Introduction to Algorithms

Meng-Tsung Tsai

12/12/2019

Beyond LP's Appearance

The convex hull problem

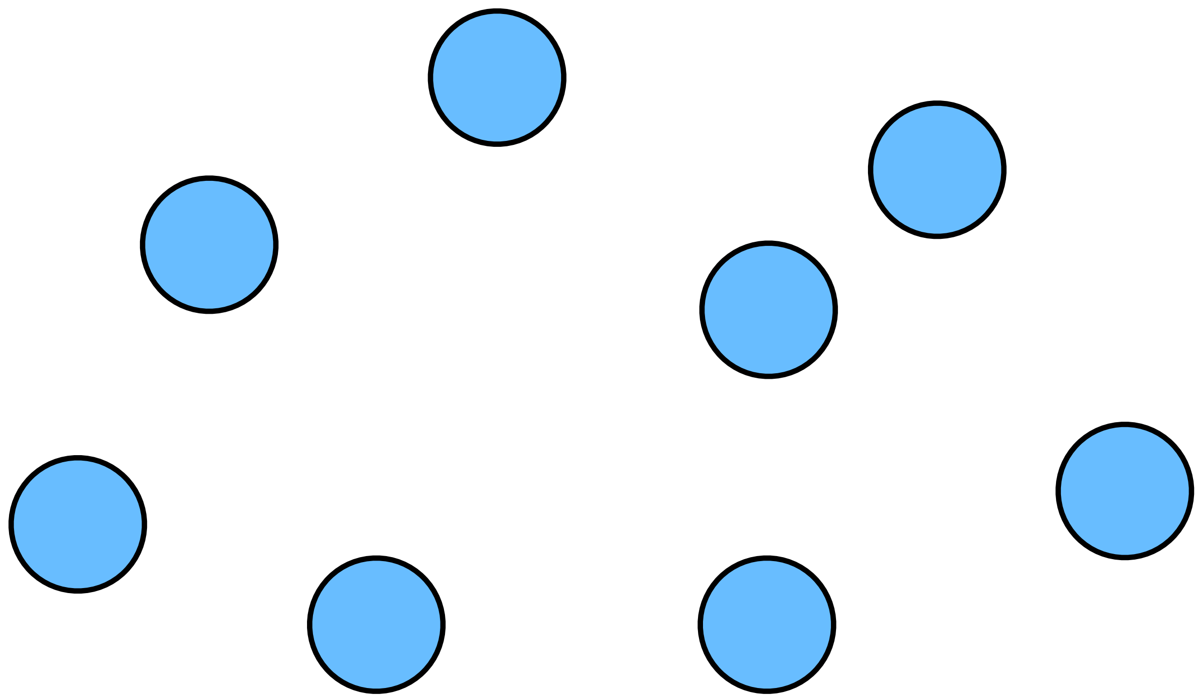
Input: given a set P of n points on a plane.

Output: the convex hull of P; that is, the smallest convex polygon that contains all points in P. We assume that the convex hull has h vertices (extreme points).

Recall that convex hulls can be computed in O(n log n) time.
By LP, we can solve it in O(n log h) time.

