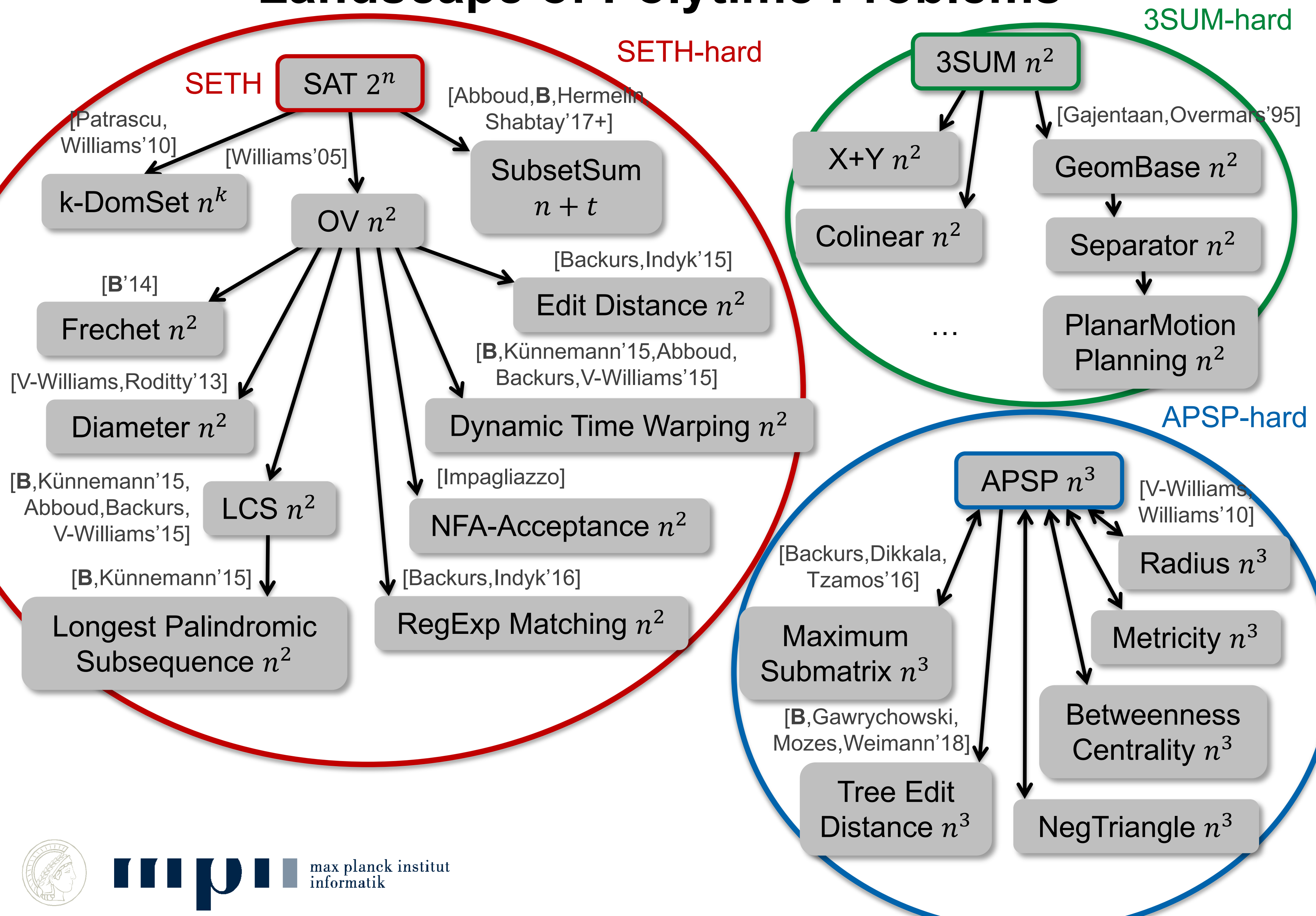
max planck institut informatik

Fine-Grained Complexity - Hardness in P

Lecture 3: 3SUM

Karl Bringmann

Landscape of Polytime Problems



max planck institut
informatik

3SUM

Problem 3SUM: Given integers $a_1, \dots, a_n, b_1, \dots, b_n, c_1, \dots, c_n$ are there i, j, k such that $a_i + b_j = c_k$?

Algorithms: Naïve: $O(n^3)$

Well-known: $O(n^2)$

3SUM-Hypothesis: $\forall \varepsilon > 0$: 3SUM has no $O(n^{2-\varepsilon})$-time algorithm

[Gajentaan,Overmars'95]

We assume that we can add/subtract/compare input integers in constant time

Can assume that the a_i, b_j, c_k are distinct and from some universe $\{1, \dots, U\}$

Proof: Set M such that $|a_i|, |b_j|, |c_k| < M$ for all i, j, k

Add: $2M$ to every a_i
$4M$ to every b_j
$6M$ to every c_k

Resulting instance is **equivalent**, has **distinct** input numbers, and **universe** $\{1, \dots, 7M\}$

3SUM

Problem 3SUM: Given integers $a_1, \dots, a_n, b_1, \dots, b_n, c_1, \dots, c_n$ are there i, j, k such that $a_i + b_j = c_k$?

Algorithms: Naïve: $O(n^3)$

Well-known: $O(n^2)$

3SUM-Hypothesis: $\forall \varepsilon > 0$: 3SUM has no $O(n^{2-\varepsilon})$-time algorithm

[Gajentaan,Overmars'95]

$O(n^2 \log n)$-time algorithm:

sort $c_1 \leq \dots \leq c_n$

for each i, j:

binary search for $a_i + b_j$ among $c_1 \leq \dots \leq c_n$

$O(n^2)$-time randomized algorithm:

put each c_k into a hashmap

for each i, j:

check whether $a_i + b_j$ is in the hashmap

Quadratic Algorithm

Given integers $a_1, \dots, a_n, b_1, \dots, b_n, c_1, \dots, c_n$

are there i, j, k such that $a_i + b_j = c_k$?

sort in increasing order: $a_1 \leq \cdots \leq a_n,\ b_1 \leq \cdots \leq b_n,\ c_1 \leq \cdots \leq c_n$

for each c_k: *check whether there are i, j s.t. $a_i + b_j = c_k$*

initialize $i = n, j = 1$

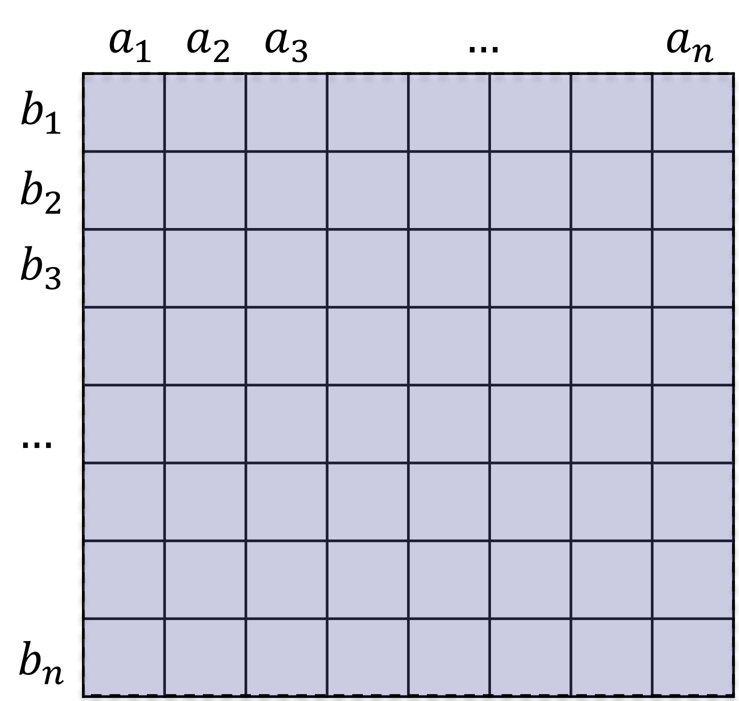
while $i > 0$ and $j \leq n$:

if $a_i + b_j = c_k$: return (a_i, b_j, c_k)

if $a_i + b_j > c_k$: $i \coloneqq i - 1$

if $a_i + b_j < c_k$: $j \coloneqq j + 1$

return “no solution”



Quadratic Algorithm

Given integers $a_1, \dots, a_n, b_1, \dots, b_n, c_1, \dots, c_n$
are there i, j, k such that $a_i + b_j = c_k$?

sort in increasing order: $a_1 \leq \cdots \leq a_n,\ b_1 \leq \cdots \leq b_n,\ c_1 \leq \cdots \leq c_n$

for each c_k: *check whether there are* i, j *s.t.* $a_i + b_j = c_k$

initialize $i = n, j = 1$

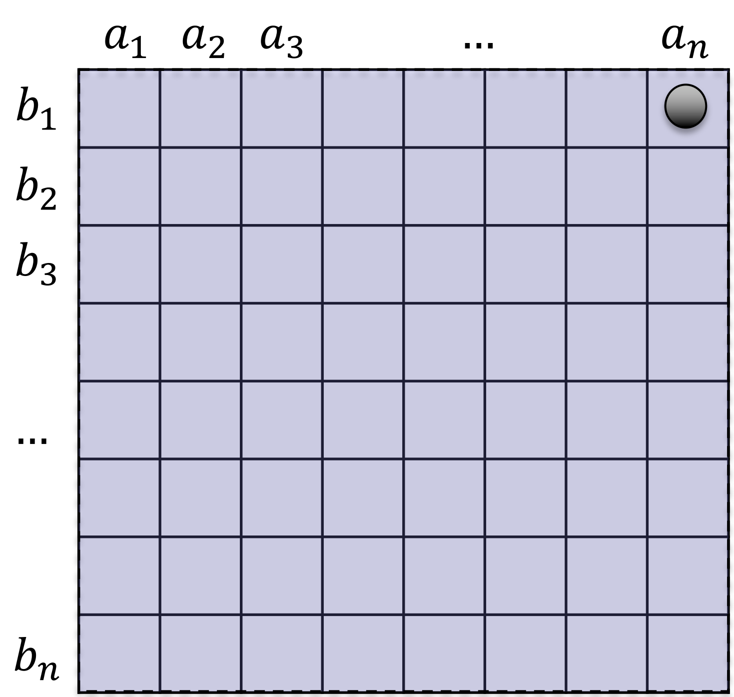
while $i > 0$ and $j \leq n$:

if $a_i + b_j = c_k$: return (a_i, b_j, c_k)

if $a_i + b_j > c_k$: $i := i - 1$

if $a_i + b_j < c_k$: $j := j + 1$

return “no solution”



Quadratic Algorithm

Given integers $a_1, \dots, a_n, b_1, \dots, b_n, c_1, \dots, c_n$

are there i, j, k such that $a_i + b_j = c_k$?

sort in increasing order: $a_1 \leq \cdots \leq a_n,\ b_1 \leq \cdots \leq b_n,\ c_1 \leq \cdots \leq c_n$

for each c_k: *check whether there are i, j s.t. $a_i + b_j = c_k$*

initialize $i = n, j = 1$

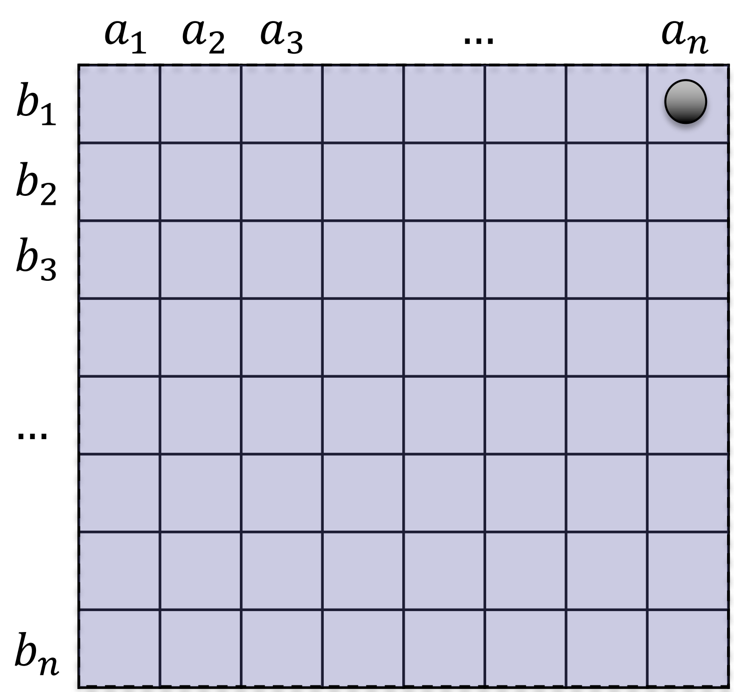
while $i > 0$ and $j \leq n$:

if $a_i + b_j = c_k$: return (a_i, b_j, c_k)

if $a_i + b_j > c_k$: $i \coloneqq i - 1$

if $a_i + b_j < c_k$: $j \coloneqq j + 1$

return "no solution"



max planck institut informatik

Quadratic Algorithm

Given integers $a_1, \dots, a_n, b_1, \dots, b_n, c_1, \dots, c_n$

are there i, j, k such that $a_i + b_j = c_k$?

