Hard problems

SAT: given a formula in conj. normal form on n variables
is it satisfiable?
conjecture: no $O(2^{(1-\varepsilon)n})$ algorithm (SETH)

OV: given n vectors in $\{0,1\}^d$ (for small d)
are any two orthogonal?
conjecture: no $O(n^{2-\varepsilon})$ algorithm

APSP: given a weighted graph with n vertices
compute the distance between any pair of vertices
conjecture: no $O(n^{3-\varepsilon})$ algorithm

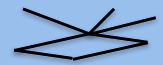
3SUM: given n integers
do any three sum to 0?
conjecture: no $O(n^{2-\varepsilon})$ algorithm

Relations = Reductions

transfer hardness of one problem to another one by reductions

problem P

instance I



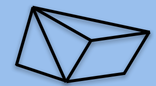
size n

reduction

$r(n)$ time

problem Q

instance J



size $s(n)$

I is a 'yes'-instance $\iff$ J is a 'yes'-instance

$t(n)$ algorithm for Q implies a $r(n) + t(s(n))$ algorithm for P

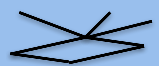
if P has no $r(n) + t(s(n))$ algorithm then Q has no $t(n)$ algorithm

Relations = Reductions

transfer hardness of one problem to another one by reductions

problem P

instance I



size n

reduction

⋮

total time $r(n)$

problem Q

instance I_1
size $s_1(n)$

⋮

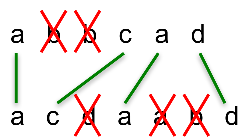
instance I_k
size $s_k(n)$

$t(n)$ algorithm for Q implies a $r(n) + \sum_{i=1}^{k} t(s_i(n))$ algorithm for P

Showcase Results

longest common subseq. edit distance, longest palindromic subsequence, Fréchet distance...	$O(n^2)$	SETH-hard $n^{2-\varepsilon}$ [B.,Künnemann'15, Abboud,Backurs,V-Williams'15]

given two strings x, y of length n,
compute the **longest string** z that
is a **subsequence** of both x and y



Showcase Results

longest common subseq. $O(n^2)$ SETH-hard $n^{2-\varepsilon}$

edit distance, longest palindromic subsequence, Fréchet distance... [B.,Künnemann'15, Abboud,Backurs,V-Williams'15]

we can stop searching for faster algorithms!

in this sense, conditional lower bounds replace NP-hardness

$O(n^{2-\varepsilon})$ algorithms are unlikely to exist

improvements are at least as hard as a **breakthrough for SAT**

Showcase Results

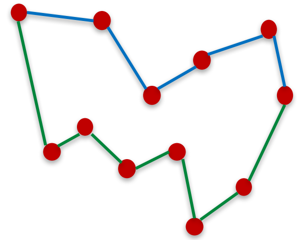
longest common subseq. $O(n^2)$ SETH-hard $n^{2-\varepsilon}$

edit distance, longest palindromic subsequence, Fréchet distance... [B.,Künnemann'15, Abboud,Backurs,V-Williams'15]

bitonic TSP $O(n^2)$ $O(n \log^4 n)$

longest increasing subsequence, matrix chain multiplication... [de Berg,Buchin,Jansen,Woeginger'16]

given n points in the plane, compute a **minimum tour connecting all points** among all tours consisting of **two x-monotone parts**



Showcase Results

longest common subseq. $O(n^2)$ SETH-hard $n^{2-\varepsilon}$

edit distance, longest palindromic subsequence, Fréchet distance... [B.,Künnemann'15, Abboud,Backurs,V-Williams'15]

bitonic TSP $O(n^2)$ $O(n \log^4 n)$

longest increasing subsequence, matrix chain multiplication... [de Berg,Buchin,Jansen,Woeginger'16]

maximum submatrix $O(n^3)$ APSP−hard $n^{3-\varepsilon}$

minimum weight triangle, graph centrality measures... [Backurs,Dikkala,Tzamos'16]

given matrix A over $\mathbb{Z}$, **choose a submatrix** (consisting of consecutive rows and columns of A) **maximizing the sum of all entries**

-3	2	-2	0
-2	5	7	-2
1	3	-1	1
3	-2	0	0

Showcase Results

longest common subseq. $O(n^2)$ SETH-hard $n^{2-\varepsilon}$

edit distance, longest palindromic subsequence, Fréchet distance... [B.,Künnemann'15, Abboud,Backurs,V-Williams'15]

bitonic TSP $O(n^2)$ $O(n \log^4 n)$

longest increasing subsequence, matrix chain multiplication... [de Berg,Buchin,Jansen,Woeginger'16]

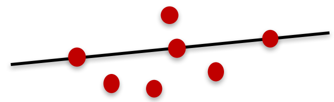
maximum submatrix $O(n^3)$ APSP−hard $n^{3-\varepsilon}$

minimum weight triangle, graph centrality measures... [Backurs,Dikkala,Tzamos'16]

colinearity $O(n^2)$ 3SUM−hard $n^{2-\varepsilon}$

motion planning, polygon containment... [Gajentaan,Overmars'95]

given n points in the plane, are any three of them on a line?