Ultimate convex hull algorithm

UH(P){ // find the upper hull



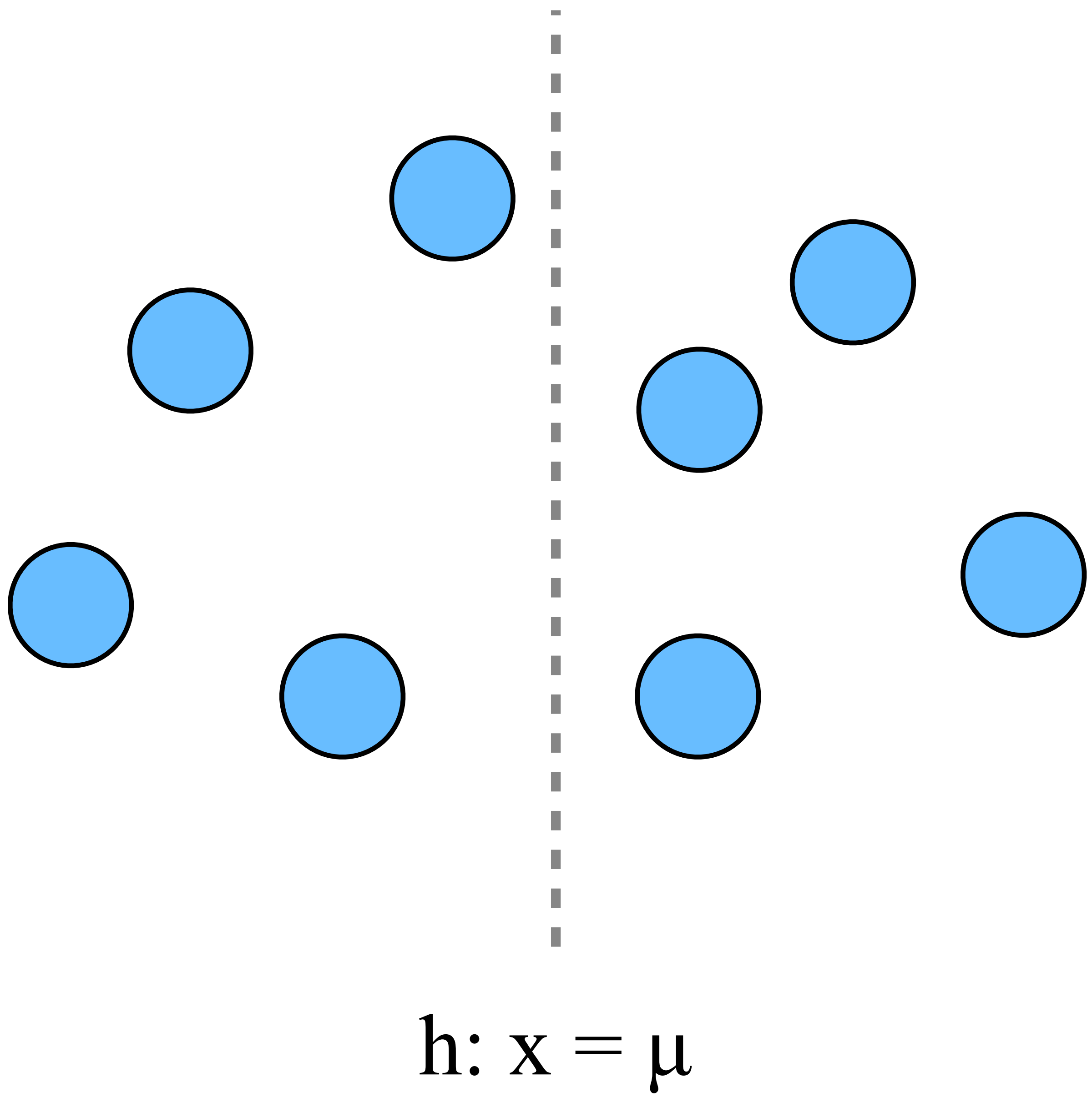
}

Ultimate convex hull algorithm

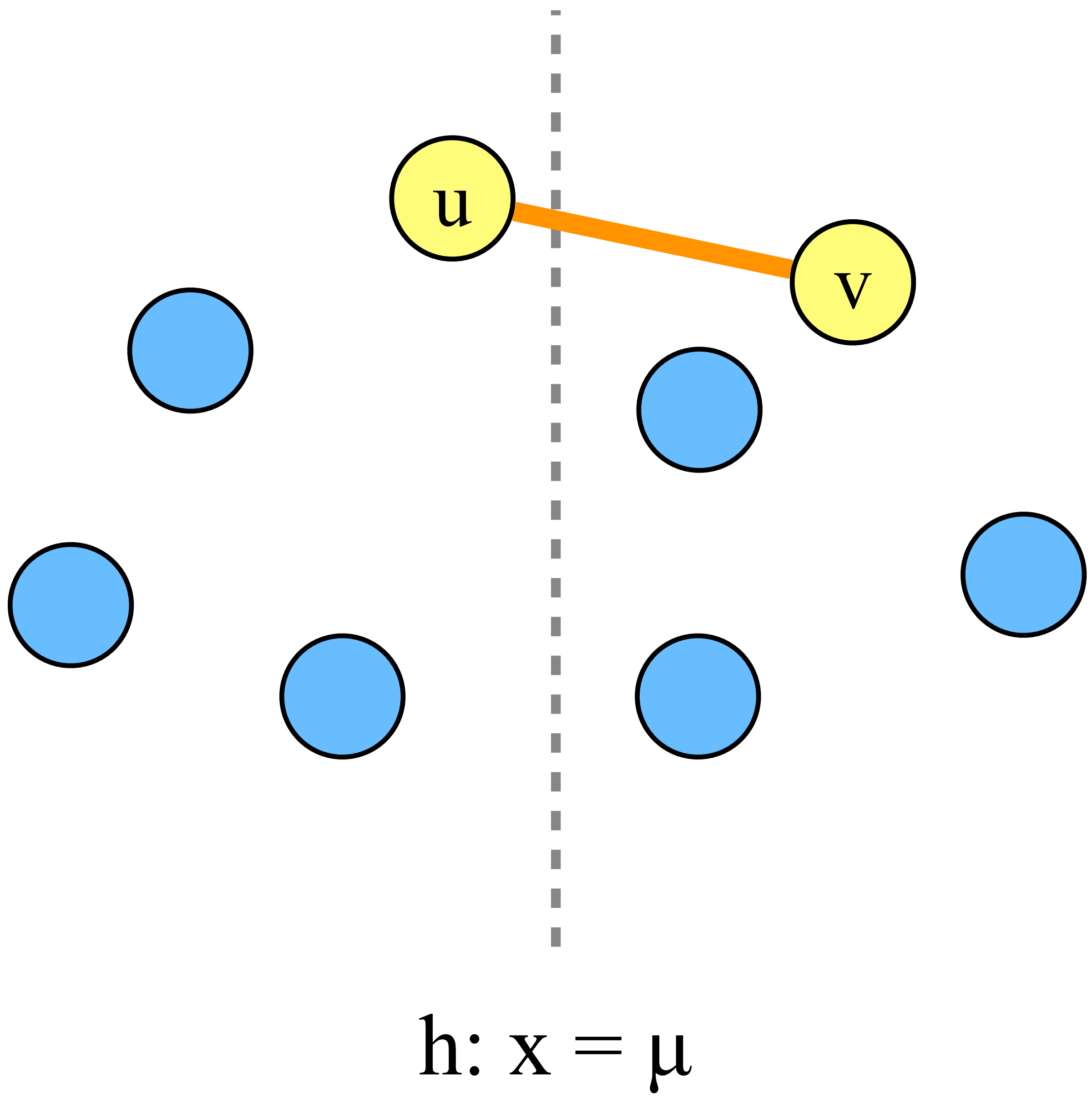
UH(P){ // find the upper hull

Step 1. Find a line h: $x = \mu$ that partitions the points in P evenly.

}



Ultimate convex hull algorithm



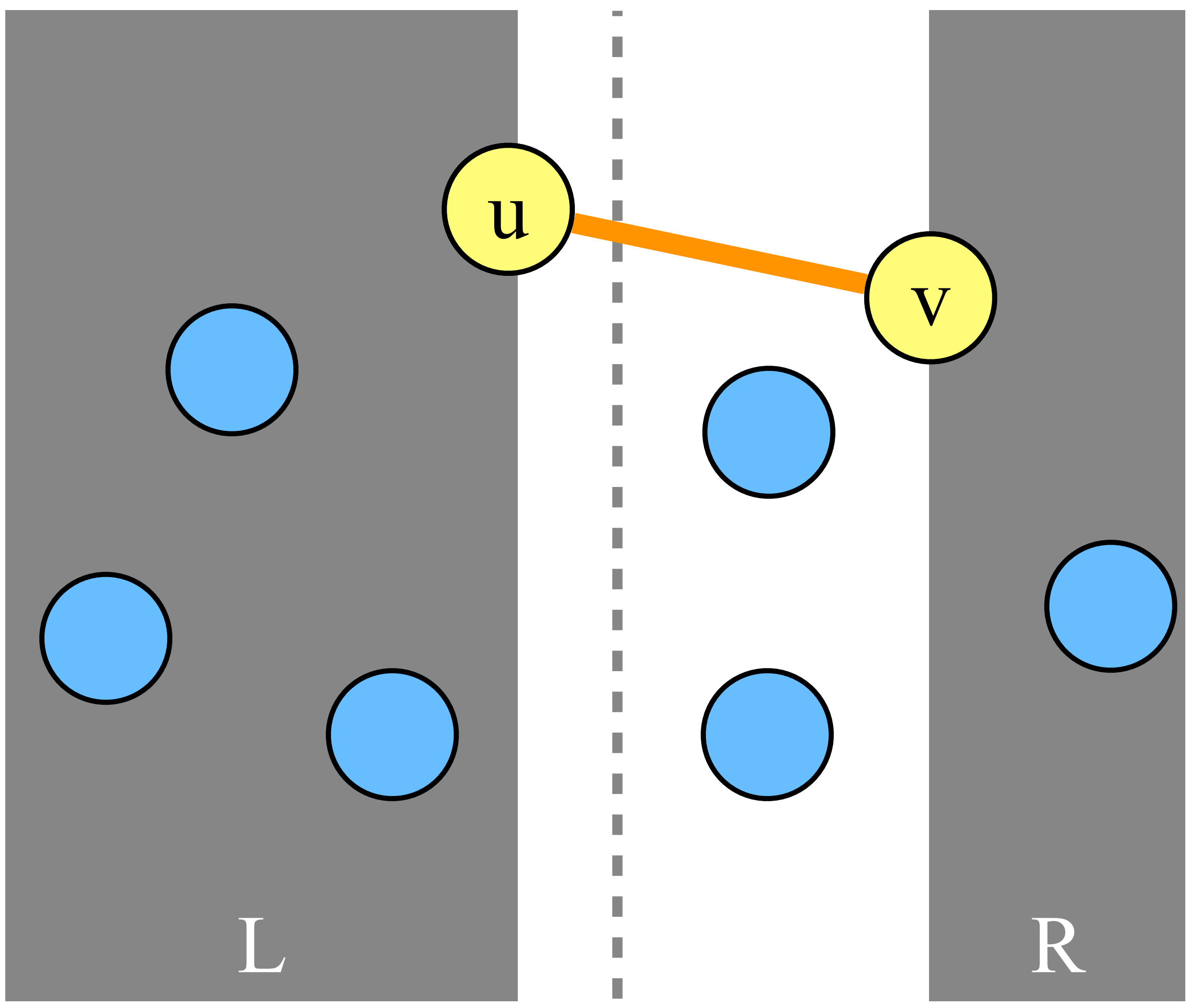
UH(P){ // find the upper hull

Step 1. Find a line h: $x = \mu$ that partitions the points in P evenly.