sort in increasing order: $a_1 \le \cdots \le a_n,\ b_1 \le \cdots \le b_n,\ c_1 \le \cdots \le c_n$

for each c_k: *check whether there are i, j s.t. $a_i + b_j = c_k$*

initialize $i = n, j = 1$

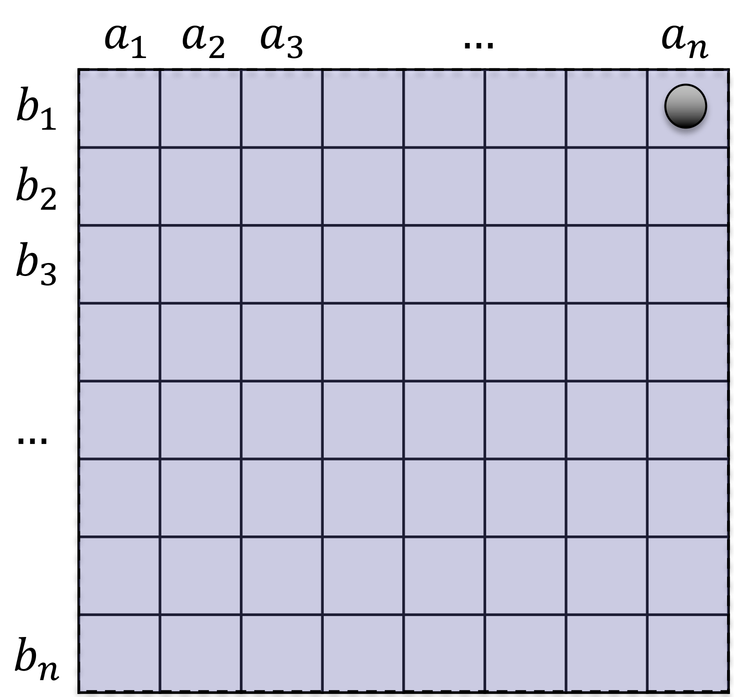
while $i > 0$ and $j \le n$:

if $a_i + b_j = c_k$: return (a_i, b_j, c_k)

if $a_i + b_j > c_k$: $i := i - 1$

if $a_i + b_j < c_k$: $j := j + 1$

return "no solution"



Quadratic Algorithm

Given integers $a_1, \ldots, a_n, b_1, \ldots, b_n, c_1, \ldots, c_n$
are there i, j, k such that $a_i + b_j = c_k$?

sort in increasing order: $a_1 \leq \cdots \leq a_n$, $b_1 \leq \cdots \leq b_n$, $c_1 \leq \cdots \leq c_n$

for each c_k: *check whether there are i, j s.t. $a_i + b_j = c_k$*

initialize $i = n, j = 1$

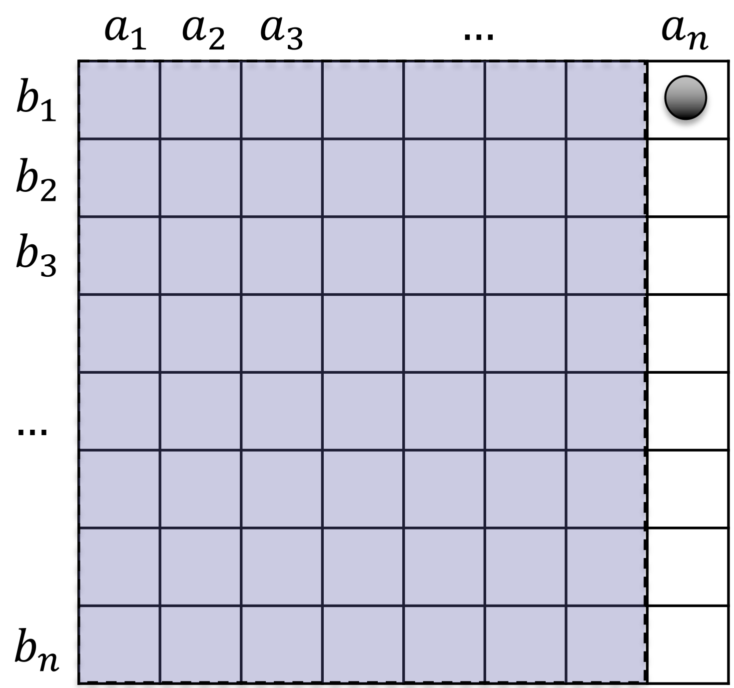
while $i > 0$ and $j \leq n$:

if $a_i + b_j = c_k$: return (a_i, b_j, c_k)

if $a_i + b_j > c_k$: $i \coloneqq i - 1$

if $a_i + b_j < c_k$: $j \coloneqq j + 1$

return "no solution"



Quadratic Algorithm

Given integers $a_1, \dots, a_n, b_1, \dots, b_n, c_1, \dots, c_n$
are there i, j, k such that $a_i + b_j = c_k$?

sort in increasing order: $a_1 \leq \cdots \leq a_n,\ b_1 \leq \cdots \leq b_n,\ c_1 \leq \cdots \leq c_n$

for each c_k: *check whether there are i, j s.t. $a_i + b_j = c_k$*

initialize $i = n, j = 1$

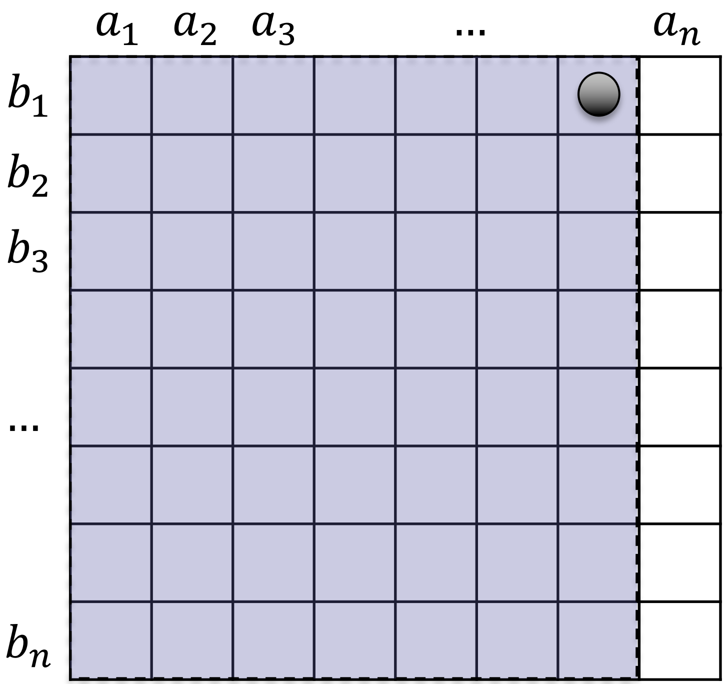
while $i > 0$ and $j \leq n$:

if $a_i + b_j = c_k$: return (a_i, b_j, c_k)

if $a_i + b_j > c_k$: $i := i - 1$

if $a_i + b_j < c_k$: $j := j + 1$

return "no solution"



Quadratic Algorithm

Given integers $a_1, \dots, a_n, b_1, \dots, b_n, c_1, \dots, c_n$

are there i, j, k such that $a_i + b_j = c_k$?

sort in increasing order: $a_1 \leq \cdots \leq a_n,\ b_1 \leq \cdots \leq b_n,\ c_1 \leq \cdots \leq c_n$

for each c_k: *check whether there are i, j s.t. $a_i + b_j = c_k$*

initialize $i = n, j = 1$

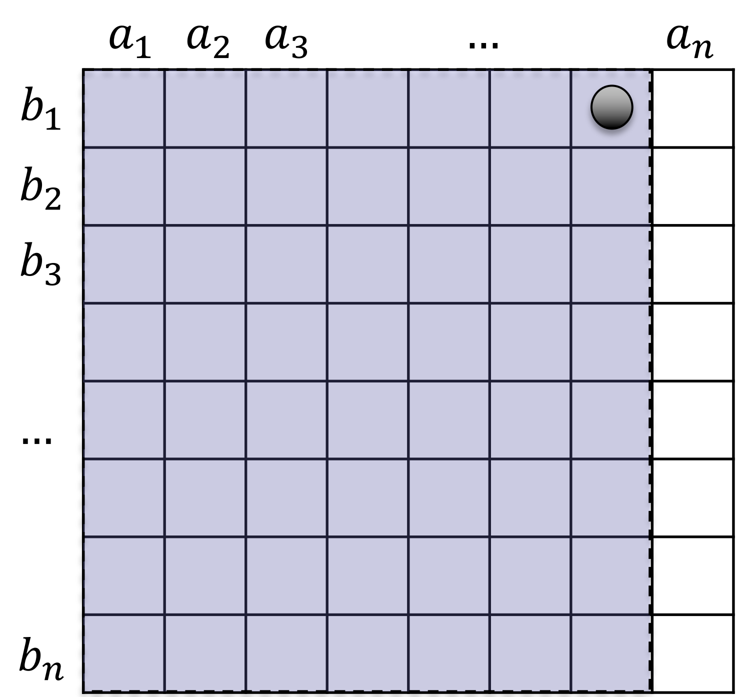
while $i > 0$ and $j \leq n$:

if $a_i + b_j = c_k$: return (a_i, b_j, c_k)

if $a_i + b_j > c_k$: $i \coloneqq i - 1$

if $a_i + b_j < c_k$: $j \coloneqq j + 1$

return "no solution"



max planck institut
informatik

Quadratic Algorithm

Given integers $a_1, \dots, a_n, b_1, \dots, b_n, c_1, \dots, c_n$

are there i, j, k such that $a_i + b_j = c_k$?

sort in increasing order: $a_1 \leq \cdots \leq a_n,\ b_1 \leq \cdots \leq b_n,\ c_1 \leq \cdots \leq c_n$

for each c_k: *check whether there are* i, j *s.t.* $a_i + b_j = c_k$

initialize $i = n, j = 1$

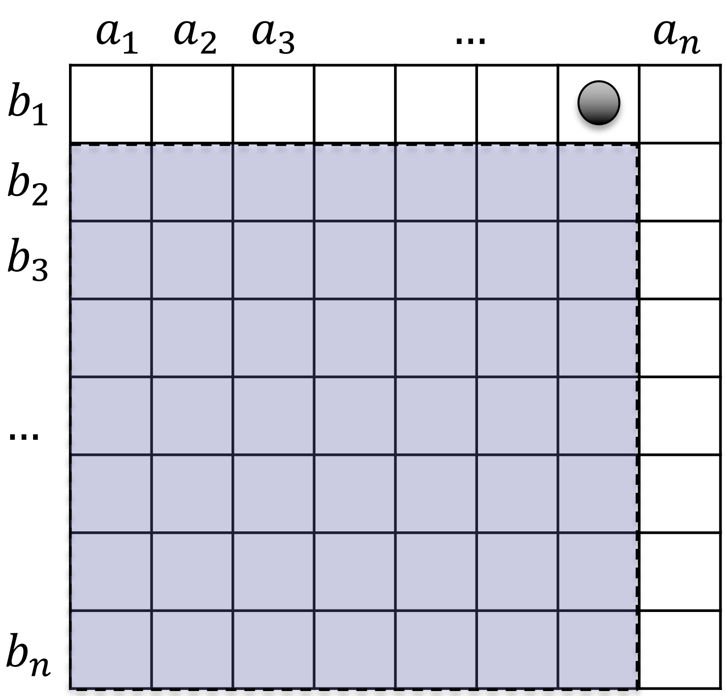
while $i > 0$ and $j \leq n$:

if $a_i + b_j = c_k$: return (a_i, b_j, c_k)

if $a_i + b_j > c_k$: $i \coloneqq i - 1$

if $a_i + b_j < c_k$: $j \coloneqq j + 1$

return "no solution"



Quadratic Algorithm

Given integers $a_1, \dots, a_n, b_1, \dots, b_n, c_1, \dots, c_n$

are there i, j, k such that $a_i + b_j = c_k$?

