Optimal Algorithm for Higher Order Voronoi Diagrams in 2D

The usefulness of nondeterminism

Timothy M. Chan, Pingan Cheng, **Da Wei Zheng**

SODA 2024

University of Illinois Urbana-Champaign & Aarhus University

Voronoi Diagrams

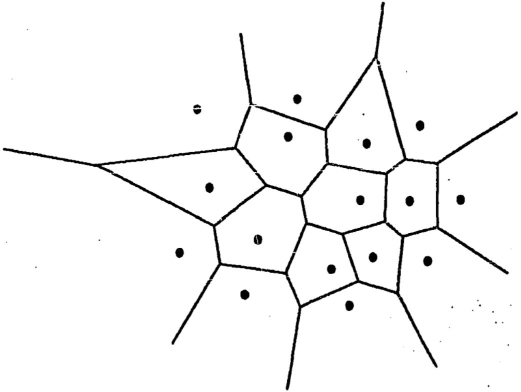


Figure 6.19: The Voronoi Diagram.

[Shamos 1978]

Order-k Voronoi Diagrams

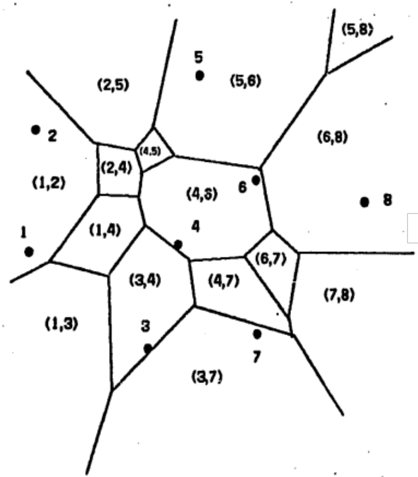


Figure 6.33: A Voronoi Diagram of Order Two.

[Shamos 1978]

Basic facts

- Fundamental (textbook) problem

Order-k Voronoi Diagrams

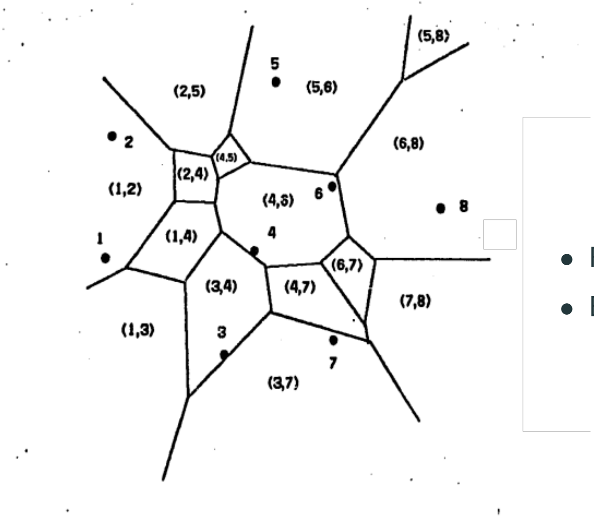


Figure 6.33: A Voronoi Diagram of Order Two.

Basic facts

- Fundamental (textbook) problem
- Each region convex polygon

[Shamos 1978]

Order-k Voronoi Diagrams

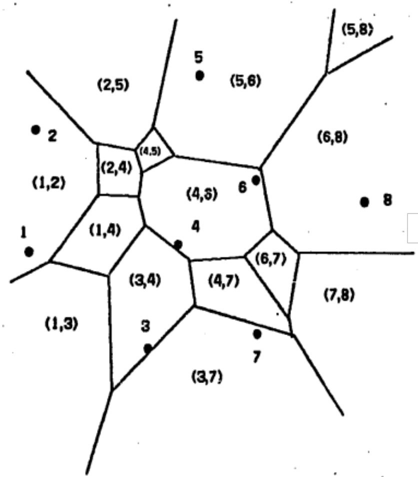


Figure 6.33: A Voronoi Diagram of Order Two.

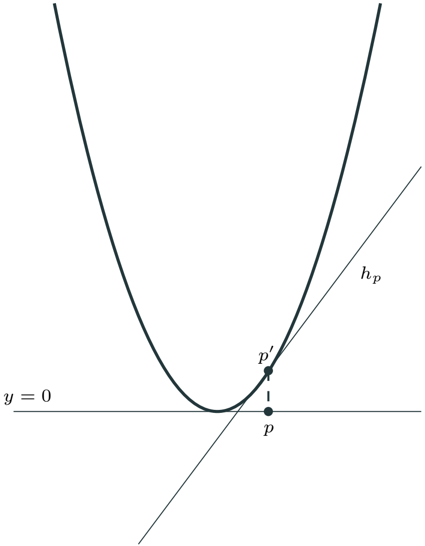
Basic facts

- Fundamental (textbook) problem
- Each region convex polygon
- Size is $\Theta(nk)$

[Shamos 1978]

Lifting and k-levels

$$p := (p_x, p_y)$$
$$p' := (p_x, p_y, p_x^2 + p_y^2)$$



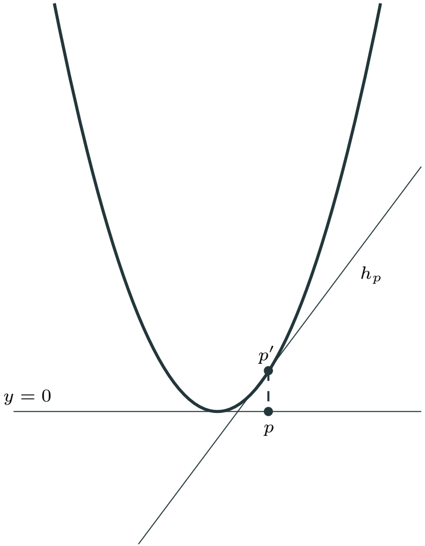
Lifting and k-levels

$$p := (p_x, p_y)$$

$$p' := (p_x, p_y, p_x^2 + p_y^2)$$

$$f_p(x, y) := 2p_x x + 2p_y y - (p_x^2 + p_y^2)$$

$$h_p := \{(x, y, z) \in \mathbb{R}^3 \mid z = f_p(x, y)\}$$

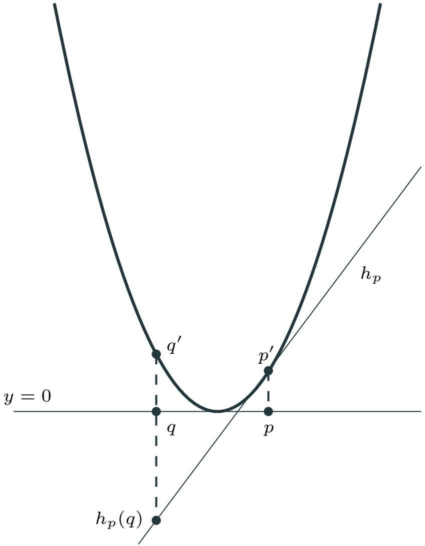


Lifting and k-levels

$$p := (p_x, p_y)$$
$$p' := (p_x, p_y, p_x^2 + p_y^2)$$

$$f_p(x, y) := 2p_x x + 2p_y y - (p_x^2 + p_y^2)$$
$$h_p := \{(x, y, z) \in \mathbb{R}^3 \mid z = f_p(x, y)\}$$

Consider $q = (q_x, q_y)$.