Step 2. Find the hull edge (u, v) that crosses h.

}

Ultimate convex hull algorithm



$h: x = \mu$

UH(P){ // find the upper hull

Step 1. Find a line $h: x = \mu$ that partitions the points in P evenly.

Step 2. Find the hull edge (u, v) that crosses h.

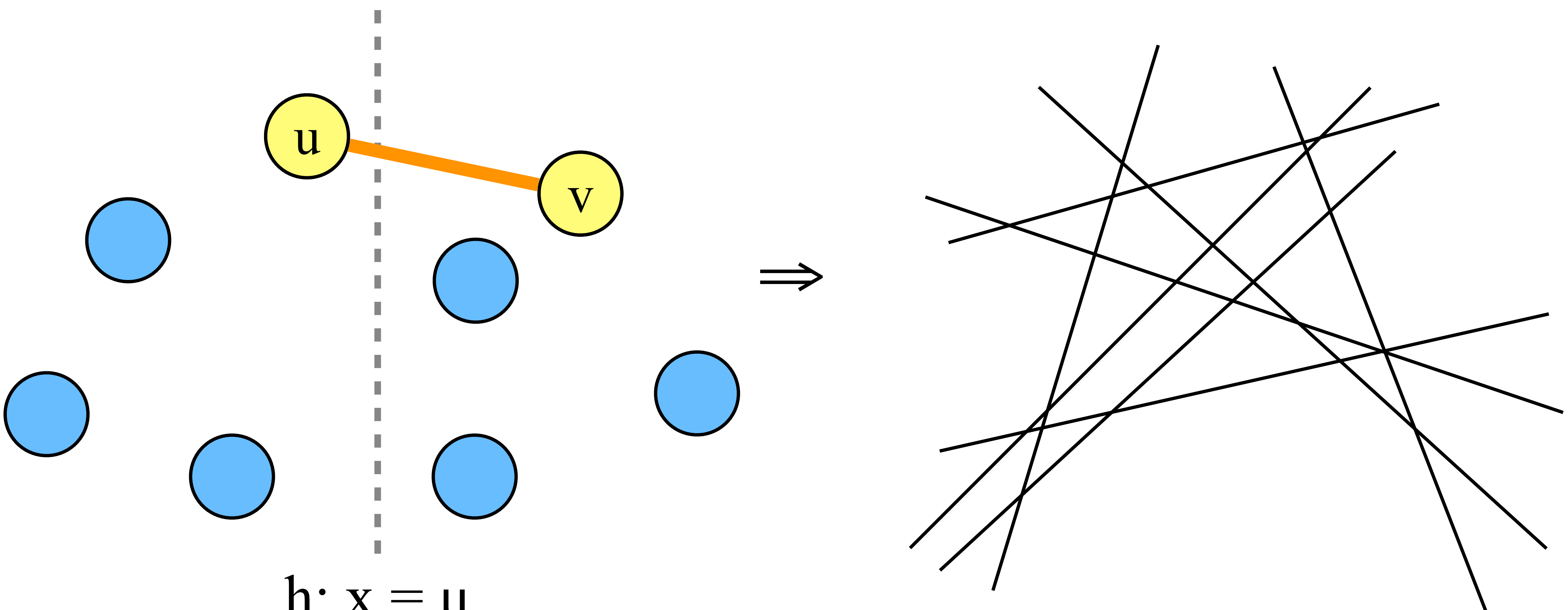
Step 3. Let
$L = \{p \text{ in } P: x(p) < x(u)\}$ and
$R = \{p \text{ in } P: x(p) > x(v)\}$.
Recurse on UH(L) and UH(R).

}

Finding the hull edge by an LP

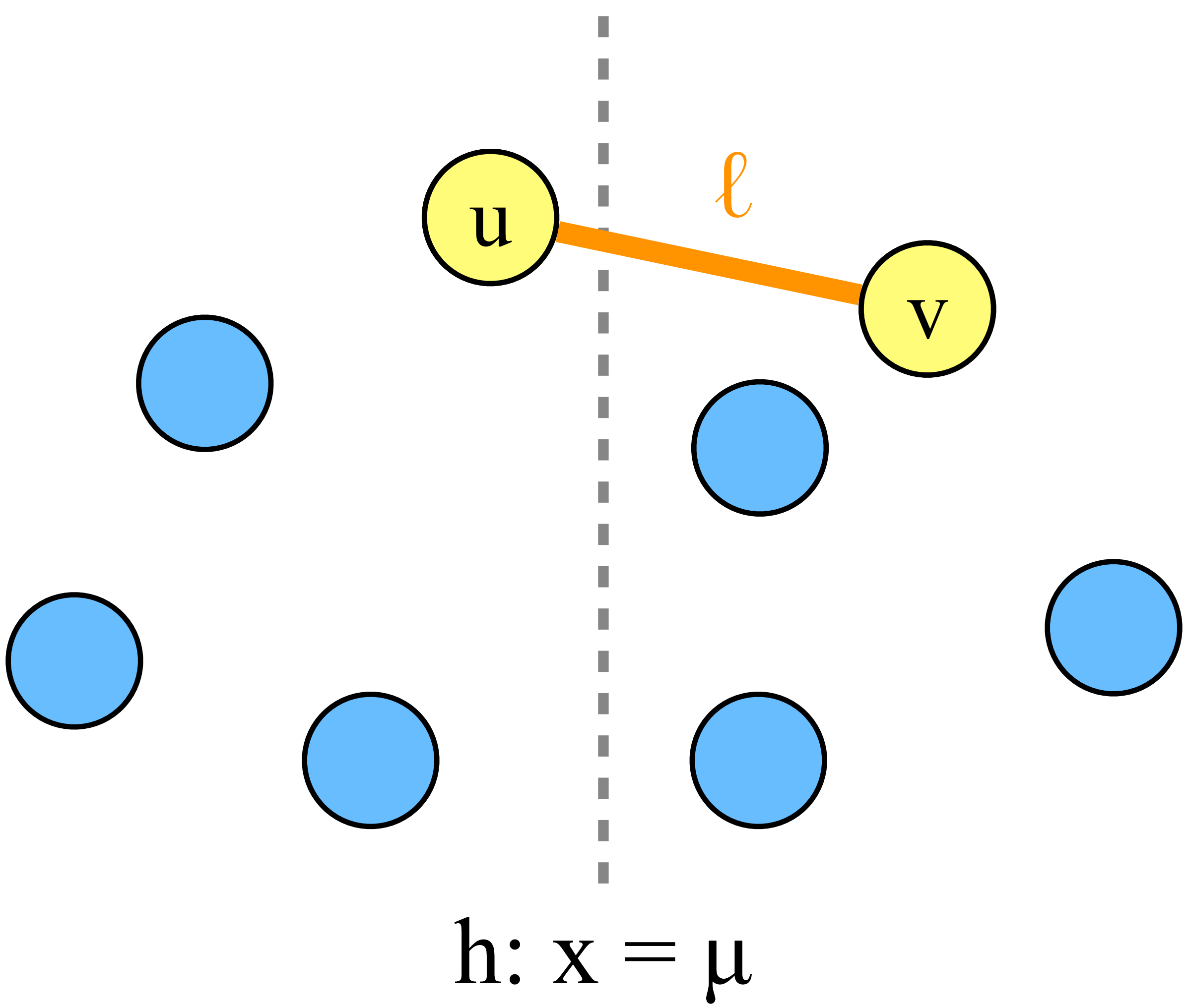
Point-Line Duality:

Map each point p = (a, b) into a line p*: y = ax - b, and map each line ℓ: y = mx - c into a point ℓ* = (m, c).

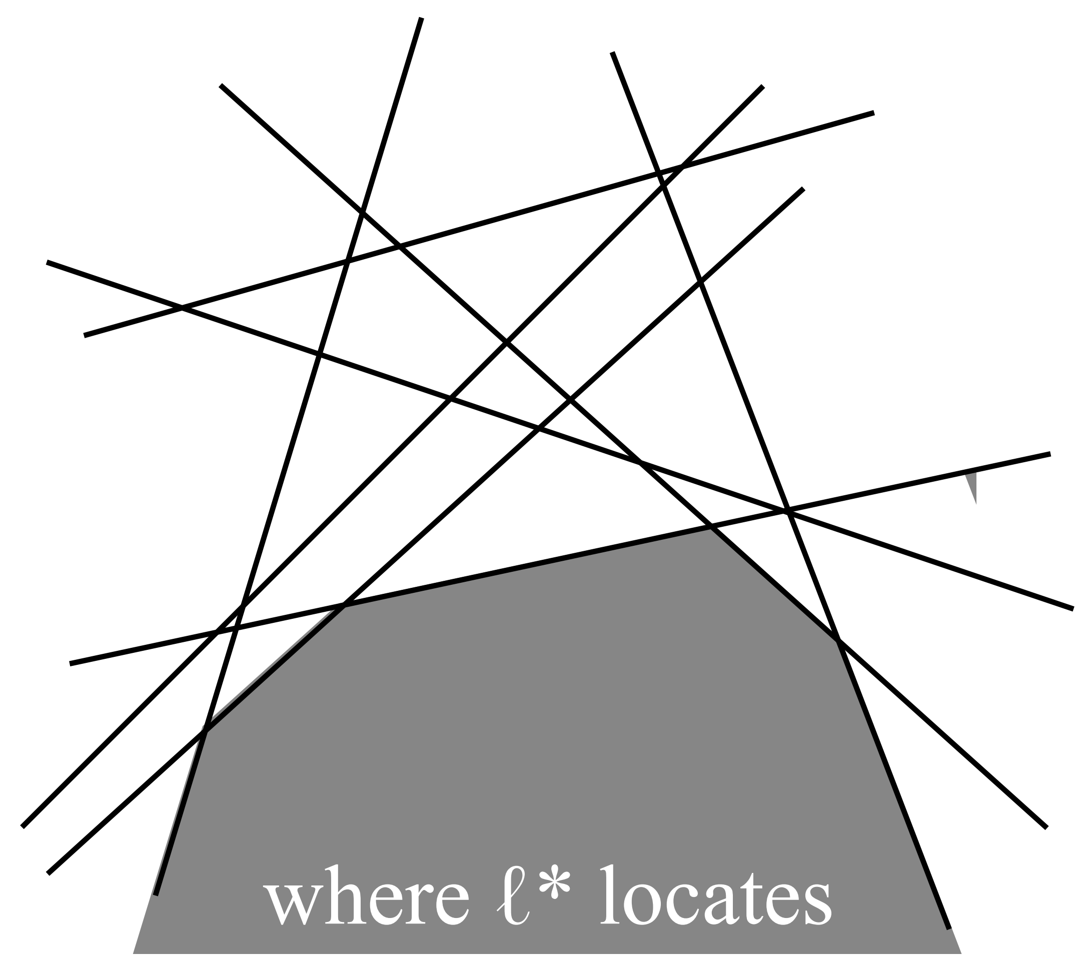


Finding the hull edge by an LP

Property: If p is below ℓ in the primal plane, then ℓ* is below p* in the dual plane. Since all points in P lie below the line ℓ that passes through u and v, ℓ* is below all p*.

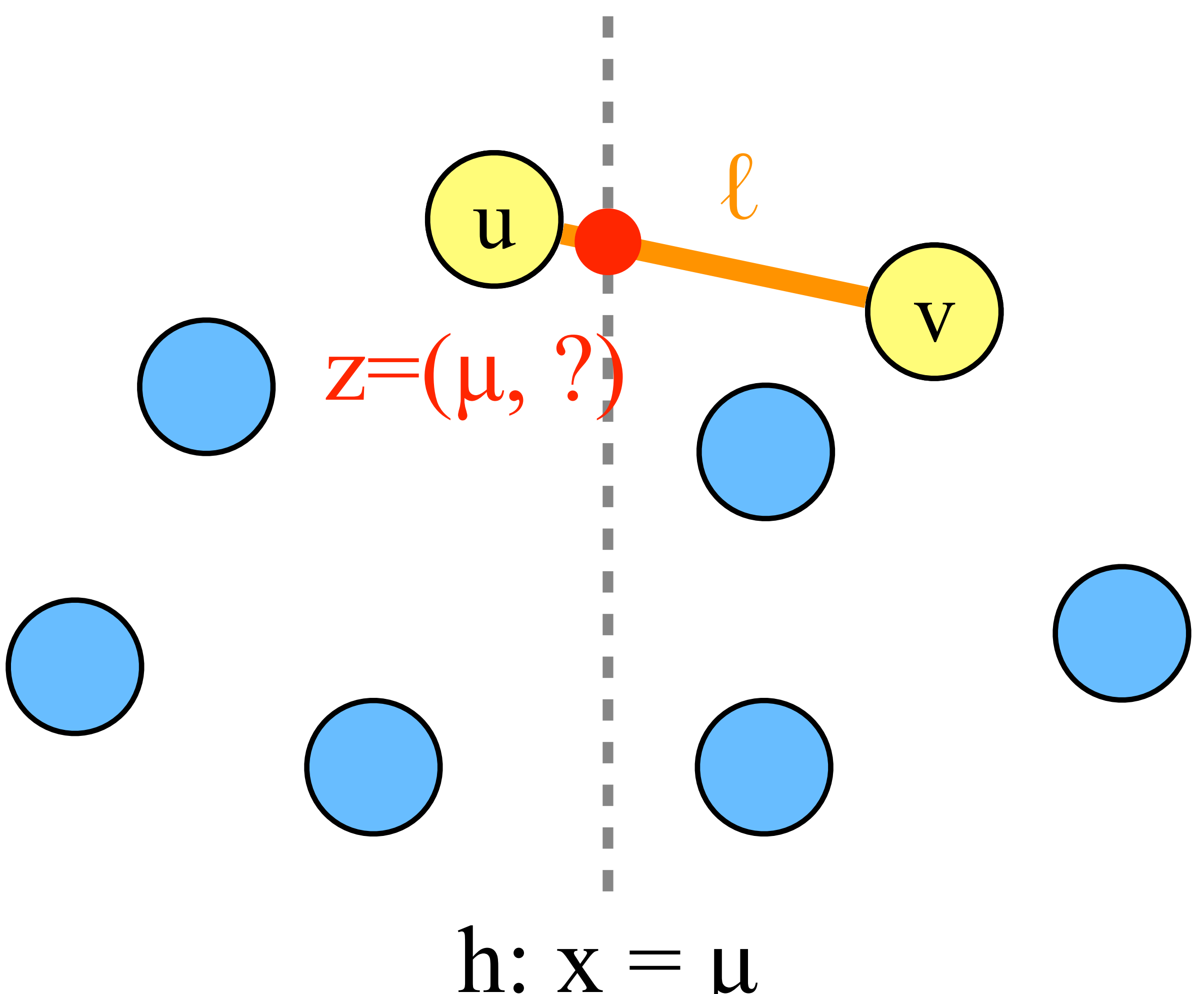


⇒

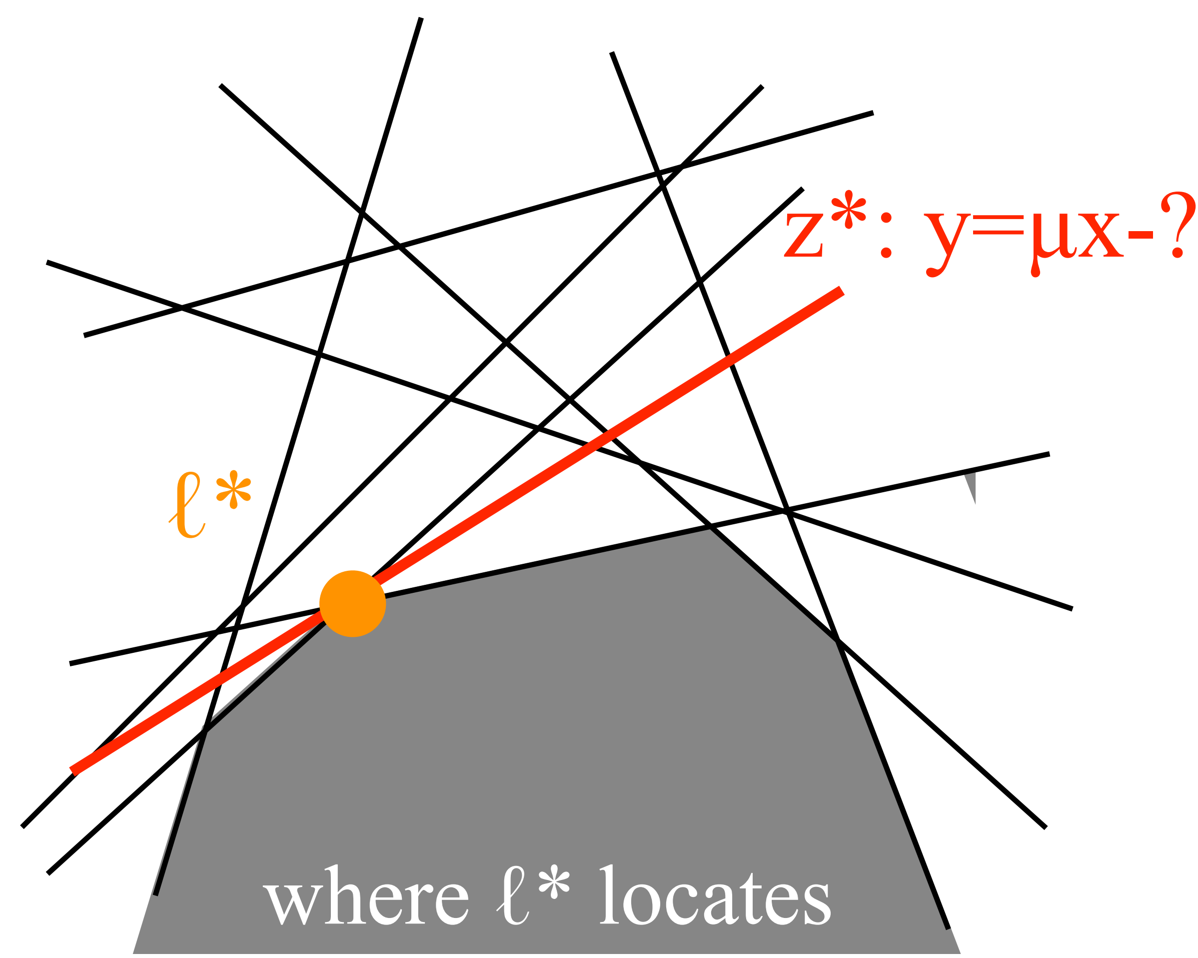


Finding the hull edge by an LP

Property: If p is on ℓ in the primal plane, then ℓ* is on p* in the dual plane. Let ℓ and h intersect at z = (μ, ?). Then, z*: y = μx - ? that touches ℓ* and ℓ* is the only point in the feasible region that can touch z*. (Why?)