sort in increasing order: $a_1 \le \dots \le a_n,\ b_1 \le \dots \le b_n,\ c_1 \le \dots \le c_n$

for each c_k: *check whether there are* i, j *s.t.* $a_i + b_j = c_k$

initialize $i = n, j = 1$

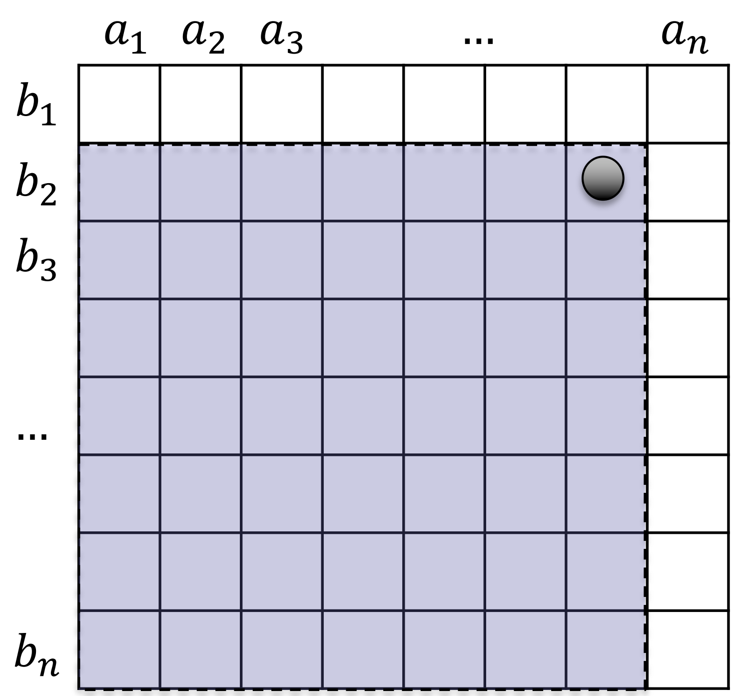
while $i > 0$ and $j \le n$:

if $a_i + b_j = c_k$: return (a_i, b_j, c_k)

if $a_i + b_j > c_k$: $i \coloneqq i - 1$

if $a_i + b_j < c_k$: $j \coloneqq j + 1$

return "no solution"



Quadratic Algorithm

Given integers $a_1, \dots, a_n, b_1, \dots, b_n, c_1, \dots, c_n$

are there i, j, k such that $a_i + b_j = c_k$?

sort in increasing order: $a_1 \le \cdots \le a_n,\ b_1 \le \cdots \le b_n,\ c_1 \le \cdots \le c_n$

for each c_k: *check whether there are i, j s.t. $a_i + b_j = c_k$*

initialize $i = n, j = 1$

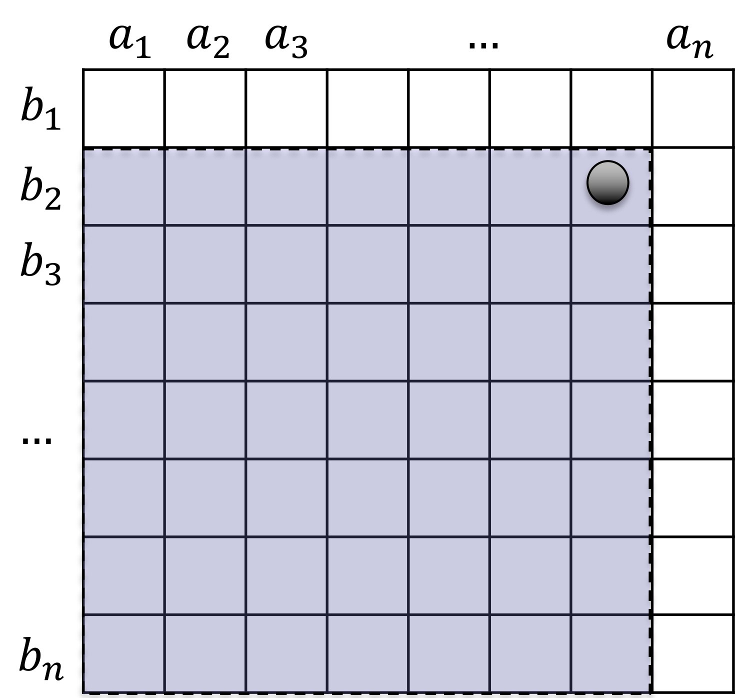
while $i > 0$ and $j \le n$:

if $a_i + b_j = c_k$: return (a_i, b_j, c_k)

if $a_i + b_j > c_k$: $i := i - 1$

if $a_i + b_j < c_k$: $j := j + 1$

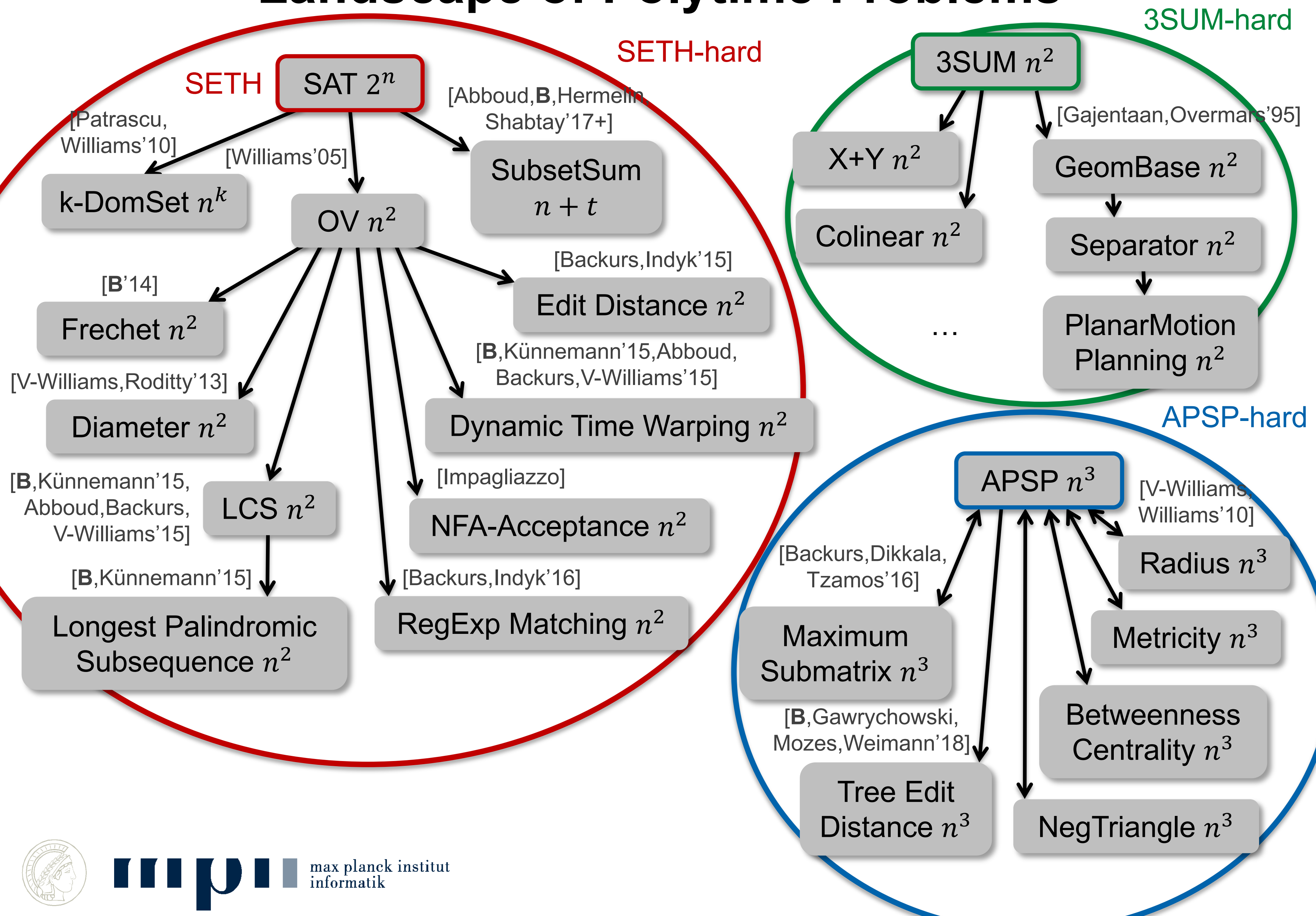
return “no solution”



time $O(n)$ per c_k

time $O(n^2)$ overall

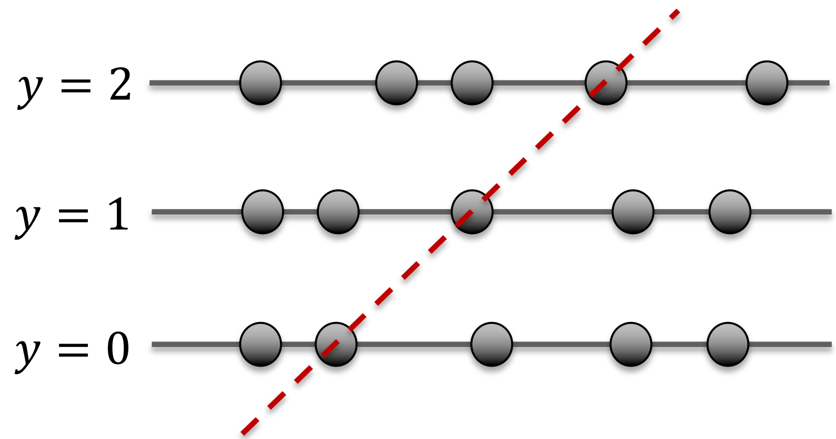
Landscape of Polytime Problems



Example: GeomBase

given a set of n points on three horinzontal lines $y = 0, y = 1, y = 2$, determine whether there exists a non-horizontal line containing three of the points

Thm: GeomBase is 3SUM-hard.



Given an instance (A, B, C) of 3SUM

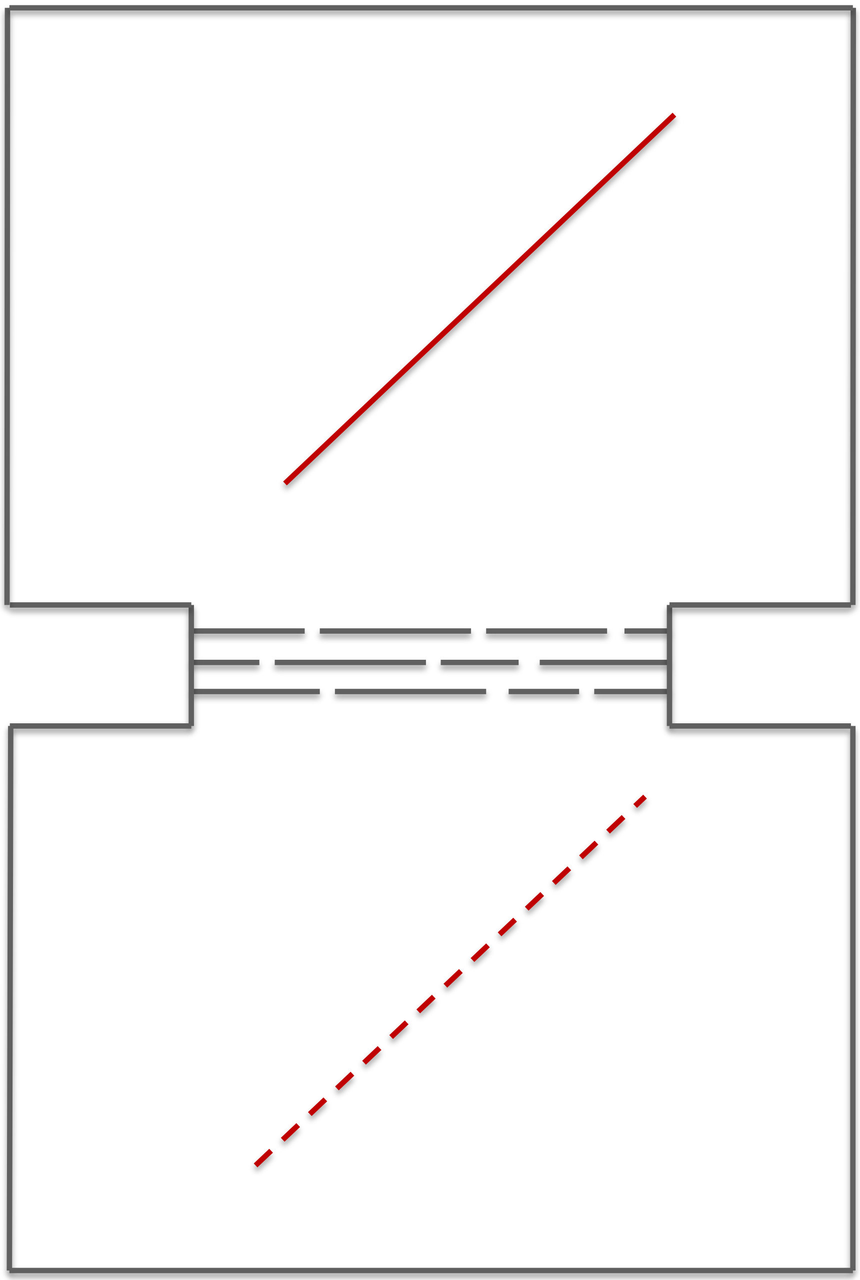
construct points:

- $(a, 0)$ for any $a \in A$
- $(b, 2)$ for any $b \in B$
- $(c/2, 1)$ for any $c \in C$