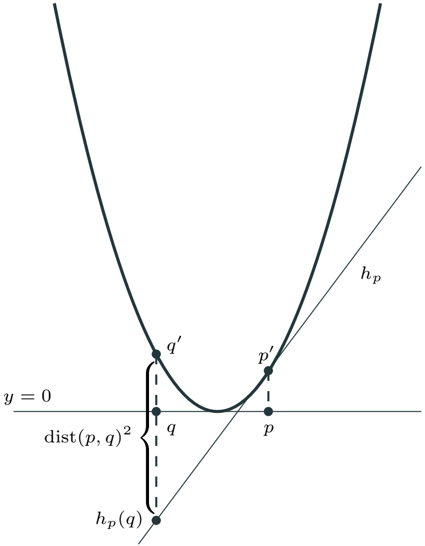


Lifting and k-levels

$$p := (p_x, p_y)$$
$$p' := (p_x, p_y, p_x^2 + p_y^2)$$

$$f_p(x, y) := 2p_x x + 2p_y y - (p_x^2 + p_y^2)$$
$$h_p := \{(x, y, z) \in \mathbb{R}^3 \mid z = f_p(x, y)\}$$

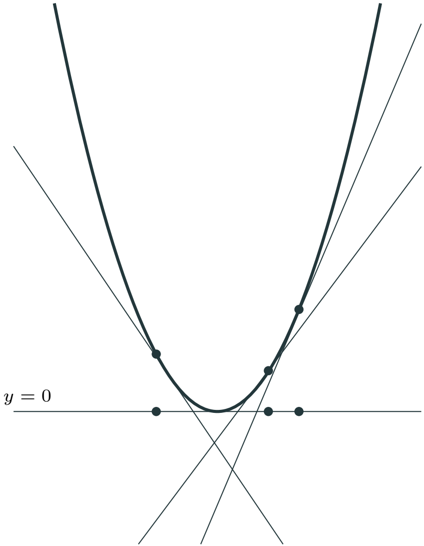
Consider $q = (q_x, q_y)$.



Lifting and k-levels

$$p := (p_x, p_y)$$
$$p' := (p_x, p_y, p_x^2 + p_y^2)$$

$$f_p(x, y) := 2p_x x + 2p_y y - (p_x^2 + p_y^2)$$
$$h_p := \{(x, y, z) \in \mathbb{R}^3 \mid z = f_p(x, y)\}$$

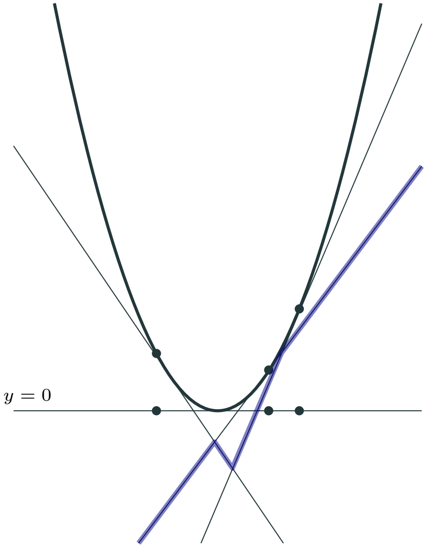


Consider the k-level of the planes.

Lifting and k-levels

$$p := (p_x, p_y)$$
$$p' := (p_x, p_y, p_x^2 + p_y^2)$$

$$f_p(x, y) := 2p_x x + 2p_y y - (p_x^2 + p_y^2)$$
$$h_p := \{(x, y, z) \in \mathbb{R}^3 \mid z = f_p(x, y)\}$$

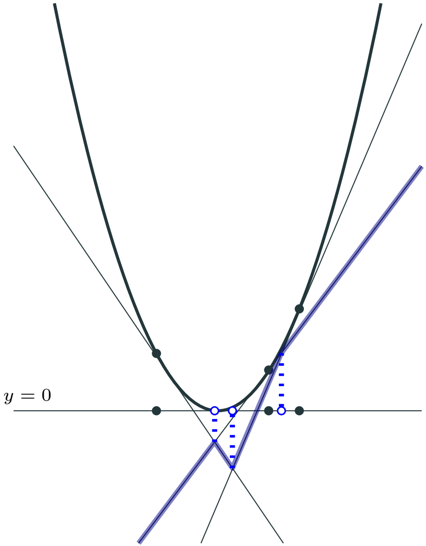


Consider the k-level of the planes.

Lifting and k-levels

$$p := (p_x, p_y)$$
$$p' := (p_x, p_y, p_x^2 + p_y^2)$$

$$f_p(x, y) := 2p_x x + 2p_y y - (p_x^2 + p_y^2)$$
$$h_p := \{(x, y, z) \in \mathbb{R}^3 \mid z = f_p(x, y)\}$$



Consider the k-level of the planes.

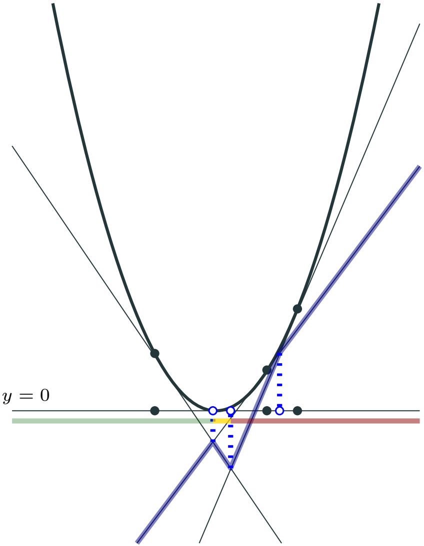
Lifting and k-levels

$$\boxed{\text{Order } k\text{-Voronoi} = k\text{-level of 3D planes}}$$

$$p := (p_x, p_y)$$
$$p' := (p_x, p_y, p_x^2 + p_y^2)$$

$$f_p(x, y) := 2p_x x + 2p_y y - (p_x^2 + p_y^2)$$
$$h_p := \{(x, y, z) \in \mathbb{R}^3 \mid z = f_p(x, y)\}$$

Consider the k-level of the planes.



