Analyzing algorithms

Vera Sacristán

Computational Geometry
Facultat de Matemàtiques i Estadística
Universitat Politècnica de Catalunya

Complexity

- Time
- Space

Other important issues: understandability, robustness, etc.

Complexity

- Time
- Space

Other important issues: understandability, robustness, etc.

Why is time so important?

Complexity

- Time
- Space

Other important issues: understandability, robustness, etc.

Why is time so important?

Execution time, assuming a speed of 1 million instructions per second

Cost	$n = 10$	$n = 20$	$n = 100$	
$\log n$	0.004 ms	0.005 ms	0.007 ms	
n	0.01 ms	0.02 ms	0.1 ms	
$n \log n$	0.033 ms	0.09 ms	0.66 ms	
n^2	0.1 ms	0.4 ms	10 ms	
n^4	10 ms	160 ms	1 min 40 sec	
2^n	1 ms	1.05 sec	2.7×10^6 UA	
$n!$	3.6 sec	76 000 years	2×10^{134} UA	
n^n	2 h 48 min	220 UA	2×10^{176} UA	

UA = age of the universe (15 thousand millions years)

Complexity

- Time
- Space

Other important issues: understandability, robustness, etc.

Why is time so important?

Execution time, assuming a speed of 1 million instructions per second

Cost	$n = 10$	$n = 20$	$n = 100$	$n = 18000$
$\log n$	0.004 ms	0.005 ms	0.007 ms	0.018 ms
n	0.01 ms	0.02 ms	0.1 ms	18 ms
$n \log n$	0.033 ms	0.09 ms	0.66 ms	254 ms
n^2	0.1 ms	0.4 ms	10 ms	5 min 24 sec
n^4	10 ms	160 ms	1 min 40 sec	3 328 years
2^n	1 ms	1.05 sec	2.7×10^6 UA	
$n!$	3.6 sec	76 000 years	2×10^{134} UA	
n^n	2 h 48 min	220 UA	2×10^{176} UA	

UA = age of the universe (15 thousand millions years)

Complexity

- Time
- Space

Other important issues: understandability, robustness, etc.

Why is time so important?

Size of the problem that can be solved in 1 hour

Cost	Current size	100 times faster	1000 times faster
n	N	$100N$	$1000N$
n^2	N	$10N$	$31.6N$
n^3	N	$4.64N$	$10N$
2^n	N	$N + 6.64$	$N + 9.97$
3^n	N	$N + 4.19$	$N + 6.29$

Complexity

- Time
- Space

Other important issues: understandability, robustness, etc.

Model of computation

The Real RAM Model:

- Each memory unit can allocate one real number, without precision limit
- Access to one memory position has unit cost
- Unit cost operations are:
  - Comparisons ($<, \leq, = \neq, >, \geq$)
  - Arithmetic operations ($+, -, *, :$)

Analytic functions (such as $\sqrt[k]{\ }, \log, \exp, \cos, \sin, \ldots$) do not have unit cost. Neither do functions floor and ceiling.

Complexity

- Time
- Space

Other important issues: understandability, robustness, etc.

Model of computation

The Real RAM Model:

- Each memory unit can allocate one real number, without precision limit
- Access to one memory position has unit cost
- Unit cost operations are:
  - Comparisons $(<, \leq, = \neq, >, \geq)$
  - Arithmetic operations $(+, -, *, :)$

Analytic functions (such as $\sqrt[k]{\ }, \log, \exp, \cos, \sin, \dots$) do not have unit cost. Neither do functions floor and ceiling.

Asymptotic analysis studies the cost of an algorithm (i.e., the number of unit cost operations performed by the algorithm) in terms of the size $n \in \mathbb{N}$ of the input of the problem.

Notation

Given $f, g : \mathbb{N} \longrightarrow \mathbb{R}^+$ increasing functions,

$$g \in O(f) \Leftrightarrow \exists n_0 \in \mathbb{N}\ \exists c \in \mathbb{R}^+\ \forall n \geq n_0\ g(n) \leq cf(n)$$

$$g \in \Omega(f) \Leftrightarrow \exists n_0 \in \mathbb{N}\ \exists c \in \mathbb{R}^+\ \forall n \geq n_0\ g(n) \geq cf(n)$$

$$g \in \Theta(f) \Leftrightarrow g \in O(f) \cap \Omega(f)$$

$$g \in o(f) \Leftrightarrow \lim_{n \to +\infty} \frac{g(n)}{f(n)} = 0$$

Notation

Given $f, g : \mathbb{N} \longrightarrow \mathbb{R}^+$ increasing functions,

$$g \in O(f) \Leftrightarrow \exists n_0 \in \mathbb{N}\ \exists c \in \mathbb{R}^+\ \forall n \geq n_0\ g(n) \leq cf(n)$$

$$g \in \Omega(f) \Leftrightarrow \exists n_0 \in \mathbb{N}\ \exists c \in \mathbb{R}^+\ \forall n \geq n_0\ g(n) \geq cf(n)$$

$$g \in \Theta(f) \Leftrightarrow g \in O(f) \cap \Omega(f)$$

$$g \in o(f) \Leftrightarrow \lim_{n \to +\infty} \frac{g(n)}{f(n)} = 0$$

Complexity of an algorithm (in a given computation model)

The worst case running time of an algorithm is $O(f)$ if the number of unit cost operations that it performs for **any** input of size n is $O(f(n))$.