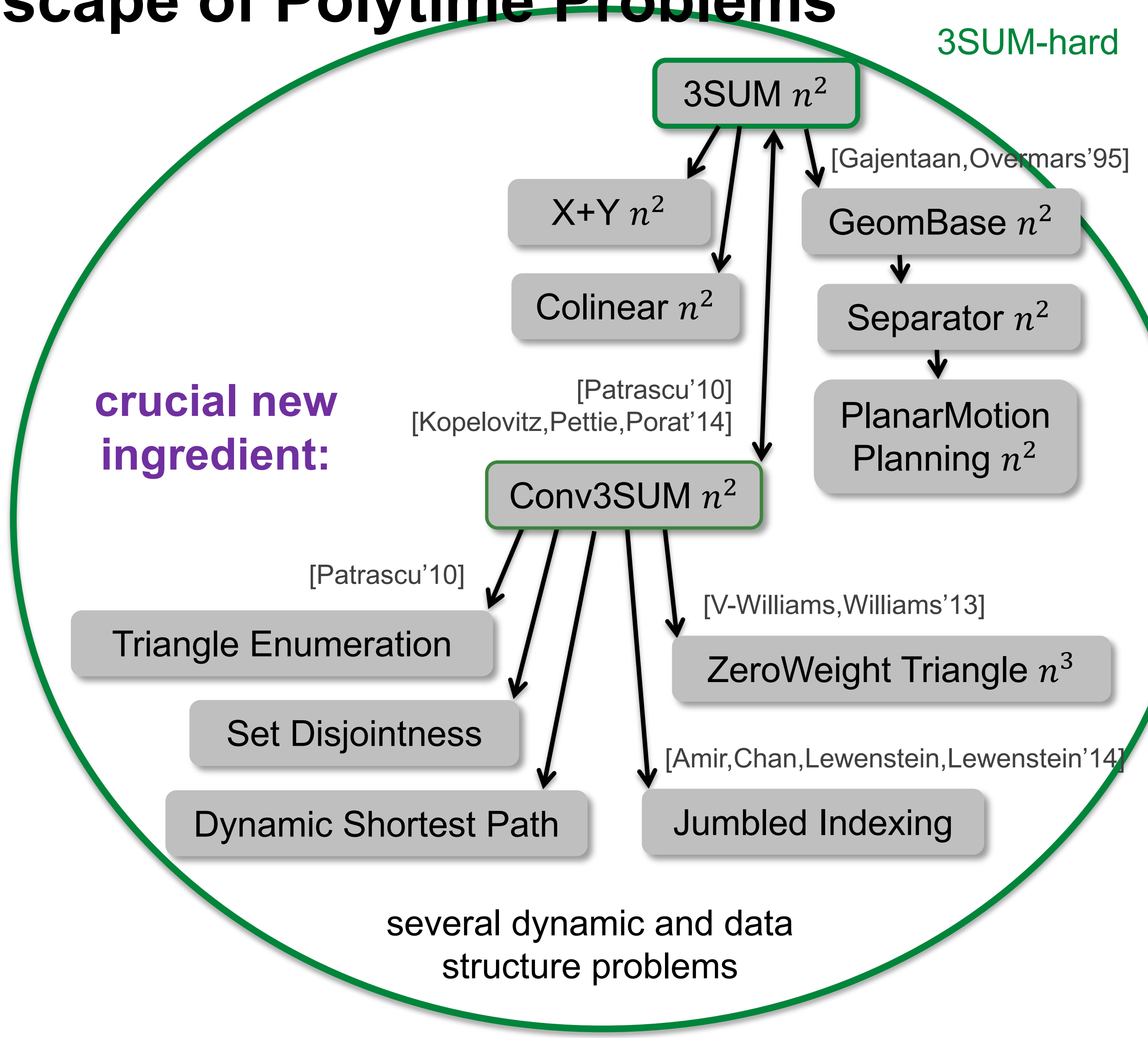
they lie on a line if $c/2 - a = b - c/2 \iff a + b = c$

Example Planar Motion Planning

Thm: PlanarMotionPlanning is 3SUM-hard.



Landscape of Polytime Problems



I. Equivalence of 3SUM and Conv3SUM

II. Subset Sum

III. Further Topics

IV. Conclusion

Equivalence of 3SUM and Conv3SUM

3SUM: given integers $a_0, \dots, a_{n-1}, b_0, \dots, b_{n-1}, c_0, \dots, c_{n-1}$
are there i, j, k such that $a_i + b_j = c_k$?

Conv3SUM: given integers $a_0, \dots, a_{n-1}, b_0, \dots, b_{n-1}, c_0, \dots, c_{n-1}$
are there i, j, k with $i + j = k$ such that $a_i + b_j = c_k$?

Thm: [Patrascu'10,Kopelovitz,Pettie,Porat'14]

1) If 3SUM is in time $T(n)$ then Conv3SUM is in time $O(T(n))$

2) If Conv3SUM is in time $T(n)$ then 3SUM is in randomized time $O(T(n))$, with one-sided error probability $\leq 1/2$

(Standard boosting yields any constant error probability $\delta > 0$)

max planck institut informatik

All hypotheses are also for randomized algorithms!

From Conv3SUM to 3SUM

Thm: 1) If 3SUM is in time $T(n)$
then Conv3SUM is in time $O(T(n))$

3SUM: $\exists i,j,k$: $a_i + b_j = c_k$?

Conv-3SUM: $\exists i+j=k$: $a_i + b_j = c_k$?

Given input $a_0, \dots, a_{n-1}, b_0, \dots, b_{n-1}, c_0, \dots, c_{n-1}$ for Conv3SUM, construct:

$a'_i := a_i \cdot 3n + i$ $\qquad$ $b'_j := b_j \cdot 3n + j$ $\qquad$ $c'_k := c_k \cdot 3n + k$

This is a YES-instance for 3SUM iff:

$$\exists i,j,k\colon\ a'_i + b'_j - c'_k = 0$$

$$\Leftrightarrow\ \exists i,j,k\colon\ 3n \cdot (a_i + b_j - c_k) + \underbrace{(i + j - k)}_{\text{divisible by } 3n \text{ iff } i+j=k} = 0$$

$$\Leftrightarrow\ \exists i,j,k\colon\ i+j=k \text{ and } a_i + b_j = c_k$$

„3SUM can simulate multiple linear equations“

Equivalent Variants of Conv3SUM

3SUM: $\exists i, j, k$: $a_i + b_j = c_k$?

Conv-3SUM: $\exists i + j = k$: $a_i + b_j = c_k$?

Given integers $a_0, \dots, a_{n-1}, b_0, \dots, b_{n-1}, c_0, \dots, c_{n-1}$
are there i, j, k with $i + j = k$ such that $a_i + b_j = c_k$?

$$a'_0, \dots, a'_{2n-1} = a_0, \dots, a_{n-1}, \infty, \dots, \infty$$
$$b'_0, \dots, b'_{2n-1} = b_0, \dots, b_{n-1}, \infty, \dots, \infty$$
$$c'_0, \dots, c'_{2n-1} = c_0, \dots, c_{n-1}, \infty, \dots, \infty$$

Assume a_i, b_j, c_k take values in $\{1, \dots, U\}$

Use $10 \cdot U$ as ∞

Given integers $a_0, \dots, a_{n-1}, b_0, \dots, b_{n-1}, c_0, \dots, c_{n-1}$
are there i, j, k with $k = (i + j) \bmod n$ such that $a_i + b_j = c_k$?

Equivalent Variants of Conv3SUM

3SUM: $\exists i, j, k$: $a_i + b_j = c_k$?

Conv-3SUM: $\exists i + j = k$: $a_i + b_j = c_k$?

Given integers $a_0, \ldots, a_{n-1}, b_0, \ldots, b_{n-1}, c_0, \ldots, c_{n-1}$ are there i, j, k with $i + j = k$ such that $a_i + b_j = c_k$?

$$a'_0, \ldots, a'_{2n-1} = a_0, \ldots, a_{n-1}, \infty, \ldots, \infty$$

$$b'_0, \ldots, b'_{2n-1} = b_0, \ldots, b_{n-1}, \infty, \ldots, \infty$$

$$c'_0, \ldots, c'_{2n-1} = c_0, \ldots, c_{n-1}, c_0, \ldots, c_{n-1}$$

Given integers $a_0, \ldots, a_{n-1}, b_0, \ldots, b_{n-1}, c_0, \ldots, c_{n-1}$ are there i, j, k with $k = (i + j) \bmod n$ such that $a_i + b_j = c_k$?

max planck institut informatik

Equivalent Variants of Conv3SUM

3SUM: $\exists i, j, k$: $a_i + b_j = c_k$?

Conv-3SUM: $\exists i + j = k$: $a_i + b_j = c_k$?

Given integers $a_0, \ldots, a_{n-1}, b_0, \ldots, b_{n-1}, c_0, \ldots, c_{n-1}$
are there i, j, k with $i + j = k$ such that $a_i + b_j = c_k$?

Given integers $a_1, \ldots, a_n$
are there i, j, k with $i + j = k$ such that $a_i + a_j + a_k = 0$?

Standard Version

Given integers $a_0, \ldots, a_{n-1}, b_0, \ldots, b_{n-1}, c_0, \ldots, c_{n-1}$
are there i, j, k with $k = (i + j) \bmod n$ such that $a_i + b_j = c_k$?

From 3SUM to Conv3SUM

Thm: 2) If Conv3SUM is in time $T(n)$ then 3SUM is in randomized time $O(T(n))$

3SUM: $\exists i,j,k$: $a_i + b_j = c_k$?

Conv-3SUM: $\exists i,j,k$: $k = (i+j) \bmod n$, $a_i + b_j = c_k$?

$h: \Omega \to \{0,..,R-1\}$

Linearity:
$h(x+y) = h(x) + h(y) \bmod R$

No overfull buckets:
$|\{x \in \Omega \mid h(x) = r\}| \leq 100n/R$

Given input $a_0, \dots, a_{n-1}, b_0, \dots, b_{n-1}, c_0, \dots, c_{n-1}$ for 3SUM

$A := \{a_0, \dots, a_{n-1}\}, B := \{b_0, \dots, b_{n-1}\}, C := \{c_0, \dots, c_{n-1}\}$,
$\Omega := A \cup B \cup C$

(Can assume that input numbers are distinct)

Assume ***magic*** hash function $h: \Omega \to \{0,..,R-1\}$ s.t.

Linearity: $h(x+y) = \big(h(x) + h(y)\big) \bmod R$ for all $x, y \in \Omega$

No overfull buckets: $|\{x \in \Omega \mid h(x) = r\}| \leq 100n/R$ for all $r \in \{0, \dots, R-1\}$

From 3SUM to Conv3SUM

Thm: 2) If Conv3SUM is in time $T(n)$ then 3SUM is in randomized time $O(T(n))$

3SUM: $\exists i,j,k$: $a_i + b_j = c_k$?

Conv-3SUM: $\exists i,j,k$: $k = (i+j) \bmod n$, $a_i + b_j = c_k$?

$h: \Omega \to \{0,..,R-1\}$

Linearity:
$h(x+y) = h(x) + h(y) \bmod R$

No overfull buckets:
$|\{x \in \Omega \mid h(x) = r\}| \le 100n/R$

Given input A, B, C for 3SUM, compute:

For any $x, y, z \in \{1, \dots, 100n/R\}$:

$a'_i :=$ the x-th element of bucket $\{a \in A \mid h(a) = i\}$

$b'_j :=$ the y-th element of bucket $\{b \in B \mid h(b) = j\}$

$c'_k :=$ the z-th element of bucket $\{c \in C \mid h(c) = k\}$

} or ∞, if there are less elements in the bucket

If $a'_0, \dots, a'_{R-1}, b'_0, \dots, b'_{R-1}, c'_0, \dots, c'_{R-1}$ is a YES-instance of Conv3SUM: return YES

Return NO

Running Time: $O\left((n/R)^3 \cdot T(R)\right)$

Setting $R := n$ we obtain time $O(T(n))$ for 3SUM

From 3SUM to Conv3SUM

Thm: 2) If Conv3SUM is in time $T(n)$ then 3SUM is in randomized time $O(T(n))$

3SUM: $\exists i, j, k$: $a_i + b_j = c_k$?

Conv-3SUM: $\exists i, j, k$: $k = (i + j) \bmod n$, $a_i + b_j = c_k$?

$h: \Omega \to \{0, .., R-1\}$

Linearity: $h(x+y) = h(x) + h(y) \bmod R$

No overfull buckets: $|\{x \in \Omega \mid h(x) = r\}| \leq 100n/R$

Given input A, B, C for 3SUM, compute:

For any $x, y, z \in \{1, \ldots, 100n/R\}$:

$a'_i :=$ the x-th element of bucket $\{a \in A \mid h(a) = i\}$

$b'_j :=$ the y-th element of bucket $\{b \in B \mid h(b) = j\}$

$c'_k :=$ the z-th element of bucket $\{c \in C \mid h(c) = k\}$

(or ∞, if there are less elements in the bucket)

If $a'_0, \ldots, a'_{R-1}, b'_0, \ldots, b'_{R-1}, c'_0, \ldots, c'_{R-1}$ is a YES-instance of Conv3SUM: return YES

Return NO

Correctness I:

Any Conv3SUM-solution $a'_i + b'_j = c'_k$ has $a'_i \in A, b'_j \in B, c'_k \in C$ (not ∞!) and thus yields a solution for 3SUM

Thus, if A, B, C has no solution, then we always return NO

From 3SUM to Conv3SUM

Thm: 2) If Conv3SUM is in time $T(n)$ then 3SUM is in randomized time $O(T(n))$

3SUM: $\exists i,j,k$: $a_i + b_j = c_k$?

Conv-3SUM: $\exists i,j,k$: $k = (i+j) \bmod n$, $a_i + b_j = c_k$?

$h: \Omega \to \{0,..,R-1\}$

Linearity: $h(x+y) = h(x) + h(y) \bmod R$

No overfull buckets: $|\{x \in \Omega \mid h(x) = r\}| \leq 100n/R$

Given input A, B, C for 3SUM, compute:

For any $x, y, z \in \{1, \dots, 100n/R\}$:

$a'_i :=$ the x-th element of bucket $\{a \in A \mid h(a) = i\}$

$b'_j :=$ the y-th element of bucket $\{b \in B \mid h(b) = j\}$

$c'_k :=$ the z-th element of bucket $\{c \in C \mid h(c) = k\}$

(or ∞, if there are less elements in the bucket)

If $a'_0, \dots, a'_{R-1}, b'_0, \dots, b'_{R-1}, c'_0, \dots, c'_{R-1}$ is a YES-instance of Conv3SUM: return YES

Return NO

Correctness II: If A, B, C has a solution $a \in A, b \in B, c \in C$ with $a + b = c$, then

Set $i := h(a), j := h(b), k := h(c)$

For some $x, y, z \in \{1, \dots, 100n/R\}$ we have $a'_i = a, b'_j = b, c'_k = c$ $\implies a'_i + b'_j = c'_k$

And $k = h(c) = h(a+b) = \big(h(a) + h(b)\big) \bmod R = (i+j) \bmod R$

Thus, if A, B, C has a solution, then we always return YES

Now Without Magic

Want: almost-linear random hash function h

$\Omega \subseteq \{1, \dots, U\}$

fix any prime $p > 2U$

pick $m \in \{1, \dots, p-1\}$ uniformly at random

$h(x) := (m \cdot x \bmod p) \bmod R$

3SUM:	$\exists i, j, k$: $a_i + b_j = c_k$?
Conv-3SUM:	$\exists i, j, k$: $k = (i + j) \bmod n$, $a_i + b_j = c_k$?

$h: \Omega \to \{0, .., R-1\}$

Linearity:
$h(x + y) = h(x) + h(y) \bmod R$

No overfull buckets:
$|\{x \in \Omega \mid h(x) = r\}| \leq 100n/R$

This random hash function $h: \{1, \dots, U\} \to \{0, .., R-1\}$ satisfies:

Almost-linearity: there is a set D of offsets, $|D| = O(1)$, s.t.

for all $x, y \in \{1, \dots, U\}$ there exists $d \in D$ s.t.

$$h(x + y) = (h(x) + h(y) + d) \bmod R$$

Unlikely overfull buckets: for any $x \in \Omega$: $\Pr[x \text{ is in overfull bucket}] \leq 1/6$

assuming $R \leq n$

$$\underbrace{x \text{ is in overfull bucket}}_{|\{y \in \Omega \mid h(y) = h(x)\}| > 100n/R}$$

max planck institut
informatik

Now Without Magic

3SUM:	$\exists i,j,k$: $a_i + b_j = c_k$?
Conv-3SUM:	$\exists i,j,k$: $k = (i+j) \bmod n$, $a_i + b_j = c_k$?

$h: \Omega \to \{0,..,R-1\}$

Almost-linearity: $h(x+y) = (h(x)+h(y)+d) \bmod R$ for some $d \in D$, $|D| = O(1)$

Unlikely overfull buckets: $\Pr[x$ in overfull bucket$] \leq 1/6$

Want: almost-linear random hash function h

$\Omega \subseteq \{1, \dots, U\}$

fix any prime $p > 2U$

pick $m \in \{1, \dots, p-1\}$ uniformly at random

$h(x) := (m \cdot x \bmod p) \bmod R$

This random hash function $h: \{1, \dots, U\} \to \{0,..,R-1\}$ satisfies:

Almost-linearity: there is a set D of offsets, $|D| = O(1)$, s.t.

for all $x, y \in \{1, \dots, U\}$ there exists $d \in D$ s.t.

$$h(x+y) = (h(x)+h(y)+d) \bmod R$$

Unlikely overfull buckets: for any $x \in \Omega$: $\Pr[x$ is in overfull bucket$] \leq 1/6$

assuming $R \leq n$

$$|\{y \in \Omega \mid h(y) = h(x)\}| > 100n/R$$

max planck institut informatik

Now Without Magic

Adapted algorithm:

Given input A, B, C for 3SUM, compute:

Pick $m \in \{1, .., p-1\}$ uniformly at random

For any $x, y, z \in \{1, \dots, 100n/R\}$ and any $d \in D$:

$a'_i :=$ the x-th element of bucket $\{a \in A \mid h(a) = i\}$

$b'_j :=$ the y-th element of bucket $\{b \in B \mid h(b) = j\}$

$c'_k :=$ the z-th element of bucket $\{c \in C \mid h(c) = (k+d) \bmod R\}$

If $a'_0, \dots, a'_{R-1}, b'_0, \dots, b'_{R-1}, c'_0, \dots, c'_{R-1}$ is a YES-instance of Conv3SUM: return YES

Return NO

3SUM: $\exists i, j, k$: $a_i + b_j = c_k$?

Conv-3SUM: $\exists i, j, k$: $k = (i + j) \bmod n$, $a_i + b_j = c_k$?

$h: \Omega \to \{0, .., R-1\}$

Almost-linearity: $h(x+y) = (h(x) + h(y) + d) \bmod R$ for some $d \in D$, $|D| = O(1)$

Unlikely overfull buckets: $\Pr[x$ in overfull bucket$] \leq 1/6$

Running Time: $O\big((n/R)^3 \cdot T(R)\big)$

Setting $R := n$ we obtain a randomized algorithm in time $O(T(n))$ for 3SUM

Now Without Magic

3SUM: $\exists i,j,k$: $a_i + b_j = c_k$?

Conv-3SUM: $\exists i,j,k$: $k = (i+j) \bmod n$, $a_i + b_j = c_k$?

$h: \Omega \to \{0,..,R-1\}$

Almost-linearity: $h(x+y) = (h(x)+h(y)+d) \bmod R$ for some $d \in D$, $|D| = O(1)$

Unlikely overfull buckets: $\Pr[x$ in overfull bucket$] \leq 1/6$

Adapted algorithm:

Given input A, B, C for 3SUM, compute:

Pick $m \in \{1,..,p-1\}$ uniformly at random

For any $x, y, z \in \{1, \dots, 100n/R\}$ and any $d \in D$:

$a'_i :=$ the x-th element of bucket $\{a \in A \mid h(a) = i\}$

$b'_j :=$ the y-th element of bucket $\{b \in B \mid h(b) = j\}$

$c'_k :=$ the z-th element of bucket $\{c \in C \mid h(c) = (k+d) \bmod R\}$

If $a'_0, \dots, a'_{R-1}, b'_0, \dots, b'_{R-1}, c'_0, \dots, c'_{R-1}$ is a YES-instance of Conv3SUM: return YES

Return NO

Correctness I:

Any Conv3SUM-solution $a'_i + b'_j = c'_k$ has $a'_i \in A, b'_j \in B, c'_k \in C$ (not ∞!) and thus yields a solution for 3SUM

Thus, if A, B, C has no solution, then we always return NO

Now Without Magic

3SUM: $\exists i,j,k$: $a_i + b_j = c_k$?

Conv-3SUM: $\exists i,j,k$: $k = (i+j) \bmod n$, $a_i + b_j = c_k$?

$h: \Omega \to \{0,..,R-1\}$

Almost-linearity: $h(x+y) = (h(x)+h(y)+d) \bmod R$ for some $d \in D$, $|D| = O(1)$

Unlikely overfull buckets: $\Pr[x$ in overfull bucket$] \leq 1/6$

Adapted algorithm:

Given input A, B, C for 3SUM, compute:

Pick $m \in \{1,..,p-1\}$ uniformly at random

For any $x, y, z \in \{1, \dots, 100n/R\}$ and any $d \in D$:

$a'_i :=$ the x-th element of bucket $\{a \in A \mid h(a) = i\}$

$b'_j :=$ the y-th element of bucket $\{b \in B \mid h(b) = j\}$

$c'_k :=$ the z-th element of bucket $\{c \in C \mid h(c) = (k+d) \bmod R\}$

If $a'_0, \dots, a'_{R-1}, b'_0, \dots, b'_{R-1}, c'_0, \dots, c'_{R-1}$ is a YES-instance of Conv3SUM: return YES

Return NO

Correctness II: If A, B, C has a solution $a \in A, b \in B, c \in C$ with $a + b = c$, then:

Error event $\mathcal{E}$: a, b or c are in an overfull bucket

By union bound, $\Pr[\mathcal{E}] \leq 3 \cdot 1/6 = 1/2$ $\quad\Longrightarrow \Pr[\overline{\mathcal{E}}] \geq 1/2$

Assume $\overline{\mathcal{E}}$ from now on

Now Without Magic

Adapted algorithm:

Given input A, B, C for 3SUM, compute:

Pick $m \in \{1, .., p-1\}$ uniformly at random

For any $x, y, z \in \{1, \dots, 100n/R\}$ and any $d \in D$:

$a'_i :=$ the x-th element of bucket $\{a \in A \mid h(a) = i\}$

$b'_j :=$ the y-th element of bucket $\{b \in B \mid h(b) = j\}$

$c'_k :=$ the z-th element of bucket $\{c \in C \mid h(c) = (k+d) \bmod R\}$

If $a'_0, \dots, a'_{R-1}, b'_0, \dots, b'_{R-1}, c'_0, \dots, c'_{R-1}$ is a YES-instance of Conv3SUM: return YES

Return NO

3SUM: $\exists i, j, k$: $a_i + b_j = c_k$?

Conv-3SUM: $\exists i, j, k$: $k = (i + j) \bmod n$, $a_i + b_j = c_k$?

$h: \Omega \to \{0, .., R-1\}$

Almost-linearity: $h(x+y) = (h(x) + h(y) + d) \bmod R$ for some $d \in D$, $|D| = O(1)$

Unlikely overfull buckets: $\Pr[x \text{ in overfull bucket}] \le 1/6$

Correctness II: If A, B, C has a solution $a \in A, b \in B, c \in C$ with $a + b = c$, then:

Let $d \in D$ s.t. $h(a+b) = (h(a) + h(b) + d) \bmod R$

Set $i := h(a), j := h(b)$, $k \in \{0, \dots, R-1\}$ s.t. $h(c) = (k+d) \bmod R$

For some $x, y, z \in \{1, \dots, 100n/R\}$ we have $a'_i = a, b'_j = b, c'_k = c \implies a'_i + b'_j = c'_k$

And $(k+d) \bmod R = h(c) = h(a+b) = (h(a) + h(b) + d) \bmod R = (i + j + d) \bmod R$

Thus $k = k \bmod R = (i+j) \bmod R$ → We return YES with probability $\ge 1/2$

Equivalence of 3SUM and Conv3SUM

3SUM: given integers $a_0, \dots, a_{n-1}, b_0, \dots, b_{n-1}, c_0, \dots, c_{n-1}$

are there i, j, k such that $a_i + b_j = c_k$?

Conv3SUM: given integers $a_0, \dots, a_{n-1}, b_0, \dots, b_{n-1}, c_0, \dots, c_{n-1}$

are there i, j, k with $i + j = k$ such that $a_i + b_j = c_k$?

Thm: [Patrascu'10,Kopelovitz,Pettie,Porat'14]

1) If 3SUM is in time $T(n)$ then Conv3SUM is in time $O(T(n))$

2) If Conv3SUM is in time $T(n)$ then 3SUM is in randomized time $O(T(n))$, with one-sided error probability $\leq 1/2$

(Standard boosting yields any constant error probability $\delta > 0$)

Hashing Analysis

fix prime $p > 2U$

pick $m \in \{1, \dots, p-1\}$ u.a.r.

$h(x) = (m \cdot x \bmod p) \bmod R$

$\Omega \subseteq \{1, \dots, U\}$

fix any prime $p > 2U$

pick $m \in \{1, \dots, p-1\}$ uniformly at random

$h(x) := (m \cdot x \bmod p) \bmod R$

This random hash function $h: \{1, \dots, U\} \to \{0, .., R-1\}$ satisfies:

Almost-linearity: there is a set D of offsets, $|D| = O(1)$, s.t.

for all $x, y \in \{1, \dots, U\}$ there exists $d \in D$ s.t.

$$h(x+y) = (h(x) + h(y) + d) \bmod R$$

Unlikely overfull buckets: for any $x \in \Omega$: $\Pr[x \text{ is in overfull bucket}] \leq 1/6$

assuming $R \leq n$

$$\underbrace{x \text{ is in overfull bucket}}_{|\{y \in \Omega \mid h(y) = h(x)\}| > 100n/R}$$

Almost-Linearity

there is a set D of offsets, $|D| = O(1)$, s.t.

for all $x, y \in \{1, \dots, U\}$ there exists $d \in D$ s.t.

$$h(x+y) = (h(x) + h(y) + d) \bmod R$$

fix prime $p > 2U$

pick $m \in \{1, \dots, p-1\}$ u.a.r.

$h(x) = (m \cdot x \bmod p) \bmod R$

$$D := \{0, -p\}$$

Proof:

$$h(x+y) = \big((m \cdot x + m \cdot y) \bmod p\big) \bmod R$$

$$= \Big(\big(\underbrace{(m \cdot x \bmod p) + (m \cdot y \bmod p)}_{\in \{0, \dots, 2(p-1)\}}\big) \bmod p\Big) \bmod R$$

$$= \big((m \cdot x \bmod p) + (m \cdot y \bmod p) + d\big) \bmod R$$

for some $d \in D := \{0, -p\}$

$$= (h(x) + h(y) + d) \bmod R$$

max planck institut informatik

Near-Universality

fix prime $p > 2U$

pick $m \in \{1, \dots, p-1\}$ u.a.r.

$h(x) = (m \cdot x \bmod p) \bmod R$

for any $x \neq y,\ x, y \in \{-U, \dots, U\}$:

$\Pr[h(x) = h(y)] \leq 4/R$

Proof: $h(x) = h(y) \iff (m \cdot x \bmod p) \bmod R = (m \cdot y \bmod p) \bmod R$

$\iff \underbrace{(m \cdot x \bmod p) - (m \cdot y \bmod p)}_{\in \{-(p-1), \dots, p-1\}} = i \cdot R$ for some $i \in \mathbb{Z}$

$\implies -p/R < i < p/R$

take $\bmod p$: $\implies \big((m \cdot x \bmod p) - (m \cdot y \bmod p)\big) \bmod p = i \cdot R \bmod p$

$\iff \big(m \cdot (x - y)\big) \bmod p = i \cdot R \bmod p$

Since $|x|, |y| \leq U < p/2$: $|x - y| < p$

Since $x \neq y$: $(x - y) \bmod p \neq 0$

Since p prime: there is inverse $(x - y)^{-1}$

max planck institut
informatik

Near-Universality

for any $x \neq y,\ x, y \in \{-U, \dots, U\}$:

$\Pr[h(x) = h(y)] \leq 4/R$

fix prime $p > 2U$

pick $m \in \{1, \dots, p-1\}$ u.a.r.

$h(x) = (m \cdot x \bmod p) \bmod R$

Proof: $h(x) = h(y) \iff (m \cdot x \bmod p) \bmod R = (m \cdot y \bmod p) \bmod R$

$\iff \underbrace{(m \cdot x \bmod p) - (m \cdot y \bmod p)}_{\in \{-(p-1), \dots, p-1\}} = i \cdot R$ for some $i \in \mathbb{Z}$

$\implies -p/R < i < p/R$

take $\bmod p$: $\implies \big((m \cdot x \bmod p) - (m \cdot y \bmod p)\big) \bmod p = i \cdot R \bmod p$

$\iff \big(m \cdot (x - y)\big) \bmod p = i \cdot R \bmod p$

multiply by $(x-y)^{-1}$: $\implies m \bmod p = i \cdot R \cdot (x-y)^{-1} \bmod p$

$= m$

So m is among the values $M = \{i \cdot R \cdot (x-y)^{-1} \bmod p \mid -p/R < i < p/R\}$ $i \neq 0$

Thus $\Pr[h(x) = h(y)] \leq \Pr[m \in M] = \frac{|M|}{p-1} \leq \frac{2p/R}{p-1} \leq 4/R$

max planck institut
informatik

Unlikely Overfull Buckets

for any $x \in \Omega$: $\Pr[\underbrace{x \text{ is in overfull bucket}}_{|\{y \in \Omega \mid h(y) = h(x)\}| > 100n/R}] \le 1/6$

fix prime $p > 2U$

pick $m \in \{1, \dots, p-1\}$ u.a.r.

$h(x) = (m \cdot x \bmod p) \bmod R$

assuming $R \le n$

Proof: Write $S(x) := |\{y \in \Omega \mid h(y) = h(x)\}|$

$$\mathbb{E}[S(x)] = \sum_{y \in \Omega} \Pr[h(x) = h(y)] = 1 + \sum_{y \in \Omega \setminus \{x\}} \Pr[h(x) = h(y)]$$

(by linearity of expectation)

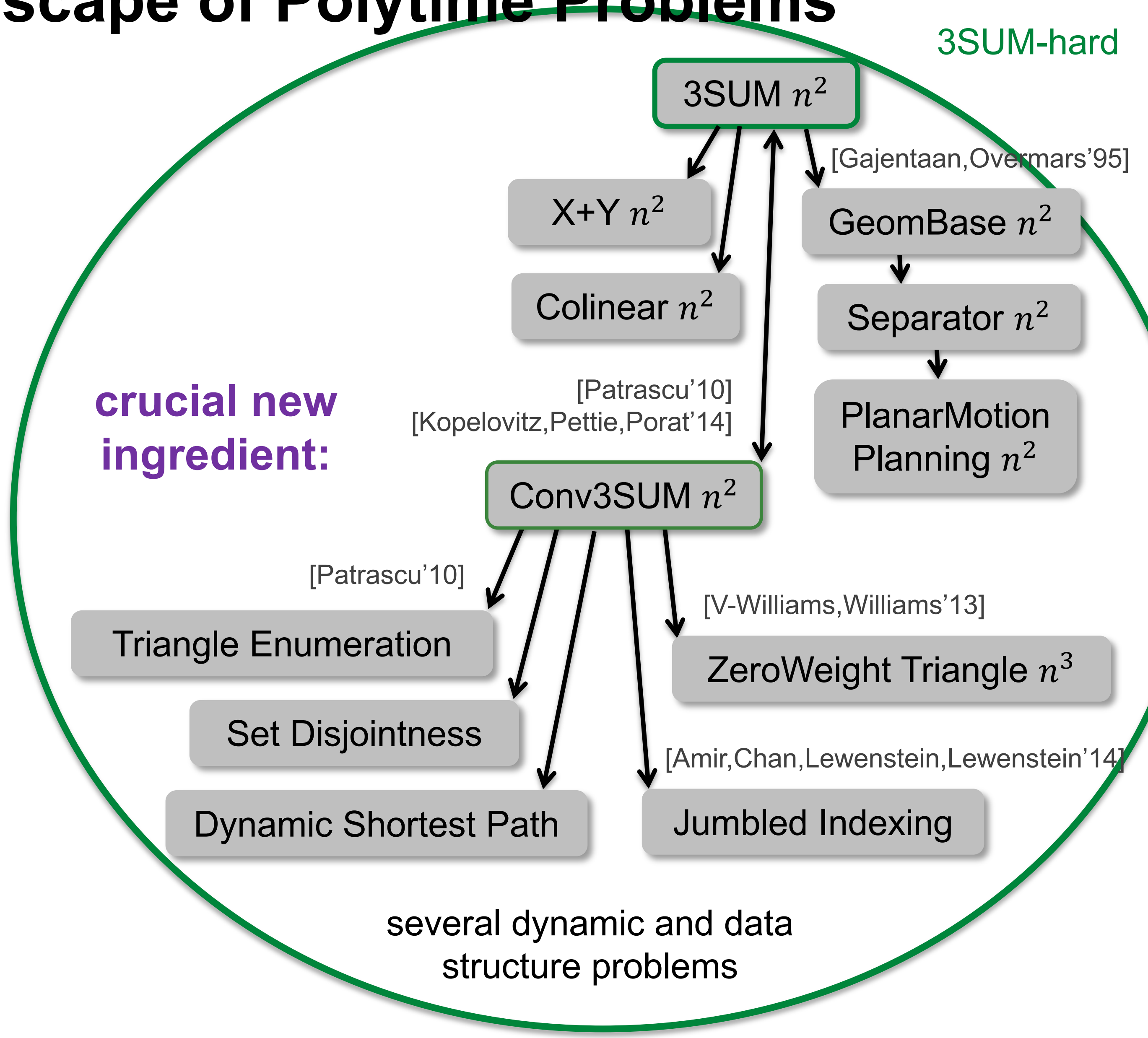
$$\le 1 + \frac{4}{R} \cdot |\Omega| \qquad \text{(by near-universality)}$$

$$\le 1 + \frac{4}{R} \cdot 3n \le 13n/R \qquad \text{(by } R \le n\text{)}$$

Markov's inequality: $\Pr[S(x) > t] \le \frac{\mathbb{E}[S(x)]}{t} \le \frac{13n/R}{t}$

In particular: $\Pr\left[S(x) > \frac{100n}{R}\right] \le \frac{13n/R}{100n/R} \le \frac{1}{6}$

Landscape of Polytime Problems



I. Equivalence of 3SUM and Conv3SUM

II. Subset Sum

III. Further Topics

IV. Conclusion

Subset Sum

Given a set X of n positive integers and a target t,

is there a subset Y of X summing to exactly t?

note: $n \le t$

many applications, connections to other problems, educational value...

pseudopolynomial time algorithm by dynamic programming: [Bellman'57]

$$T[i,s] := T[i-1,s] \vee T[i-1,s-x_i] \qquad X = \{x_1, \dots, x_n\}$$

time $O(nt)$, space $O(t)$

Attempts to break $O(nt)$

Is time $O(nt)$ optimal? Is there an $\tilde{O}(t)$ algorithm?

use basic Word RAM parallelism, word size w: $O(nt/w)$ [Pisinger'03]

consider $s := \max X$; we can assume $s \leq t$: $O(ns)$ [Pisinger'99]

recent breakthrough: $\tilde{O}(\sqrt{n} \cdot t)$ [Koiliaris,Xu Arxiv'15/SODA'17]

all previous algorithms are deterministic

Thm: [B. SODA'17]

Subset Sum is in randomized time $\tilde{O}(t)$.

one-sided error probability $1/n$, time $O(t \log t \log^5 n)$

$\tilde{O}(t)$-Algorithm - Preliminaries

A, B sets of non-negative integers

sumset: $A \oplus B := \{\, a + b \mid a \in A \cup \{0\},\ b \in B \cup \{0\} \,\}$

***t*-capped sumset:** $A \oplus_t B := (A \oplus B) \cap \{0, \dots, t\}$

Fact: $A \oplus_t B$ can be computed in time $O(t \log t)$

how to use „$\oplus_t$": $X \oplus_t X$ contains forbidden sums $x + x$ ☹

however, for a **partitioning** $X = X_1 \cup X_2$:

$X_1 \oplus_t X_2$ contains only valid subset sums of X

New goal: compute all valid subset sums: $\{\, \Sigma(Y) \mid Y \subseteq X \,\} \cap \{0, \dots, t\}$

where $\Sigma(Y) := \sum_{y \in Y} y$

$\tilde{O}(t)$-Algorithm - Step 1: Color Coding

we use color-coding to detect sums of **small** subsets:

ColorCoding(X, t, k):

for $r = 1, \ldots, O(\log n)$:

consider a **random** partitioning $X = X_1 \cup \cdots \cup X_{k^2}$

compute $S_r := X_1 \oplus_t \ldots \oplus_t X_{k^2}$

return $\bigcup_r S_r$

for a solution Y, we say that the partitioning *splits* Y if $|Y \cap X_i| \leq 1$ for all i

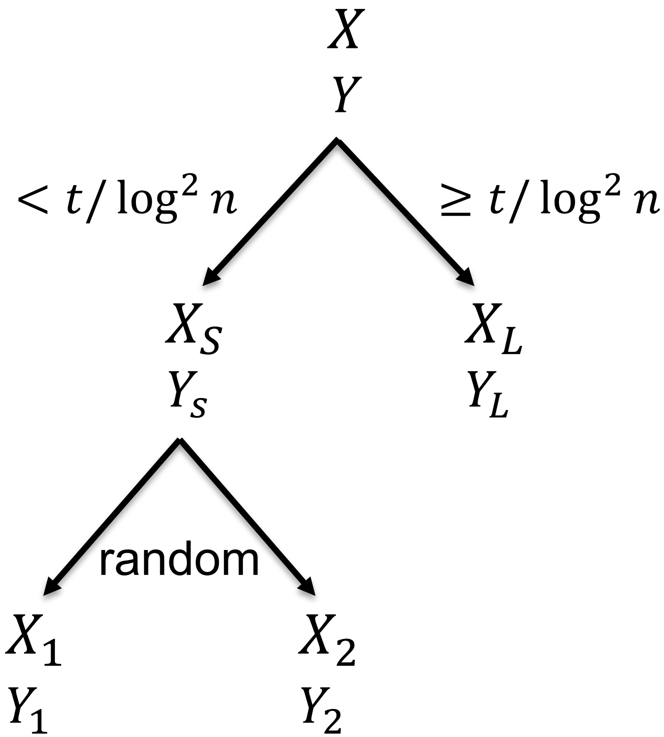
if the partitioning splits $\boldsymbol{Y}$ then $\boldsymbol{S_r}$ contains $\boldsymbol{\Sigma(Y)}$

since we can choose the element in $Y \cap X_i$ (or 0) in each X_i to obtain $\Sigma(Y)$

$\Pr$[random partitioning splits Y] $\geq 1/e$ by birthday paradox

correctness w.h.p., running time $\tilde{O}(t \cdot k^2)$

$\tilde{O}(t)$-Algorithm – Step 2: Recursion



fix solution Y of instance (X, t)

S_L = ColorCoding$(X_L, t, \log^2 n)$ contains Y_L w.h.p.

$\Sigma(Y_i) \leq (1+\varepsilon)t/2$ for $\varepsilon = O(1/\log n)$ w.h.p.

since either $\Sigma(Y_S) \leq t/2$ or $|Y_S| = \Omega(\log^2 n)$

$|X_i| \leq (1+\varepsilon)n/2$ w.h.p.

recursively solve $(X_1, (1+\varepsilon)t/2) \to S_1$

recursively solve $(X_2, (1+\varepsilon)t/2) \to S_2$

return $S_1 \oplus_t S_2 \oplus_t S_L$

time $T(n,t) \leq \tilde{O}(t) + 2T((1+\varepsilon)n/2, (1+\varepsilon)t/2)$

$$\approx \tilde{O}(t) + 2T(n/2, t/2) \quad = \tilde{O}(t)$$

Conditional Lower Bounds

Thm: [B. SODA'17]

Subset Sum is in randomized time $\tilde{O}(t)$.

Is time $\tilde{O}(t)$ optimal? Can we prove a conditional lower bound?

Thm: [Abboud,B.,Hermelin,Shabtay'17+]

Subset Sum is not in time $t^{1-\varepsilon}2^{o(n)}$ unless SETH fails.

Strong Exponential Time Hypothesis:

$\forall\varepsilon > 0$: $\exists k$: k-SAT is not in time $O(2^{(1-\varepsilon)n})$

max planck institut
informatik

Conditional Lower Bounds

Thm: [B. SODA'17]

Subset Sum is in randomized time $\tilde{O}(t)$.

Is time $\tilde{O}(t)$ optimal? Can we prove a conditional lower bound?

Thm: [Abboud,B.,Hermelin,Shabtay'17+]

Subset Sum is not in time $t^{1-\varepsilon}2^{o(n)}$ unless SETH fails.

---

k-Sum problem: Given set A, are there $a_1, \dots, a_k \in A$ with $a_1 + \dots + a_k = 0$?

Recall: k-Sum is in time $O(n + t \operatorname{polylog} t)$ and in time $O\left(n^{\lceil k/2 \rceil} \log n\right)$ (for const. k)

Cor: [Abboud,B.,Hermelin,Shabtay'17+]

k-Sum is not in time $t^{1-\varepsilon}n^{o(k)}$ unless SETH fails.

Ingredient: k-sum-free Sets

$S \subseteq \{1,\ldots,U\}$ is called k-sum-free if $\forall \ell \leq k$: $\forall x_1,\ldots,x_\ell, x \in S$:

$$x_1 + \cdots + x_\ell = \ell \cdot x \quad \Longrightarrow \quad x_1 = \cdots = x_\ell = x$$

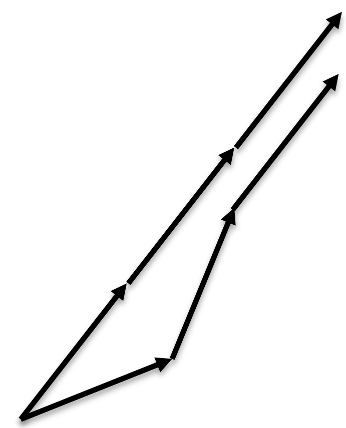
Consistency constraint

Thm: For any k, n and $\varepsilon > 0$ there exists a k-sum-free set S of size n over universe $U = n^{1+\varepsilon} k^{O(1/\varepsilon)}$ [Behrend'46]

Proof: $R := \{y \in [b]^r \mid \|y\| = z\}$ is k-sum-free

since $\|\ell \cdot y\| = \ell \cdot z$

but $\|y_1 + \cdots + y_\ell\| < \ell \cdot z$ if $y_i \neq y_j$ for some i,j



embed R into the integers:

$S := \left\{\sum_{i=1}^{r} y[i] \cdot (kb)^{i-1} \,\middle|\, y \in R\right\}$ is k-sum-free, since there is no overflow

$0 \ldots 0\ y[r]$	$\ldots$	$0 \ldots 0\ y[2]$	$0 \ldots 0\ y[1]$

SETH-Hardness of Subset Sum I

k-SAT: n variables, m clauses

time $O(2^{(1-\varepsilon/2)n})$

sparsification lemma [Impagliazzo,Paturi,Zane'01]

$\Uparrow$

k-SAT: n variables, $O_{k,\varepsilon}(n)$ clauses,
each variable appears in $O_{k,\varepsilon}(1)$ clauses

time $O(2^{(1-\varepsilon)n})$

block a variables to a supervariables

block $O_{k,\varepsilon}(a)$ clauses to a superconstraint

$\Uparrow$

Constraint Satisfaction Problem:

$n' = n/a$ variables over $[2^a]$, n/a constraints

each variable appears in $\leq d$ clauses

each constraint touches $\leq d$ variables

$d = O_{k,\varepsilon}(\text{poly}(a))$

time $O(2^{(1-\varepsilon)n'a})$

SETH-Hardness of Subset Sum II

Subset Sum instance:

highest bits … lowest bits

	$2\log n$ bits		$3n/a$ bits		
target t:	$2n/a$	$0 \dots 0$	$1 \dots 1$	$1 \dots 1$	$0 \dots 0$
item (x, α):	1	$0 \dots 0$	0010	$0 \dots 0$	$0 \dots 0$
item $(C, \alpha_1, \dots, \alpha_s)$:	1	$0 \dots 0$	$0 \dots 0$	0100	$0 \dots 0$

for any variable x, assignment $\alpha \in [2^a]$

(item (x, α): 0010 at position corresponding to x)

for any constraint $C = C(x_1, \dots, x_s)$, satisfying assignment $\alpha_1, \dots, \alpha_s \in [2^a]$

(item $(C, \alpha_1, \dots, \alpha_s)$: 0100 at position corresponding to C)

$\Downarrow$ choose exactly $2n/a$ items $\Longrightarrow$ $\Downarrow$ choose exactly one item for each variable and each clause

max planck institut informatik

SETH-Hardness of Subset Sum II

Subset Sum instance:

highest bits … lowest bits

	$2\log n$ bits		$3n/a$ bits			$n/a \cdot \log(2\,dU+1)$ bits		
target t:	$2n/a$	$0 \dots 0$	$1 \dots 1$	$1 \dots 1$	$0 \dots 0$	…	dU	…
item (x, α):	1	$0 \dots 0$	0010	$0 \dots 0$	$0 \dots 0$	$0 \dots 0$	$dU - d_x S_\alpha$	$0 \dots 0$
item $(C, \alpha_1, \dots, \alpha_s)$:	1	$0 \dots 0$	$0 \dots 0$	0100	$0 \dots 0$	…	S_{α_i}	…

for any variable x, assignment $\alpha \in [2^a]$

d_x = #constraints touching x

for any constraint $C = C(x_1, \dots, x_s)$, satisfying assignment $\alpha_1, \dots, \alpha_s \in [2^a]$

(S_{α_i}) in x_i-block, 0 otherwise

construct d-sum-free set $S \subseteq [U]$ of size 2^a with $U = 2^{(1+\varepsilon)a} d^{O(1/\varepsilon)}$

write $S = \{S_1, \dots, S_{2^a}\}$, construction time $T(a, k, \varepsilon) = O(1)$

SETH-Hardness of Subset Sum II

Subset Sum instance:

	highest bits					lowest bits		
	$2\log n$ bits		$3n/a$ bits			$n/a \cdot \log(2\,dU+1)$ bits		
target t:	$2n/a$	$0\ldots0$	$1\ldots1$	$1\ldots1$	$0\ldots0$	…	dU	…
item (x,α):	1	$0\ldots0$	0010	$0\ldots0$	$0\ldots0$	$0\ldots0$	$dU - d_x S_\alpha$	$0\ldots0$
item $(C,\alpha_1,\ldots,\alpha_s)$:	1	$0\ldots0$	$0\ldots0$	0100	$0\ldots0$	…	S_{α_i}	…

for any variable x, assignment $\alpha \in [2^a]$

d_x = #constraints touching x

for any constraint $C = C(x_1,\ldots,x_s)$, satisfying assignment $\alpha_1,\ldots,\alpha_s \in [2^a]$

S_{α_i}: in x_i-block, 0 otherwise

- block sum $= dU \Leftrightarrow \sum S_{\alpha_i} = d_x S_\alpha \Leftrightarrow$ all assignments for x are consistent
- no overflow between blocks since $d_x \le d$ and $S_\alpha \le U$
- *consistent choice of assignments satisfing all constraints*

SETH-Hardness of Subset Sum II

Subset Sum instance:

highest bits ... lowest bits

	$2\log n$ bits		$3n/a$ bits			$n/a \cdot \log(2\,dU+1)$ bits		
target t:	$2n/a$	$0 \dots 0$	$1 \dots 1$	$1 \dots 1$	$0 \dots 0$	...	dU	...
item (x, α):	1	$0 \dots 0$	0010	$0 \dots 0$	$0 \dots 0$	$0 \dots 0$	$dU - d_x S_\alpha$	$0 \dots 0$
item $(C, \alpha_1, \dots, \alpha_s)$:	1	$0 \dots 0$	$0 \dots 0$	0100	$0 \dots 0$	...	S_{α_i}	...

For item (x, α): for any variable x, assignment $\alpha \in [2^a]$

For item $(C, \alpha_1, \dots, \alpha_s)$: for any constraint $C = C(x_1, \dots, x_s)$, satisfying assignment $\alpha_1, \dots, \alpha_s \in [2^a]$

recall:

$$d = O_{k,\varepsilon}(\text{poly}(a))$$

$$U = 2^{(1+\varepsilon)a} d^{O(1/\varepsilon)}$$

$$\#\text{bits} = n/a \cdot \log\big(O(dU)\big)$$
$$= (1+\varepsilon)n + O_{k,\varepsilon}(n \log(a)/a)$$
$$\le (1+2\varepsilon)n$$

for sufficiently large $a = a(k, \varepsilon)$

$$\#\text{items} = O_{k,\varepsilon}(n)$$

$t^{1-4\varepsilon} 2^{o(n)}$ algorithm for Subset Sum would break SETH

Conditional Lower Bounds

Thm: [B. SODA'17]

Subset Sum is in randomized time $\tilde{O}(t)$.

Is time $\tilde{O}(t)$ optimal? Can we prove a conditional lower bound?

Thm: [Abboud,B.,Hermelin,Shabtay'17+]

Subset Sum is not in time $t^{1-\varepsilon}2^{o(n)}$ unless SETH fails.

I. Equivalence of 3SUM and Conv3SUM

II. Subset Sum

III. Further Topics

IV. Conclusion

More Algorithms for 3SUM

trivial: $O(n^3)$

well-known: $O(n^2)$

using Word RAM bit-tricks: $O\left(n^2 \cdot \frac{\log^2 w}{w}\right)$, $O\left(n^2 \cdot \frac{(\log \log n)^2}{\log^2 n}\right)$
(cell size $w = \Omega(\log n)$,
each number fits in a cell)

[Baran,Demaine,Patrascu'05]

no bit-tricks: $O\left(n^2 \cdot \frac{(\log \log n)^{O(1)}}{\log^2 n}\right)$

[Gronlund,Pettie'14]
[Gold,Sharir'15]
[Chan'17]

---

decision tree complexity: $O\left(n^{\frac{3}{2}} \cdot \sqrt{\log n}\right)$ [Gronlund,Pettie'14]

$O(n \cdot \log^2 n)$ [Kane,Lovett,Moran'18]

max planck institut
informatik

Strong 3SUM Hypothesis

using FFT: 3SUM is in time $O(n + U \operatorname{polylog} U)$ over universe $\{1, \ldots, U\}$

3SUM over any universe $\{1, \ldots, n^c\}$ is equivalent to 3SUM over universe $\{1, \ldots, n^3\}$

via hashing, follows from [Baran,Demaine,Patrascu'05]

Strong 3SUM Hypothesis: 3SUM over universe $\{1, \ldots, n^2\}$ is not in time $O(n^{2-\varepsilon})$

max planck institut
informatik

(min,+)-Convolution

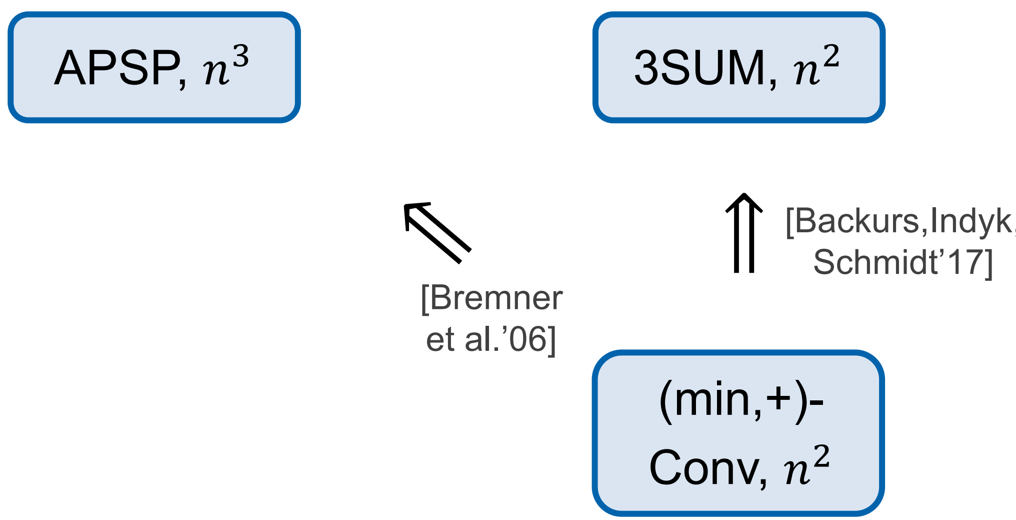
Problem (min,+)-Convolution:

Given integers $a_1, \dots, a_n, b_1, \dots, b_n$, compute $c_0, \dots, c_{n-1}$ with

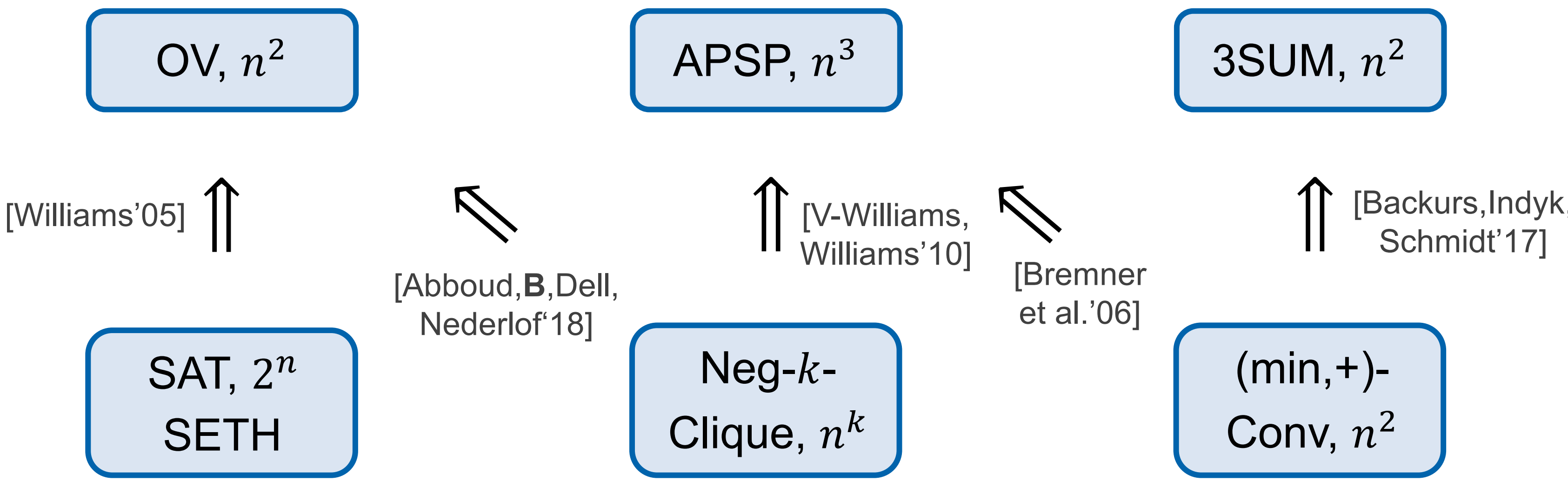
$$c_k := \min_{1 \le i \le k} a_i + b_{k-i}$$

(min,+)-Conv-Hypothesis:

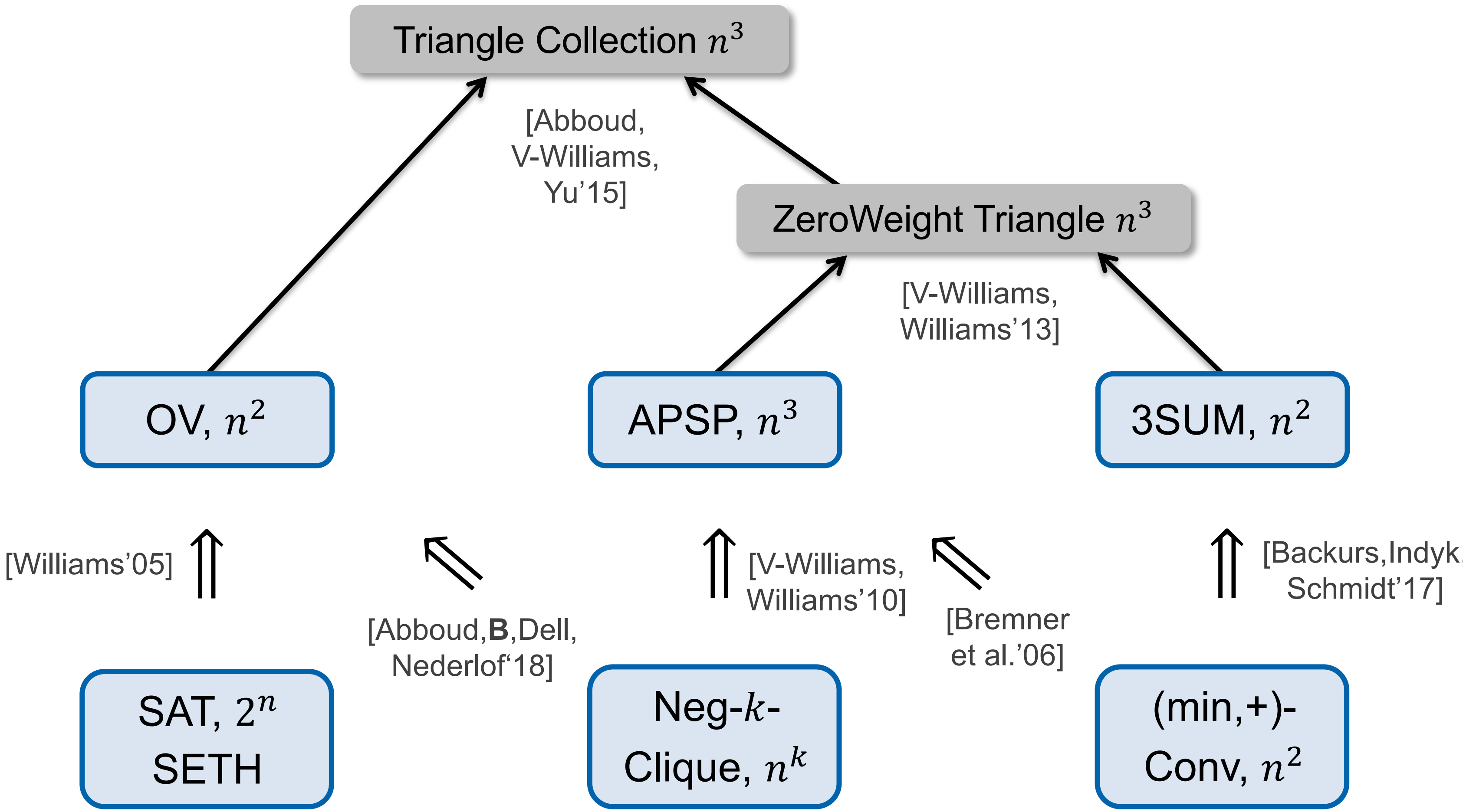
$\forall \varepsilon > 0$: (min,+)-Conv has no $O(n^{2-\varepsilon})$-time algorithm



Current Landscape of Hypotheses



Harder Problems



max planck institut
informatik

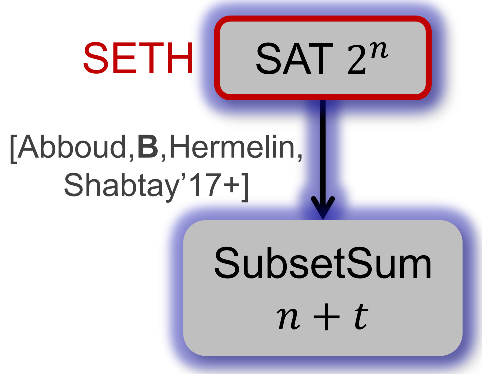
I. Equivalence of 3SUM and Conv3SUM

II. Subset Sum

III. Further Topics

IV. Conclusion

Conv3SUM

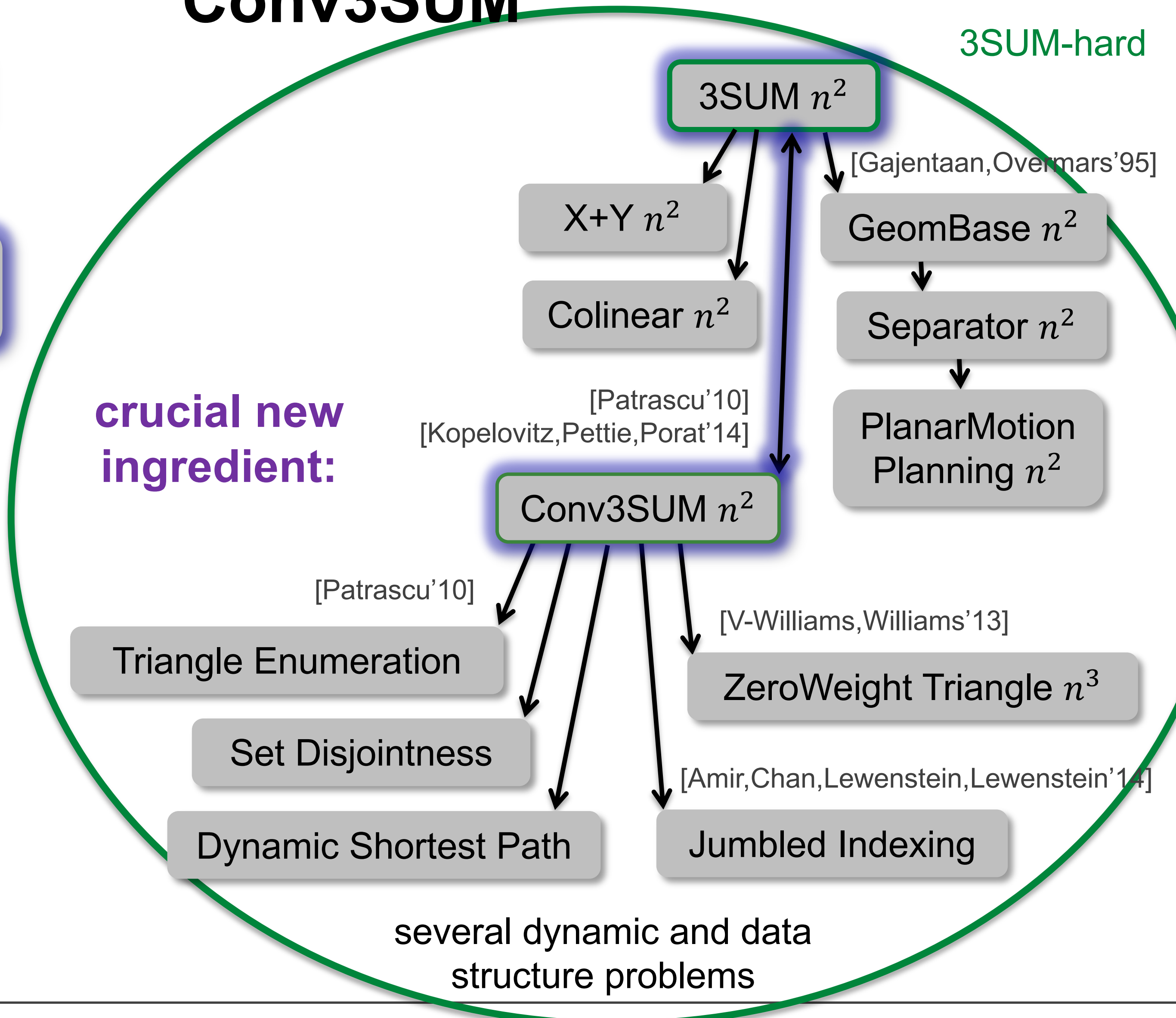


we have seen:

k-sum-free sets

almost-linear hashing

further topics, e.g. (min,+)-Conv



Open: **Knapsack**: improve time $O(nW)$ to $O(n^2 + W)$?

Is **3SUM** over universe $\{1, \dots, n^2\}$ equivalent to 3SUM?