Constructing the $\{1, \ldots, k\}$-level

$O(nk^2 \log n)$	Lee '82
$O(n^3)$	Edelsbrunner, O'Rourke, and Seidel '83

Algorithms for constructing Order-k Voronoi diagrams

Constructing the $\{1, \ldots, k\}$-level

$O(nk^2 \log n)$	Lee '82
$O(n^3)$	Edelsbrunner, O'Rourke, and Seidel '83
$O(nk^2 + n \log n)$	Aggarwal, Guibas, Saxe, and Shor '87

Constructing only the k-level

Algorithms for constructing Order-k Voronoi diagrams

Constructing only the k-level

$O(nk\sqrt{n}\log n)$	Edelsbrunner '86
$O(nk\log^2 n + n^2)$	Chazelle and Edelsbrunner '85

Algorithms for constructing Order-k Voronoi diagrams

Constructing only the k-level

$O(nk\sqrt{n}\log n)$	Edelsbrunner '86
$O(nk\log^2 n + n^2)$	Chazelle and Edelsbrunner '85
$O(n^{1+\varepsilon}k)$ rand.	Clarkson '86

Constructing only the k-level

$O(nk\sqrt{n}\log n)$	Edelsbrunner '86
$O(nk\log^2 n + n^2)$	Chazelle and Edelsbrunner '85
$O(n^{1+\varepsilon}k)$ rand.	Clarkson '86
$O(nk\log n + n\log^3 n)$ rand. inc.	Agarwal, de Berg, Matoušek, and Schwarzkopf '94

Algorithms for constructing Order-k Voronoi diagrams

Constructing only the k-level

$O(nk\sqrt{n}\log n)$	Edelsbrunner '86
$O(nk\log^2 n + n^2)$	Chazelle and Edelsbrunner '85
$O(n^{1+\varepsilon}k)$ rand.	Clarkson '86
$O(nk\log n + n\log^3 n)$ rand. inc.	Agarwal, de Berg, Matoušek, and Schwarzkopf '94
$O(n^{1+\varepsilon}k)$	Agarwal and Matoušek '95

Constructing only the k-level

$O(nk\sqrt{n}\log n)$	Edelsbrunner '86
$O(nk\log^2 n + n^2)$	Chazelle and Edelsbrunner '85
$O(n^{1+\varepsilon}k)$ rand.	Clarkson '86
$O(nk\log n + n\log^3 n)$ rand. inc.	Agarwal, de Berg, Matoušek, and Schwarzkopf '94
$O(n^{1+\varepsilon}k)$	Agarwal and Matoušek '95
$O(nk\log k + n\log n)$	Chan '98, Chan and Tsakalidis '15

Algorithms for constructing Order-k Voronoi diagrams

Constructing only the k-level

$O(nk\sqrt{n}\log n)$	Edelsbrunner '86
$O(nk\log^2 n + n^2)$	Chazelle and Edelsbrunner '85
$O(n^{1+\varepsilon}k)$ rand.	Clarkson '86
$O(nk\log n + n\log^3 n)$ rand. inc.	Agarwal, de Berg, Matoušek, and Schwarzkopf '94
$O(n^{1+\varepsilon}k)$	Agarwal and Matoušek '95
$O(nk\log k + n\log n)$	Chan '98, Chan and Tsakalidis '15
$\boxed{O(nk2^{O(\log^* k)} + n\log n) \text{ rand.}}$	Ramos '99

Constructing only the k-level

$O(nk\sqrt{n}\log n)$	Edelsbrunner '86
$O(nk\log^2 n + n^2)$	Chazelle and Edelsbrunner '85
$O(n^{1+\varepsilon}k)$ rand.	Clarkson '86
$O(nk\log n + n\log^3 n)$ rand. inc.	Agarwal, de Berg, Matoušek, and Schwarzkopf '94
$O(n^{1+\varepsilon}k)$	Agarwal and Matoušek '95
$O(nk\log k + n\log n)$	Chan '98, Chan and Tsakalidis '15
$\boxed{O(nk2^{O(\log^* k)} + n\log n) \text{ rand.}}$	Ramos '99

This paper:

$\boxed{O(nk + n\log n) \text{ rand.}}$ Optimal!

Self reductions

[Ramos '99] Reduces in $O(n^2)$ time rand. to $O(n^2/\log^2 n)$ problems of size $\log n$:

$$T(n) \leq O(n^2/\log^2 n)\, T(\log n) + O(n^2)$$

Self reductions

[Ramos '99] Reduces in $O(n^2)$ time rand. to $O(n^2/\log^2 n)$ problems of size $\log n$:

$T(n) \leq O(n^2/\log^2 n)\, T(\log n) + O(n^2)$ (Idea: [AdBMS '94], cuttings for k-level)

Self reductions

[Ramos '99] Reduces in $O(n^2)$ time rand. to $O(n^2/\log^2 n)$ problems of size $\log n$:

$T(n) \leq O(n^2/\log^2 n)\, T(\log n) + O(n^2)$ (Idea: [AdBMS '94], cuttings for k-level)

$T(n) \leq O(n^2 2^{O(\log^* n)})$ by repeating Ramos divide and conquer.

Self reductions

[Ramos '99] Reduces in $O(n^2)$ time rand. to $O(n^2/\log^2 n)$ problems of size $\log n$:

$T(n) \leq O(n^2/\log^2 n)\, T(\log n) + O(n^2)$ (Idea: [AdBMS '94], cuttings for k-level)

$T(n) \leq O(n^2 2^{O(\log^* n)})$ by repeating Ramos divide and conquer.

[Chan '98] Reduces in $O(n \log n)$ time to $O(n/k)$ problems of size $O(k)$:

$T(n,k) \leq O(n/k)\, T(k) + O(n \log n)$

Self reductions

[Ramos '99] Reduces in $O(n^2)$ time rand. to $O(n^2/\log^2 n)$ problems of size $\log n$:

$$T(n) \leq O(n^2/\log^2 n)\, T(\log n) + O(n^2)$$

(Idea: [AdBMS '94], cuttings for k-level)

$T(n) \leq O(n^2 2^{O(\log^* n)})$ by repeating Ramos divide and conquer.

[Chan '98] Reduces in $O(n \log n)$ time to $O(n/k)$ problems of size $O(k)$:

$$T(n,k) \leq O(n/k)\, T(k) + O(n \log n)$$

(Idea: Shallow cuttings, can be made det.)

Self reductions

[Ramos '99] Reduces in $O(n^2)$ time rand. to $O(n^2/\log^2 n)$ problems of size $\log n$:

$T(n) \leq O(n^2/\log^2 n)\, T(\log n) + O(n^2)$ (Idea: [AdBMS '94], cuttings for k-level)

$T(n) \leq O(n^2 2^{O(\log^* n)})$ by repeating Ramos divide and conquer.

[Chan '98] Reduces in $O(n \log n)$ time to $O(n/k)$ problems of size $O(k)$:

$T(n,k) \leq O(n/k)\, T(k) + O(n \log n)$ (Idea: Shallow cuttings, can be made det.)

Self reductions

[Ramos '99] Reduces in $O(n^2)$ time rand. to $O(n^2/\log^2 n)$ problems of size $\log n$:

$$T(n) \le O(n^2/\log^2 n)\, T(\log n) + O(n^2) \qquad \text{(Idea: [AdBMS '94], cuttings for } k\text{-level)}$$

$T(n) \le O(n^2 2^{O(\log^* n)})$ by repeating Ramos divide and conquer.