The worst case running time of an algorithm is $\Omega(f)$ if the number of unit cost operations that it performs is $\Omega(f(n))$ for **some** input of size n.

Notation

Given $f, g : \mathbb{N} \longrightarrow \mathbb{R}^+$ increasing functions,

$$g \in O(f) \Leftrightarrow \exists n_0 \in \mathbb{N}\ \exists c \in \mathbb{R}^+\ \forall n \geq n_0\ g(n) \leq cf(n)$$

$$g \in \Omega(f) \Leftrightarrow \exists n_0 \in \mathbb{N}\ \exists c \in \mathbb{R}^+\ \forall n \geq n_0\ g(n) \geq cf(n)$$

$$g \in \Theta(f) \Leftrightarrow g \in O(f) \cap \Omega(f)$$

$$g \in o(f) \Leftrightarrow \lim_{n \to +\infty} \frac{g(n)}{f(n)} = 0$$

Complexity of an algorithm (in a given computation model)

The worst case running time of an algorithm is $O(f)$ if the number of unit cost operations that it performs for **any** input of size n is $O(f(n))$.

The worst case running time of an algorithm is $\Omega(f)$ if the number of unit cost operations that it performs is $\Omega(f(n))$ for **some** input of size n.

Complexity of a problem (in a given computation model)

The (time) complexity of a problem is $O(f)$ if **there exists** an algorithm solving it in $O(f)$ running time.

The (time) complexity of a problem is $\Omega(f)$ **all** algorithms solving it run in $\Omega(f(n))$ time.

Lower bounds

Theorem (Ben-Or): Let X be a semi-algebraic subset of R^d (i.e., X is the set of points in dimensions d satisfying a set of algebraic equations and/or inequations). The membership decision problem associated with X has the following lower bound:

$$\Omega(\log(\max(cc(X), cc(\mathbb{R}^d \setminus X))) - d),$$

where $cc(Y)$ stands for the number of connected components of the set Y.

Lower bounds

Theorem (Ben-Or): Let X be a semi-algebraic subset of R^d (i.e., X is the set of points in dimensions d satisfying a set of algebraic equations and/or inequations). The membership decision problem associated with X has the following lower bound:

$$\Omega(\log(\max(cc(X), cc(\mathbb{R}^d \setminus X))) - d),$$

where $cc(Y)$ stands for the number of connected components of the set Y.

Some known lower bounds. The following problems are $\Omega(n \log n)$ in the Real RAM computation model:

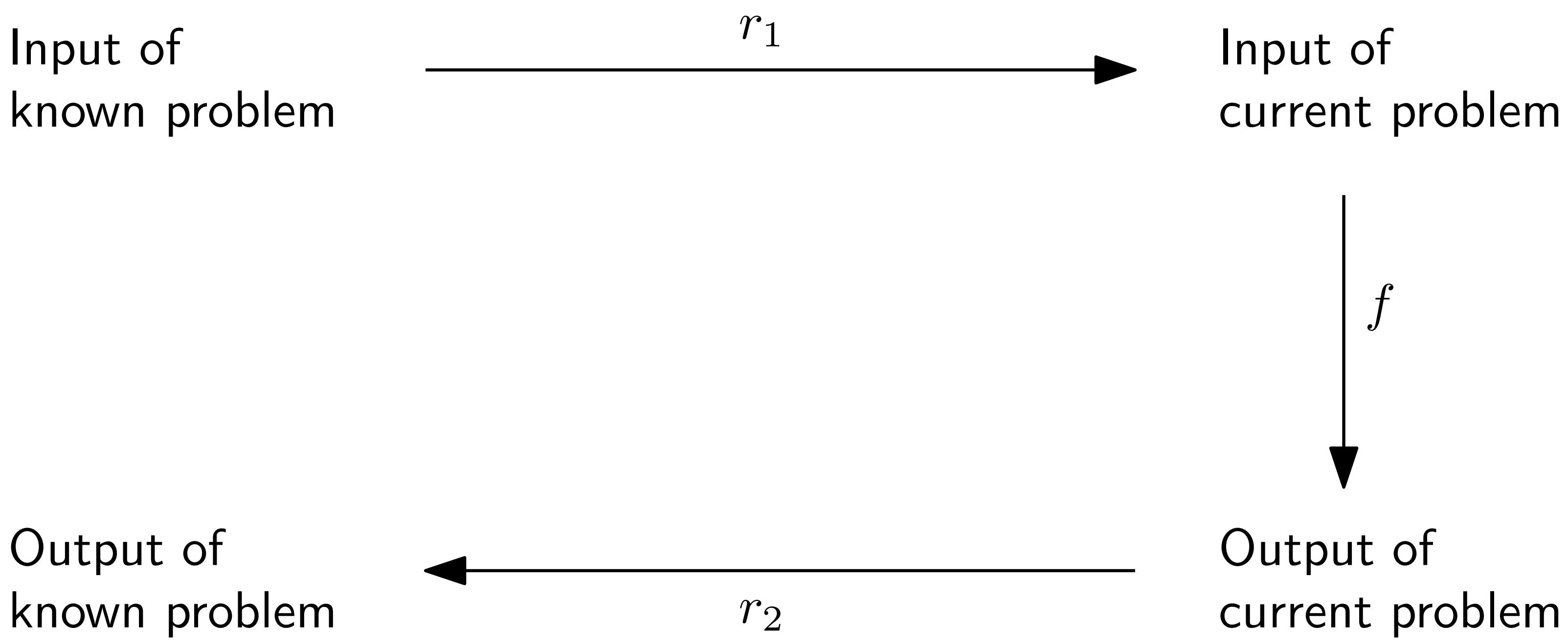
- Sorting n real (integer) numbers.
- Element uniqueness: deciding whether n given real (integer) numbers are all distinct.
- Max-gap: computing the maximum distance between two consecutive numbers from a set of n real (integer) numbers.
- Set disjointness: deciding whether two given sets of n real (integer) numbers are disjoint.
- Set equality: deciding whether two given sets of n real (integer) numbers are equal.

