
  “Can I leave this scooter here?”
  “Do these virtual objects collide?”

- ***“Is point p inside of region P?”***







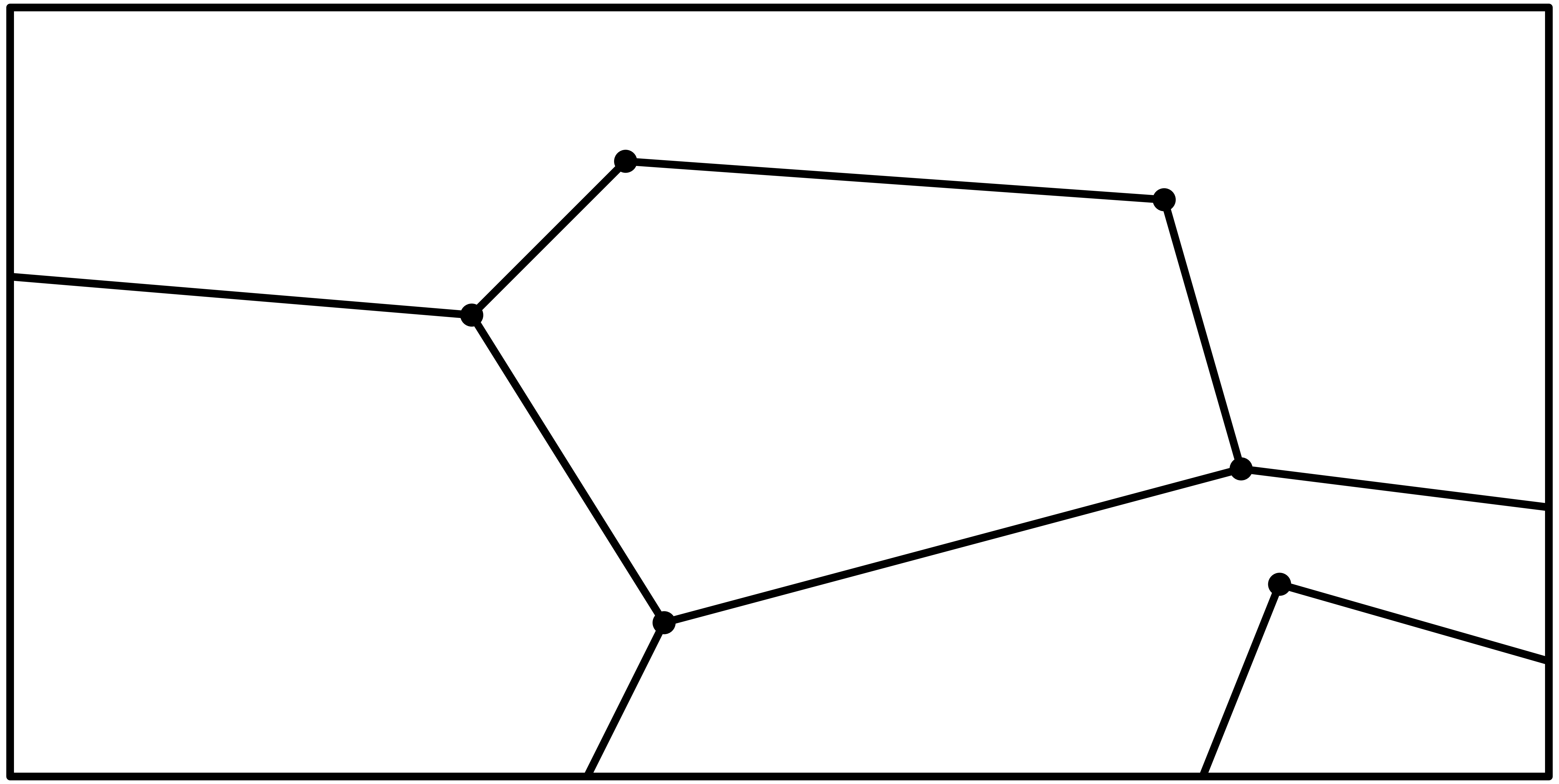
Applications: Geofencing, Navigation, Simulation Software, Outlier Detection, …

Geometric path planning

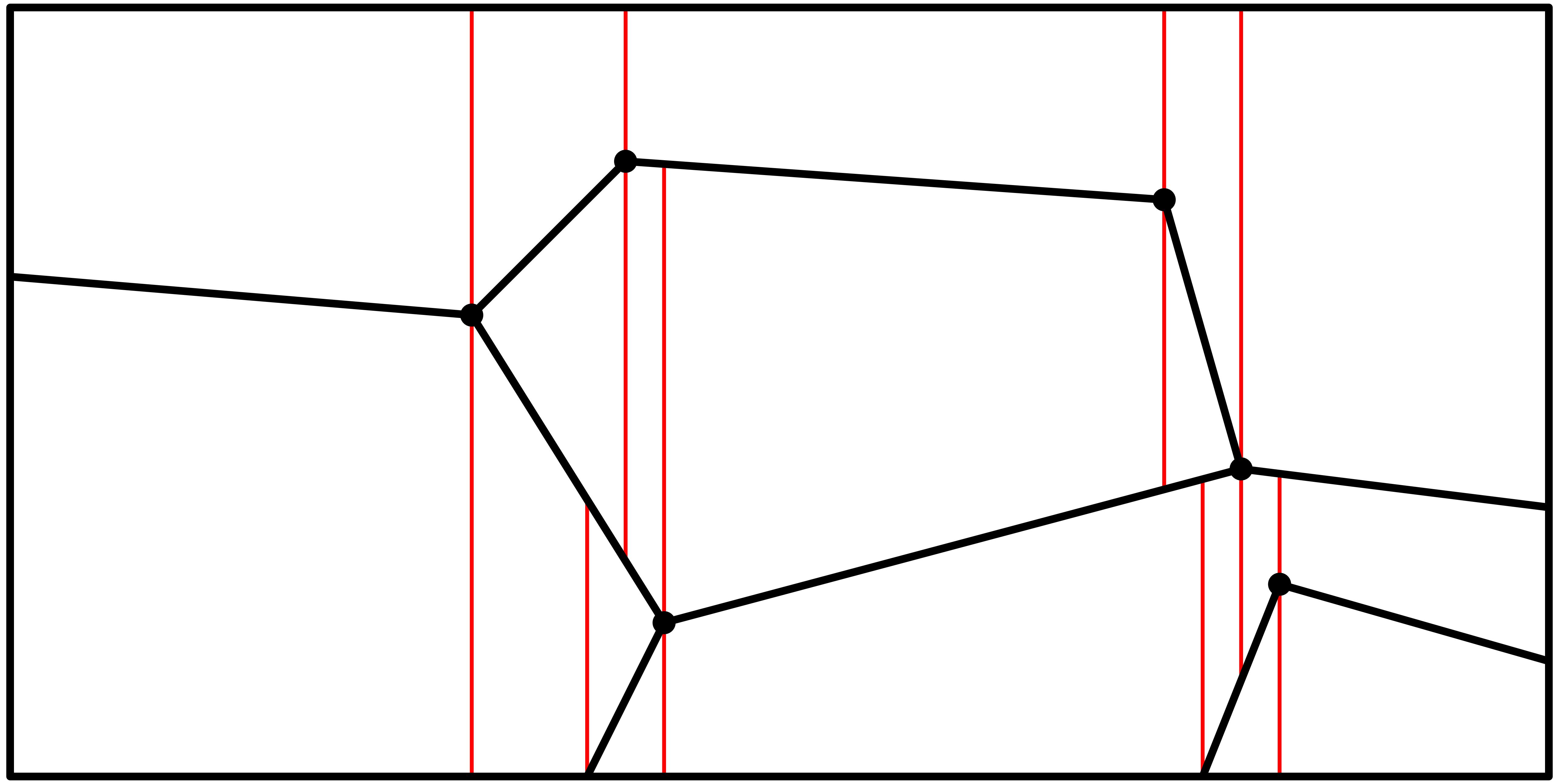
Trapezoidal maps

Peter Kramer | January 23rd, 2025

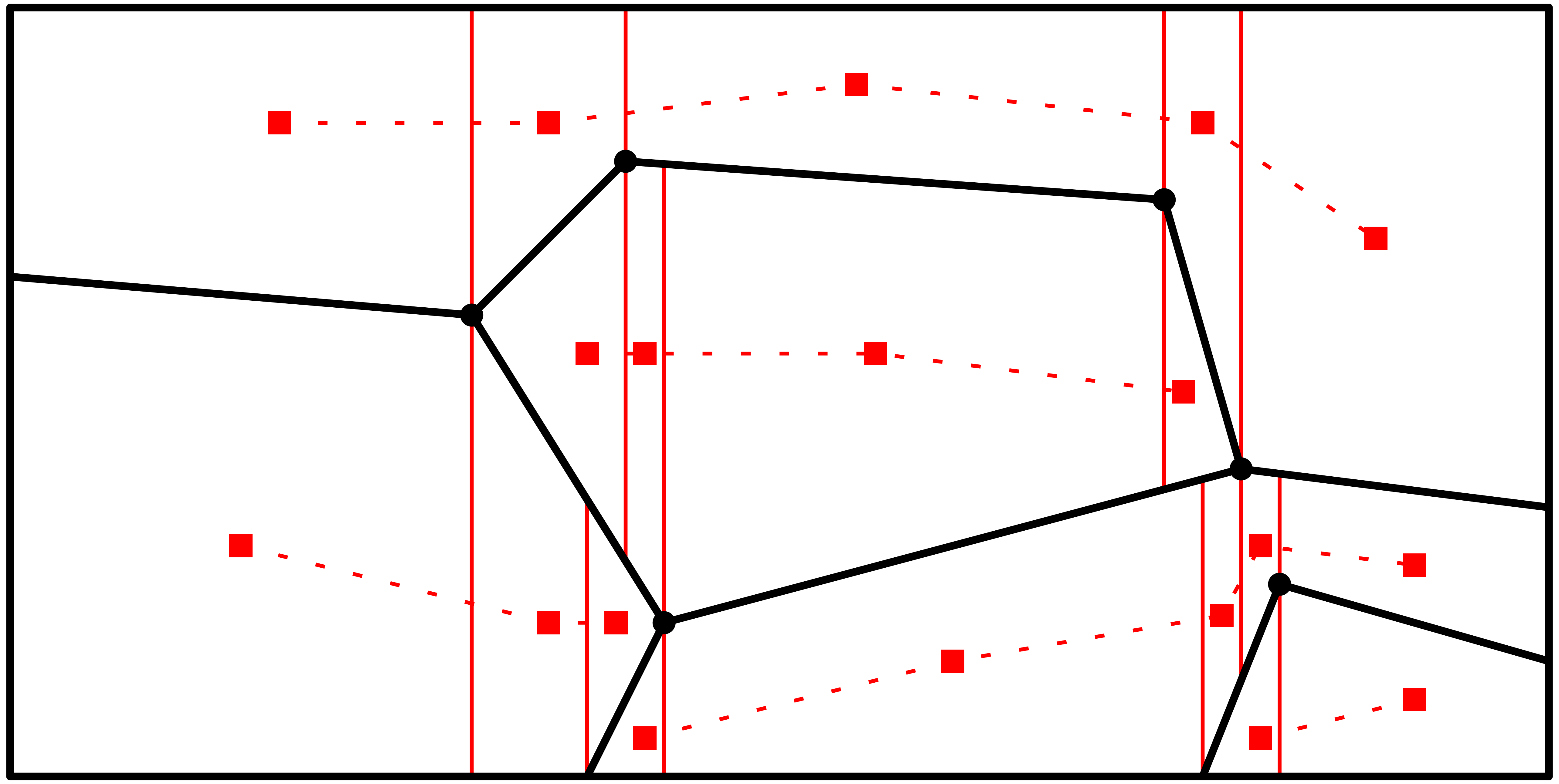
Trapezoidal maps



Trapezoidal maps

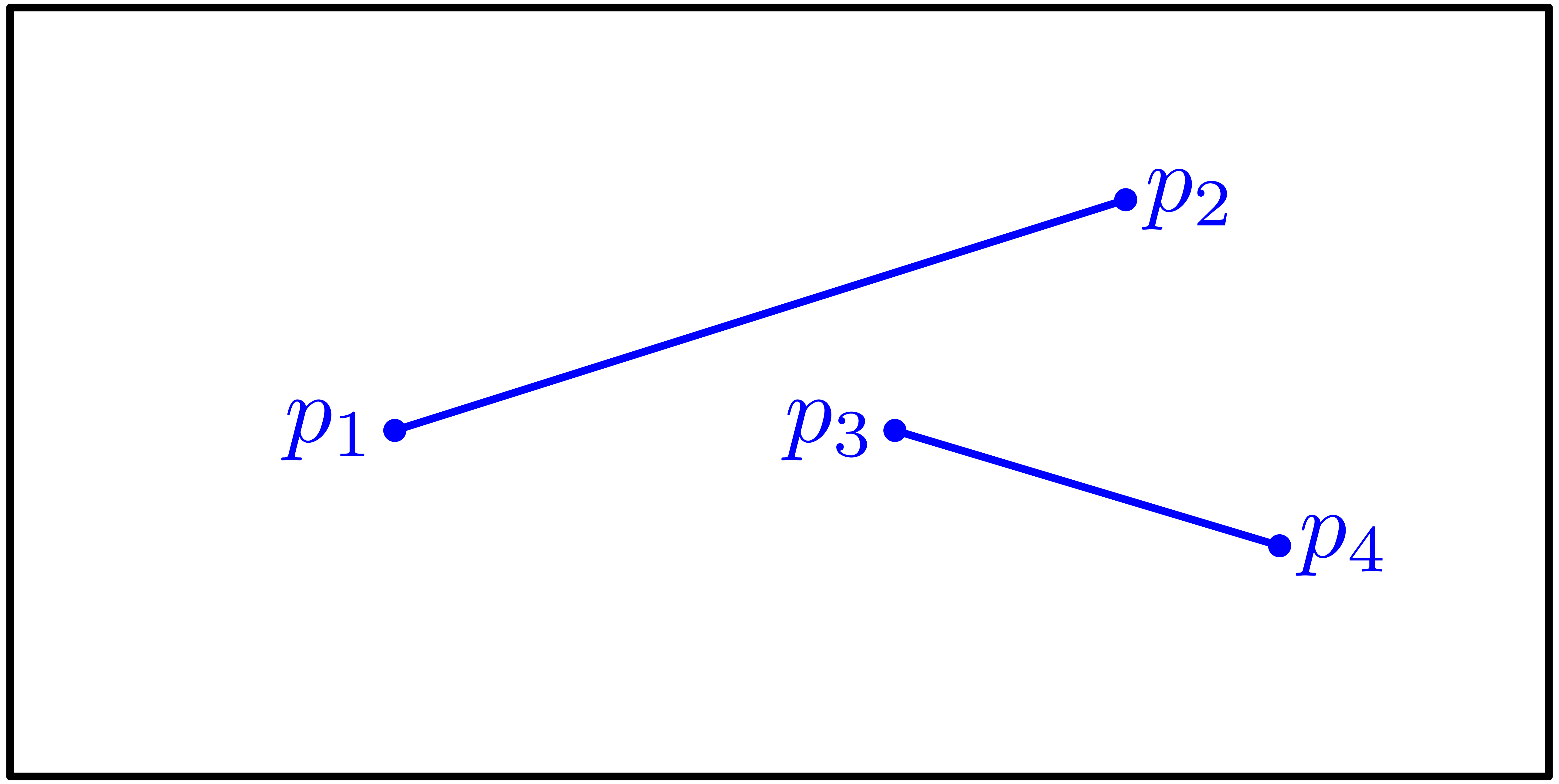


Trapezoidal maps

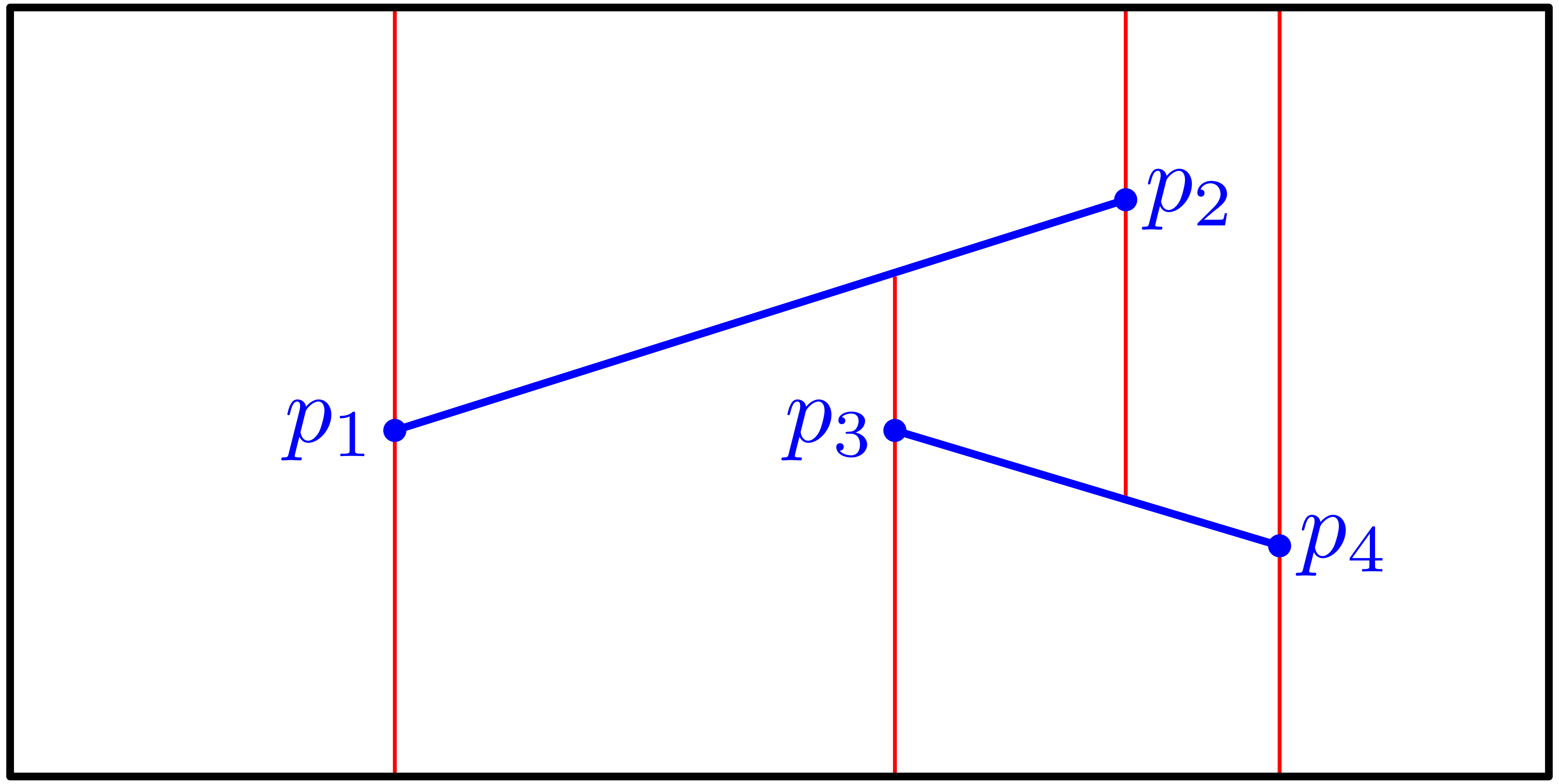


Trapezoidal maps

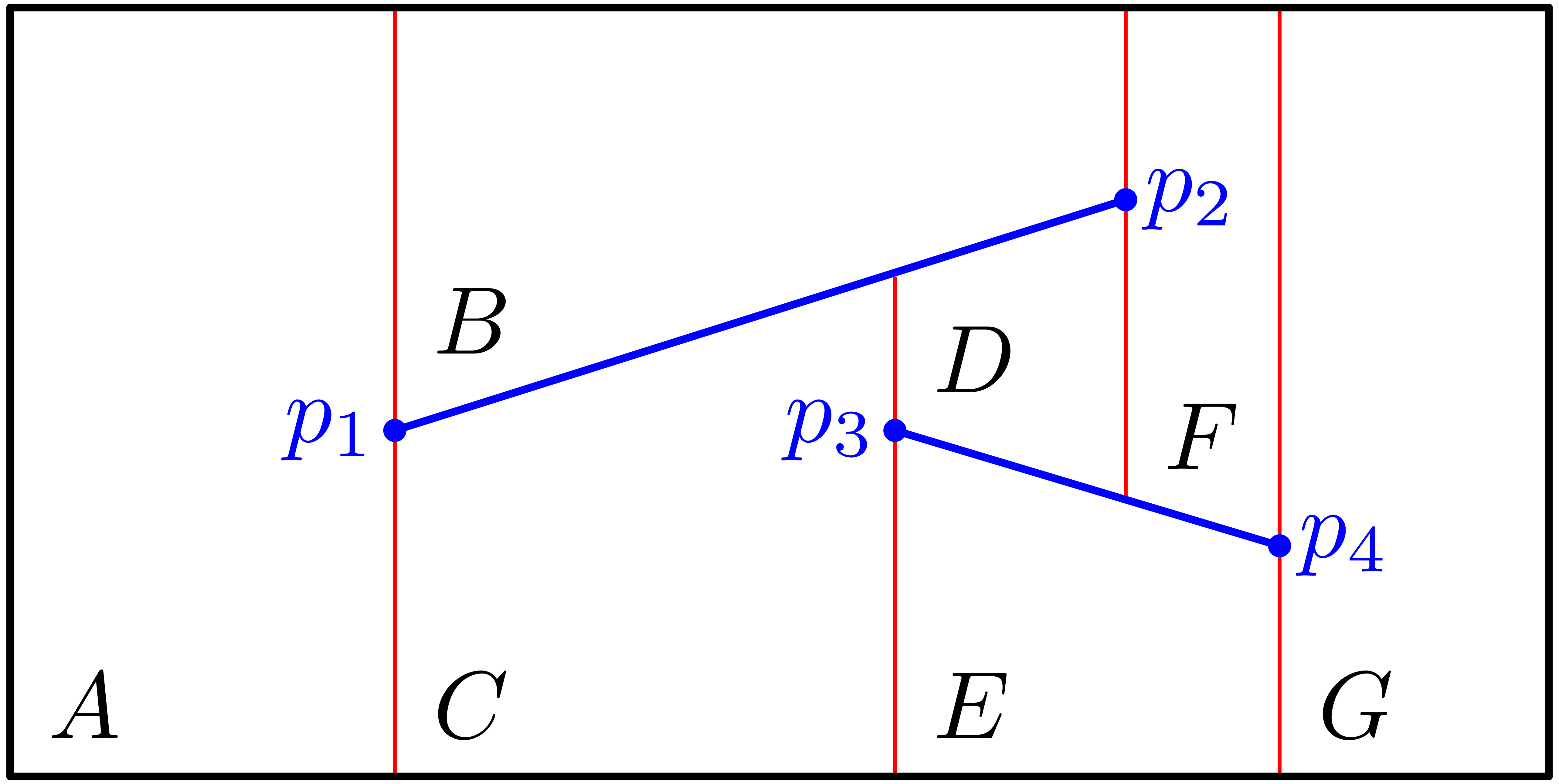
Trapezoidal maps

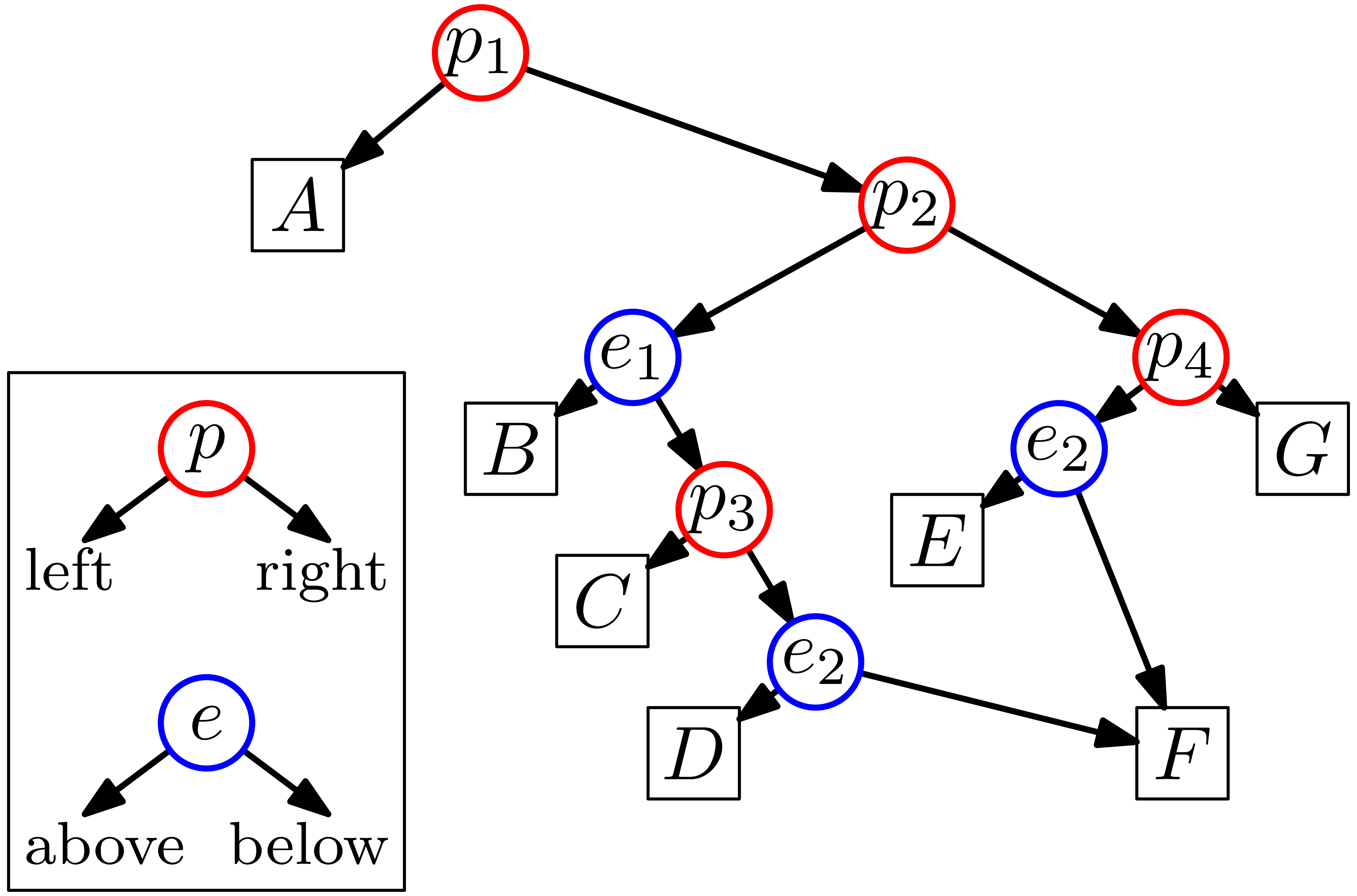


Trapezoidal maps