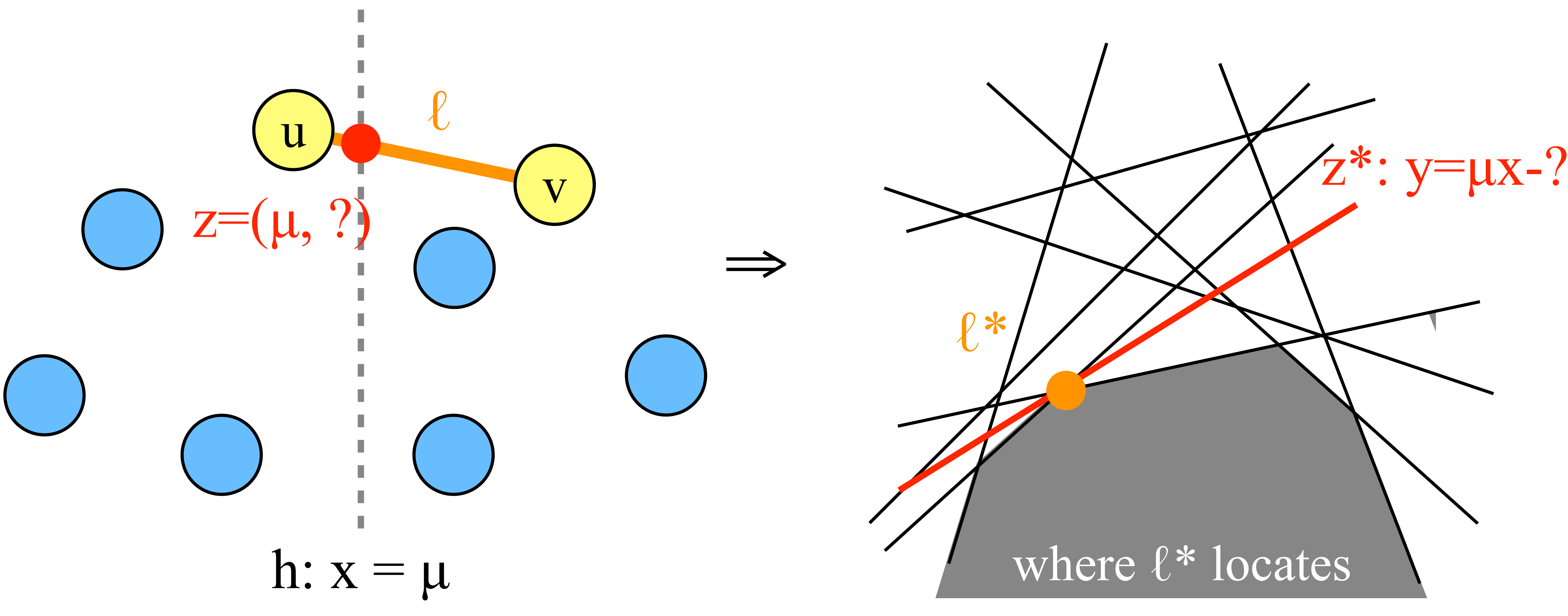


⇒

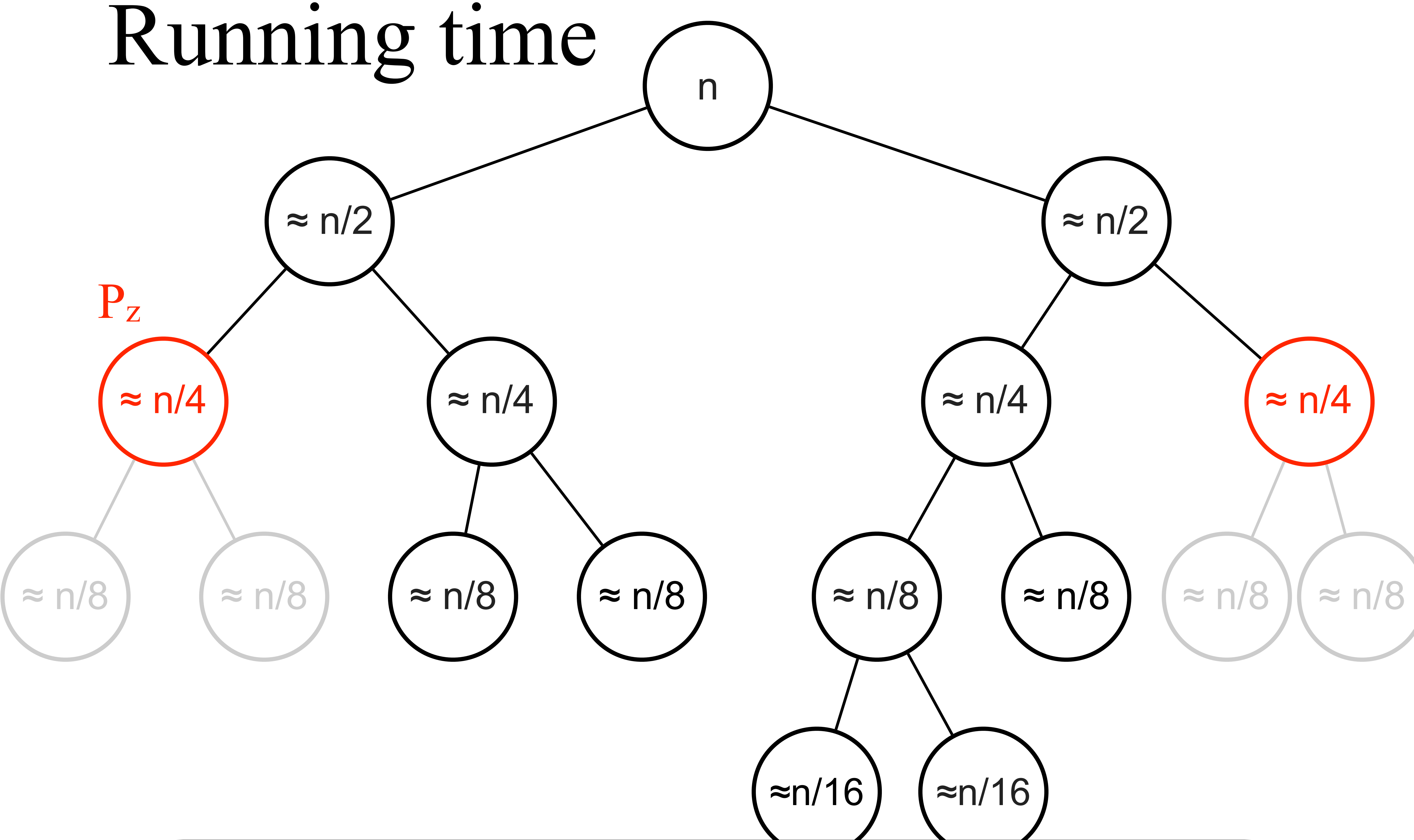


Finding the hull edge by an LP

As a result, finding the hull edge in the UH that crosses a line is equivalent to solving a linear program.

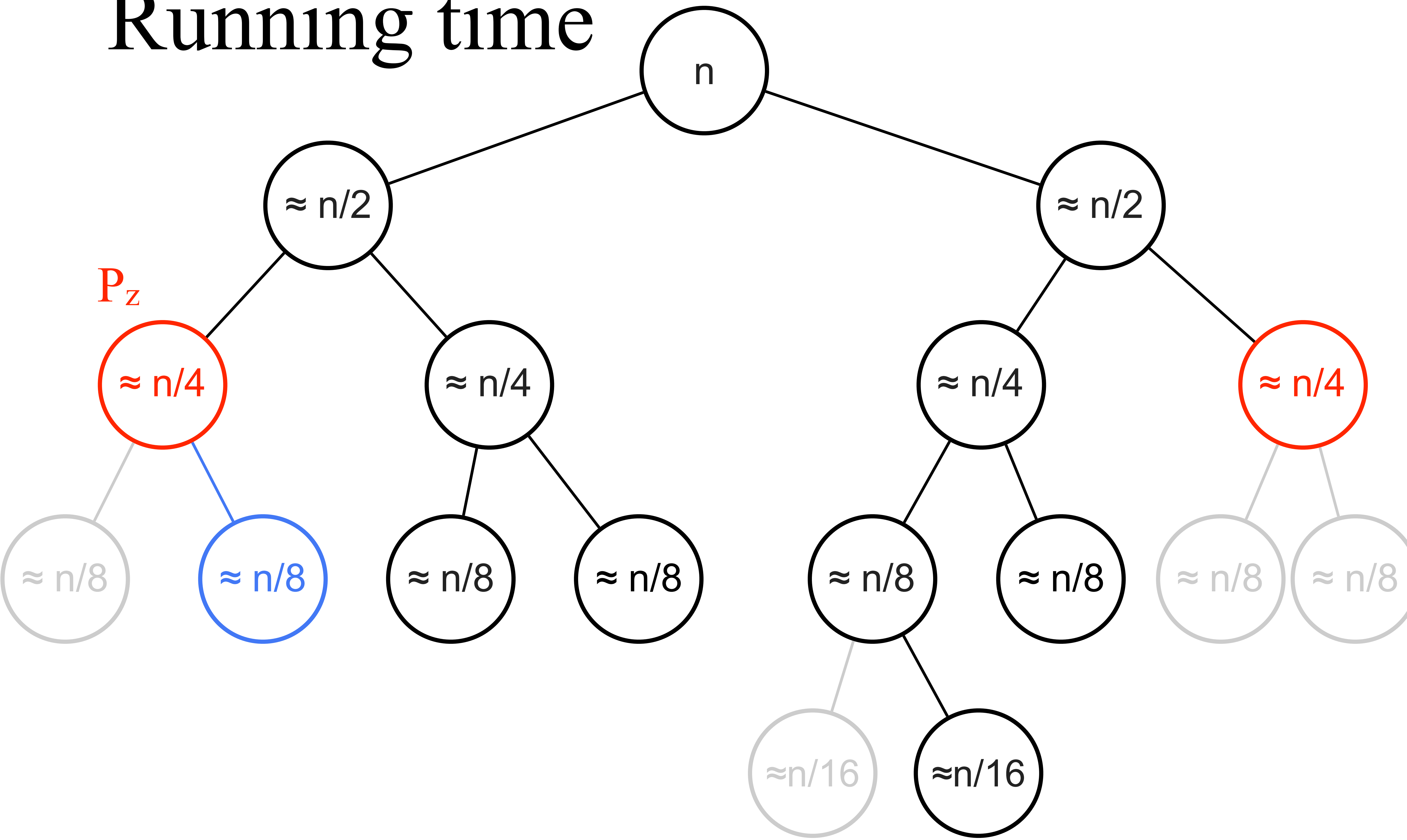


Running time

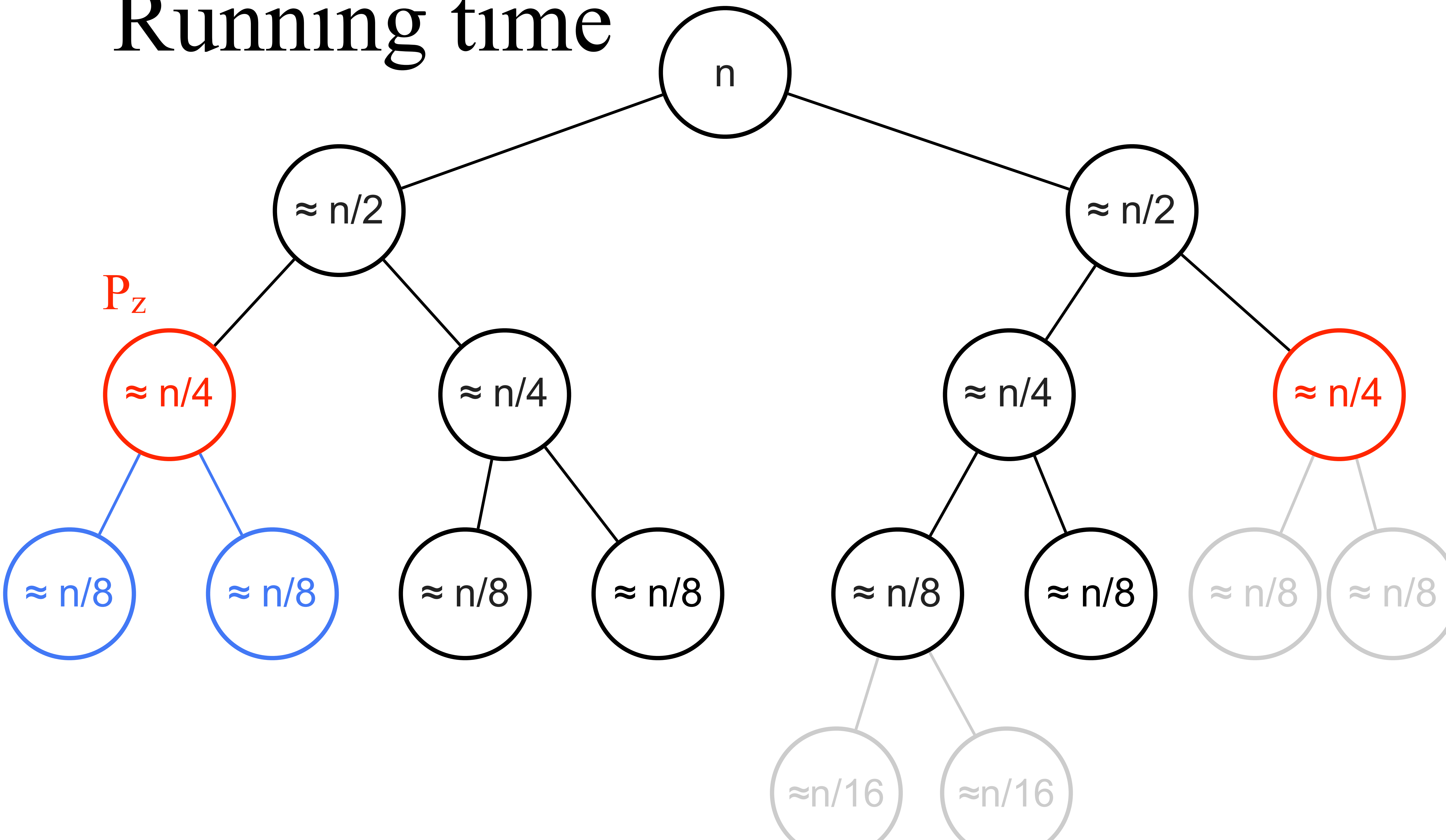


There are O(h) subproblems. If a subproblem P_z doesn't output a hull edge, then P_z's two child subproblems are done.

Running time

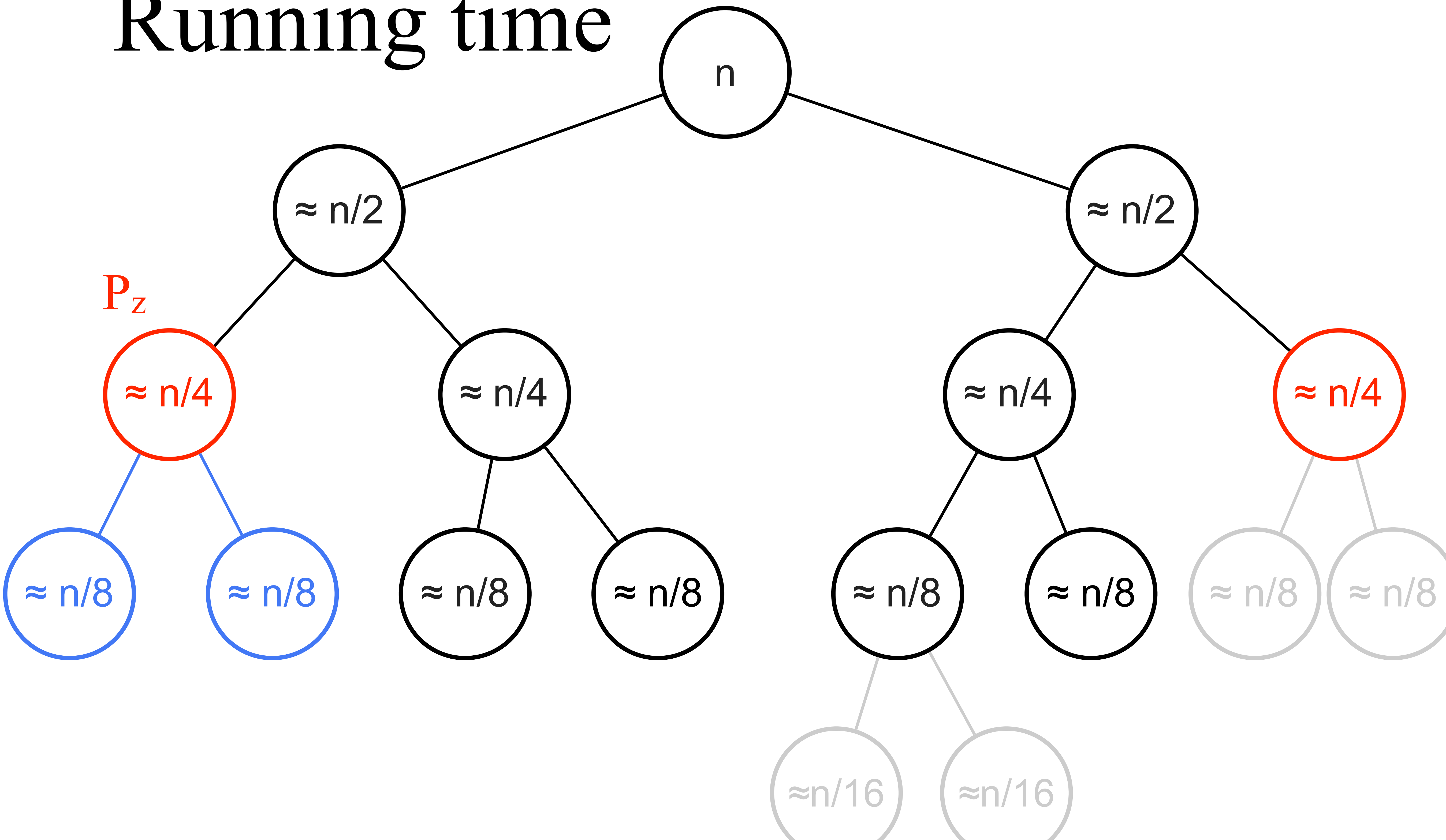


Running time



The computation cost of a node of larger depth can be upper-bounded by that of a node of smaller depth.

Running time



Hence, the computation cost is upper-bounded by O(n log h).

LP Duality and Totally Unimodularity

LP Duality

$\max\{ c^T x : Ax \leq b, x \geq 0, x \in R^n\}$

$=$

$\min\{ b^T y : A^T y \geq c, y \geq 0, y \in R^m\}$.

Totally Unimodularity

A matrix M is totally unimodular if every square submatrix (could be not contiguous) has determinant +1, 0, or -1.

It is known that the incidence matrix M of a bipartite graph is totally unimodular. Clearly, the transpose of M is also totally unimodular.

In an indicence matrix, $M(i, j) = 1$ indicates that node v_i is incident by edge e_j.

If M is totally unimodular and b is integral, then all vertices in polyhedron $P = \{Mx \leq b\}$ are integral points.

Frational matching and fractional vertex cover

$\max\{\ 1^T e :\ Ae \leq 1,\ e \geq 0,\ e \in R^n\ \}$ (frational matching)

= (by LP duality)

$\min\{\ 1^T v : A^T v \geq 1,\ v \geq 0,\ v \in R^m\ \}$ (fractional vertex cover)

Integral matching and integral vertex cover

$\max\{ 1^Te : Ae \leq 1, e \geq 0, e \in Z^n \}$ (integral matching)

$\leq$ (because $Z^n \subset R^n$)

$\max\{ 1^Te : Ae \leq 1, e \geq 0, e \in R^n \}$ (fractional matching)

$=$ (by LP duality)

$\min\{ 1^Tv : A^Tv \geq 1, v \geq 0, v \in R^m \}$ (fractional vertex cover)

$\leq$ (because $Z^m \subset R^m$)

$\min\{ 1^Tv : A^Tv \geq 1, v \geq 0, v \in Z^m \}$ (integral vertex cover)

Integral bipartite matching and integral bipartite vertex cover

$\max\{\ 1^T e :\ Ae \leq 1,\ e \geq 0,\ e \in Z^n\ \}$ (integral matching)

= (because A is totally unimodular)

$\max\{\ 1^T e :\ Ae \leq 1,\ e \geq 0,\ e \in R^n\ \}$ (fractional matching)

= (by LP duality)

aka König Theorem.

$\min\{\ 1^T v :\ A^T v \geq 1,\ v \geq 0,\ v \in R^m\ \}$ (fractional vertex cover)

= (because A is totally unimodular)

$\min\{\ 1^T v :\ A^T v \geq 1,\ v \geq 0,\ v \in Z^m\ \}$ (integral vertex cover)

Exercise

Given a set of n disks in a plane, each of radius 1. Construct an intersection graph G so that each node in G represents a disk in the input. Then, connect two nodes in G with an edge if and only if two disks intersect (including touch). Devise a polynomial-time algorithm to find the largest clique in G.

(Hint. "Disk Graphs: A Short Survey" by Fishkin)

Exercise

Let $G = (U \cup V, E)$ be an undirected bipartite connected graph, prove that:

$\sum_{v \text{ in } N(u)} 1/\deg(v) = 1$ for all node u in U, where N(u) denotes the set of u's neighbor nodes, then

G has a perfect matching.

(Hint. König Theorem)

Exercise

In Chap 29.2, you could find how to use LP to solve the graph problems, including shortest paths, maximum flow, and min-cost maximum flow. By googling, you could find how to use LP to solve minimum spanning trees as well.

Why do we have to learn algorithms for each of the above
rather than
simply to use LP to solve them?