Lower bounds

Reduction

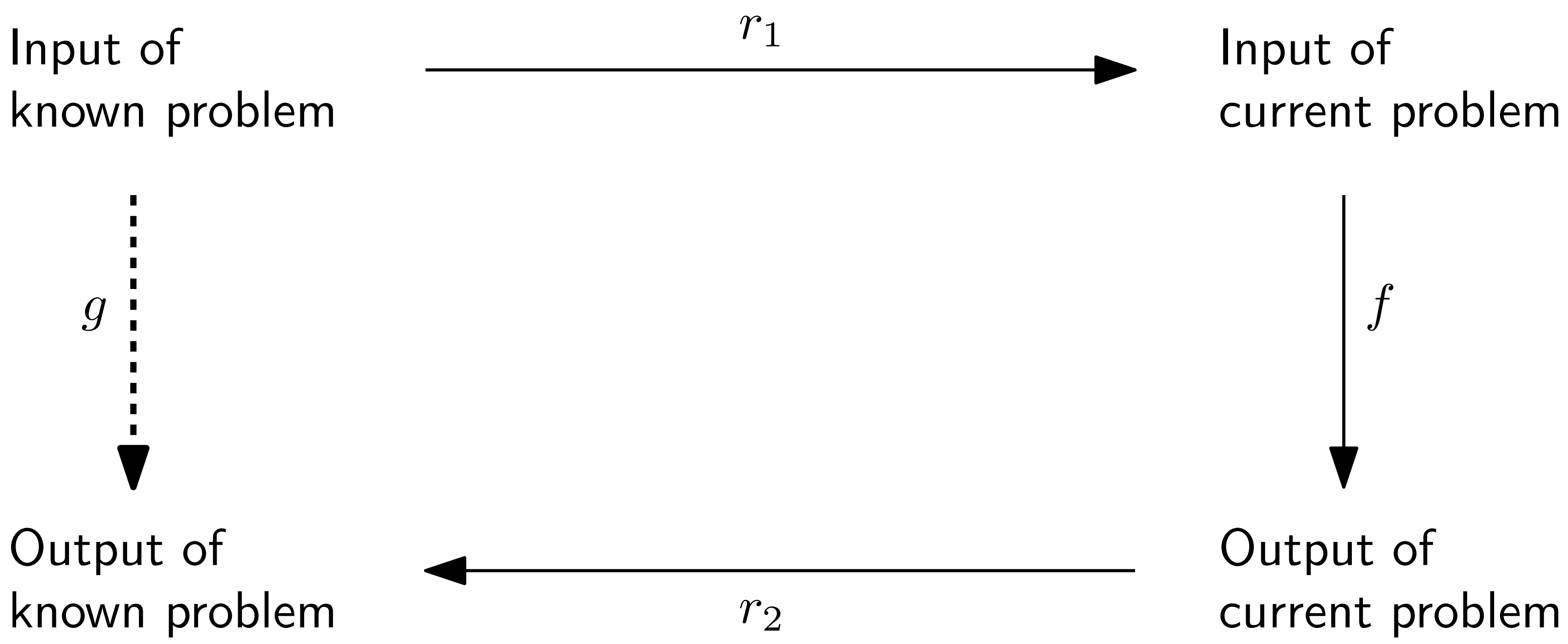
Lower bounds

Reduction



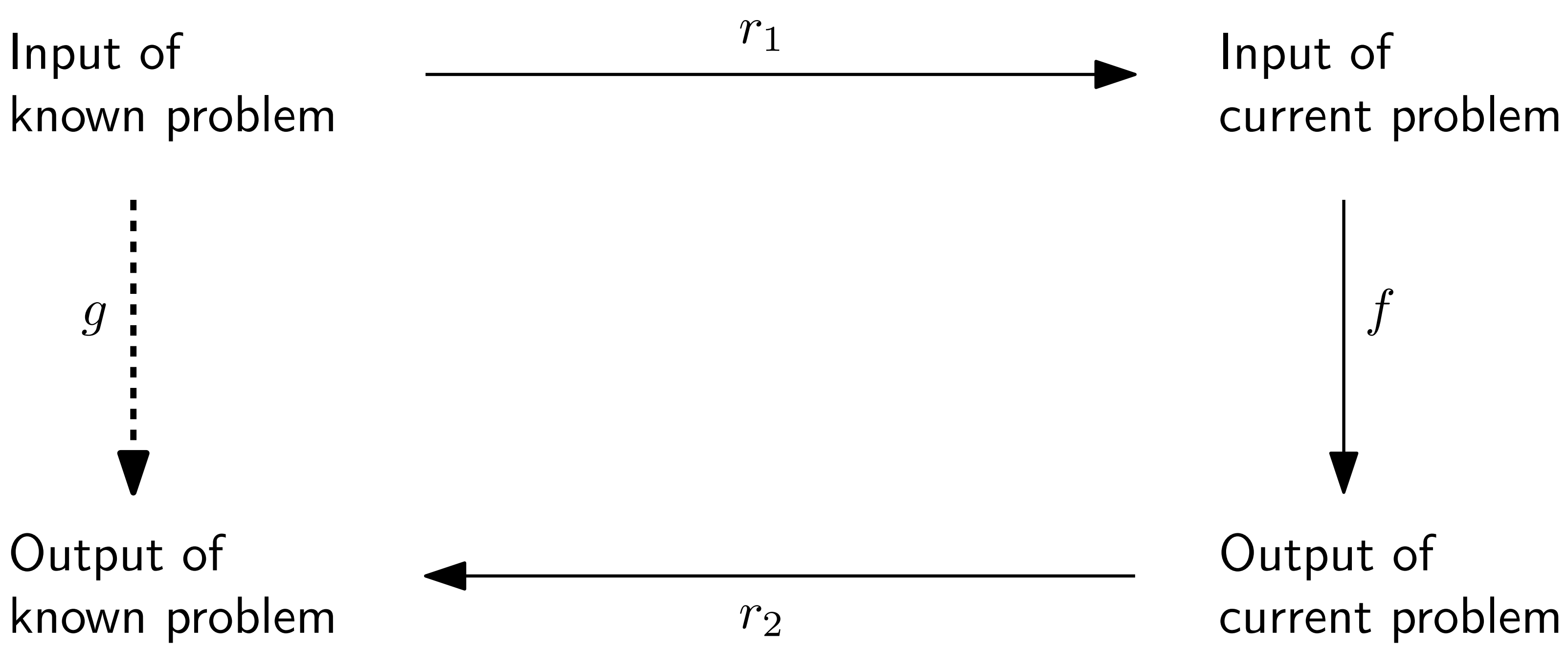
Lower bounds

Reduction



Lower bounds

Reduction



$$\left.\begin{array}{r} g \in \Omega(h(n)) \\ r_1, r_2 \in o(h(n)) \end{array}\right\} \Longrightarrow f \in \Omega(h(n))$$

Lower bounds

Reduction example

Lower bounds

Reduction example

The convex hull problem is $\Omega(n \log n)$

Lower bounds

Reduction example

The convex hull problem is $\Omega(n \log n)$

Input: $p_1, \ldots, p_n \in \mathbb{R}^2$

Output: $v_1, \ldots, v_k$ the vertices of $ch(\{p_1, \ldots, p_n\})$ in counterclockwise order

Lower bounds

Reduction example

The convex hull problem is $\Omega(n \log n)$

Input: $p_1, \ldots, p_n \in \mathbb{R}^2$

Output: $v_1, \ldots, v_k$ the vertices of $ch(\{p_1, \ldots, p_n\})$ in counterclockwise order

