

# PARALLEL ALGORITHMS FOR SOME PROBLEMS IN PLANAR POINT SETS

Gur Saran Adhar\*  
Department of Computer Science  
Univ. of North Carolina Wilmington, USA

\*This work was completed during research reassignment at the  
Dayalbagh Educational Institute, India

+

+



$P_7$



$P_{12}$



$P_3$



$P_{10}$



$P_4$



$P_{11}$



$P_1$



$P_8$



$P_9$



$P_5$



$P_6$



$P_2$

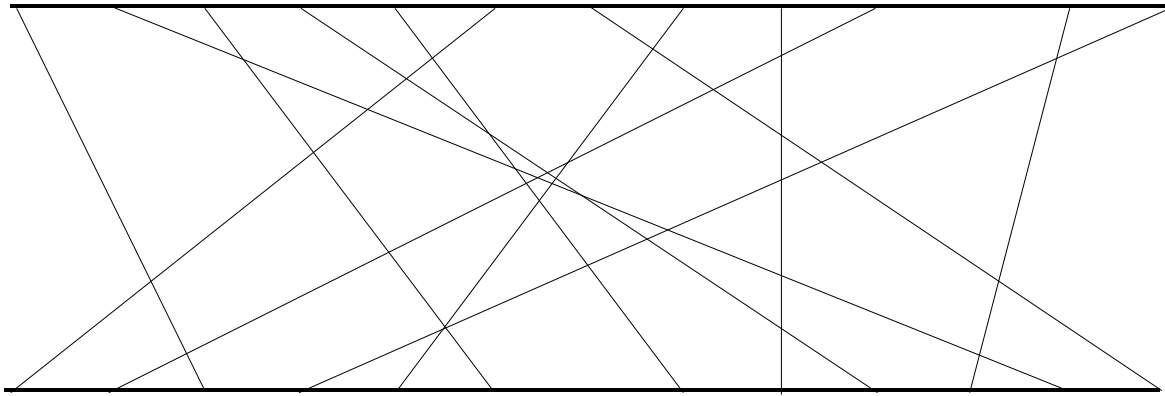
+

1

+

+

0 1 2 3 4 5 6 7 8 9 10 11 12



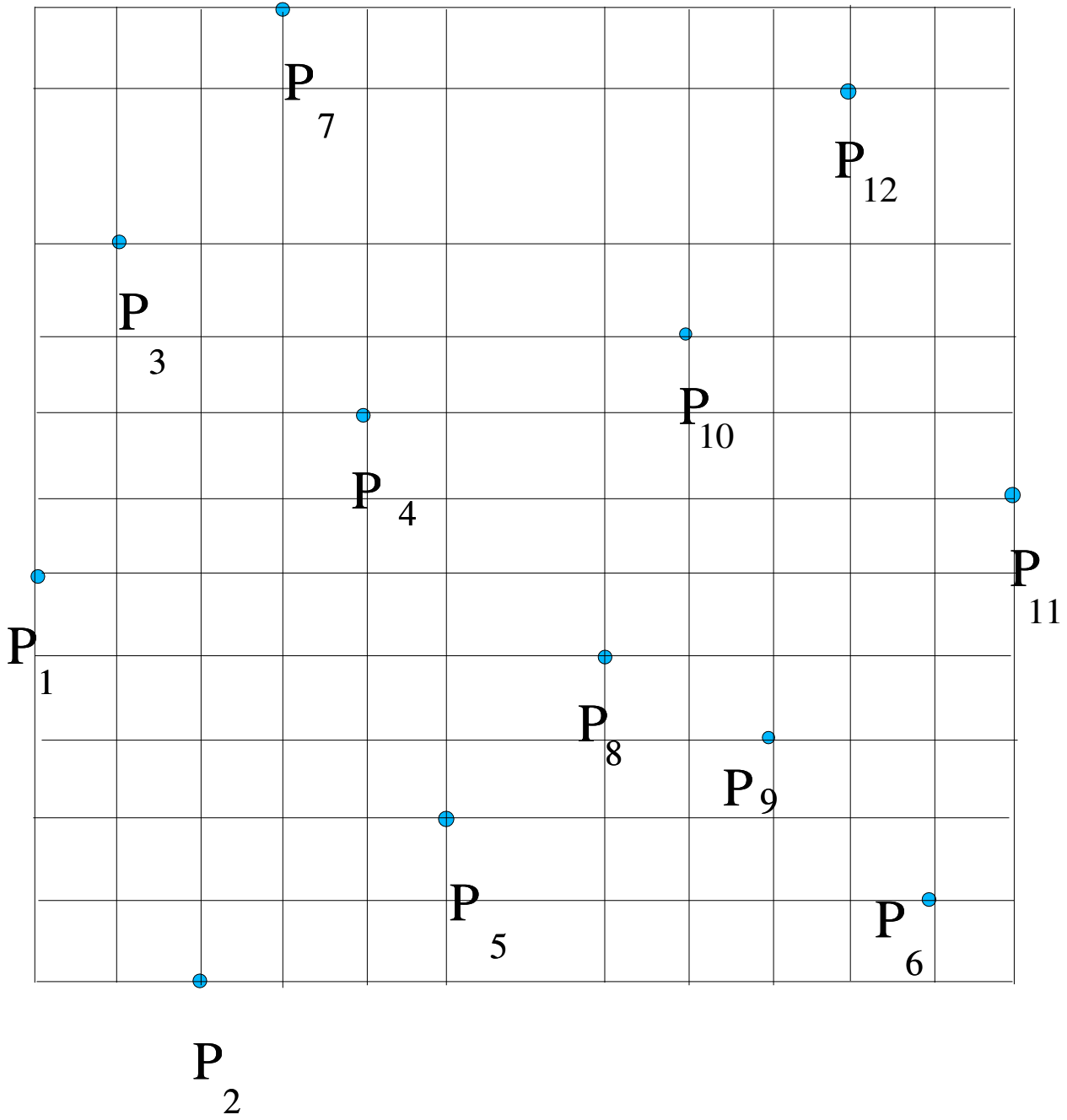
0 1 2 3 4 5 6 7 8 9 10 11 12

+

2

+

+

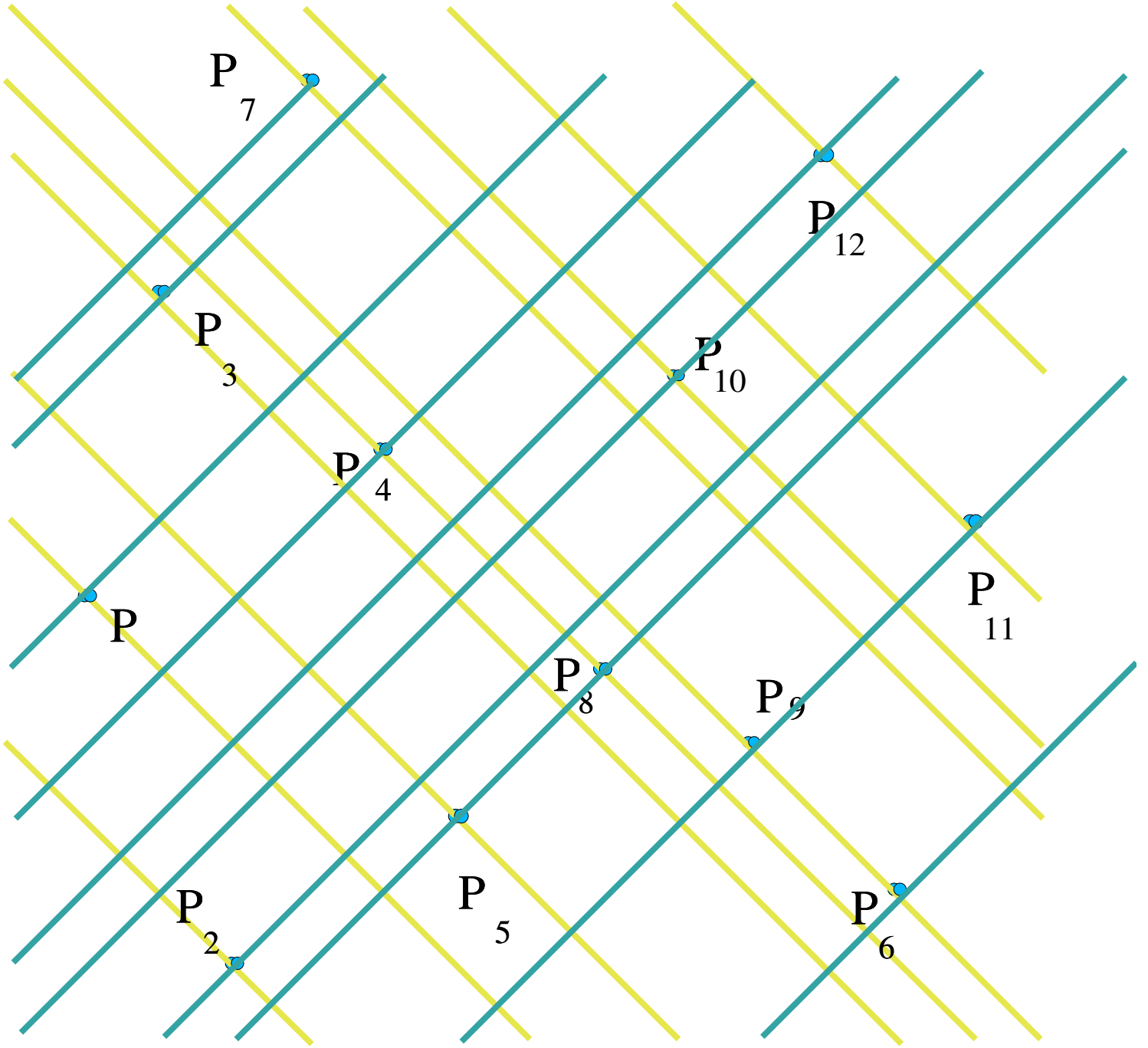


+

3

+

+

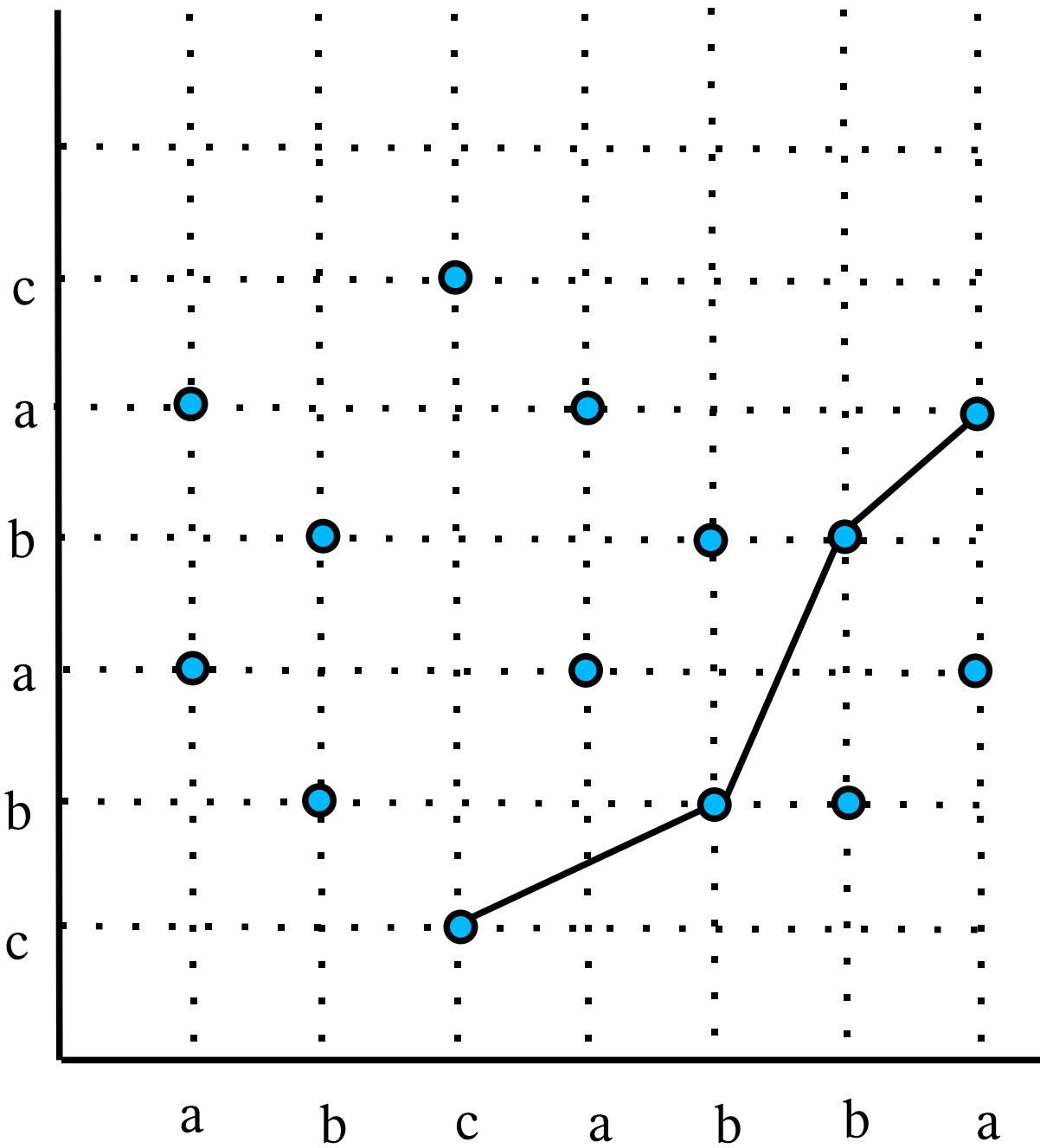


+

4

+

+



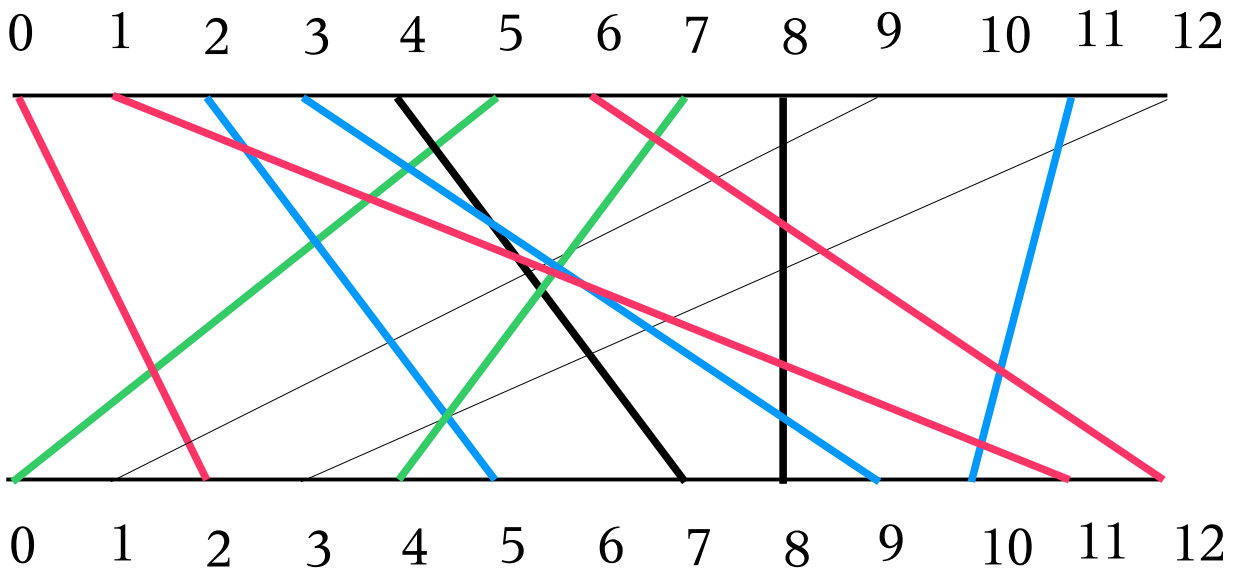
Longest monotonically increasing line represents longest Common Subsequence (Hunt-Szymanski)

+

+

+

+



Signal nets assigned to separate layers in VLSI

+

6

+

+

## Definition: Point Domination

A point  $p = (x, y)$  **dominates** another point  $q = (x', y')$  iff  $x' \leq x$  and  $y' \leq y$  (i.e., point  $q$  lies to the left and below point  $p$ ) denoted as  $q \leq p$ .

This definition is consistent with the concept of *vector dominance* where a vector  $\mathbf{v}$  *dominates*  $\mathbf{w}$  if and only if for all indices  $i$ ,  $v_i \geq w_i$ .

+

7

+

+



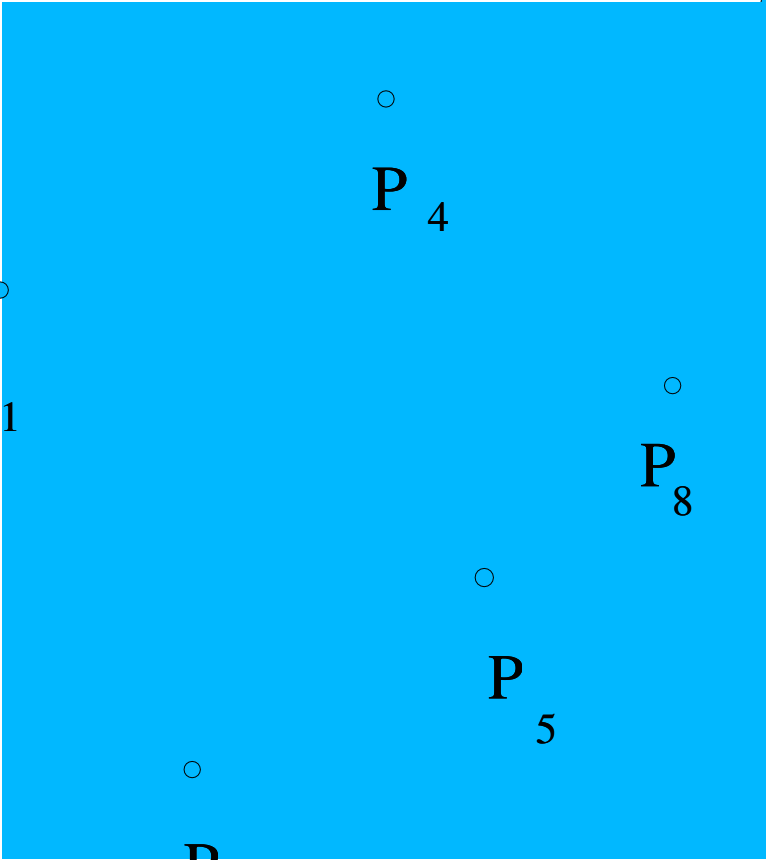
$P_7$



$P_{12}$



$P_3$



$P_1$



$P_4$



$P_{10}$



$P_8$



$P_5$



$P_2$



$P_{11}$



$P_9$



$P_6$

+

2

8

+

+

## Definition: Chain

A sequence of points  $\sigma = (p_1, p_2, \dots, p_n)$  is an *increasing* sequence of points iff point  $p_i$  dominates point  $p_{i-1}$  for all  $1 < i \leq n$ . Such a sequence is called a **chain** set or simply a chain.

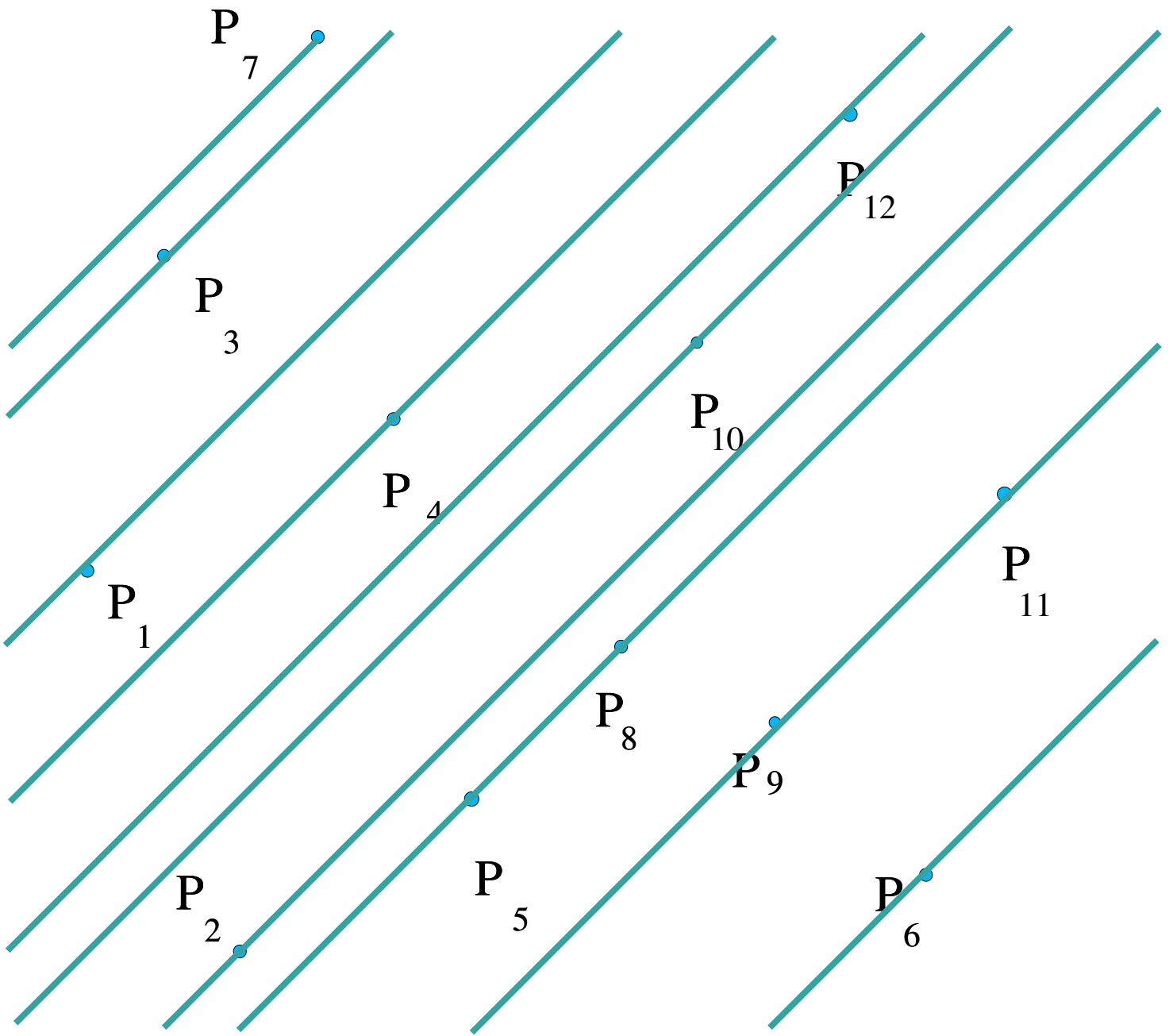
A chain  $\sigma = (p_1, p_2, \dots, p_n)$  is a maximum chain if it is of maximum length.

+

9

+

+

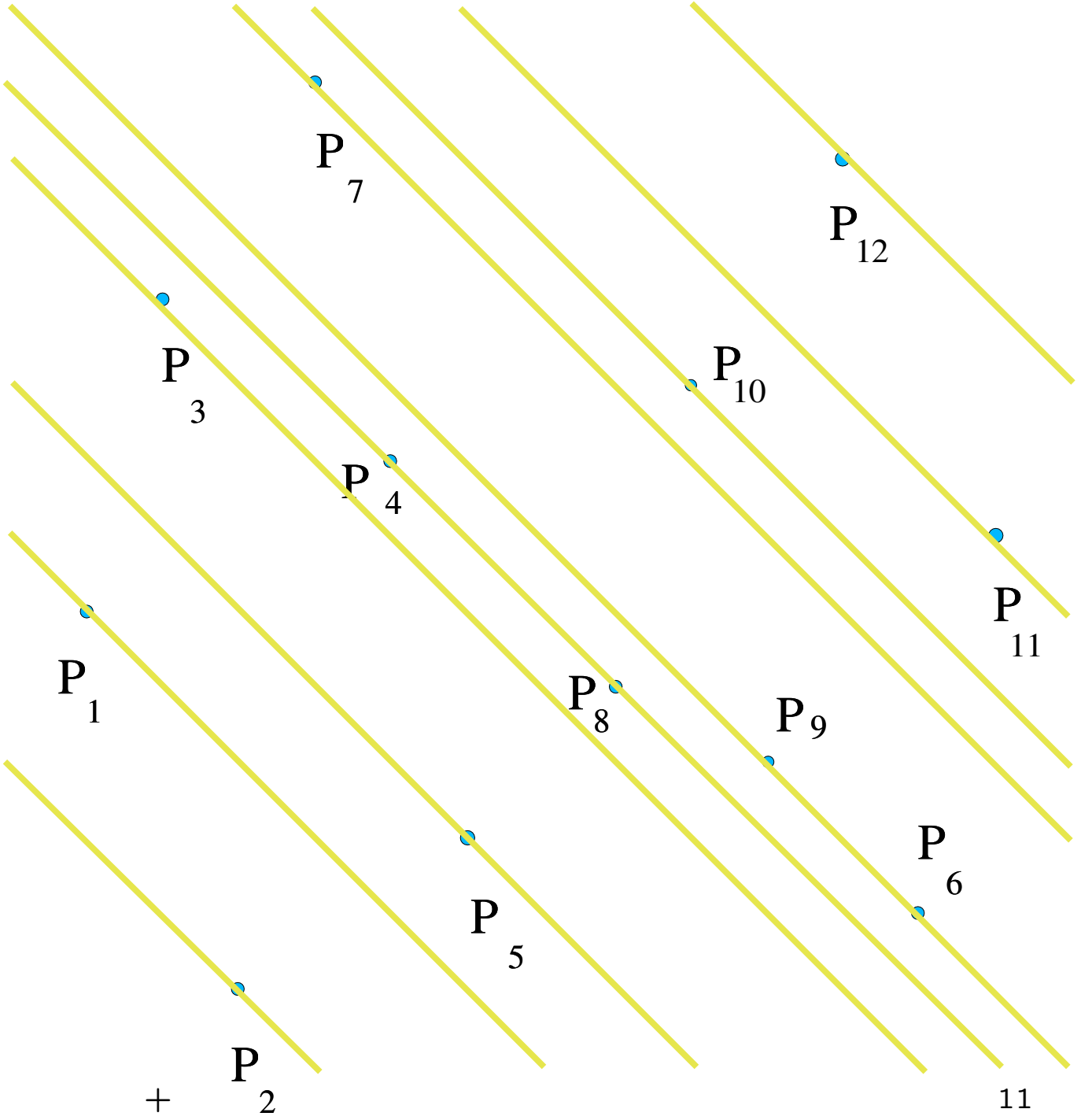


+

10

+

+



+

+

## Definition: Height of a Point

The **height**  $ht(p)$  **of a point**  $p$  is defined as the size (length) of a maximum chain in the subset  $\{q \in P \mid q \leq p\}$ , (i.e, size of maximum chain in the subset of points smaller than  $p$  that terminates at  $p$ ) and the **height for a set of points**  $P$  is the size of the maximum chain in  $P$ .

+

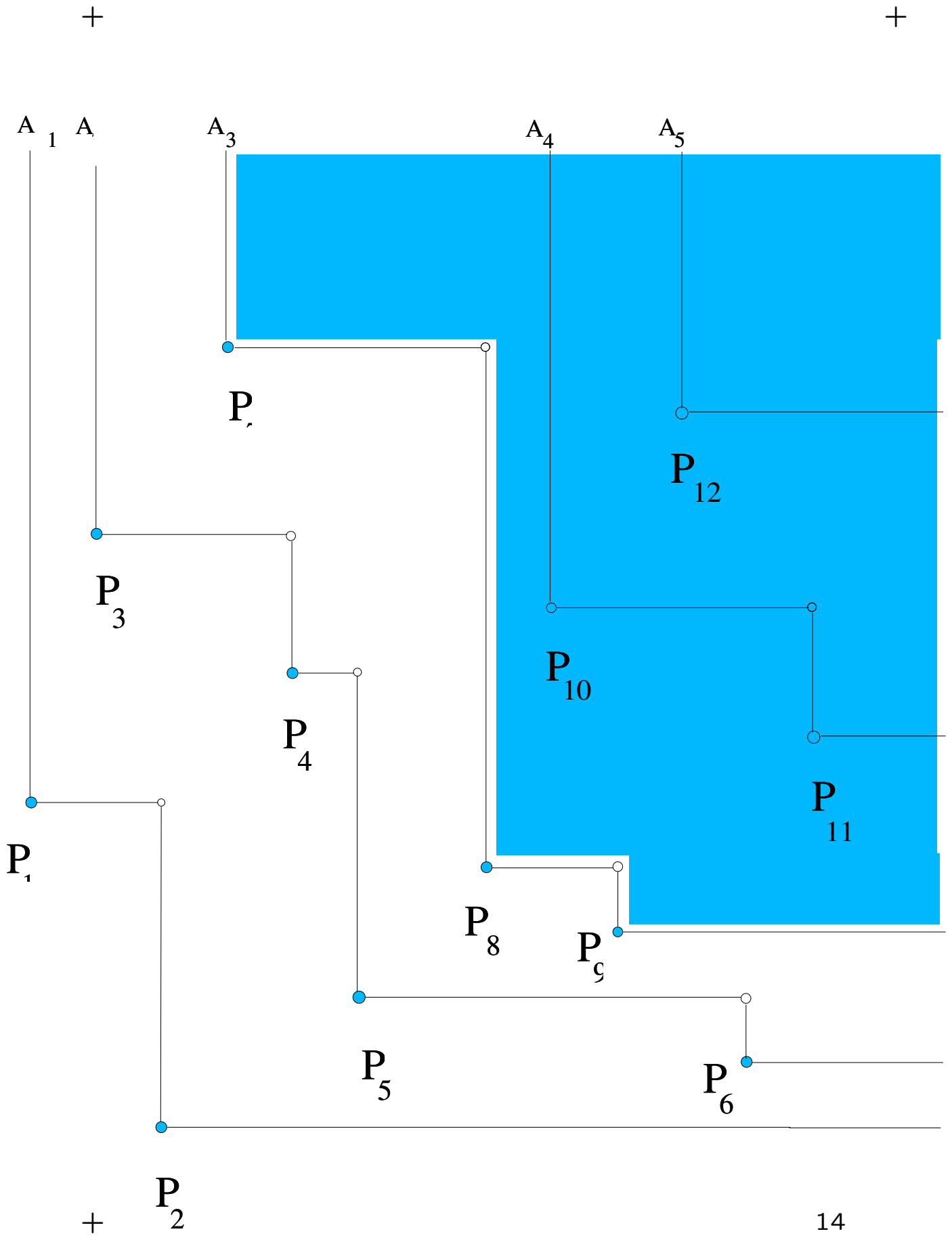
+

+

**Definition:**  
**Canonical Anti-Chain Partition**

A **canonical anti-chain partition**  $A_1, \dots, A_\lambda$  of  $P$ , where  $\lambda$  is the height of the set  $P$ , is the family of anti-chains such that each anti-chain  $A_i$  contains elements with the same height  $i$ .

+



+

+

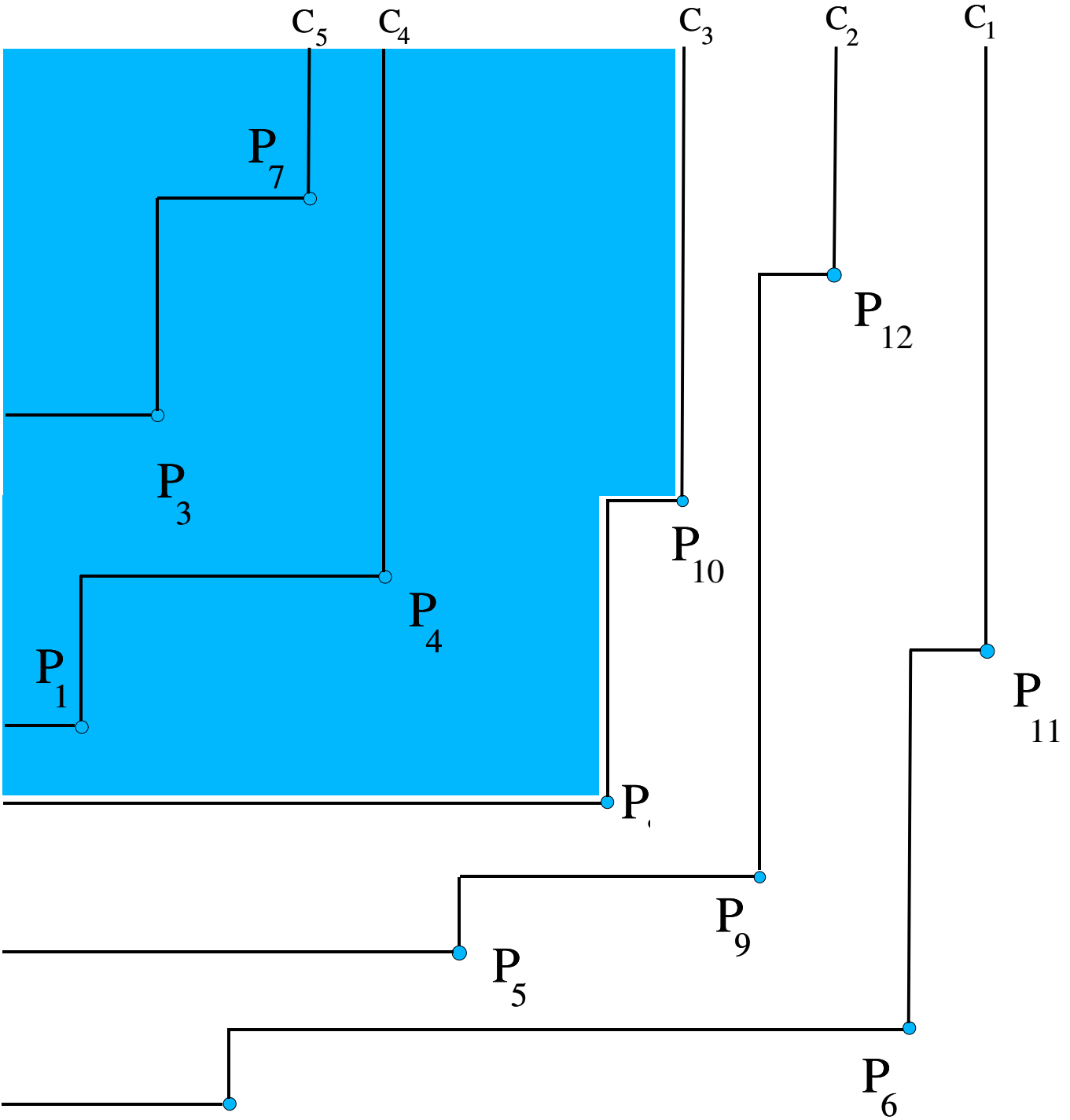
**Definition:**  
**Canonical Chain Partition**

A **canonical chain partition** of a set of points  $P$  is the unique chain partition  $C_1, \dots, C_k$  with all points of  $\cup_{j \geq i} C_j$  in the shadow of  $C_i$  (light from lower right corner), for  $1 \leq i \leq k$ .

+

+

+



+  $P_2$

+

+

## Side Remark

Permutation graphs are Perfect

A Chain of points corresponds to an Independent set in the Permutation graph

An Anti-chain of points corresponds to a Clique in the Permutation graph

Which is why Signal nets corresponding to a chain can therefore all be safely assigned to the same layer in VLSI.

The partial order defined on points is a bi-order.

+

+

+

## Definition: Affine Transformation $\delta$

If  $x_{max}$  represent the maximum x-co-ordinate value of points in  $P$  then **affine transformation**  $\delta$  maps a point  $p = (x, y)$  to another point  $\delta(p)$  by the following function:

$$\delta(p) = (y, (x_{max} - x)) \quad (1)$$

In terms of matrix operations, the  $\delta$  transformation corresponds to first multiplying the point vector  $\mathbf{p} = \begin{pmatrix} x \\ y \end{pmatrix}$  with transformation matrix:

$$\mathbf{m} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$$

followed by addition of the vector:

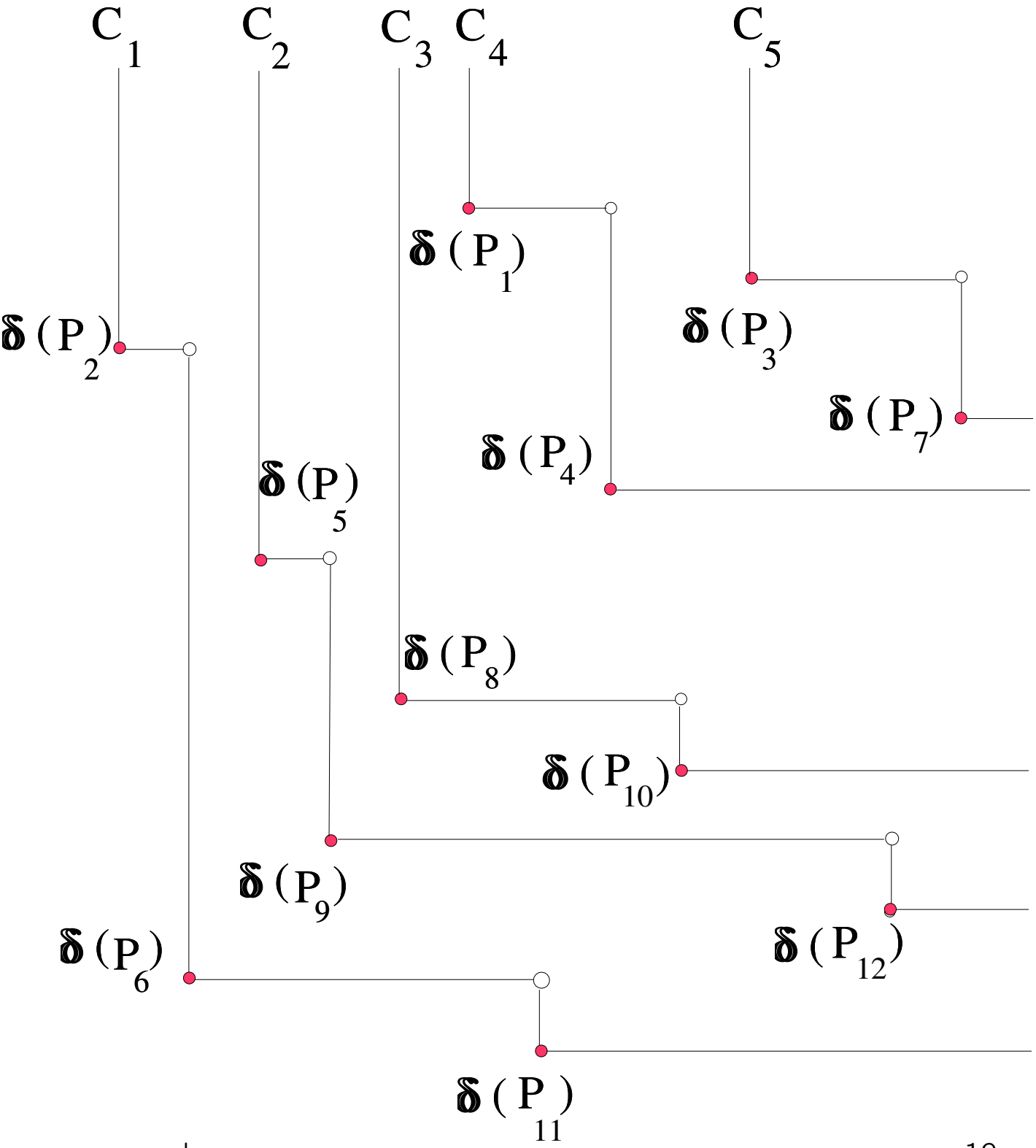
$$\mathbf{a} = \begin{pmatrix} 0 \\ x_{max} \end{pmatrix}$$

$$\delta(\mathbf{p}) = \mathbf{m} \cdot \mathbf{p} + \mathbf{a} \quad (2)$$

+

+

+



+

+

+

## Lemma

If two points  $p, q$  in the set  $P$  are comparable  $p \leq q$  or  $q \leq p$  then their mappings  $\delta(p)$  and  $\delta(q)$  are incomparable.

**Proof:** Without loss of generality let  $p = (x, y) \leq q = (x', y')$ . Therefore  $\delta(p) = (y, (x_{max} - x))$  and  $\delta(q) = (y', (x_{max} - x'))$ . Hence,  $\delta(q) \leq \delta(p)$ . ♣

+

+

+

## Theorem

Following statements are equivalent:

- A chain  $C$  in the set of points  $P$  is an anti-chain  $A$  in  $\delta(P)$ .
- An anti-chain  $A$  in the set of points  $P$  is a chain  $C$  in  $\delta(P)$ .



+

+

+

## Proposition

If  $k$  is the length of the maximal canonical chain for a set of points in a plane then the number of canonical anti-chains which partition the set of points is also  $k$ .

**Proof:** Each point on the  $k$ -maximal chain belongs to a distinct anti-chain giving the result. ♣

+

+

+

## Algorithm Chains and Anti-Chains

### Algorithm ChainsAndAntiChains

(input  $P$ : set\_of\_points;

output  $A$ : family of anti-chains;  $C$ : family of chains)

**begin**

i **height**( $P$ );

ii **for all points**  $p \in P$  **do** /\* in parallel \*/

$A_{ht(p)} \leftarrow A_{ht(p)} \cup \{p\}$

/\* add point  $p$  to the anti-chain with the index  
same as height of  $p$  \*/

iii  $\delta(P) \leftarrow$  **transform**( $P$ );

iv **height**( $\delta(P)$ );

v **for all points**  $p \in \delta(P)$  **do** /\* in parallel \*/

$C_{ht(\delta(p))} \leftarrow C_{ht(\delta(p))} \cup \{\delta(p)\}$

/\* add point  $\delta(p)$  to the chain with the index  
same as height of  $\delta(p)$  \*/

**end.**

+

+

+

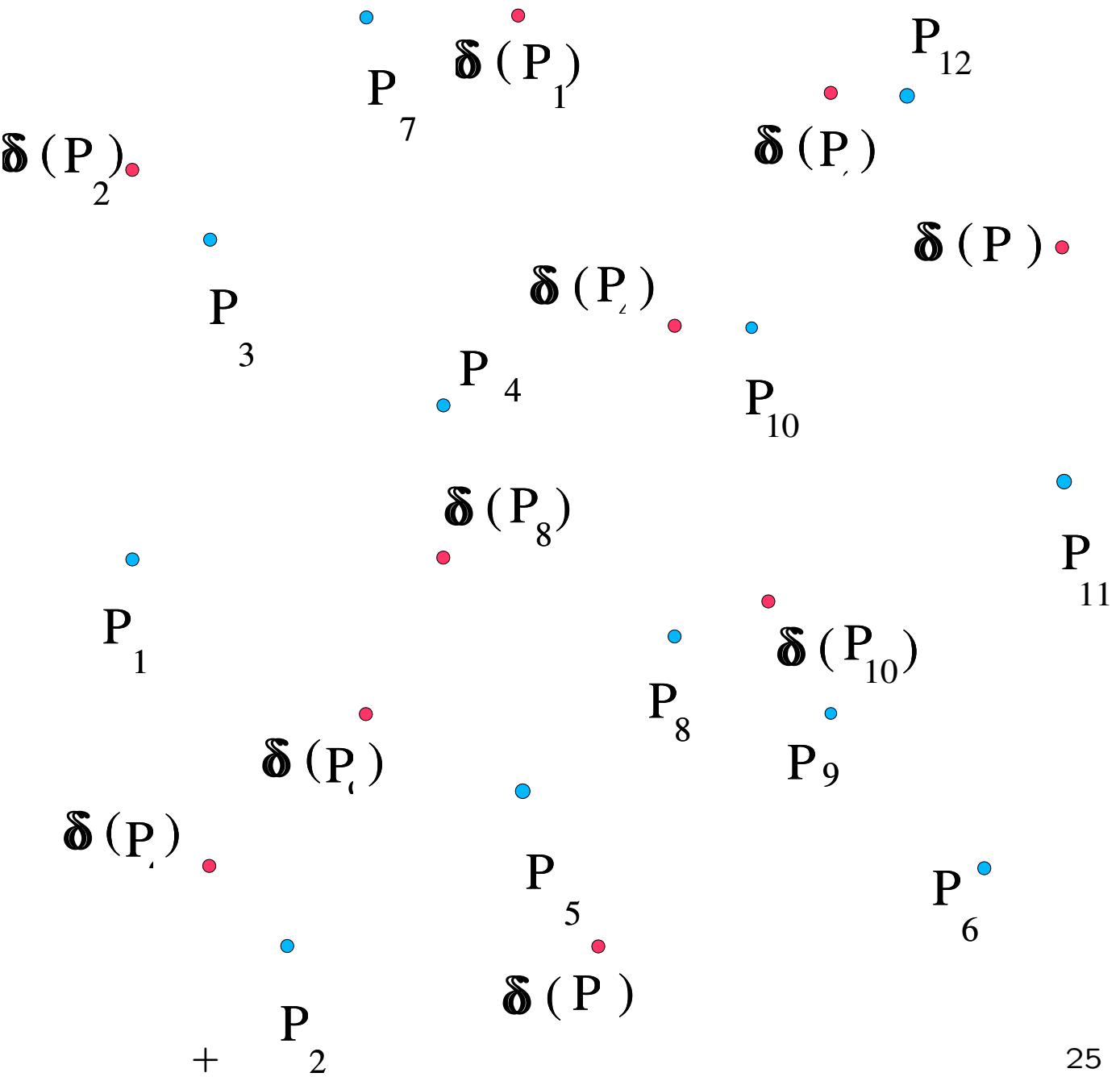
## Algorithm Transform

```
function transform  
(input P:set_of_points;  
output  $\delta(P)$ : transformed set of points)  
begin  
  
  i Initialize  
       $m = \{\{0, 1\}, \{-1, 0\}\}; a = \{0, x_{max}\};$   
  ii for all points  $p \in P$  do /* in parallel */  
       $\delta(p) \leftarrow m.p + a$   
  
end.
```

+

+

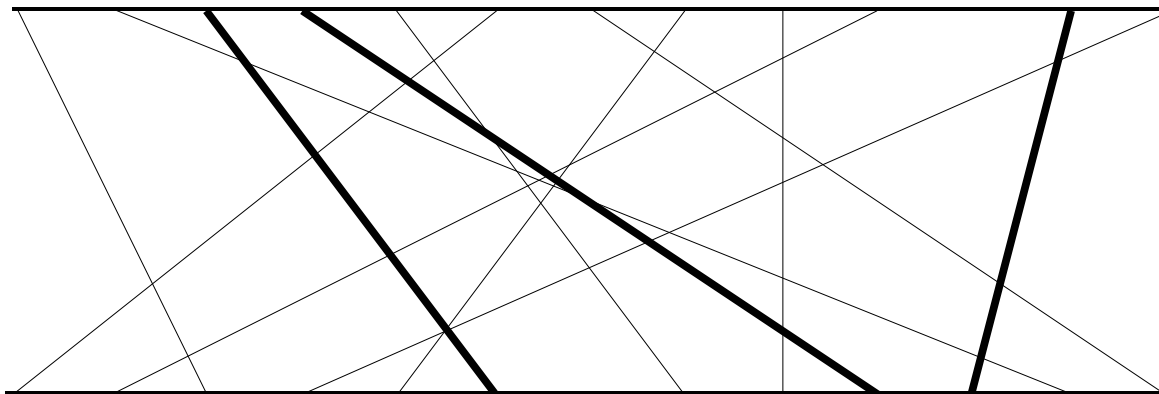
+



+

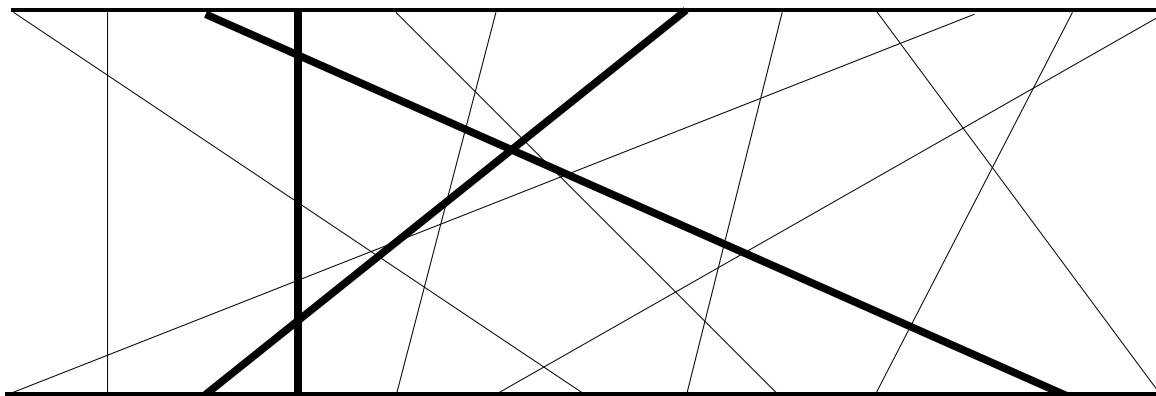
25

$P_2$   $P_6$   $P_5$   $P_9$   $P_8$   $P$   $P_{11}$   $P_4$   $P_{10}$   $P_3$   $P_{12}$   $P_7$  +  
 0 1 2 3 4 5 6 7 8 9 10 11 12



0 1 2 3 4 5 6 7 8 9 10 11 12  
 $P_1$   $P_3$   $P_2$   $P_7$   $P_4$   $P_5$   $P_8$   $P_{10}$   $P_9$   $P_{12}$   $P_6$   $P_{11}$

$\delta(P)_{11}$   $\delta(P)_6$   $\delta(P)_{12}$   $\delta(P)_9$   $\delta(P)_{10}$   $\delta(P)_8$   $\delta(P)_5$   $\delta(P)_4$   $\delta(P)_7$   $\delta(P)_2$   $\delta(P)_3$   $\delta(P)_1$   
 0 1 2 3 4 5 6 7 8 9 10 11 12



0 1 2 3 4 5 6 7 8 9 10 11 12  
 $\delta(P)_2$   $\delta(P)_6$   $\delta(P)_5$   $\delta(P)_9$   $\delta(P)_8$   $\delta(P)_1$   $\delta(P)_{11}$   $\delta(P)_4$   $\delta(P)_{10}$   $\delta(P)_3$   $\delta(P)_{12}$   $\delta(P)_7$

+

+

+

## Algorithm Height (sequential)

```
function  
height(sequential)(P:set_of_points)  
begin  
  
  i Initialize  $Cumulative\_set \leftarrow Current\_Set \leftarrow p_s$ ;  
     $ht(p_s) \leftarrow i \leftarrow 0$ ;  
  ii while  $(P - Cumulative\_set) \neq \phi$  do  
  
  iii for each  $(z \notin Cumulative\_set)$   
    if  $lt(z) \subseteq Cumulative\_set$  then  
       $Current\_Set \leftarrow Current\_Set \cup z$ ;  
       $ht(z) \leftarrow i$   
  
  iv  
  
     $Cumulative\_Set \leftarrow Cumulative\_Set \cup Current\_Set$ ;  
     $i \leftarrow i + 1$ ;  
  
end.
```