[Chan '98] Reduces in $O(n \log n)$ time to $O(n/k)$ problems of size $O(k)$:

$$T(n,k) \le O(n/k)\, T(k) + O(n \log n) \qquad \text{(Idea: Shallow cuttings, can be made det.)}$$

$$T(n,k) \le O(n/k)\left(k^2 2^{O(\log^* k)}\right) + O(n \log n) = O(nk 2^{O(\log^* k)} + n \log n)$$

Self reductions

[Ramos '99] Reduces in $O(n^2)$ time rand. to $O(n^2/\log^2 n)$ problems of size $\log n$:

$$T(n) \leq O(n^2/\log^2 n)\, T(\log n) + O(n^2)$$

(Idea: [AdBMS '94], cuttings for k-level)

$T(n) \leq O(n^2 2^{O(\log^* n)})$ by repeating Ramos divide and conquer.

[Chan '98] Reduces in $O(n \log n)$ time to $O(n/k)$ problems of size $O(k)$:

$$T(n,k) \leq O(n/k)\, T(k) + O(n \log n)$$

(Idea: Shallow cuttings, can be made det.)

$$T(n,k) \leq O(n/k)\left(k^2 2^{O(\log^* k)}\right) + O(n \log n) = O(nk2^{O(\log^* k)} + n \log n)$$

Goal: Find an $O(n^2)$ time algorithm!

Another consequence of Ramos' divide and conquer

Repeat Ramos' divide and conquer until problems are of size $b = \log\log\log n$.

$$\begin{aligned} T(n) &\leq O(n^2/(\log n)^2) \cdot T(\log n) + O(n^2) \\ &\leq O(n^2/(\log\log n)^2) \cdot T(\log\log n) + O(n^2) \\ &\leq O(n^2/(\log\log\log n)^2) \cdot T(\log\log\log n) + O(n^2) \end{aligned}$$

Another consequence of Ramos' divide and conquer

Repeat Ramos' divide and conquer until problems are of size $b = \log\log\log n$.

$$
\begin{aligned}
T(n) &\leq O(n^2/(\log n)^2) \cdot T(\log n) + O(n^2) \\
&\leq O(n^2/(\log\log n)^2) \cdot T(\log\log n) + O(n^2) \\
&\leq O(n^2/(\log\log\log n)^2) \cdot T(\log\log\log n) + O(n^2)
\end{aligned}
$$

Build an algebraic decision tree for problems of size b (Can take $2^{2^{O(b)}}$ time).

Another consequence of Ramos' divide and conquer

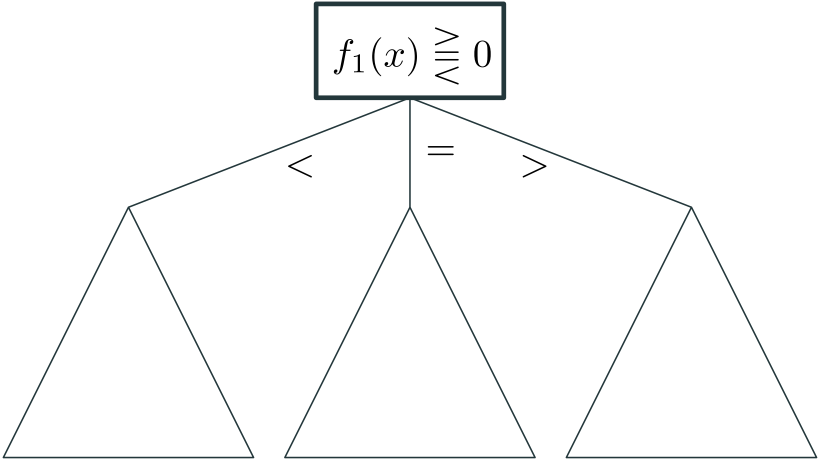
Repeat Ramos' divide and conquer until problems are of size $b = \log\log\log n$.

$$\begin{aligned} T(n) &\leq O(n^2/(\log n)^2) \cdot T(\log n) + O(n^2) \\ &\leq O(n^2/(\log\log n)^2) \cdot T(\log\log n) + O(n^2) \\ &\leq O(n^2/(\log\log\log n)^2) \cdot T(\log\log\log n) + O(n^2) \end{aligned}$$

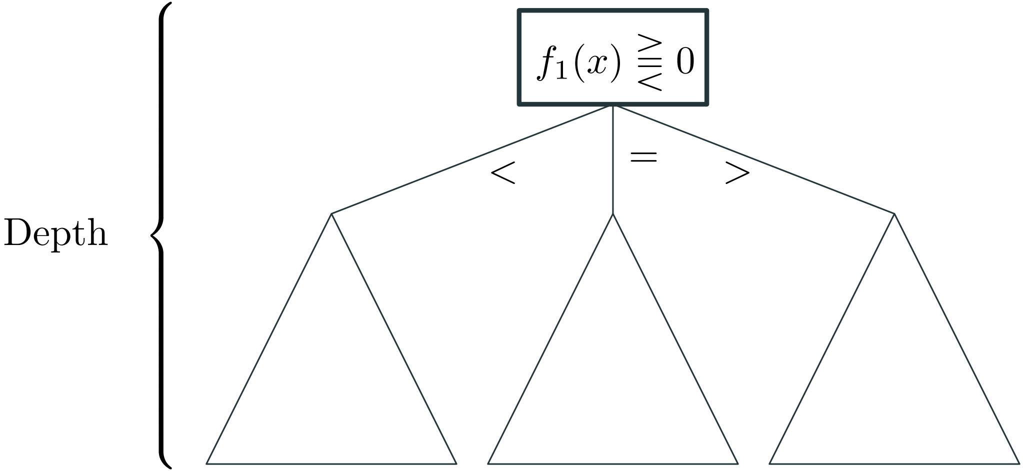
Build an algebraic decision tree for problems of size b (Can take $2^{2^{O(b)}}$ time).

New goal: Find a quadratic depth decision tree!

Aside: On (algebraic) decision trees



Aside: On (algebraic) decision trees



Outline of paper

Outline of paper

0. Reduce to decision tree problem.

Outline of paper

0. Reduce to decision tree problem.
1. Reduce from finding k-level to verifying a k-level.

Outline of paper

0. Reduce to decision tree problem.
1. Reduce from finding k-level to verifying a k-level.
   (Idea: "Guess" entire k-level!)

Outline of paper

0. Reduce to decision tree problem.
1. Reduce from finding k-level to verifying a k-level.
   (Idea: "Guess" entire k-level!)
2. Solve verification problem in $O(n^2)$ time.

Outline of paper

0. Reduce to decision tree problem.
1. Reduce from finding k-level to verifying a k-level.
   (Idea: "Guess" entire k-level!)
2. Solve verification problem in $O(n^2)$ time.
   (Idea: "standard" alg. w/ planar separators, recursion, dynamic 3d conv. hulls)

Going to higher dimensions

(Idea 1: Use similar ideas in removing $2^{O(\log^* n)}$ factors for Hopcroft's problem)

Going to higher dimensions

(Idea 1: Use similar ideas in removing $2^{O(\log^* n)}$ factors for Hopcroft's problem)

- View input as a vector $x \in \mathbb{R}^{3n}$.

Going to higher dimensions

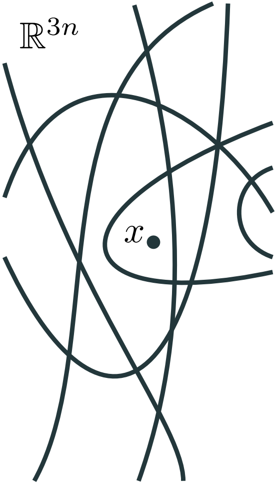
(Idea 1: Use similar ideas in removing $2^{O(\log^* n)}$ factors for Hopcroft's problem)

- View input as a vector $x \in \mathbb{R}^{3n}$.
- k-level determined by vertices in arrangement of planes and above/below relations with other planes, i.e. completely by comparisons of four planes h_1, h_2, h_3, h_4.

Going to higher dimensions

(Idea 1: Use similar ideas in removing $2^{O(\log^* n)}$ factors for Hopcroft's problem)

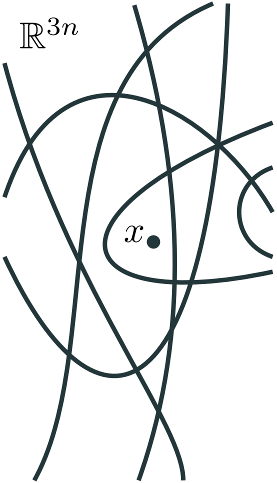
- View input as a vector $x \in \mathbb{R}^{3n}$.
- k-level determined by vertices in arrangement of planes and above/below relations with other planes, i.e. completely by comparisons of four planes h_1, h_2, h_3, h_4.
- Each comparison is a high dimensional algebraic surface.



Going to higher dimensions

(Idea 1: Use similar ideas in removing $2^{O(\log^* n)}$ factors for Hopcroft's problem)

- View input as a vector $x \in \mathbb{R}^{3n}$.
- k-level determined by vertices in arrangement of planes and above/below relations with other planes, i.e. completely by comparisons of four planes h_1, h_2, h_3, h_4.
- Each comparison is a high dimensional algebraic surface.
- By Milnor-Thom, there are $(3n)^{O(3n)} = n^{O(n)}$ different cells in arrangement.



Reduction to verification problem

Reduction to verification problem

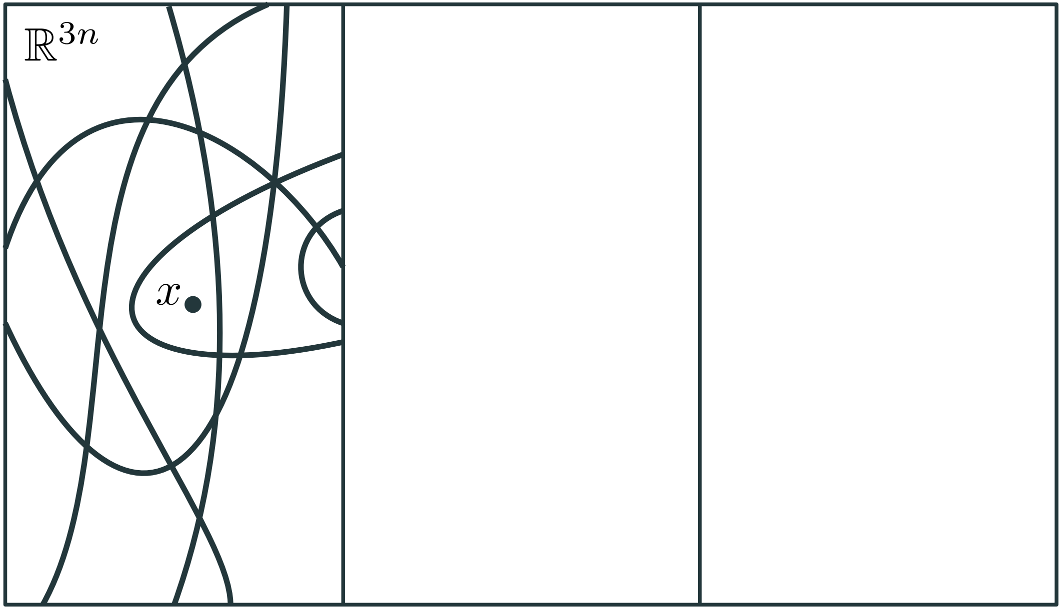
(Idea 2: Use Ramos' divide and conquer again!)

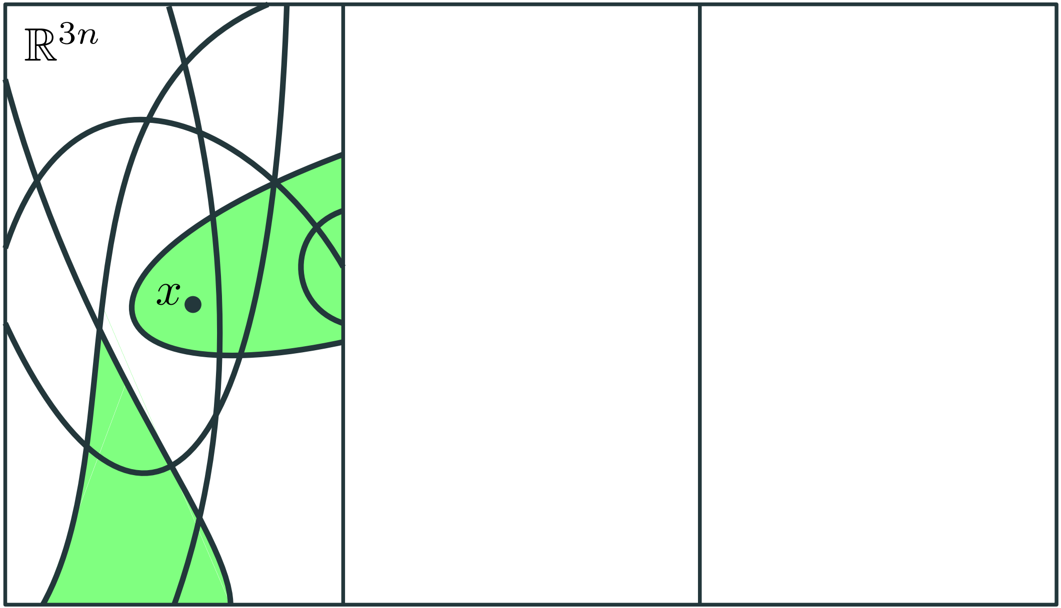
Reduction to verification problem

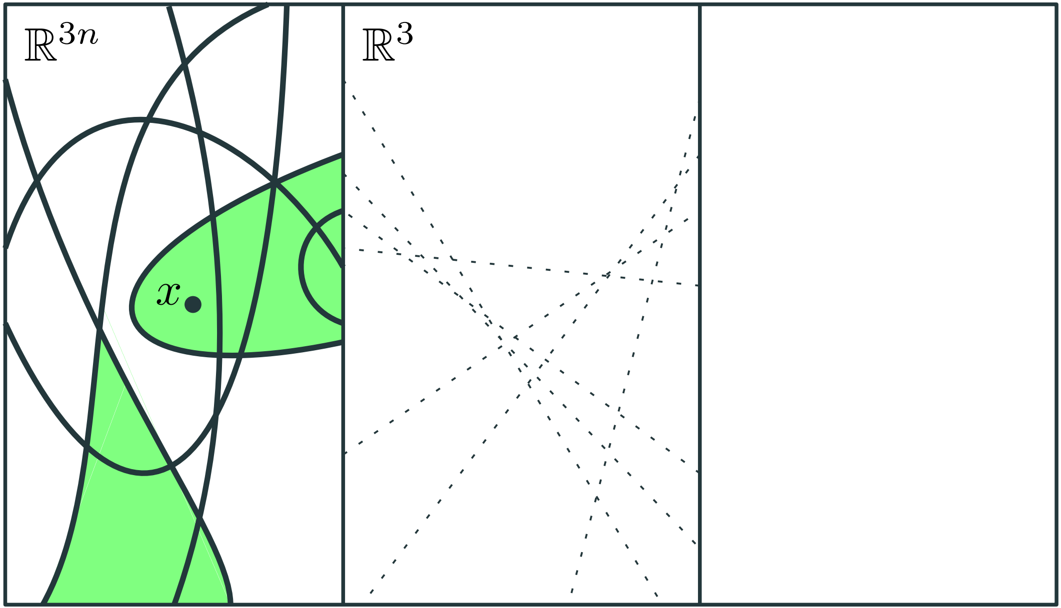
(Idea 2: Use Ramos' divide and conquer again!)

Use just one round to get problems of size $b = O(\log n)$.

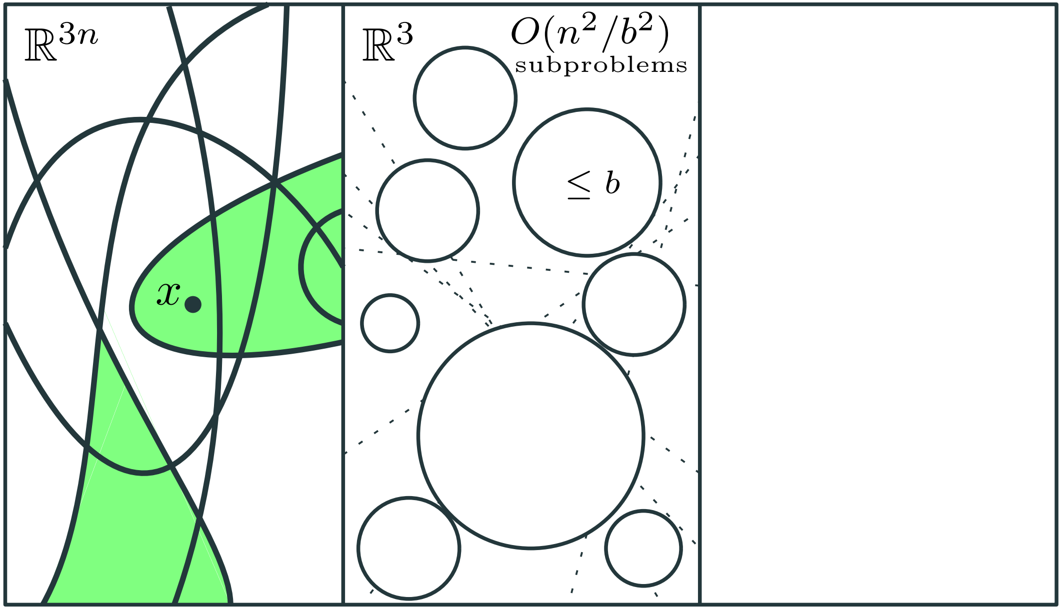
$$T(n) \leq O(n^2/b) \cdot T(b) + O(n^2)$$

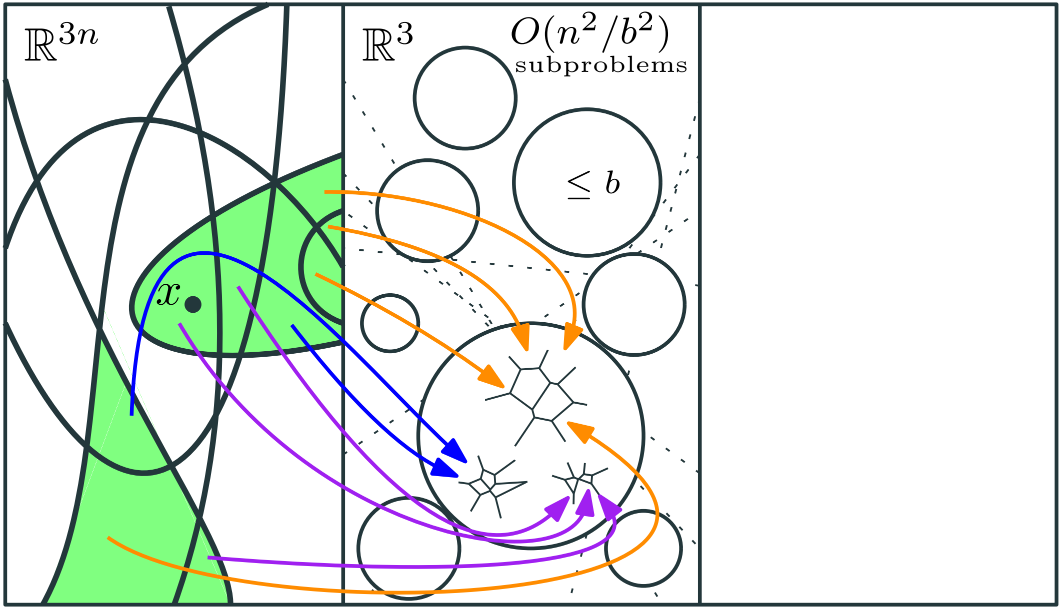

$\mathbb{R}^{3n}$
x

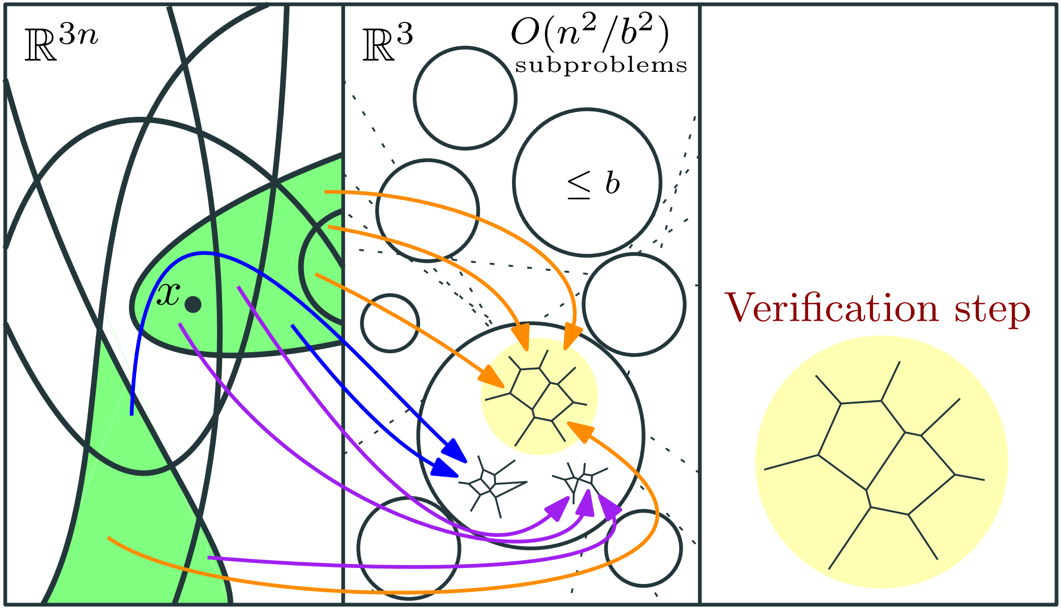

$\mathbb{R}^{3n}$
x



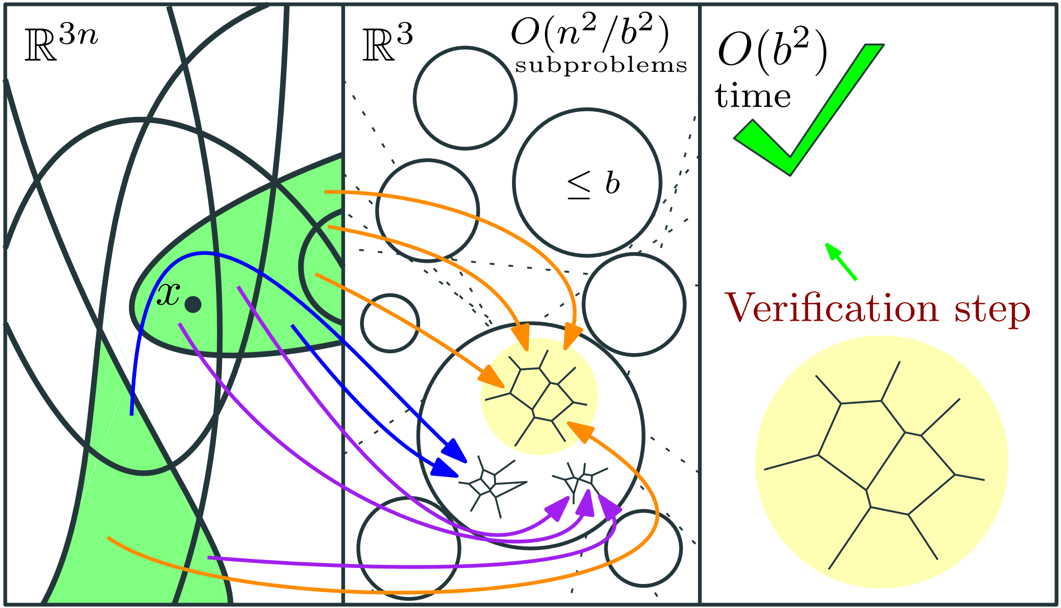
$\mathbb{R}^{3n}$
x
$\mathbb{R}^{3}$

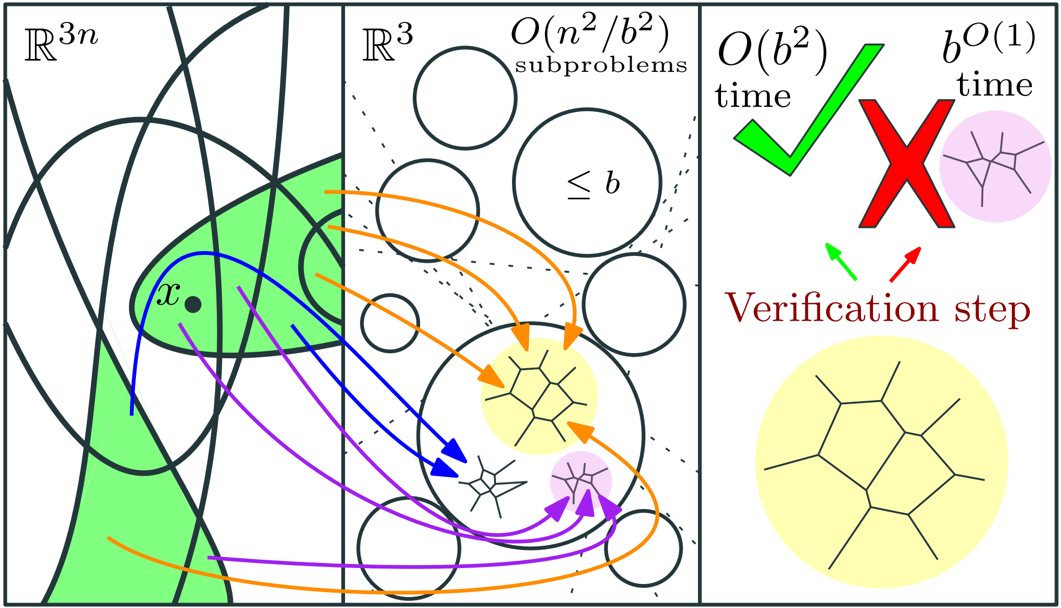

$\mathbb{R}^{3n}$
x
$\mathbb{R}^3$
$O(n^2/b^2)$
subproblems
$\leq b$


$\mathbb{R}^{3n}$
x
$\mathbb{R}^3$
$O(n^2/b^2)$
subproblems
$\leq b$

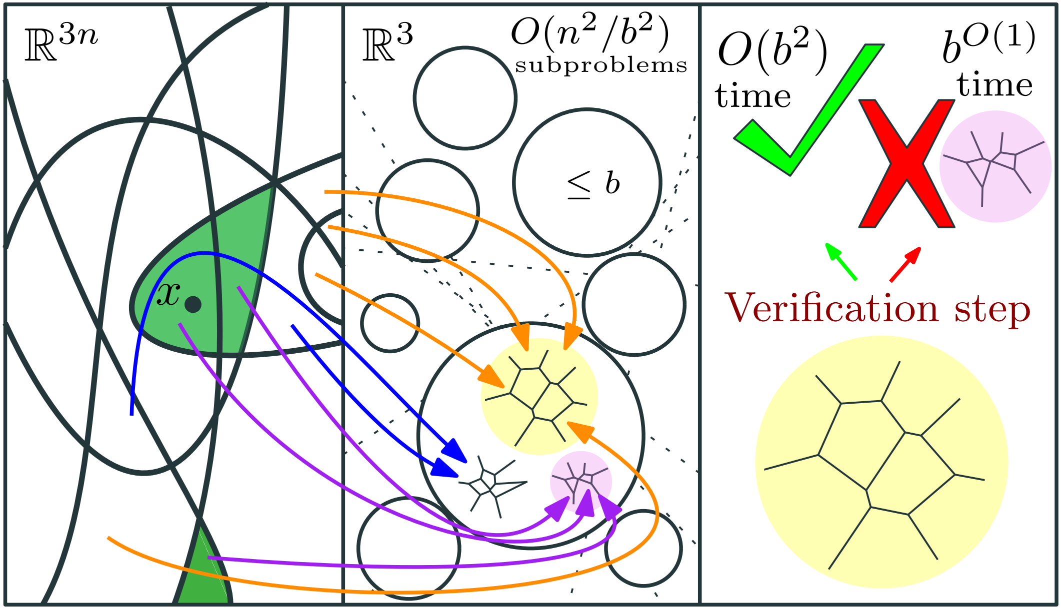


$\mathbb{R}^{3n}$
x
$\mathbb{R}^3$
$O(n^2/b^2)$
subproblems
$\leq b$
Verification step


$\mathbb{R}^{3n}$
x
$\mathbb{R}^3$
$O(n^2/b^2)$
subproblems
$\leq b$
$O(b^2)$
time
Verification step

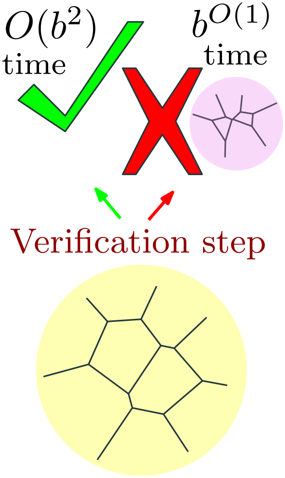


$\mathbb{R}^{3n}$
x
$\mathbb{R}^3$
$O(n^2/b^2)$
subproblems
$\leq b$
$O(b^2)$
time
$b^{O(1)}$
time
Verification step


$\mathbb{R}^{3n}$
x
$\mathbb{R}^3$
$O(n^2/b^2)$
subproblems
$\leq b$
$O(b^2)$
time
$b^{O(1)}$
time
Verification step

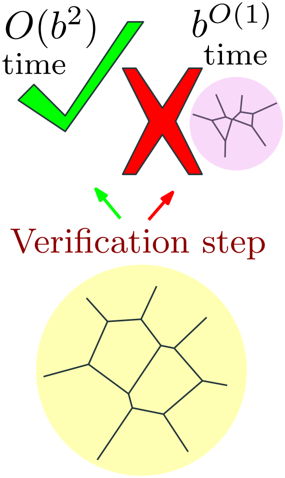
Overall runtime

- If we guessed wrong, the answer might be the second most popular option.



Overall runtime

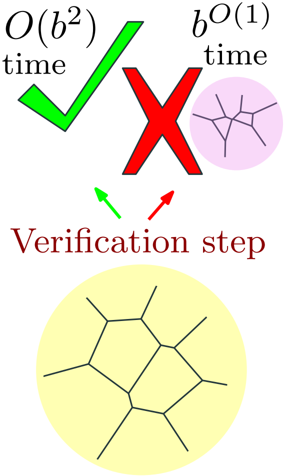
- If we guessed wrong, the answer might be the second most popular option.
- Number of active cells decreases by factor of $1/2$.



Overall runtime

- If we guessed wrong, the answer might be the second most popular option.
- Number of active cells decreases by factor of 1/2.

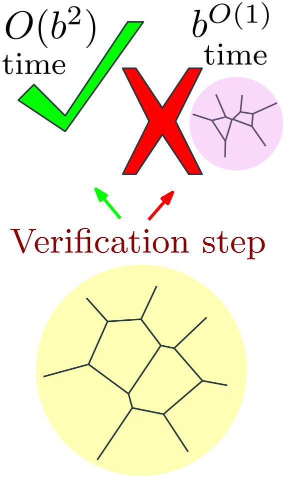
$$T(n) \leq O(n^2/b^2) \cdot O(b^2) + b^{O(1)} \cdot \log\left(n^{O(n)}\right)$$



Overall runtime

- If we guessed wrong, the answer might be the second most popular option.
- Number of active cells decreases by factor of $1/2$.

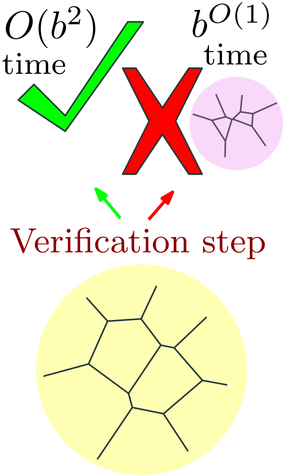
$$
\begin{aligned}
T(n) &\leq O(n^2/b^2) \cdot O(b^2) + b^{O(1)} \cdot \log\left(n^{O(n)}\right) \\
&\leq O\left(n^2 + b^{O(1)} n \log n\right)
\end{aligned}
$$



Overall runtime

- If we guessed wrong, the answer might be the second most popular option.
- Number of active cells decreases by factor of $1/2$.

$$
\begin{aligned}
T(n) &\leq O(n^2/b^2) \cdot O(b^2) + b^{O(1)} \cdot \log\left(n^{O(n)}\right) \\
&\leq O\left(n^2 + b^{O(1)} n \log n\right) \\
&\leq O(n^2)
\end{aligned}
$$



Open Problems

Conclusion

Open Problems

- Simpler algorithm?

Open Problems

- Simpler algorithm? (Decision trees impractical)

Open Problems

- Simpler algorithm? (Decision trees impractical)
- Optimal deterministic algorithm?

Open Problems

- Simpler algorithm? (Decision trees impractical)
- Optimal deterministic algorithm? (Ramos' randomized self-reduction)

Open Problems

- Simpler algorithm? (Decision trees impractical)
- Optimal deterministic algorithm? (Ramos' randomized self-reduction)
- Extensions of decision trees and non-determinism?

Open Problems

- Simpler algorithm? (Decision trees impractical)
- Optimal deterministic algorithm? (Ramos' randomized self-reduction)
- Extensions of decision trees and non-determinism? (Need self-reduction)

Conclusion

Open Problems

- Simpler algorithm? (Decision trees impractical)
- Optimal deterministic algorithm? (Ramos' randomized self-reduction)
- Extensions of decision trees and non-determinism? (Need self-reduction)
  - Hopcroft's problem in $o(n^{4/3})$ time?

Open Problems

- Simpler algorithm? (Decision trees impractical)
- Optimal deterministic algorithm? (Ramos' randomized self-reduction)
- Extensions of decision trees and non-determinism? (Need self-reduction)
    - Hopcroft's problem in $o(n^{4/3})$ time?
    - Detecting collinear triples in $\mathbb{R}^2$ in $o(n^2)$ time?

Conclusion

Open Problems

- Simpler algorithm? (Decision trees impractical)
- Optimal deterministic algorithm? (Ramos' randomized self-reduction)
- Extensions of decision trees and non-determinism? (Need self-reduction)
  - Hopcroft's problem in $o(n^{4/3})$ time?
  - Detecting collinear triples in $\mathbb{R}^2$ in $o(n^2)$ time?
  - How to verify no instances?

Thank you for listening