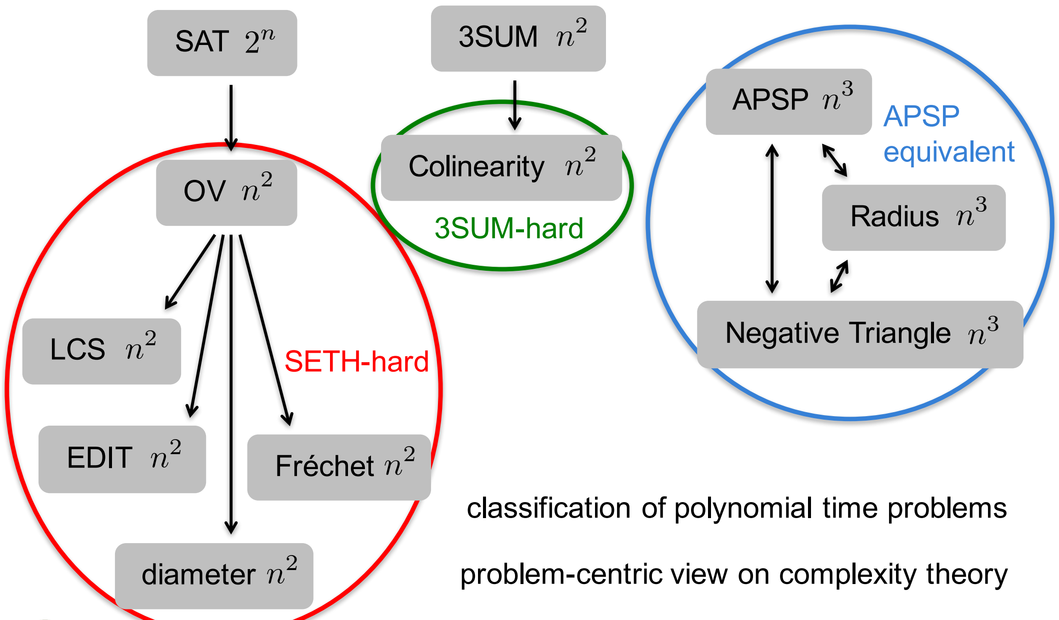


Showcase Results

longest common subseq.	$O(n^2)$	SETH-hard $n^{2-\varepsilon}$	
edit distance, longest palindromic subsequence, Fréchet distance...		[B.,Künnemann'15, Abboud,Backurs,V-Williams'15]	
bitonic TSP	$O(n^2)$	$O(n \log^4 n)$	
longest increasing subsequence, matrix chain multiplication...		[de Berg,Buchin,Jansen,Woeginger'16]	
maximum submatrix	$O(n^3)$	APSP−hard $n^{3-\varepsilon}$	
minimum weight triangle, graph centrality measures...		[Backurs,Dikkala,Tzamos'16]	
colinearity	$O(n^2)$	3SUM−hard $n^{2-\varepsilon}$	
motion planning, polygon containment...		[Gajentaan,Overmars'95]	

Open: optimal binary search tree $O(n^2)$
knapsack $O(nW)$
many more...

Complexity Inside P



classification of polynomial time problems

problem-centric view on complexity theory

II. An Example for OV-hardness

Orthogonal Vectors Hypothesis

Input: Sets $A, B \subseteq \{0,1\}^d$ of size n

Task: Decide whether there are $a \in A, b \in B$ such that $a \perp b$

$\Leftrightarrow \sum_{i=1}^{d} a_i \cdot b_i = 0$

$\Leftrightarrow$ for all $1 \le i \le d$: $a_i = 0$ or $b_i = 0$

$A = \{(1,1,1), (1,1,0), (1,0,1), (0,0,1)\}$

$B = \{(0,1,0), (0,1,1), (1,0,1), (1,1,1)\}$

trivial $O(n^2 d)$ algorithm

best known algorithm: $O(n^{2-1/O(\log c)})$ where $d = c \log n$ [Lecture 03]

OV-Hypothesis: no $O(n^{2-\varepsilon}\mathrm{poly}(d))$ algorithm for any $\varepsilon > 0$

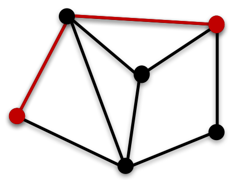
„OV has no $O(n^{2-\varepsilon})$ algorithm, even if $d = \mathrm{polylog}\, n$"

Graph Diameter Problem

Input: An unweighted graph $G = (V, E)$

Task: Compute the largest distance between any pair of vertices

$$= \max_{u,v \in V} d_G(u,v)$$



diameter 2

Easy algorithm:

Single-source-shortest-paths:

Dijkstra's algorithm: $O(m + n \log n)$

All-pairs-shortest-paths:

Dijkstra from every node: $O\big(n(m + n \log n)\big) \le O(n\, m \log n)$

from this information we can compute the diameter in time $O(n^2)$

OV-Hardness Result

OV: sets $A, B \subseteq \{0,1\}^d$ of size n → reduction, time $O(dn)$ → **diameter**: graph G, $O(n)$ vertices, $O(dn)$ edges