+

+

+



$P_7$



$P_{12}$



$P_3$



$P_{10}$



$P_4$



$P_{11}$

$P_1$



$P_5$



$P_8$



$P_9$



$P_2$



$P_6$



$P_c$

+

+

+



$P_7$



$P_{12}$



$P_3$



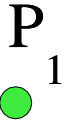
$P_{10}$



$P_4$



$P_{11}$



$P_1$



$P_8$



$P_9$



$P_5$



$P_6$



$P_2$

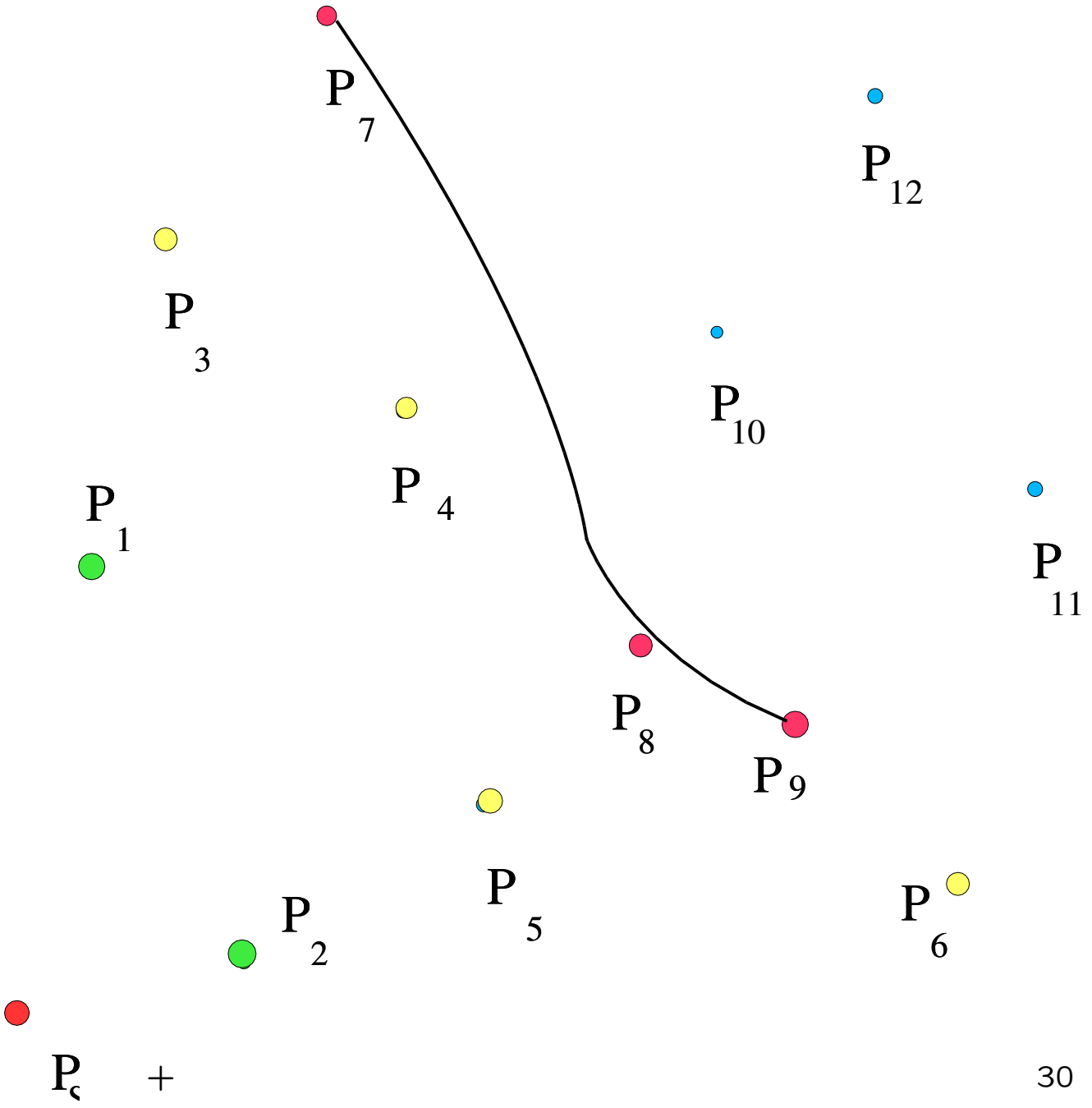


$P_c$

+

+

+



+

+

$P_7$

$P_{12}$

$P_3$

$P_{10}$

$P_1$

$P_4$

$P_{11}$

$P_8$

$P_9$

$P_2$

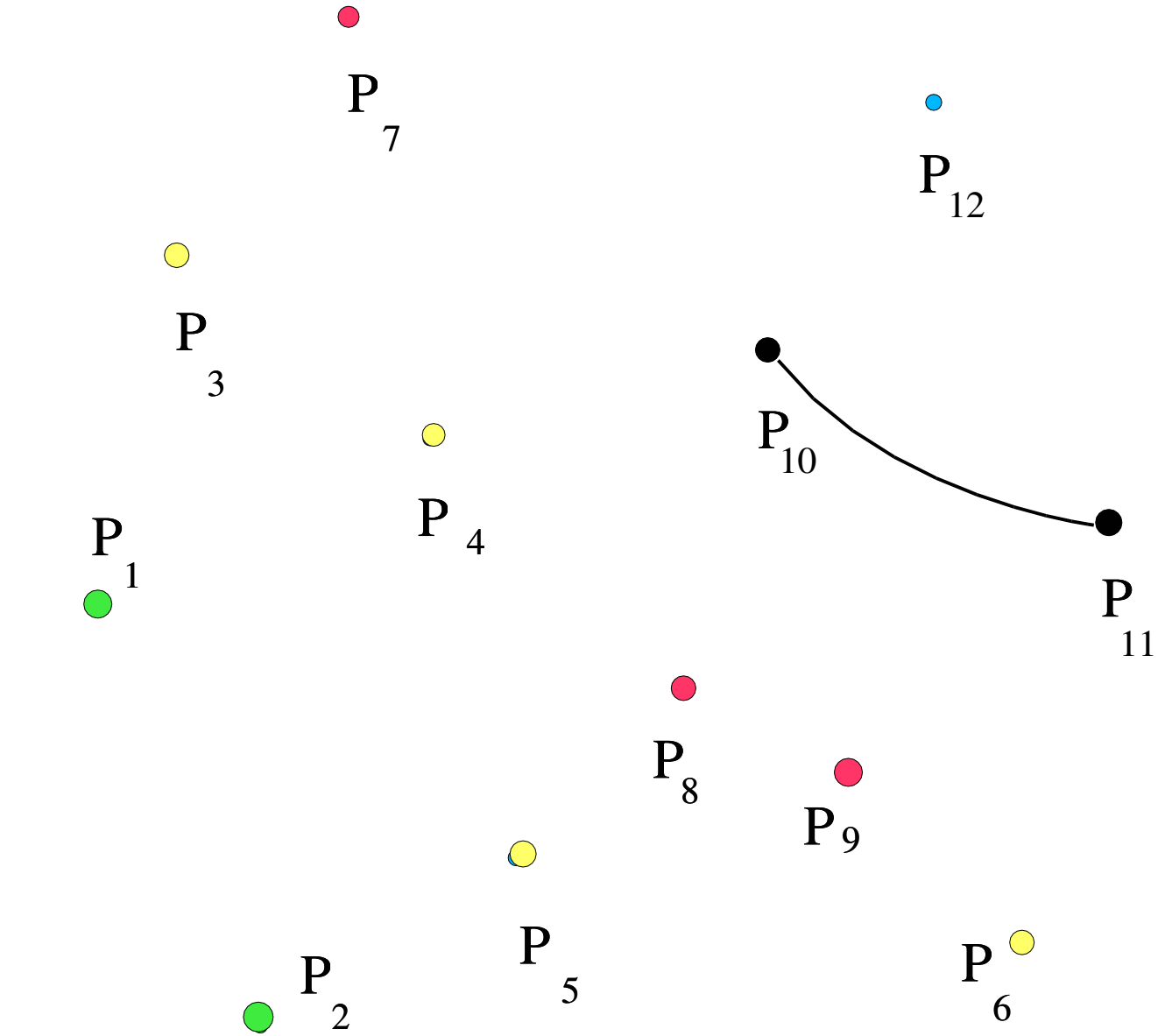
$P_5$

$P_6$

$P_c$

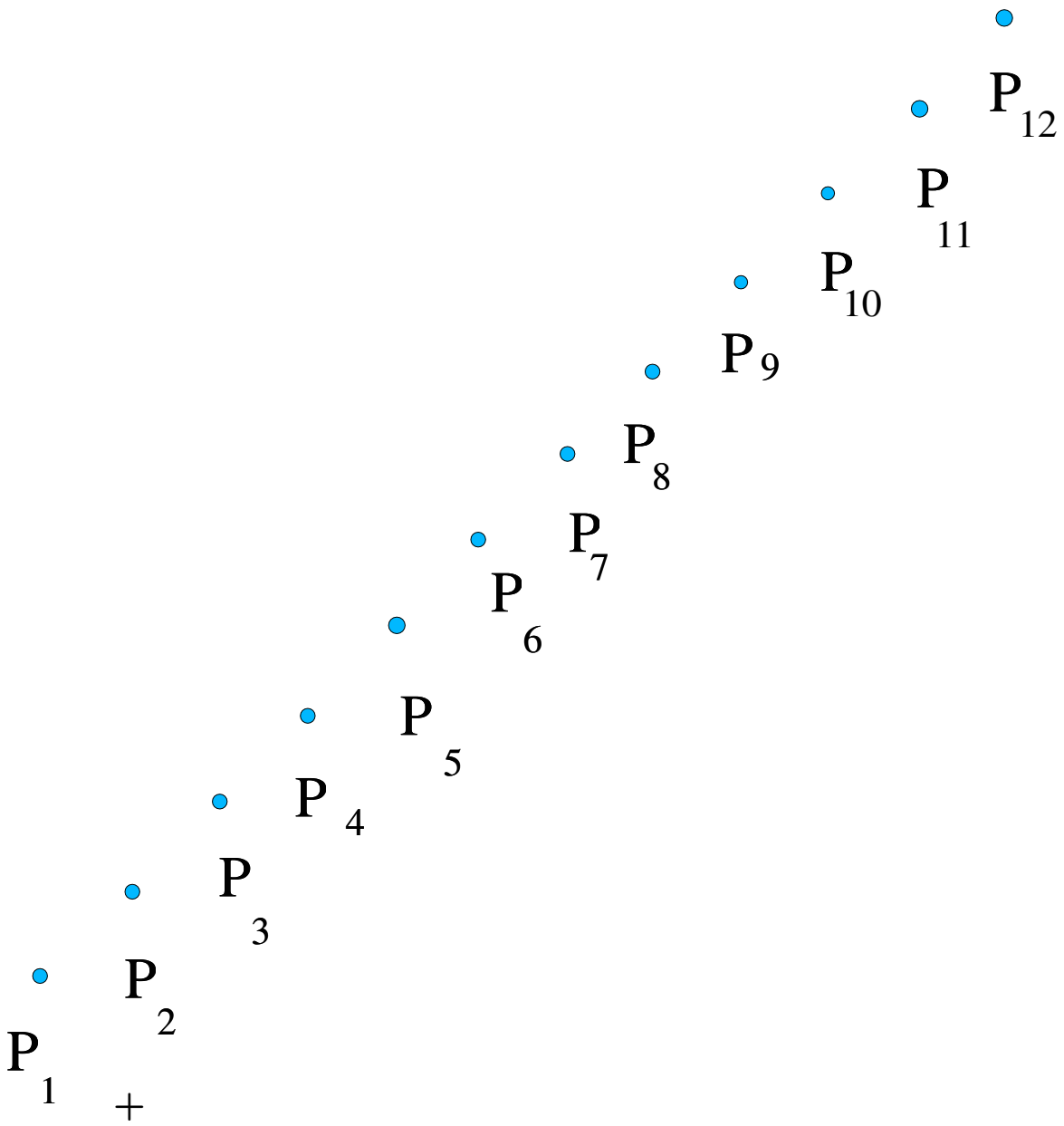
+

31



+

+



+

+

## Definition: Direct Domination

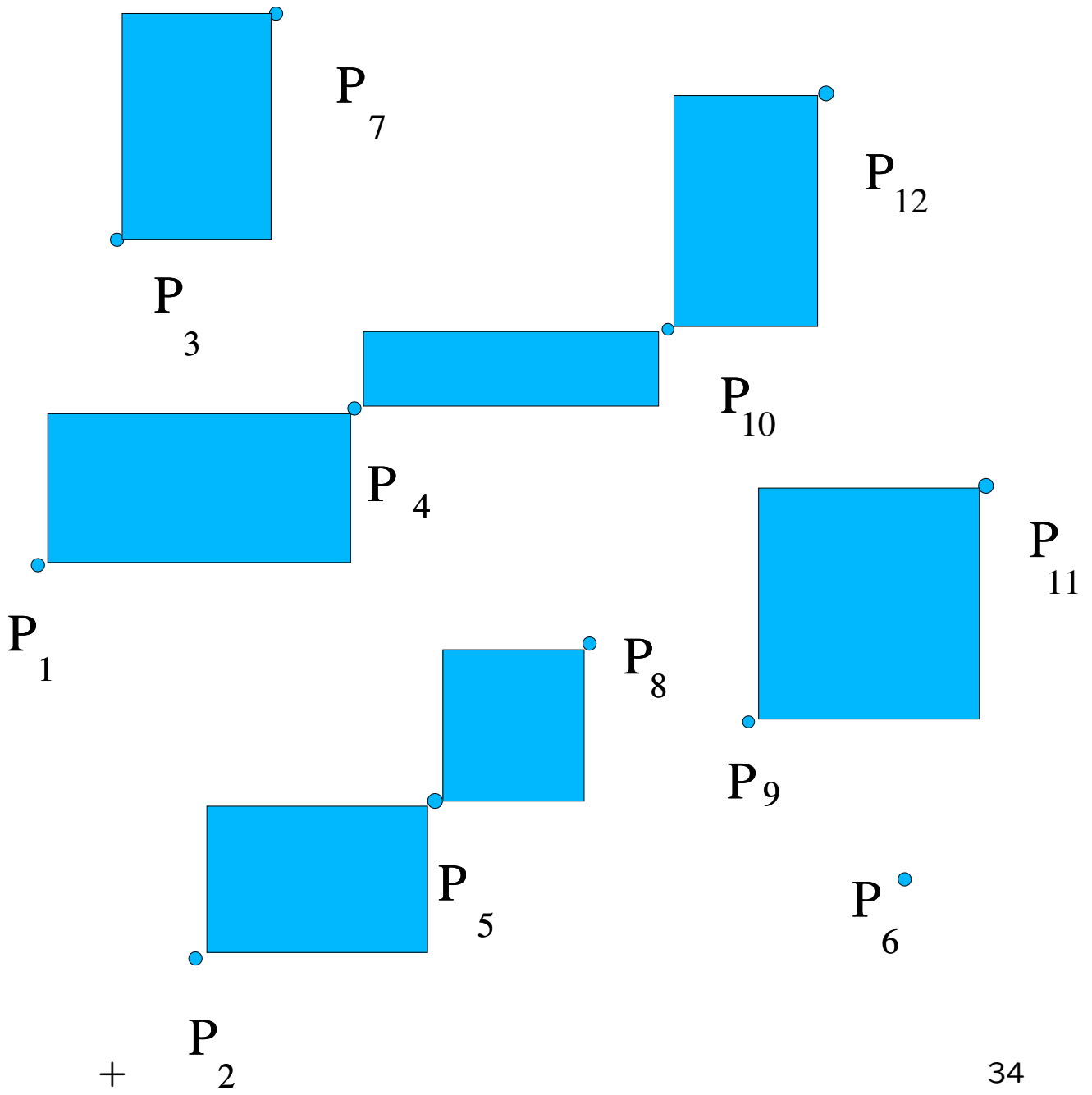
A point  $p$  **directly dominates** another point  $q$  iff  $q \leq p$  and there is no other point  $k$  such that  $q \leq k \leq p$ .

Another interpretation of direct dominance is that a point  $p$  directly dominates point  $q$  if there is no other point inside the rectangle defined by the two points  $p$  and  $q$  as diagonally opposite corners and  $q \leq p$ .

+

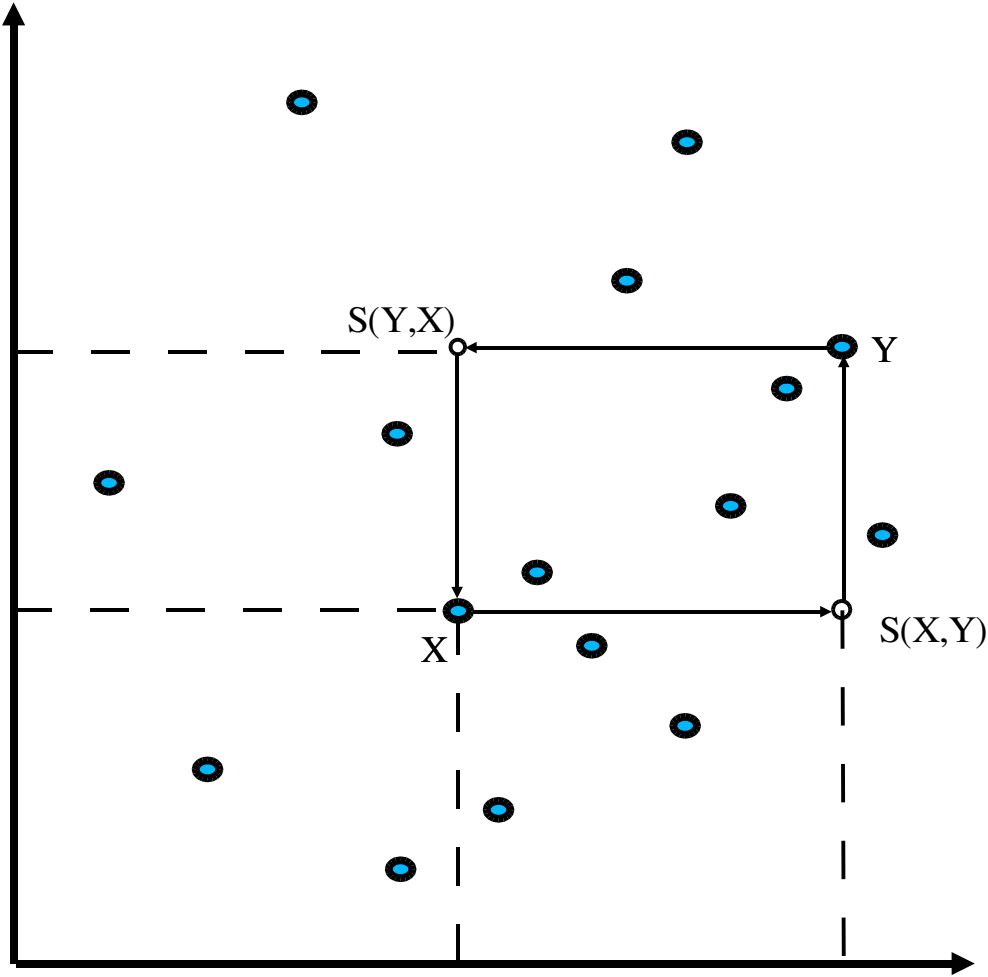
+

+



+

+



+

+

+

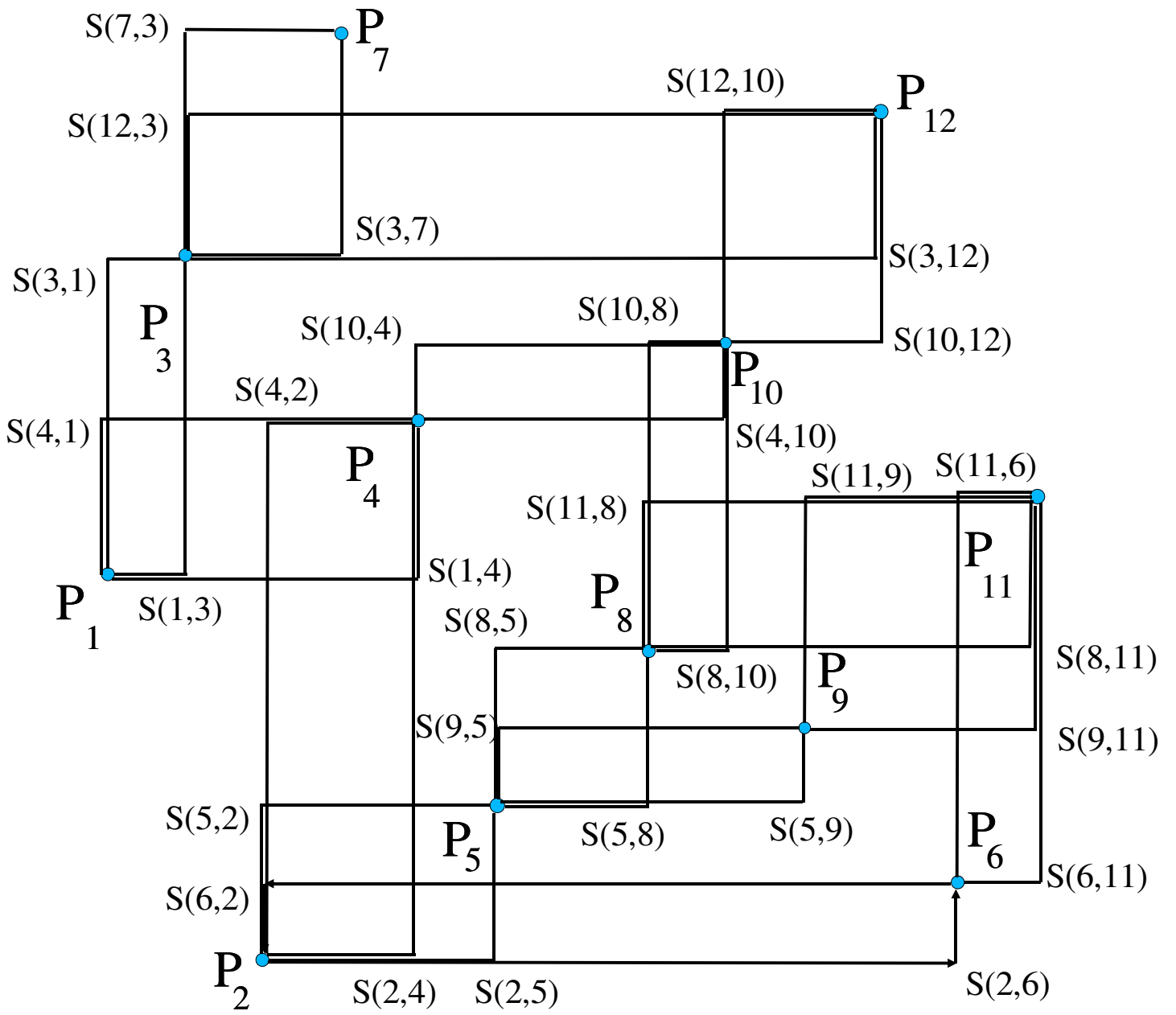
## Algorithm Direct Domination

```
function  
Direct_Domination(P:set_of_points)  
begin  
  
  i for all pairs of points  $x, y, x \in lt(y)$  do  
    /* in parallel */  
    if  $SizeOf\_lt(y) - SizeOf\_lt(S(x, y)) -$   
       $SizeOf\_lt(S(y, x)) + SizeOf\_lt(x) == 0$   
      /* the rectangle defined by corners  $x$  and  $y$  is  
      empty */  
      then  $x \rightarrow y$  /*  $y$  directly dominates  $x$  */  
  
end.
```

+

+

+



+

+

+

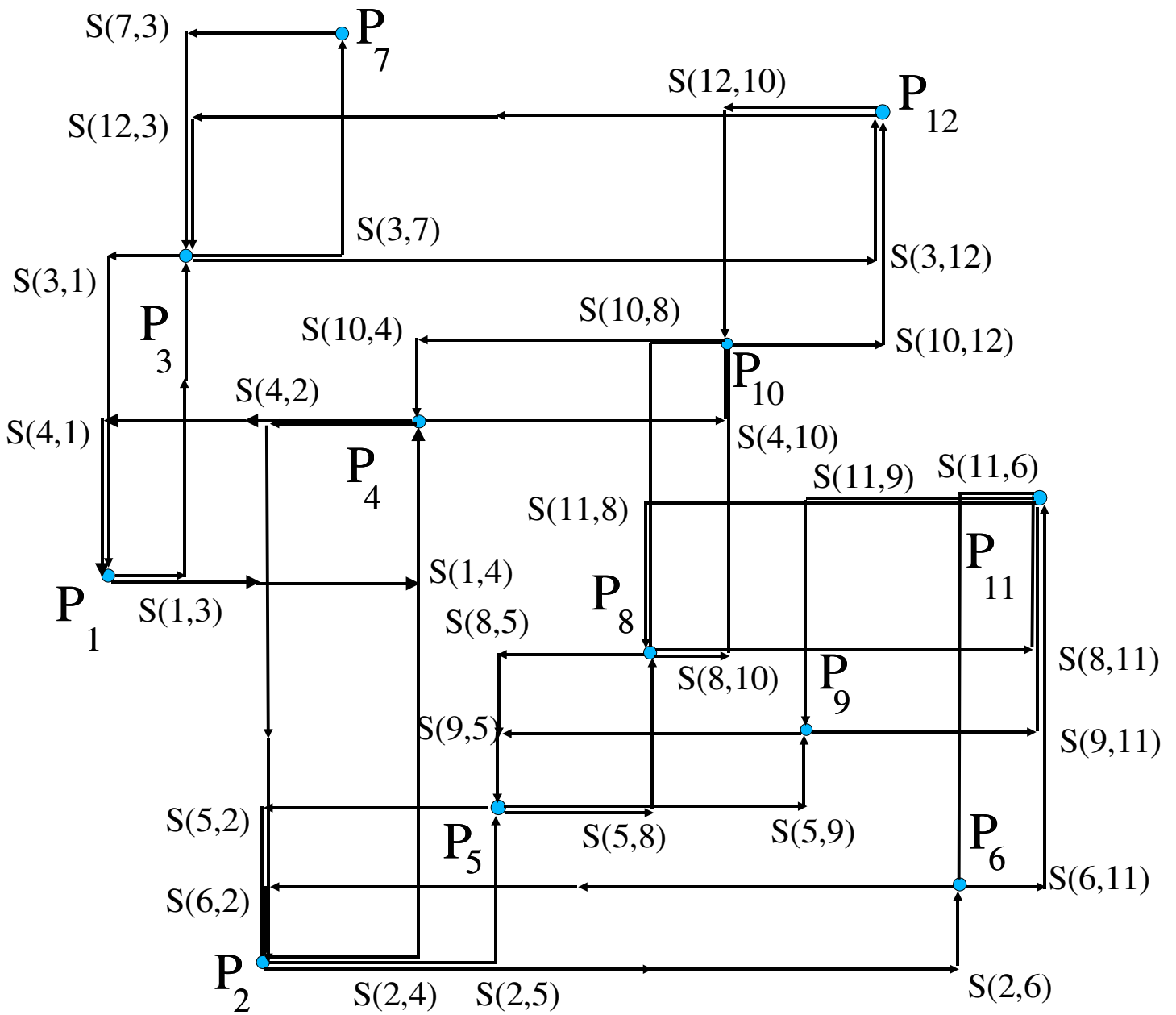
## Algorithm Height (parallel)

```
function  
height(parallel)(P:set_of_points)  
begin  
  
iv for every point  $p$  compute  $lt(p)$ ; and  
    $SizeOf\_lt(p)$   
iv for every pair of points  $(x, y) \in P$  do in parallel  $Direct\_Domination(P)$ ;  
iv Orient the rectangles counter-clockwise with  $x$   
   and  $y$  as diagonally opposite corners;  
iv Assign weight of zero to all horizontal segments  
   and a weight of +1 and -1 to vertical segments  
   going north and south respectively.  
iv for each point  $p$  in parallel stitch the rectan-  
   gles which share  $p$ .  
iv Height of a point is the sum of the +1's.  
  
end.
```

+

+

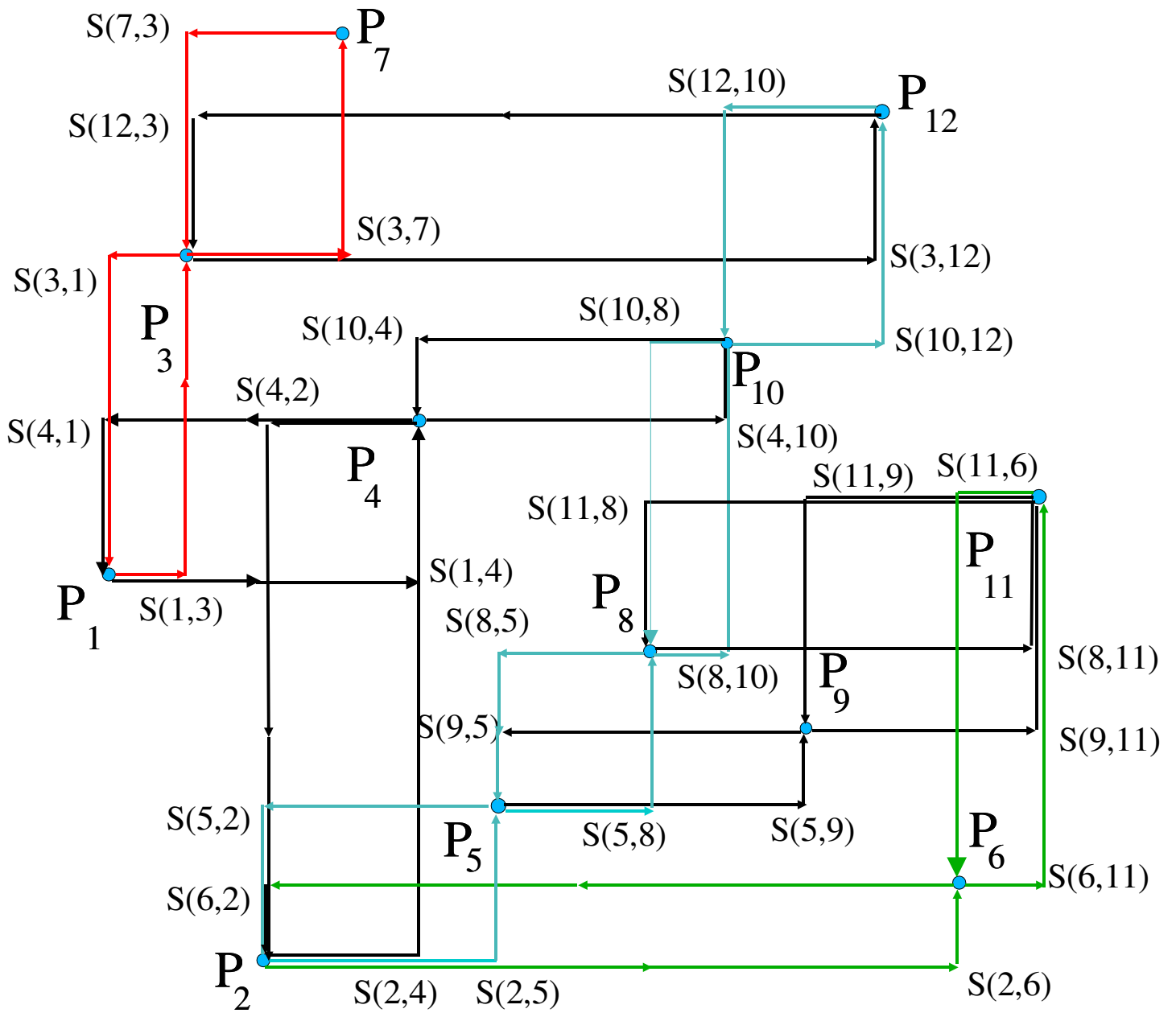
+



+

+

+

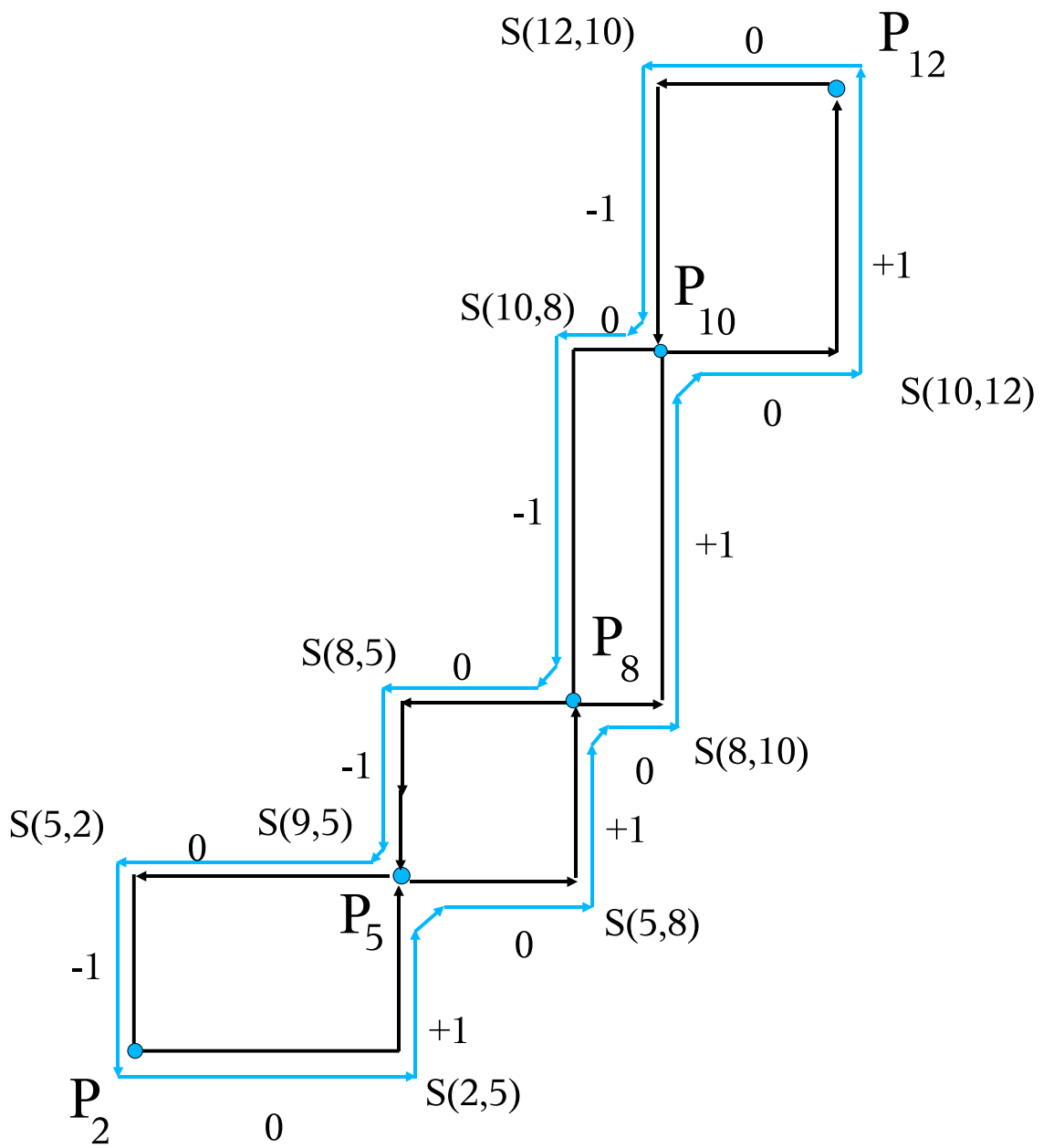


+

40

+

+



+

+

+

## Function LtAndSizeOf\_Lt

General Prefix Problem:  $f(m)$  and  $y(m)$  be given sequence of integers  $1 \leq m \leq n$ .

There is a binary associative operator '\*' on the f-elements;

the y-elements can be compared by a linear order '<'.

The general prefix problem is to compute the sequence of prefixes:

$D_m = f(j_1) * f(j_2) * f(j_3) * \dots * f(j_k)$  where  $j_1 < j_2 < \dots < j_k$  and  $\{j_1, \dots, j_k\}$  is the set of indices  $j < m$  for which  $y(j) < y(m)$ , where  $m$  is each index from 1 to  $n$ .

If the  $f(m)$  is set to **reference of point**  $m$  for all the points,  $y(m)$  is set to the y-coordinates of the point  $m$  and the \* operation is set to the union operation ' $\cup$ ' (which is associative) then the general prefix function [SS89] computes the  $lt(m)$  for each point  $m$ .

+

+

+

**How these ideas can be useful**

+

43

+

+

## Computing Maximum Chain

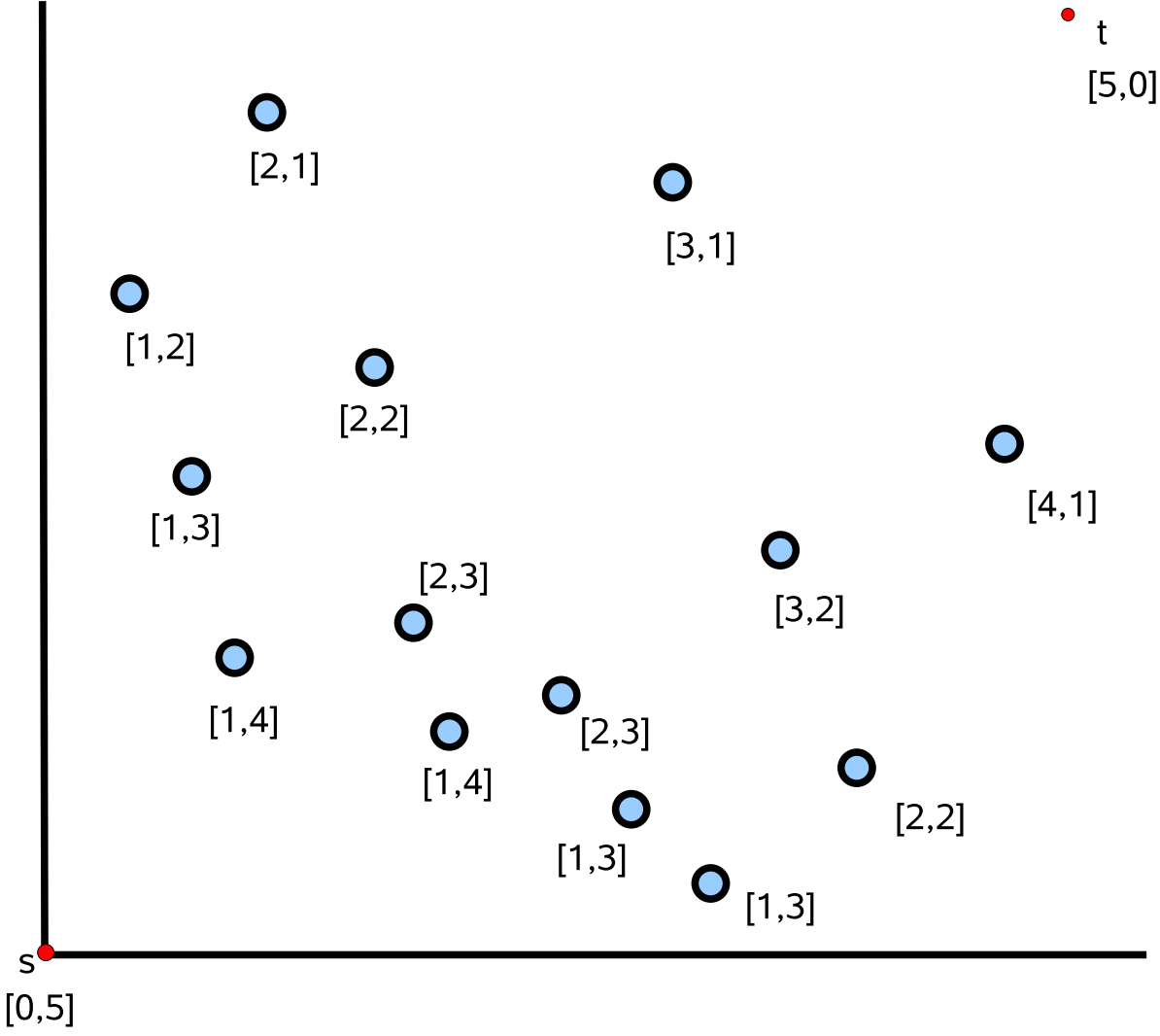
### Definition: Depth of a Point

The **depth**  $dt(p)$  **of a point**  $p$  is defined as the size (length) of a maximum chain in the subset  $\{q \in P \mid q > p\}$ , (i.e, size of maximum chain in the subset of points larger than  $p$  that begin at  $p$ ).

+

+

+

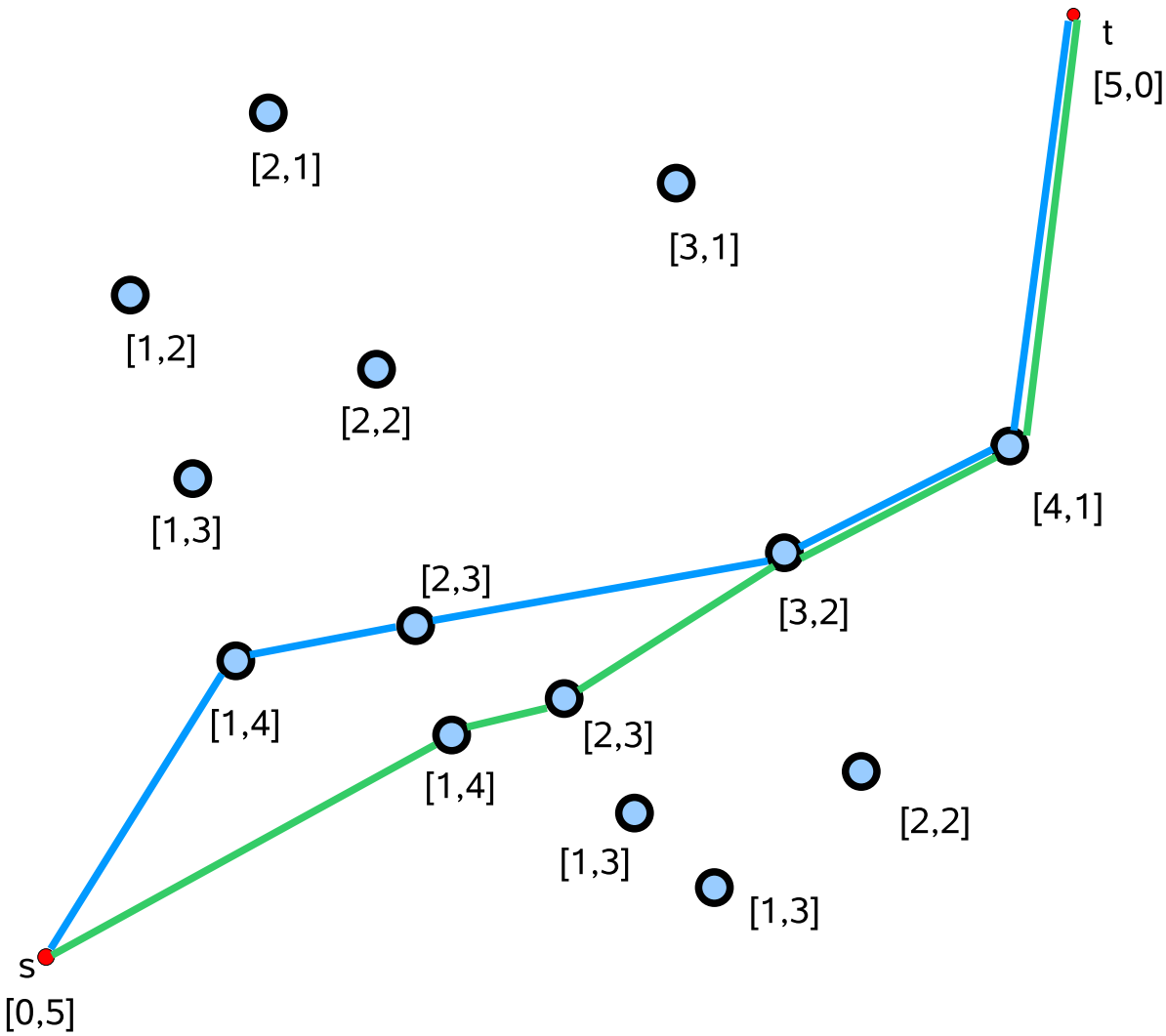


[Height,Depth] of points

+

+

+



+

+

+

## **Automatic Routing in Well-shaped Meshes**

*Meshes are well-shaped if the angle and/or the aspect ratios of their elements are bounded within some values.*

*Most of the meshes used in scientific simulations are well-shaped.*

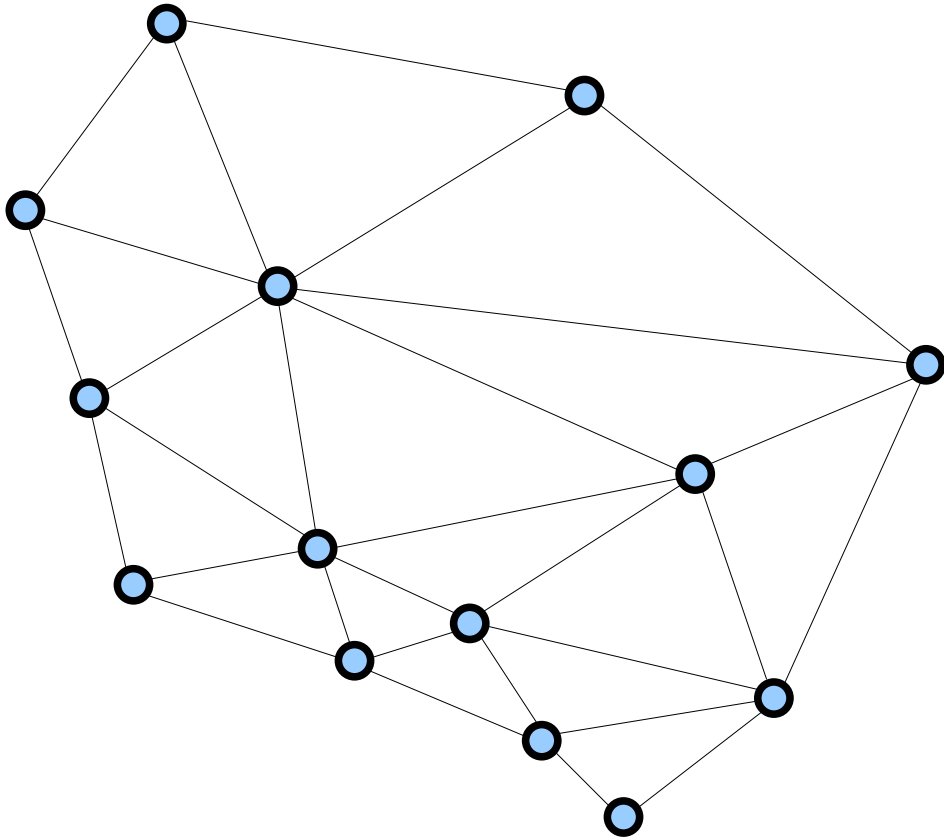
*Overlap graphs [Miller93] contain well-shaped meshes and planar graphs.*

+

47

+

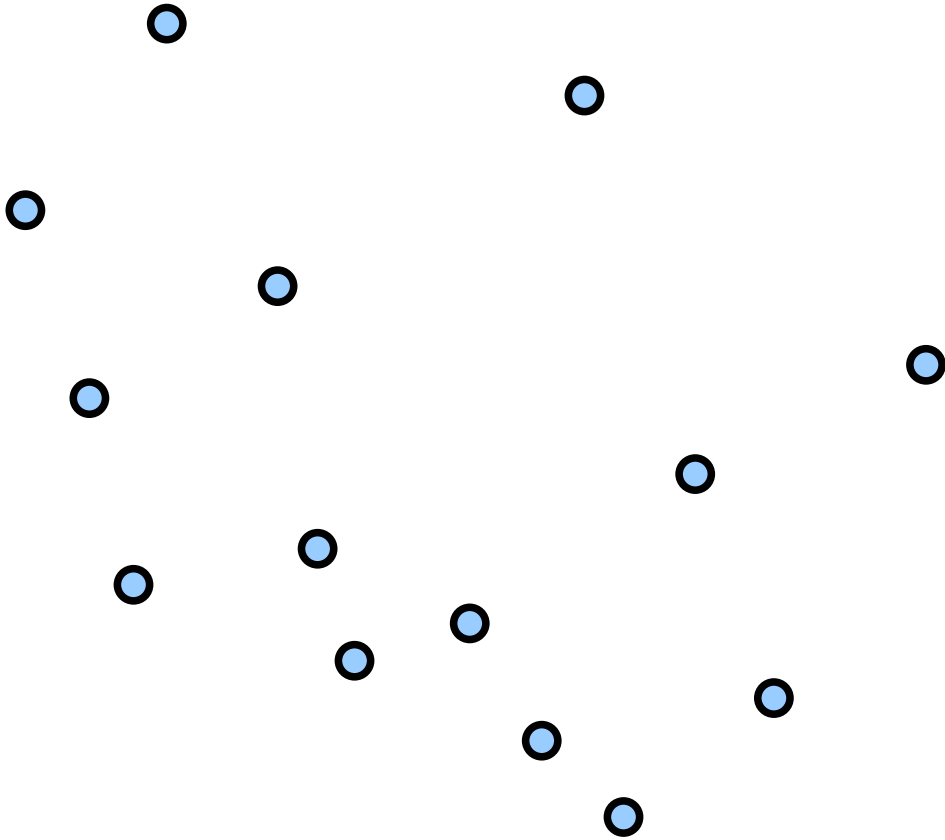
+



+

+

+



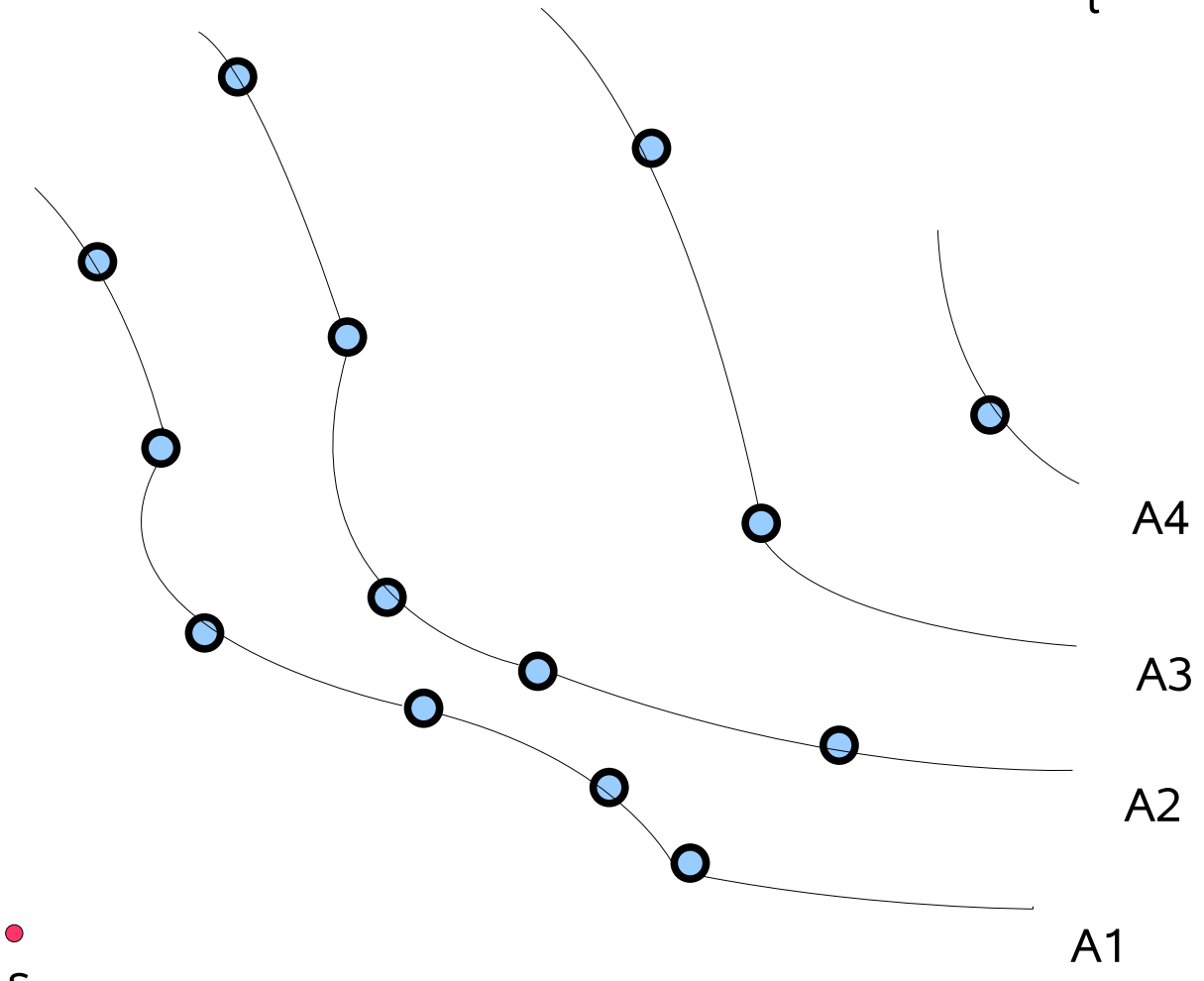
Nodes of a finite element mesh

+

+

+

t



s

A4

A3

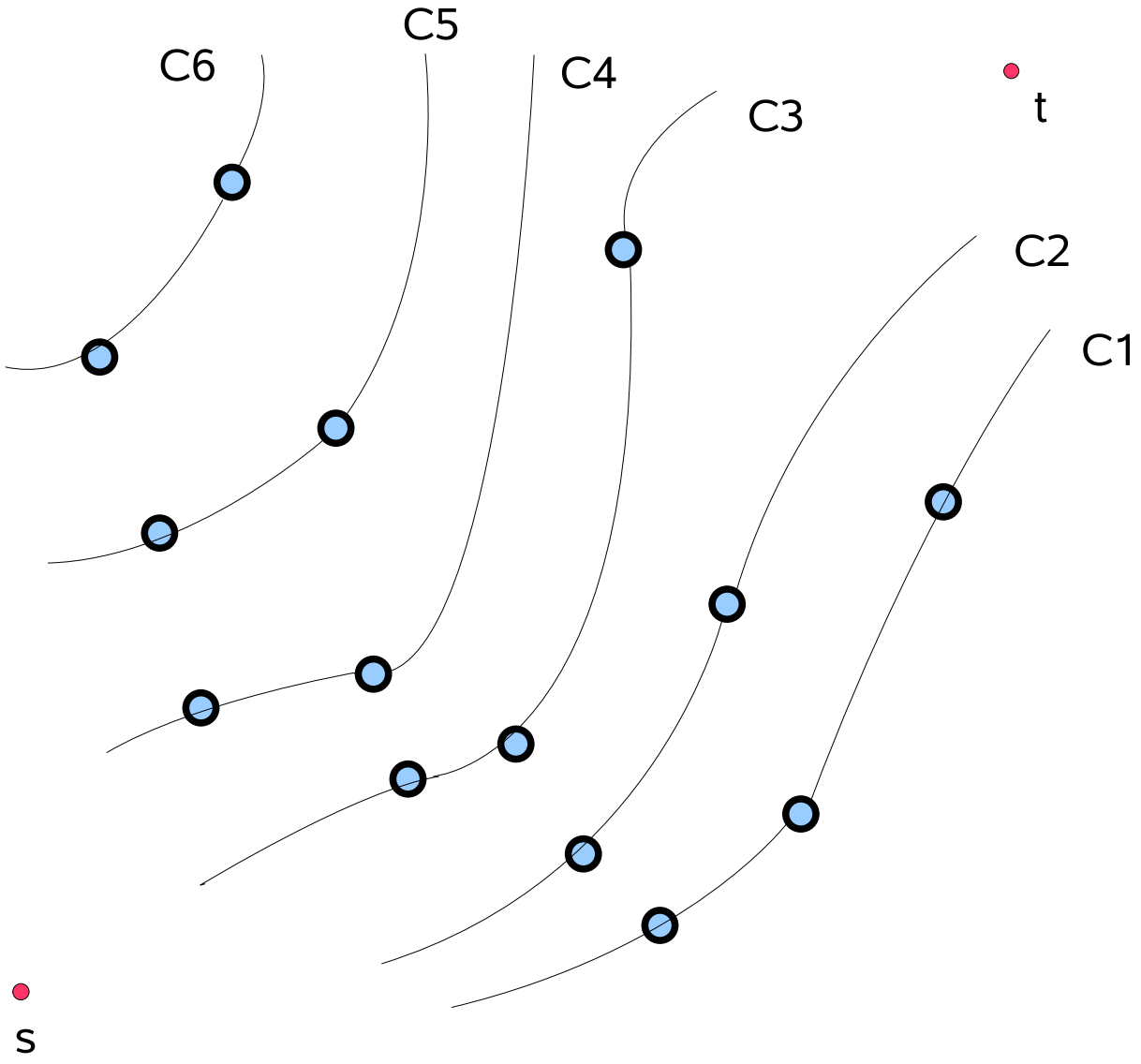
A2

A1

+

+

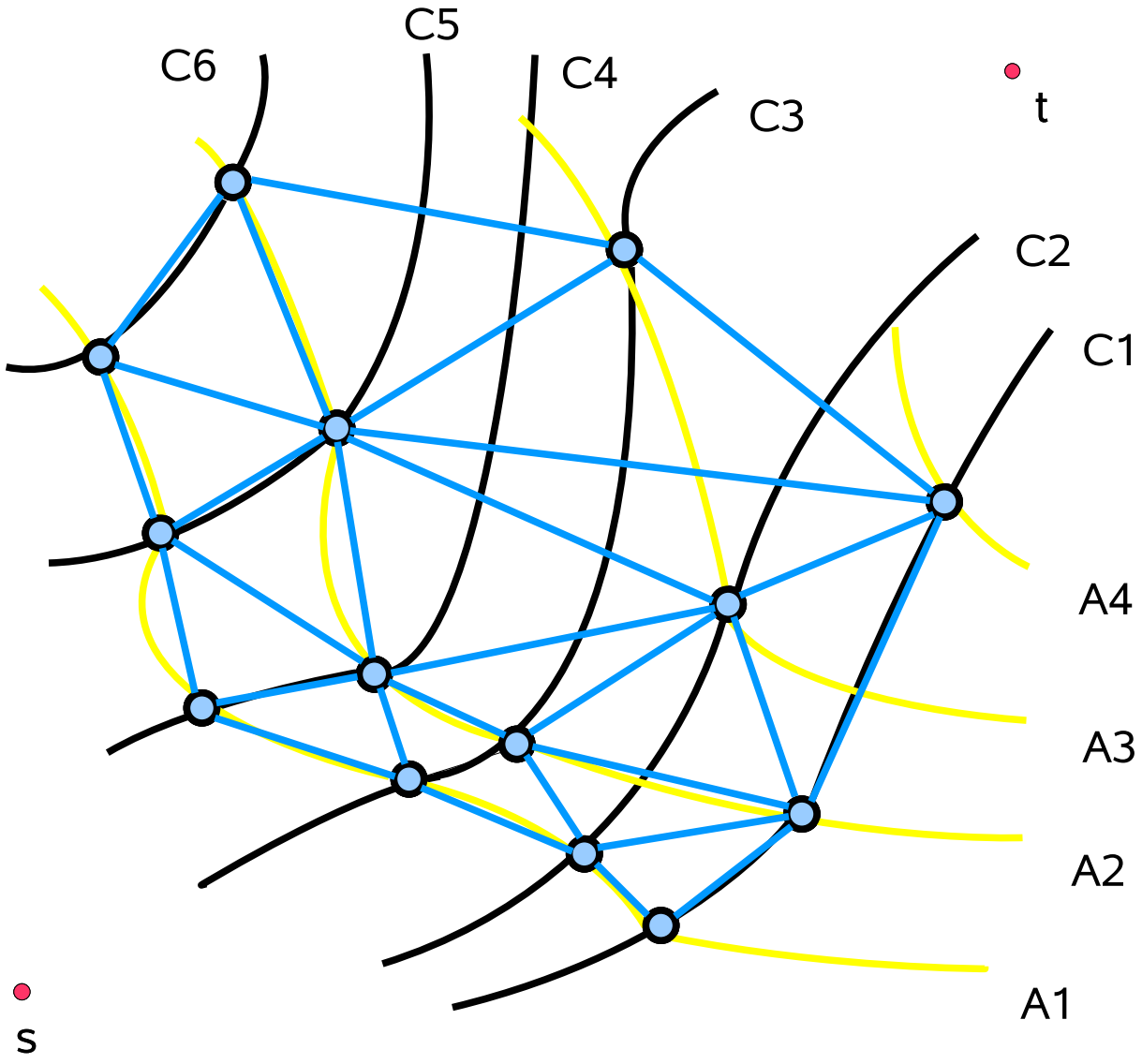
+



+

+

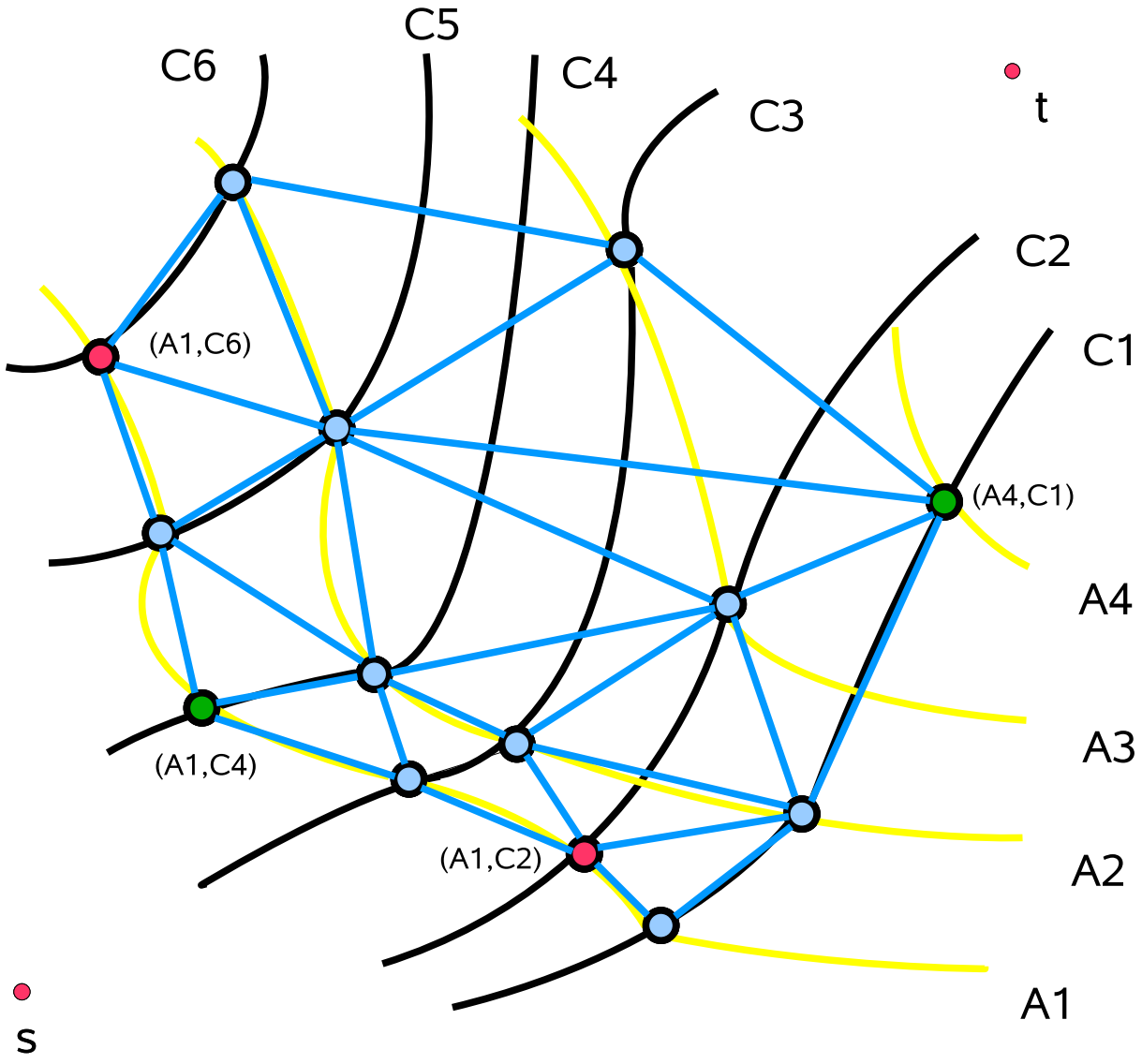
+



+

+

+



+

+

+

## **Conclusion**

Parallel Algorithms are presented to find canonical anti-chain and chain partition in planar point sets. These structures provide a basis for solving other problems for points in a plane.

+

+

+

## References

**FW93** Stefen Felsner and Lorenz Wernisch, Maximum k-chains in planar point sets: Combinatorial structures and Algorithms, ACM Symposium on Theory of Computing, 1993.

**MW92** J. Matousek and E. Welzl, Good Splitters for counting points in a triangle, Journal of Algorithms, 1992.

**SS89** Fredrick Springsteel and Ivan Stojmenovic, Parallel General Prefix Computations with Geometric, Algebraic and other Applications, International Journal of Parallel Programming, 1989.

**Vin84** G. Viennot, Chains and Anti-chains Families, Grids and Young Tableaux, North Holland Math. Stud. 1984.

**Miller93** G.L. Miller, S. H. Teng, W. Thurston, and S.A. Vavasis, Automatic Mesh Partitioning, Graph Theory and Sparse Matrix Computation, 57-84, Springer Verlag, Heidelberg and New York, 1993.

+