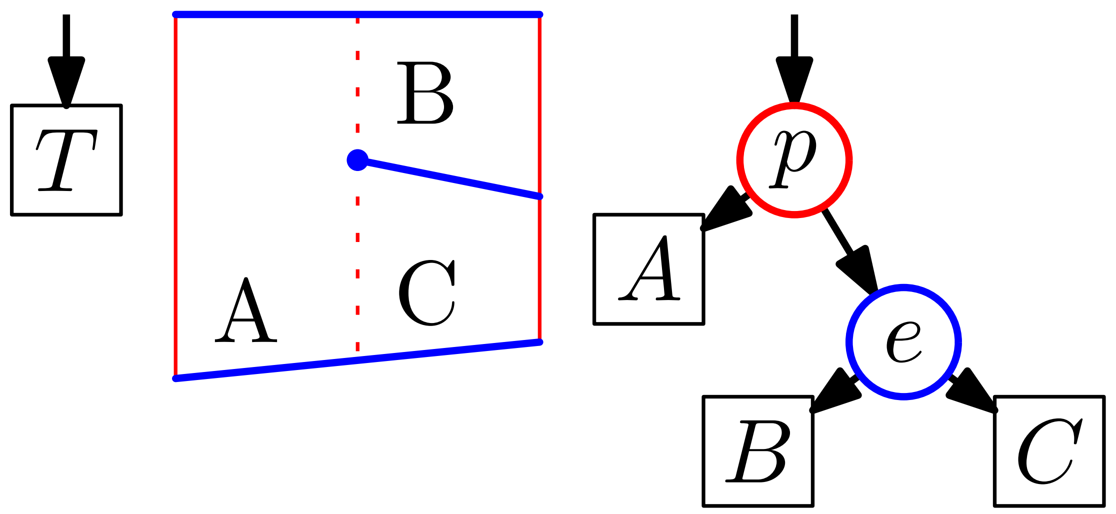
Trapezoidal maps

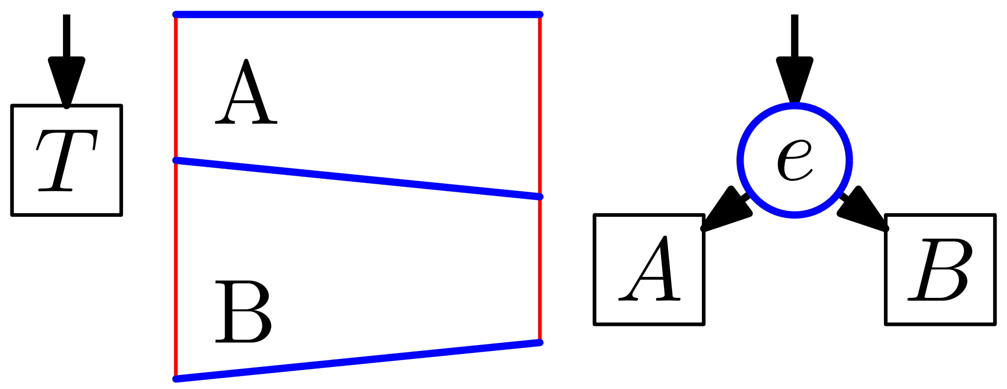


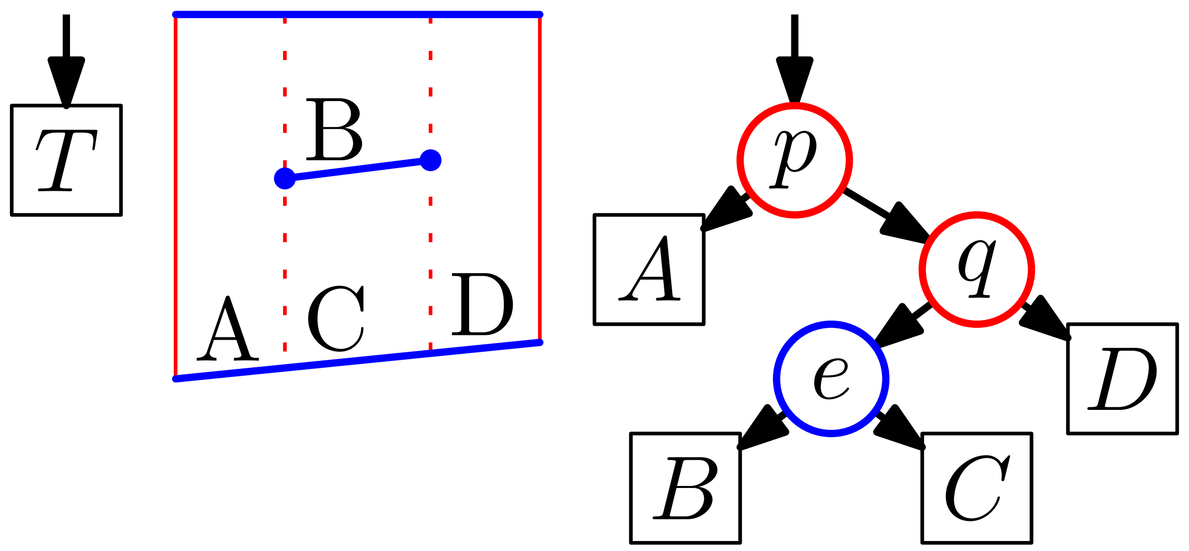


Trapezoidal maps

Incremental construction

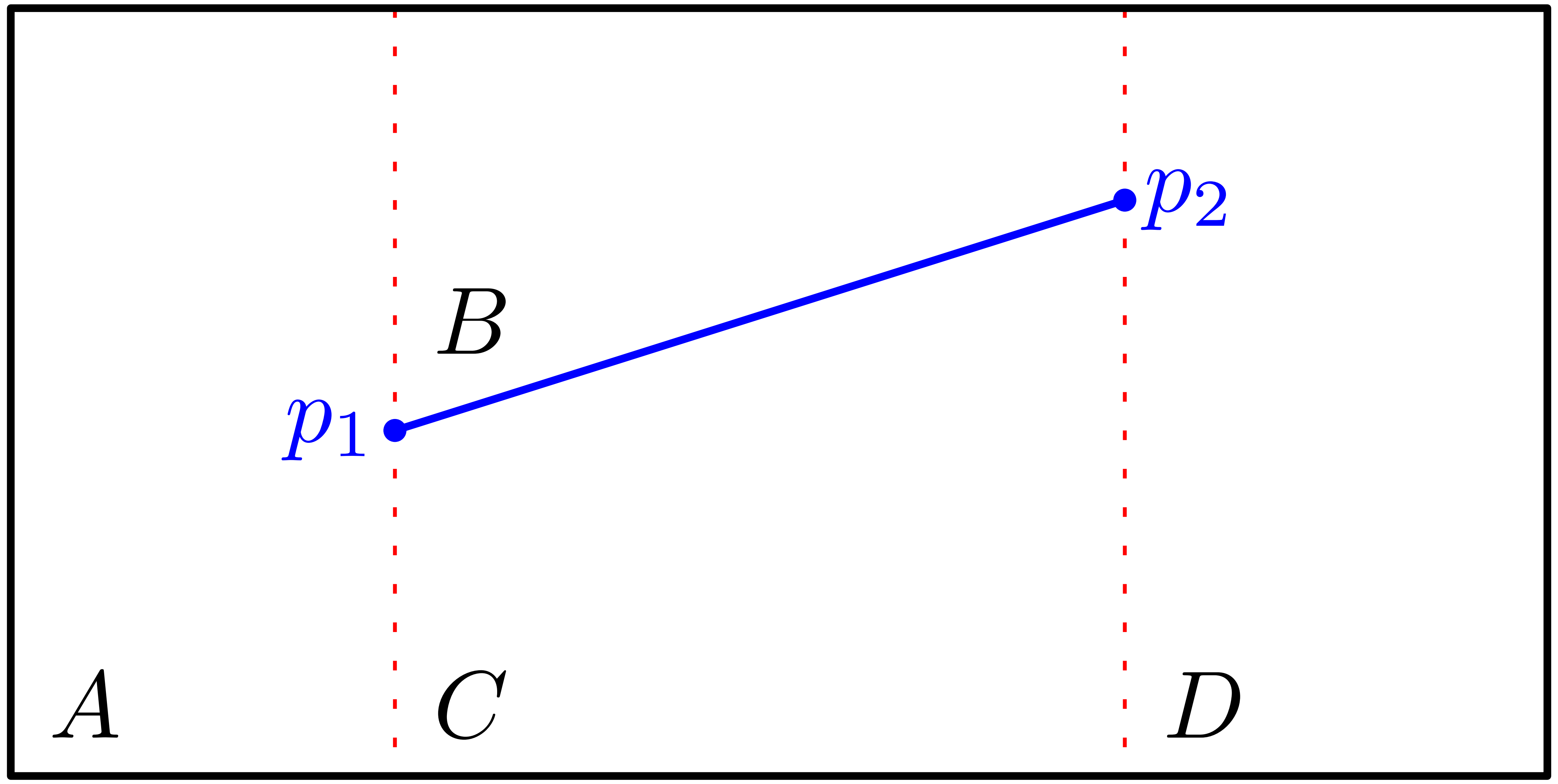


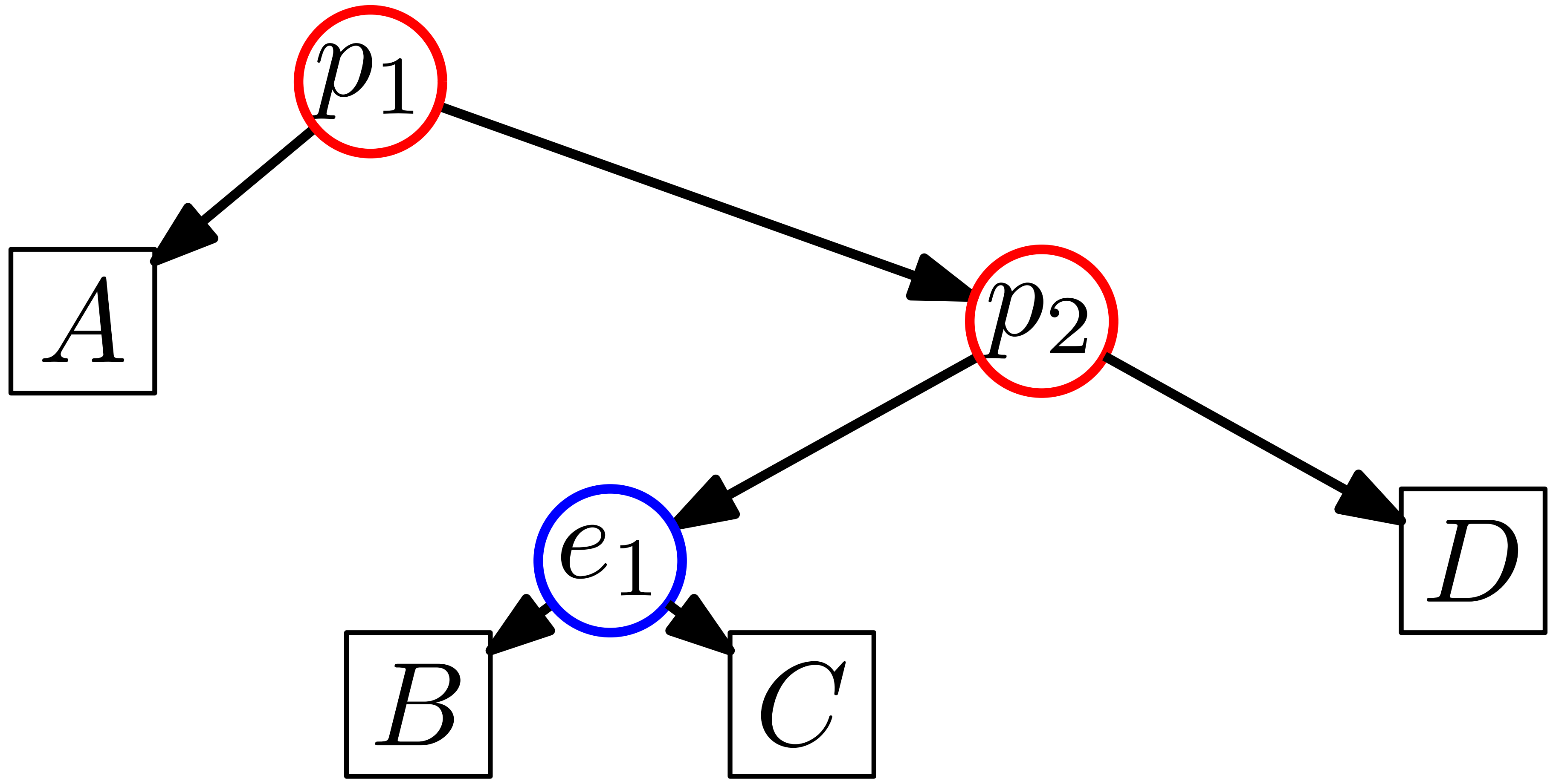




Trapezoidal maps

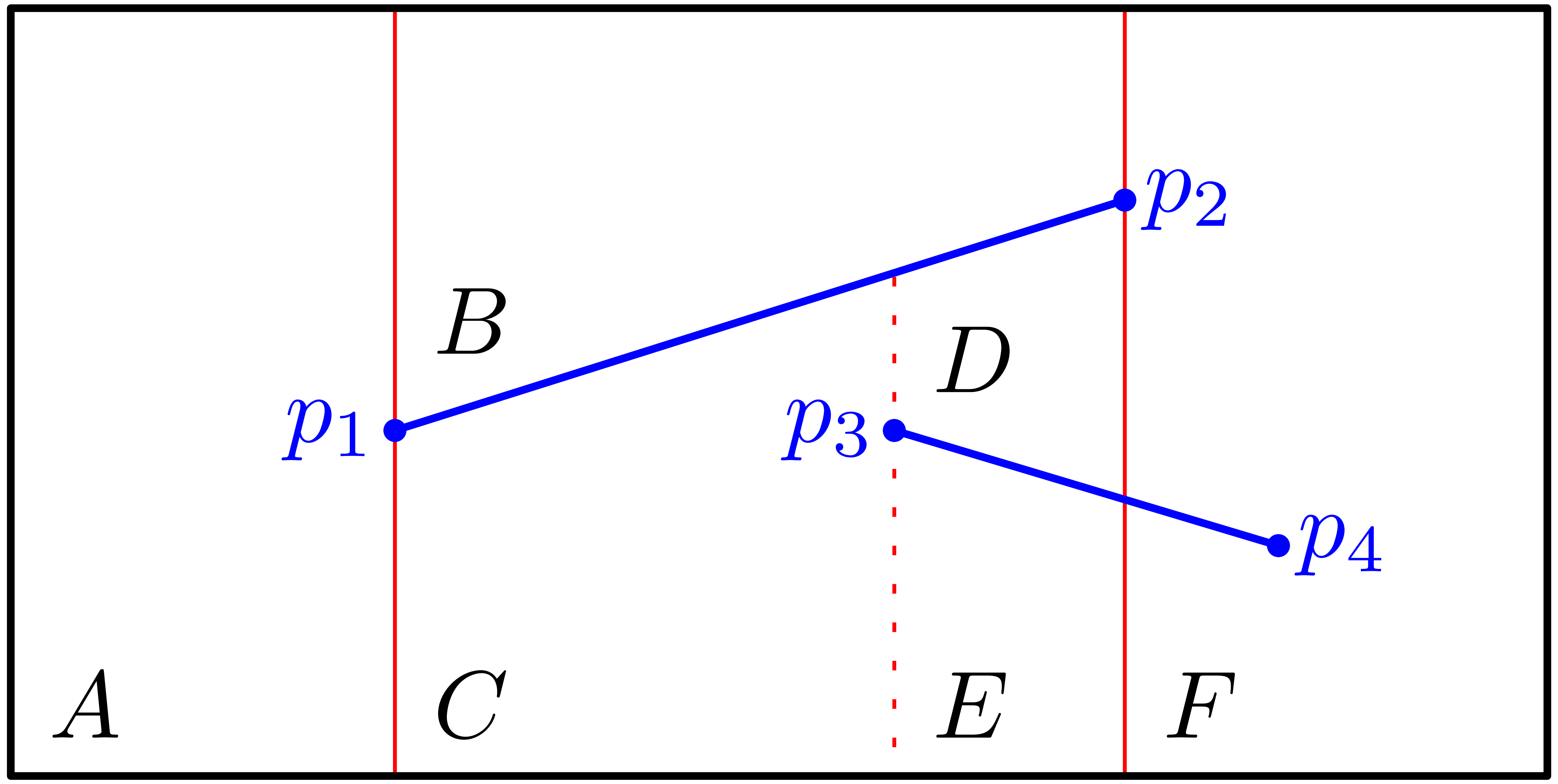
Incremental construction

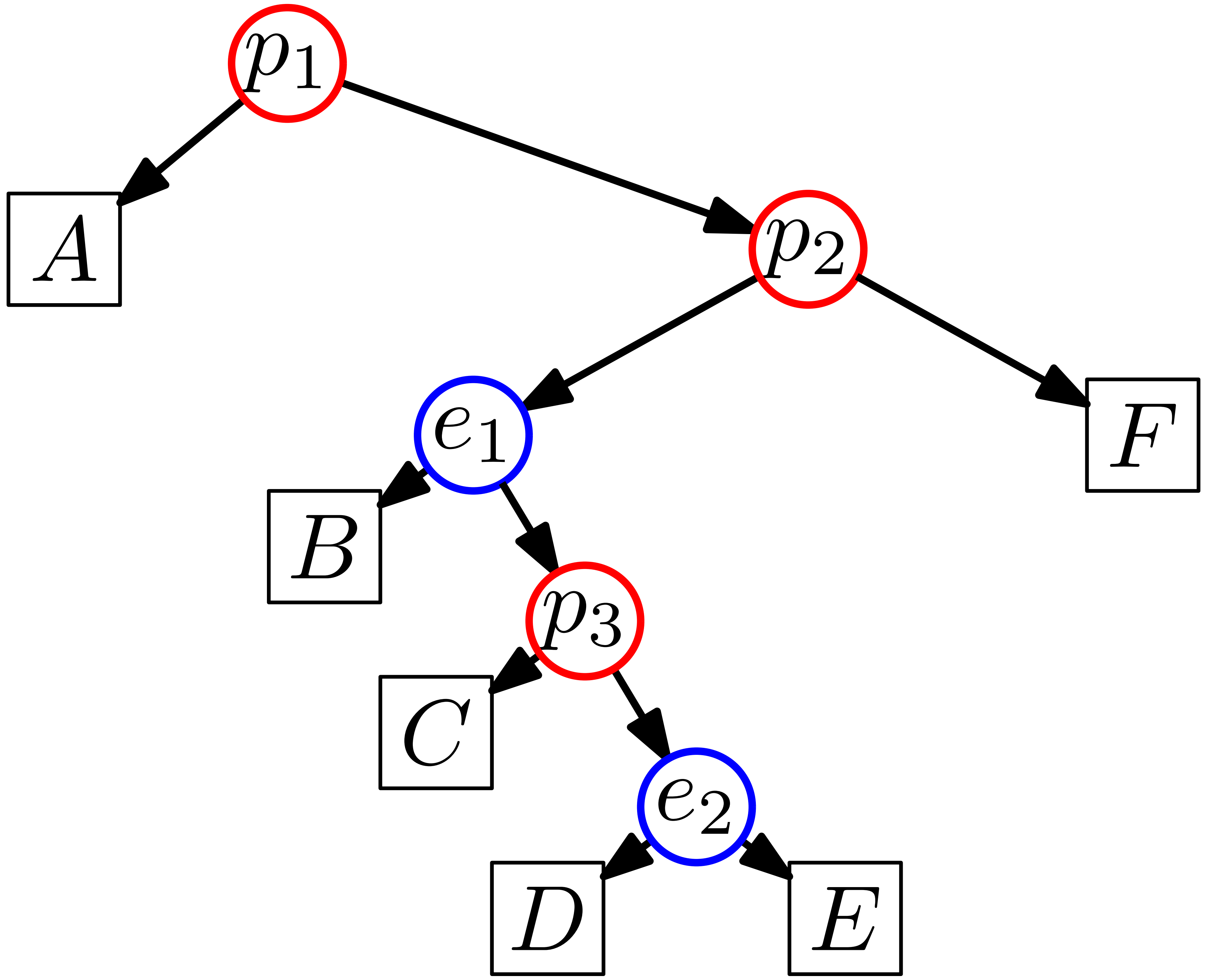




Trapezoidal maps

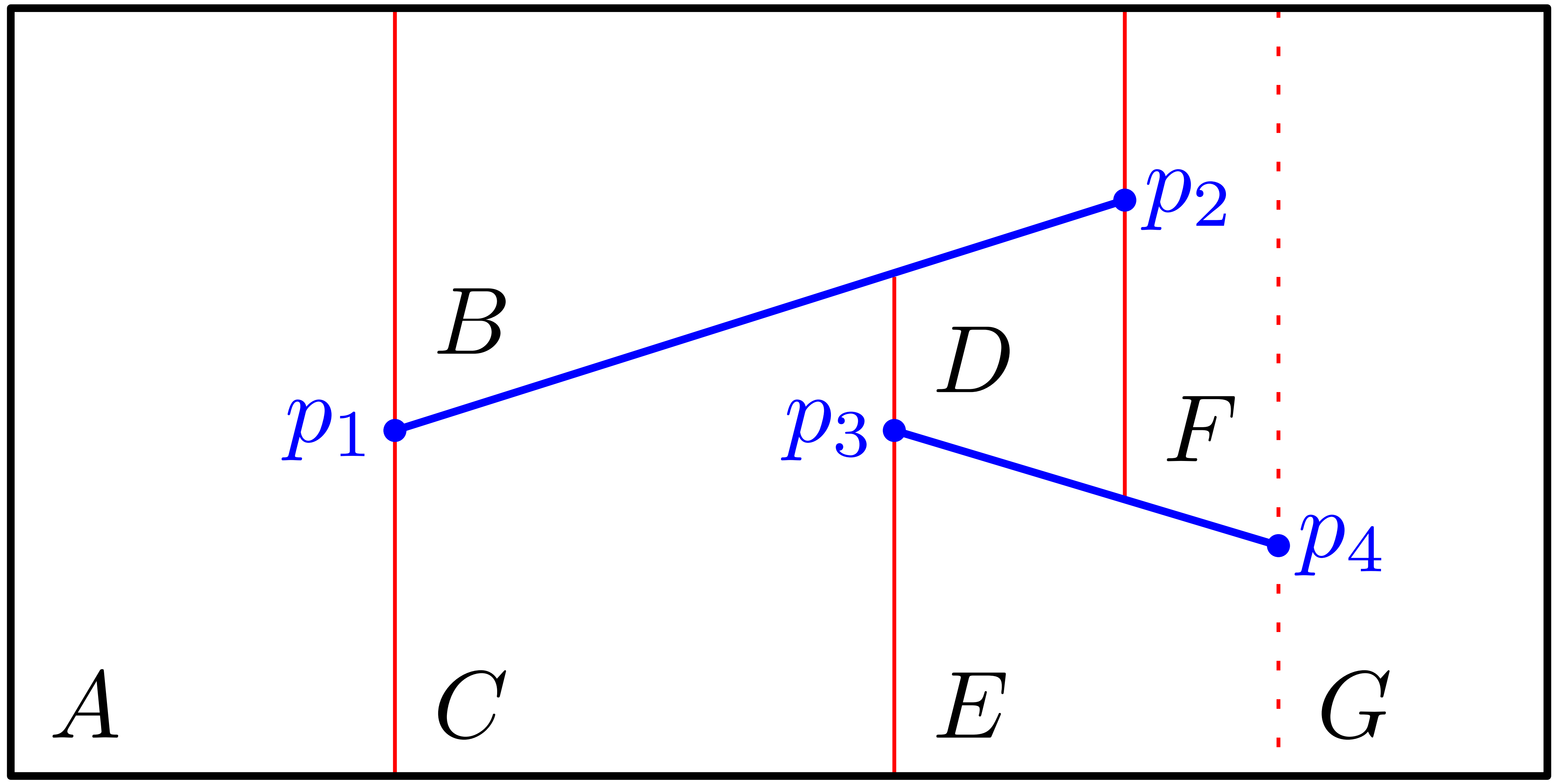