$x_1, \ldots, x_n \in \mathbb{R}$

Lower bounds

Reduction example

The convex hull problem is $\Omega(n \log n)$

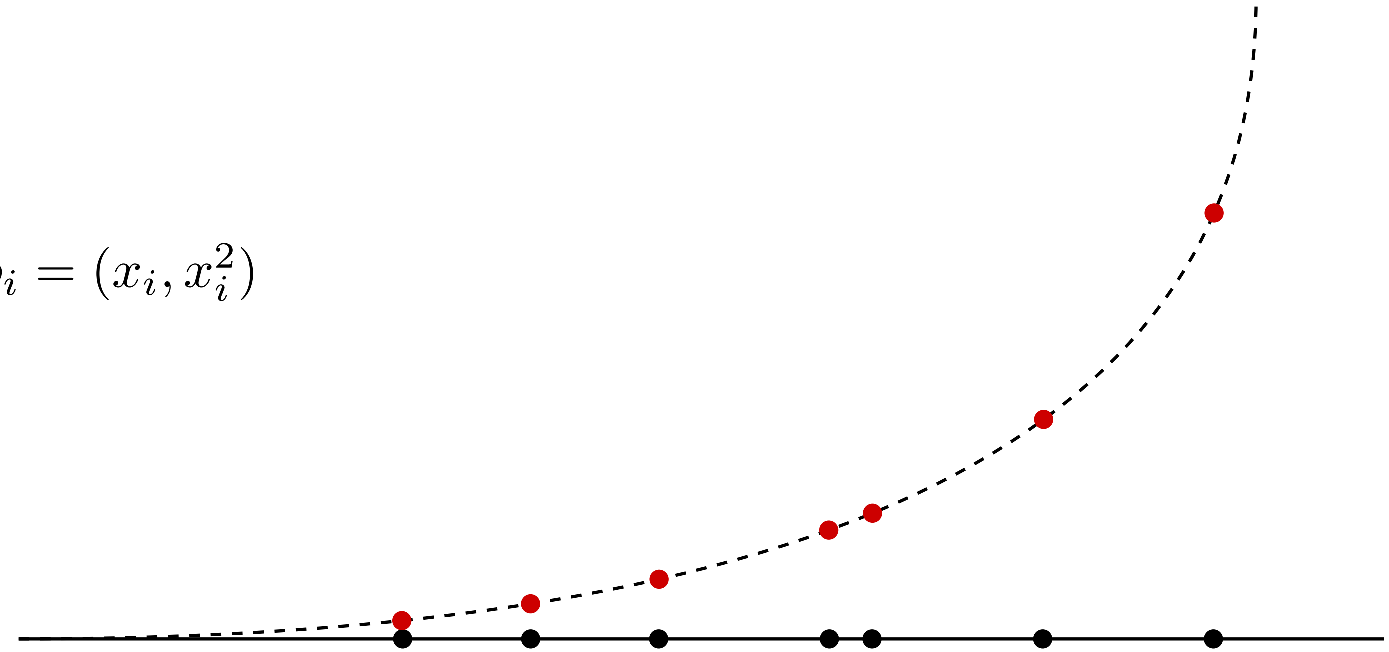
Input: $p_1, \ldots, p_n \in \mathbb{R}^2$

Output: $v_1, \ldots, v_k$ the vertices of $ch(\{p_1, \ldots, p_n\})$ in counterclockwise order

$x_1, \ldots, x_n \in \mathbb{R}$

↓

$p_1, \ldots, p_n \in \mathbb{R}^2$, $p_i = (x_i, x_i^2)$



Lower bounds

Reduction example

The convex hull problem is $\Omega(n \log n)$

Input: $p_1, \ldots, p_n \in \mathbb{R}^2$

Output: $v_1, \ldots, v_k$ the vertices of $ch(\{p_1, \ldots, p_n\})$ in counterclockwise order