$O(n^{2-\varepsilon}\text{poly}(d))$ algorithm $\Longleftarrow$ $O((nm)^{1-\varepsilon})$ algorithm

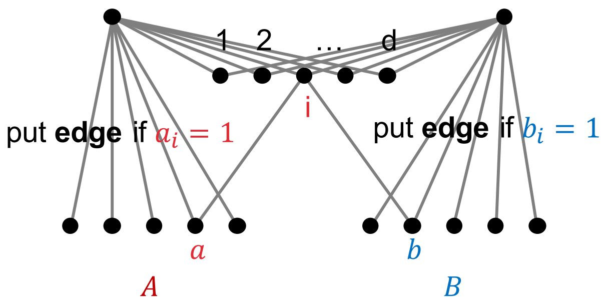
Thm: Diameter has no $O((nm)^{1-\varepsilon})$ algorithm unless the OV-Hypothesis fails. [Roditty,V-Williams'13]

max planck institut informatik

Proof

OV: sets $A, B \subseteq \{0,1\}^d$ of size n → reduction, time $O(dn)$ → **diameter**: graph G, $O(n)$ vertices, $O(dn)$ edges

Proof: can assume: every vector has at least one ‚1'



$d(a, b) = 2 \Leftrightarrow a, b$ not orthogonal

diameter = 3 ⇔ there exists an orthogonal pair

max planck institut informatik

Proof

OV: sets $A, B \subseteq \{0,1\}^d$ of size n → reduction, time $O(dn)$ → **diameter**: graph G, $O(n)$ vertices, $O(dn)$ edges

Remark: Even deciding whether the diameter is ≤ 2 or ≥ 3 has no $O((nm)^{1-\varepsilon})$ algorithm unless OVH fails.

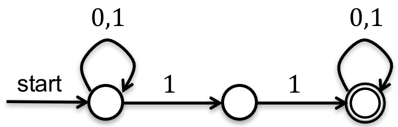
There is no $(3/2 - \varepsilon)$-approximation for the diameter in time $O((nm)^{1-\varepsilon})$ unless OVH fails.

max planck institut informatik

III. Another Example for OV-hardness

NFA Acceptance Problem

nondeterministic finite automaton G accepts input string s if there is a walk in G from starting state to some accepting state, labelled with s



string: 01011010

dynamic programming algorithm in time $O(|s||G|)$:

$T[i] :=$ *set of states reachable via walks labelled with* $s[1..i]$

$T[0] := \{\text{starting state}\}$

$T[i] := \{v \mid \exists u \in T[i-1] \text{ and } \exists \text{ transition } u \to v \text{ labelled } s[i]\}$

OV-Hardness Result

OV
sets $A, B \subseteq \{0,1\}^d$
of size n

reduction →
time $O(dn)$

NFA acceptance
grammar G, string s
$|G| = O(dn)$
$|s| = O(dn)$

$O(n^{2-\varepsilon}\text{poly}(d))$ algorithm $\Longleftarrow$ $O((|s|\ |G|)^{1-\varepsilon})$ algorithm

Thm: NFA acceptance has no $O((|s|\ |G|)^{1-\varepsilon})$ algorithm unless OVH fails. [Impagliazzo]

Proof

OV
sets $A, B \subseteq \{0,1\}^d$
of size n

reduction →
time $O(dn)$

NFA acceptance
grammar G, string s
$|G| = O(dn)$
$|s| = O(dn)$

Proof:

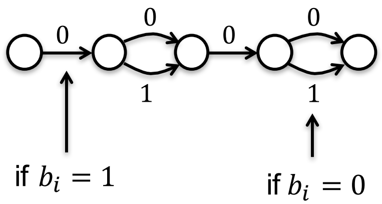
fix some $a \in A$:

in string s:

0011
↑
$= a_1 a_2 \ldots a_d$

fix some $b \in B$:

in NFA G:



Proof

OV

sets $A, B \subseteq \{0,1\}^d$

of size n

reduction → time $O(dn)$

NFA acceptance

grammar G, string s

$|G| = O(dn)$

$|s| = O(dn)$

Proof:

fix some $a \in A$:

in string s:

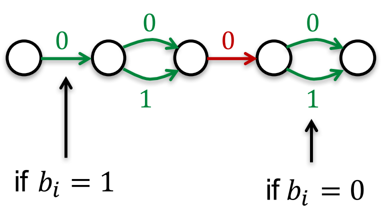
0011

↑

$= a_1 a_2 \ldots a_d$

fix some $b \in B$:

in NFA G:



Proof

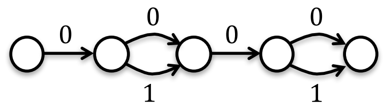
fix some $a \in A$:

in string s:

0011

fix some $b \in B$:

in NFA G:



---

string s $= \$1100\$0110\$ \ldots \$0011\$$

(for all $a \in A$)

✔ equivalent to OV instance

✔ size $|s| = |G| = O(dn)$

NFA G: