- initial algorithm

list L[1], . . ., L[N]

(1) compute maximum score MAX

(2) for I from 1 to N do

    L[I] := L[I] * 100 / MAX

- initial algorithm

  list L[1], . . ., L[N]

  (1) compute maximum score MAX
  (2) for I from 1 to N do
  L[I] := L[I] * 100 / MAX

- improved algorithm

  (1) compute maximum score MAX
  (2) FACTOR := 100/MAX
  (3) for I from 1 to N do
  L[I] := L[I] * FACTOR

- binary search (Kaldewaij, page 100)

```
x,y := 1,N;
do x≠y →
   z := (x+y) div 2;
   if L[z]≤a → x := z □ L[z]>a → y := z fi
od
```

$f, g \colon \mathbb{N}_{>0} \to \mathbb{N}_{>0}$

$f \in O(g)$

$$\iff \quad \lim_{n\to\infty} \frac{f(n)}{g(n)} < \infty$$

$$\iff \quad \exists c\, \forall n\colon\; f(n) \le c \cdot g(n)$$

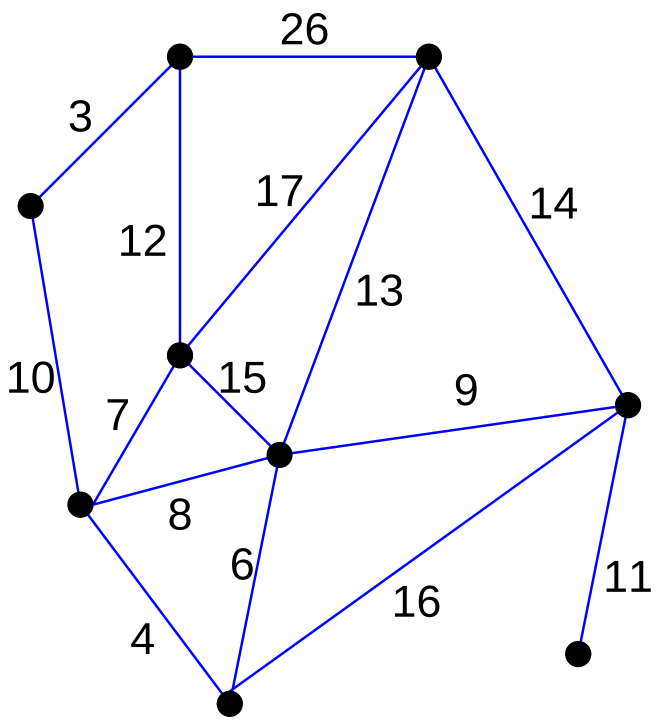
$f \in \Theta(g)$

$$\iff \quad 0 < \lim_{n\to\infty} \frac{f(n)}{g(n)} < \infty$$

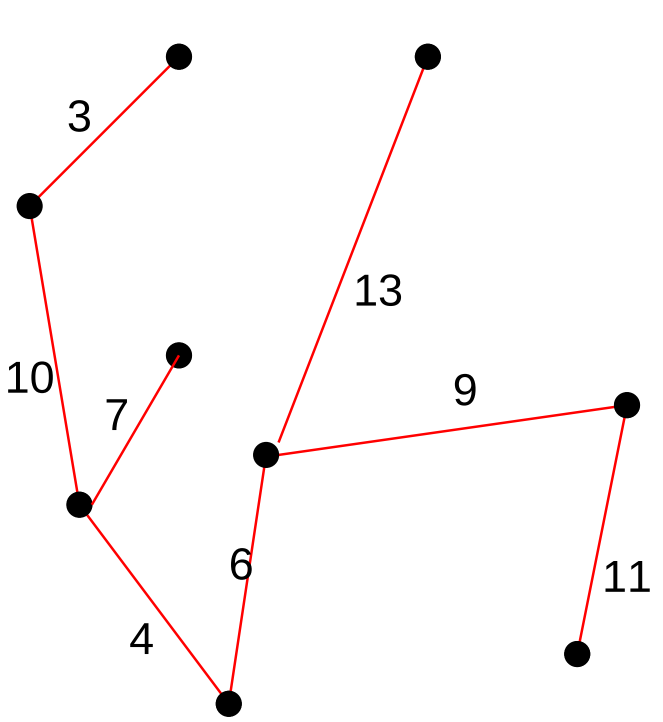
$$\iff \quad \exists c_1, c_2\, \forall n\colon\; c_1 \cdot g(n) \le f(n) \le c_2 \cdot g(n)$$

N	$1 + \log_2 N$
10	4
100	7
1000	10
a million	20
a billion	30
a billion billion	60