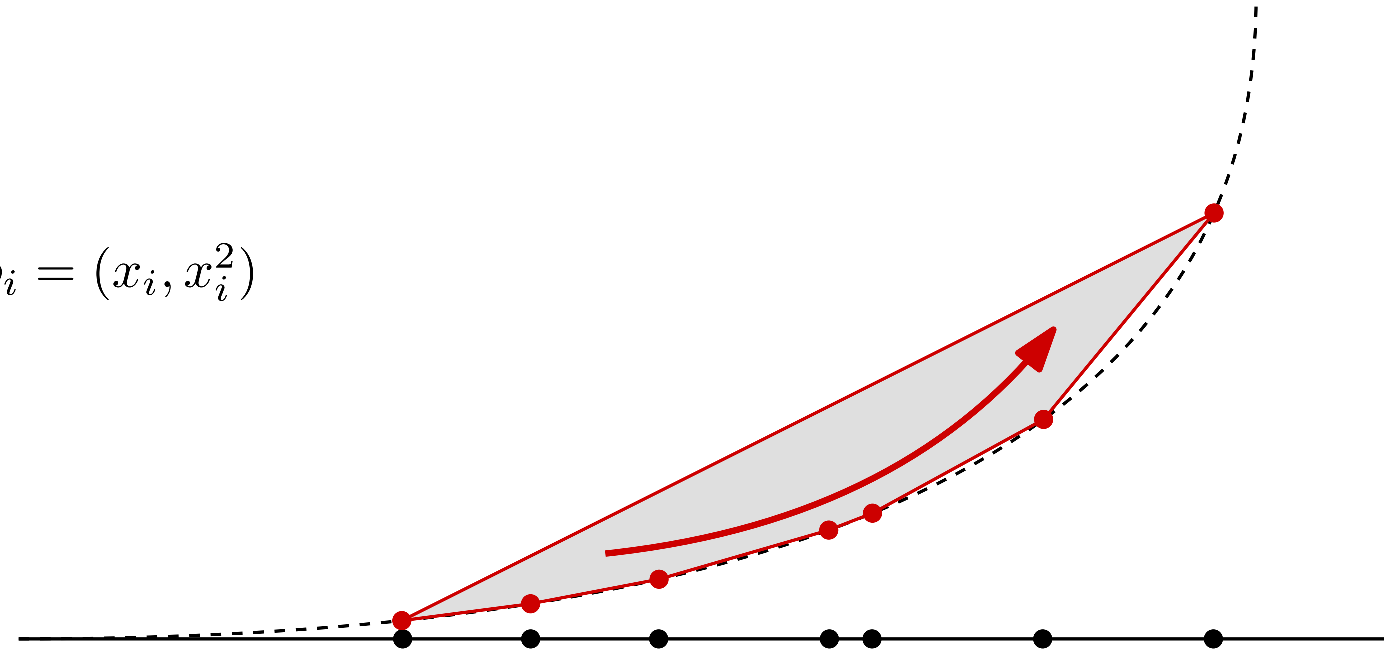
$x_1, \ldots, x_n \in \mathbb{R}$

↓

$p_1, \ldots, p_n \in \mathbb{R}^2$, $p_i = (x_i, x_i^2)$

↓

$ch(\{p_1, \ldots, p_n\})$



Lower bounds

Reduction example

The convex hull problem is $\Omega(n \log n)$

Input: $p_1, \ldots, p_n \in \mathbb{R}^2$

Output: $v_1, \ldots, v_k$ the vertices of $ch(\{p_1, \ldots, p_n\})$ in counterclockwise order

$x_1, \ldots, x_n \in \mathbb{R}$

↓

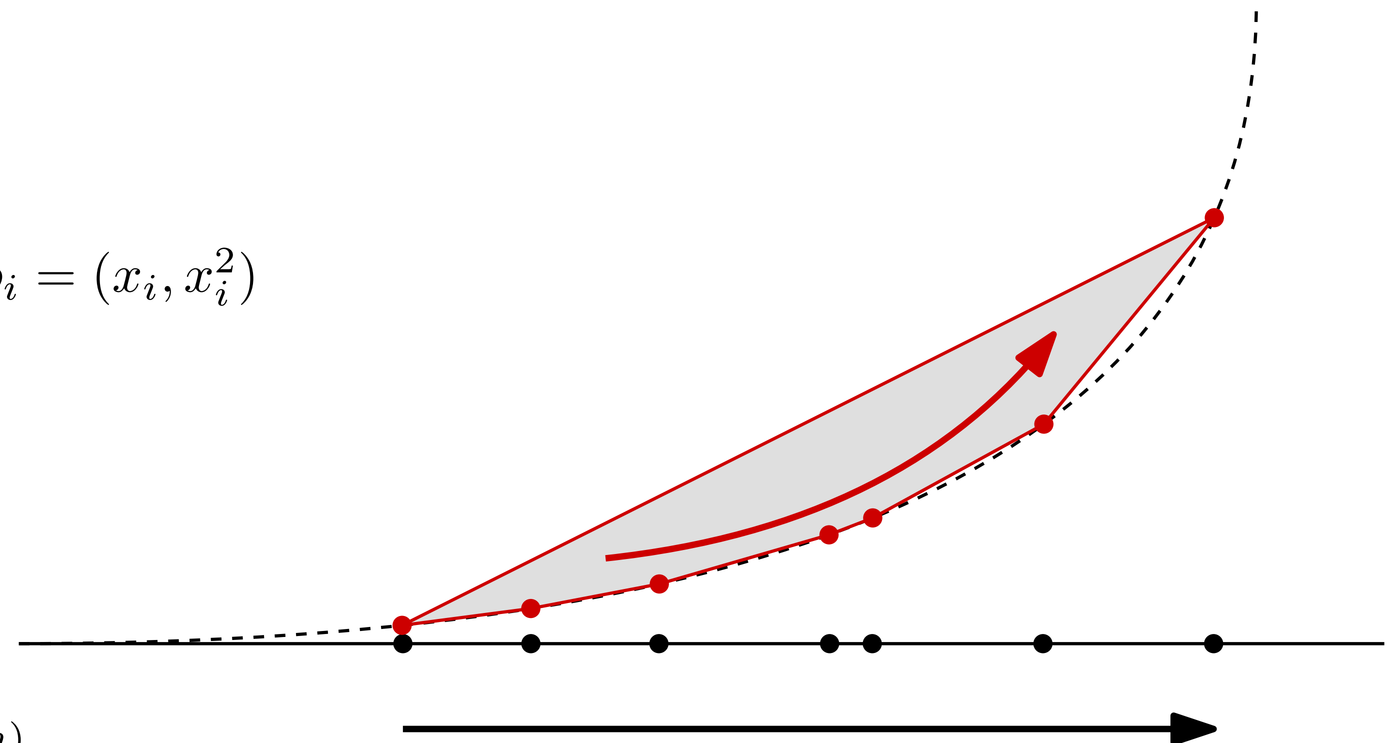
$p_1, \ldots, p_n \in \mathbb{R}^2$, $p_i = (x_i, x_i^2)$

↓

$ch(\{p_1, \ldots, p_n\})$

↓

$x_{\sigma(1)} \leq \cdots \leq x_{\sigma(n)}$



Lower bounds

Reduction example

The convex hull problem is $\Omega(n \log n)$

Input: $p_1, \dots, p_n \in \mathbb{R}^2$

Output: $v_1, \dots, v_k$ the vertices of $ch(\{p_1, \dots, p_n\})$ in counterclockwise order

$x_1, \dots, x_n \in \mathbb{R}$

↓ $O(n)$

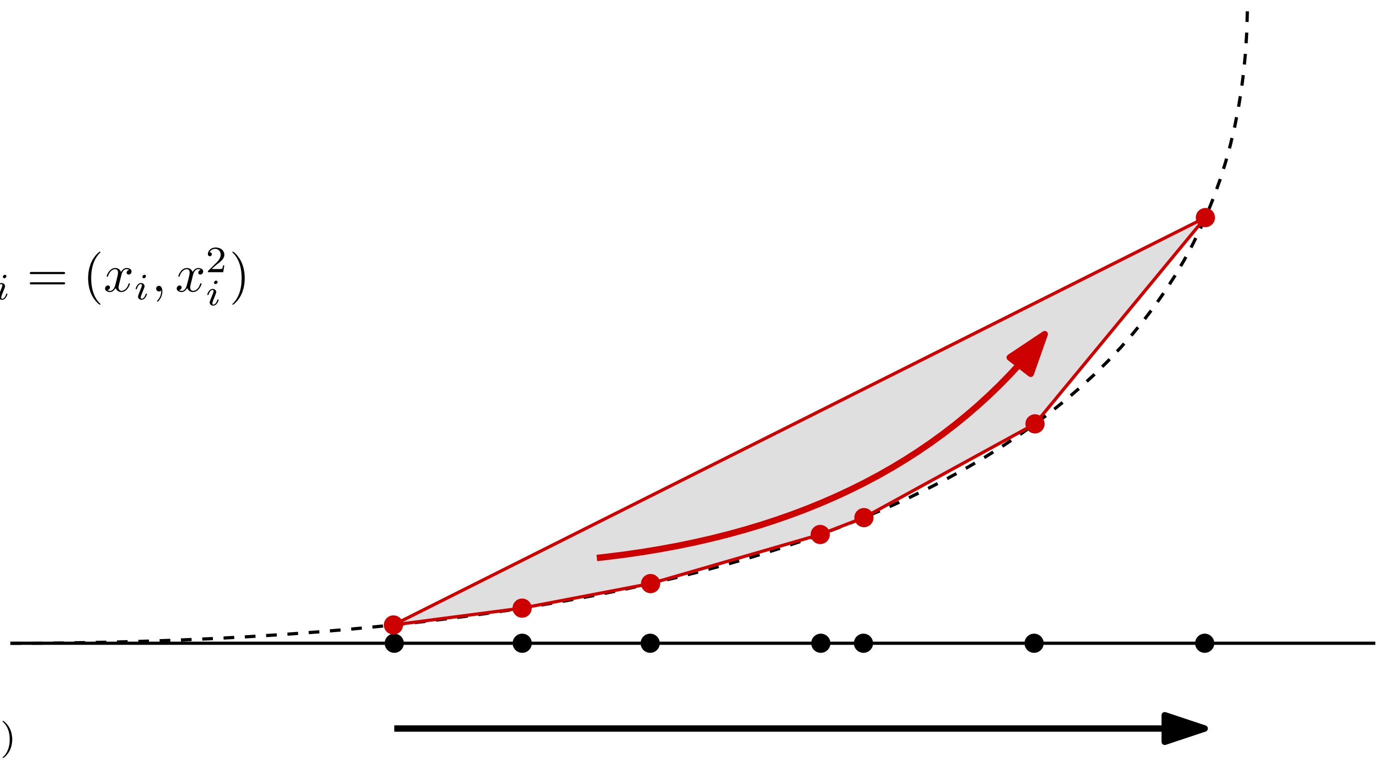
$p_1, \dots, p_n \in \mathbb{R}^2$, $p_i = (x_i, x_i^2)$

↓

$ch(\{p_1, \dots, p_n\})$

↓ $O(n)$

$x_{\sigma(1)} \leq \cdots \leq x_{\sigma(n)}$



Lower bounds

Reduction example

The convex hull problem is $\Omega(n \log n)$

Input: $p_1, \ldots, p_n \in \mathbb{R}^2$

Output: $v_1, \ldots, v_k$ the vertices of $ch(\{p_1, \ldots, p_n\})$ in counterclockwise order

$x_1, \ldots, x_n \in \mathbb{R}$

↓ $O(n)$

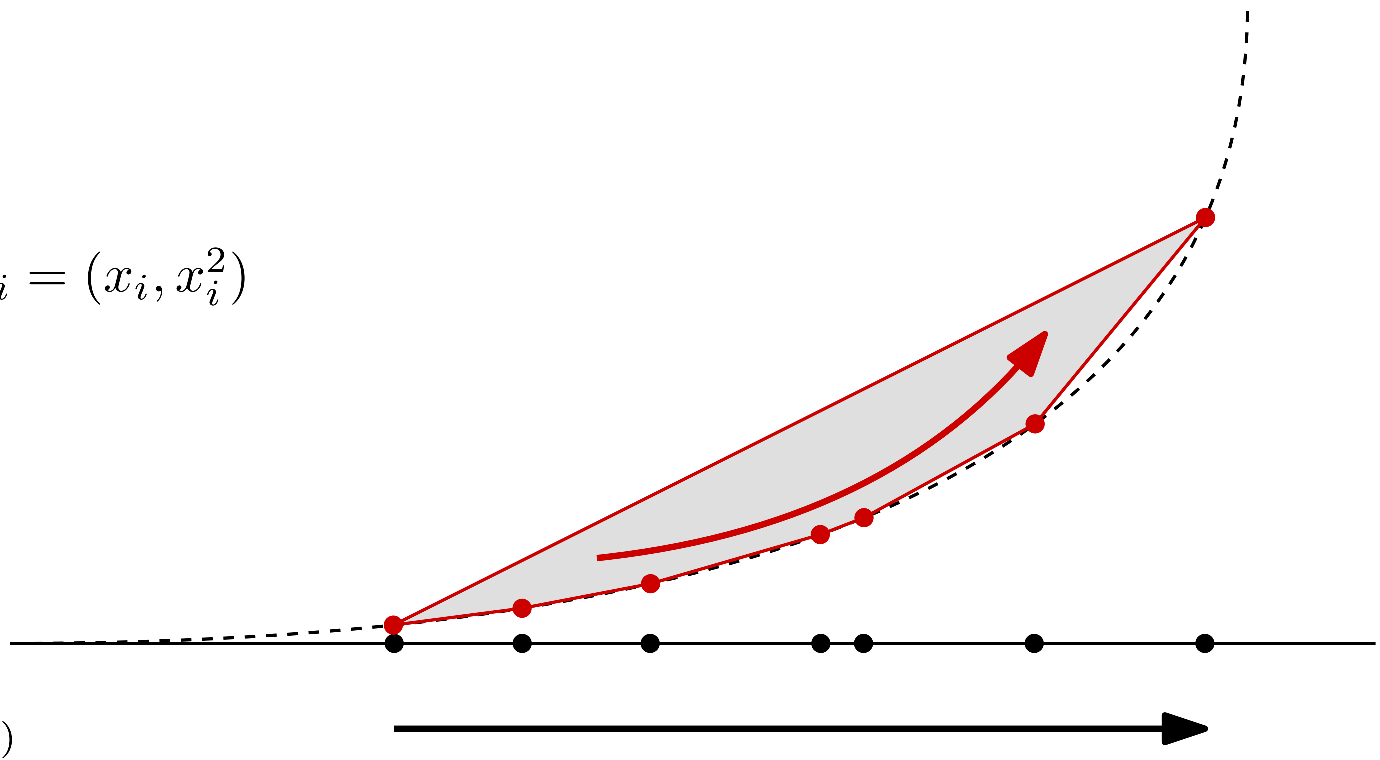
$p_1, \ldots, p_n \in \mathbb{R}^2$, $p_i = (x_i, x_i^2)$

↓ $\Omega(n \log n)$

$ch(\{p_1, \ldots, p_n\})$

↓ $O(n)$

$x_{\sigma(1)} \leq \cdots \leq x_{\sigma(n)}$



FURTHER READING

F. P.Preparata and M. I Shamos
Computational Geometry: An Introduction
Springer-Verlag, 1985.

J-D. Boissonnat and M. Yvinec
Algorithmic Geometry
Cambridge University Press, 1997.