Incremental construction

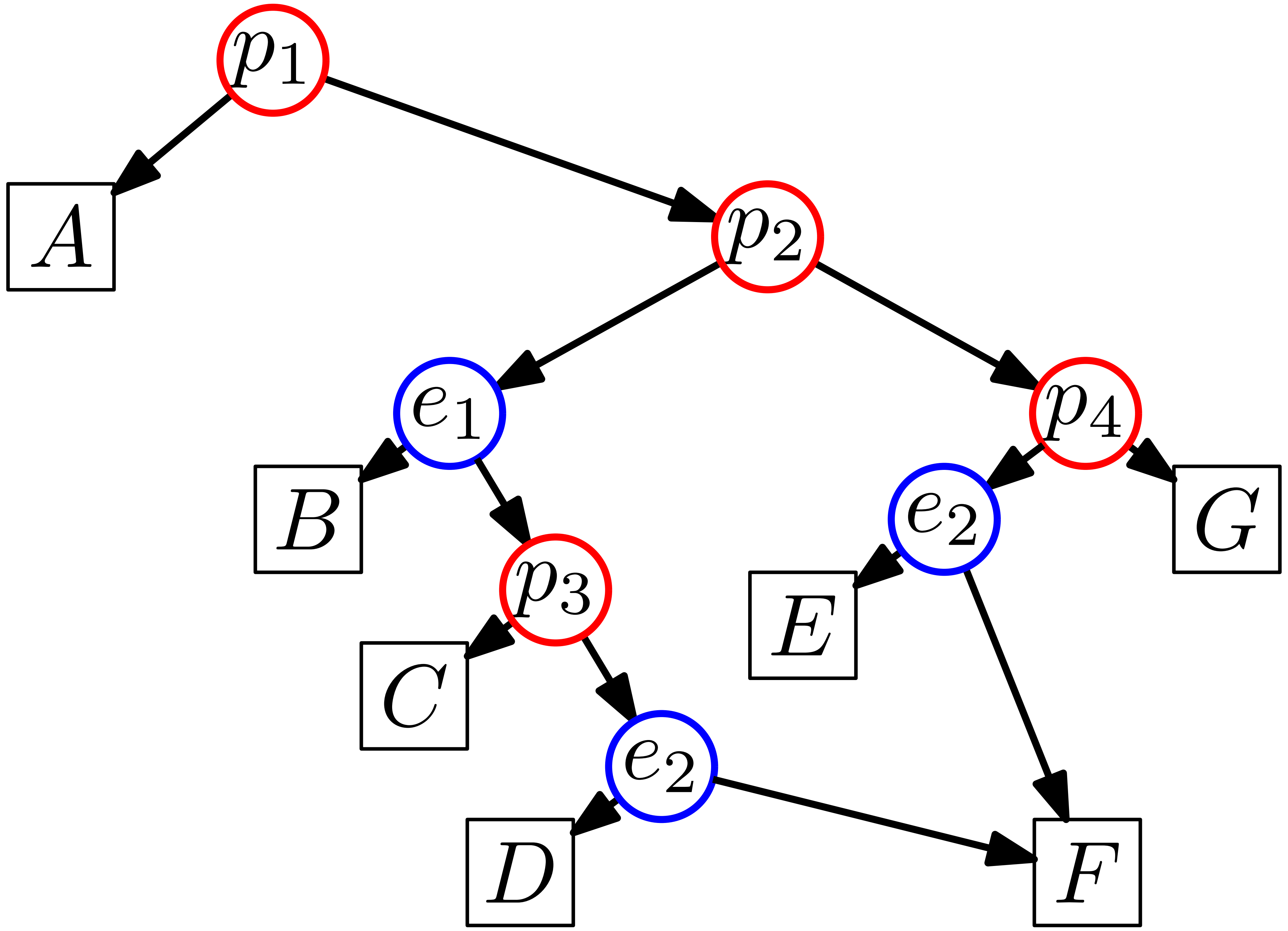




Trapezoidal maps

Incremental construction





Thank you