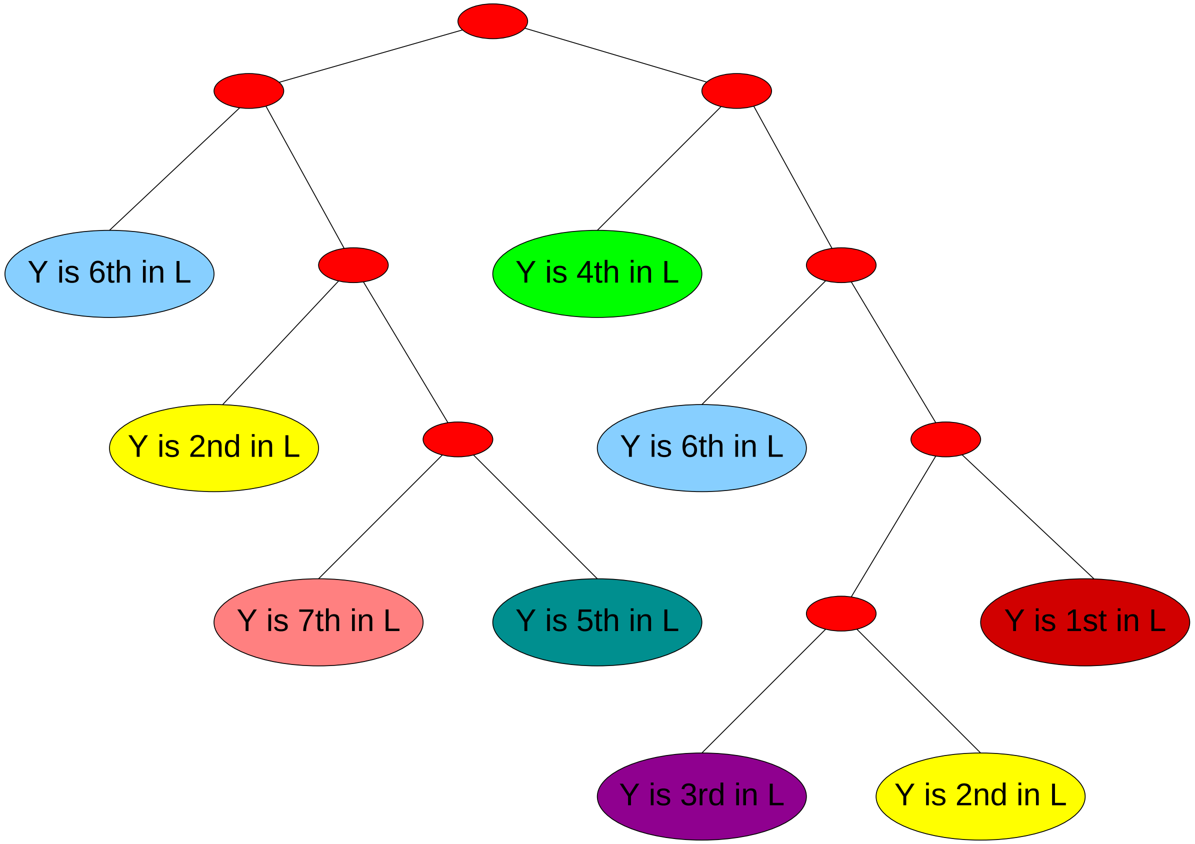
- city network and minimal spanning tree



(not drawn to scale)

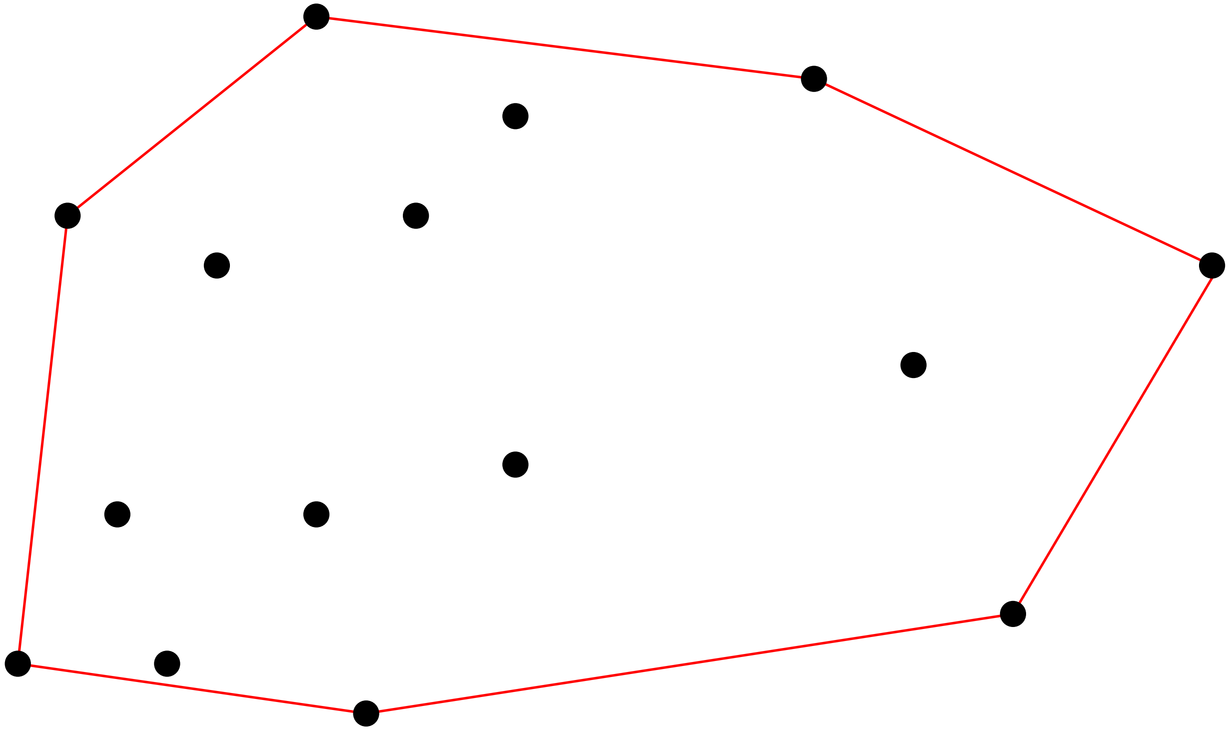


total cost: 63



binary tree of n nodes has height at least $\log_2 n$

- a convex hull



- O(N x log N) algorithm

  (1) find lowest pont P_1

  (2) sort remaining points by angle $P_2, P_3, \ldots$

  (3) add P_1, P_2 to current hull

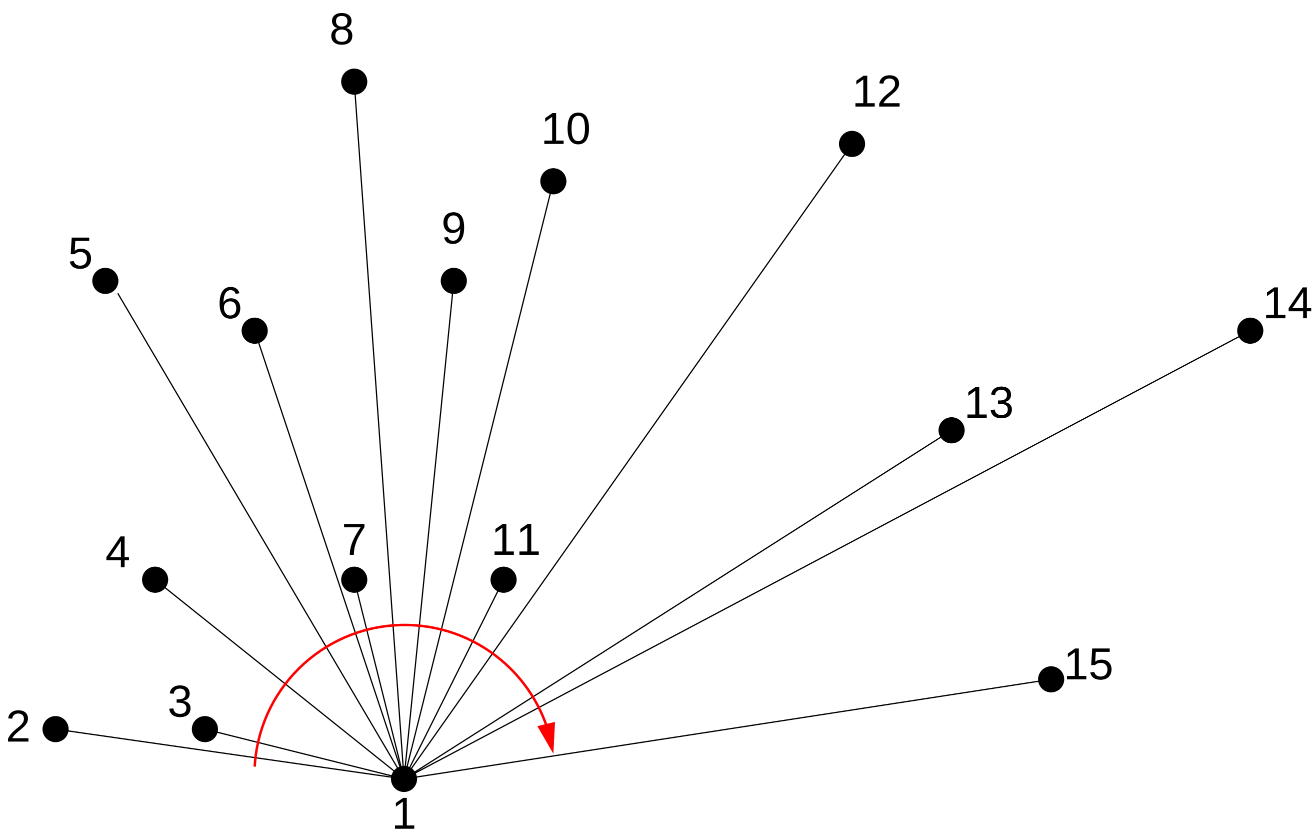
  (4) for I from 3 to N do

  add P_I to current hull

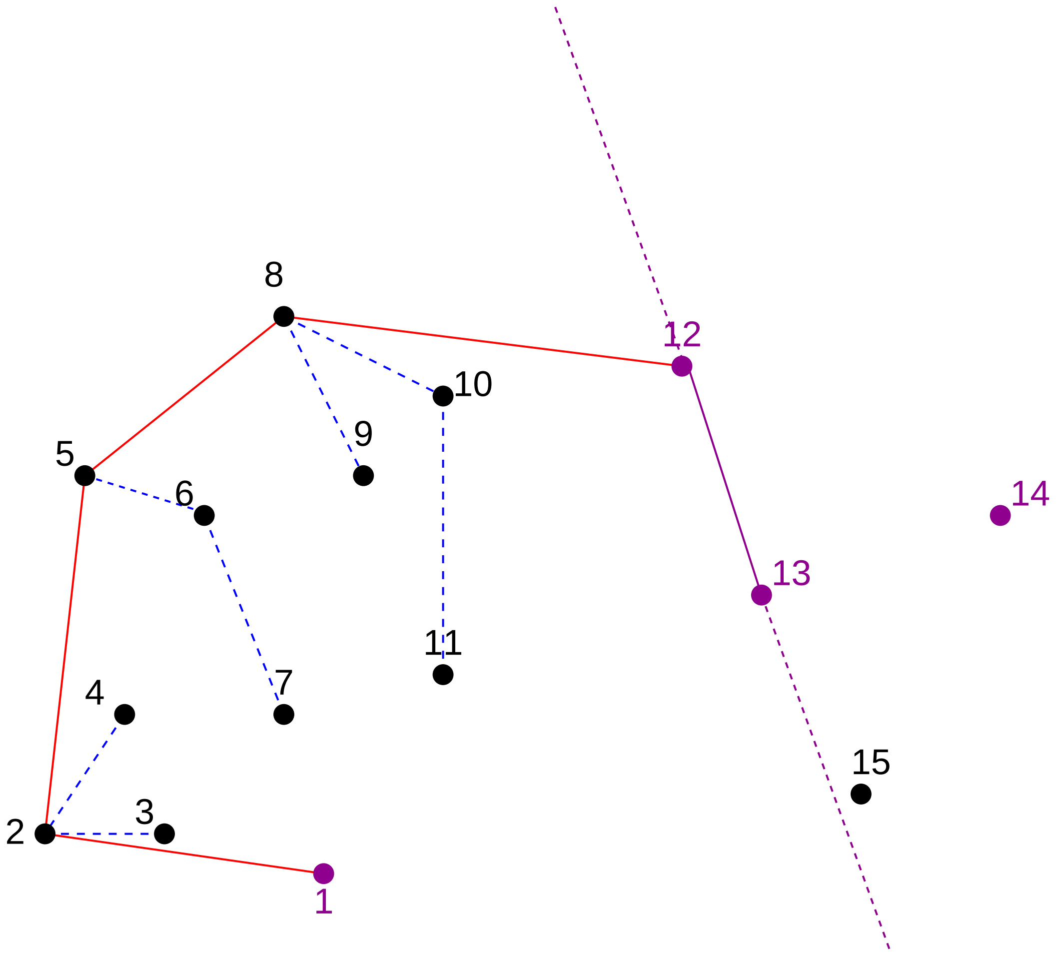
  remove backwards wrong points P_J

  until a point is OK

- points sorted by angle



- the first few fences



- O(N x log N) algorithm

(1)	find lowest pont P_1	O(N)
(2)	sort remaining points by angle $P_2, P_3, \ldots$	O(N x log N)
(3)	add P_1, P_2 to current hull	O(1)
(4)	for I from 3 to N do	O(N)
	add P_I to current hull	
	remove backwards wrong points P_J	
	until a point is OK	

- sorting via convex hull

  values $x_1, x_2, x_3, \ldots$

  points $(x_1, x_1^2), (x_2, x_2^2), (x_3, x_3^2), \ldots$