3SUM and Related Problems

Timothy Chan
(UIUC)
The 3SUM Problem

- Given sets $A$, $B$, $C$ of $n$ real numbers, decide $\exists a \in A, b \in B, c \in C$ with
  \[ a + b = c \]

- **Standard Sol’n 1:**
  sort $A + B$, sort $C$, merge

  $\Rightarrow O(n^2 \log n)$ time
The 3SUM Problem

- **Standard Sol’n 2:**
  
  **Preprocessing:** sort $A$ & $B$
  
  **For each** $c \in C$: test if $c \in A + B$ in $O(n)$ time
The 3SUM Problem

• Standard Sol’n 2:

  Preprocessing: sort $A$ & $B$
  For each $c \in C$: test if $c \in A + B$ in $O(n)$ time

$\Rightarrow O(n^2)$ time (better??)
Variants

• monochromatic version: \( A = B = C \)

• integer version

• convolution 3SUM: given sequences \( A, B, C \), decide \( \exists i, k \) s.t. \( A[i] + B[k - i] = C[k] \)

• \( k \)SUM
  \( (O(n^{[k/2]}) \) time for \( k \) odd, \( O(n^{k/2} \log n) \) time for \( k \) even)
3SUM requires $\Omega(n^{2-\varepsilon})$ time.

(or more strongly, 3SUM for integers (in $[n^2]$) requires $\Omega(n^{2-\varepsilon})$ time.)

useful for proving conditional polynomial lower bound for other problems, by reductions from 3SUM.
Exs of 3SUM-Hard Problems (started by Gajentaan–Overmars’93)

- 3COLLINEAR: given $n$ points in 2D, decide $\exists$ 3 collinear points
Exs of 3SUM-Hard Problems (started by Gajentaan–Overmars’93)

- **3COLLINEAR**: given $n$ points in 2D, decide $\exists$ 3 collinear points

- **3CONCURRENT**: given $n$ lines in 2D, decide $\exists$ 3 concurrent lines
Exs of 3SUM-Hard Problems (started by Gajentaan–Overmars’95)

- given $n$ points in 2D, find 3 points defining smallest triangle area
- given $n$ triangles in 2D, decide if union covers $[0, 1]^2$
- given $n$ halfplanes in 2D, find the deepest point
- given $n$ red/blue points in 2D, find smallest # pts to remove s.t. $\exists$ line separating red from blue points
- given $n$ line segment obstacles in 2D and initial/final position of a rod, decide if motion exists for the rod
- given $n$ points in 3D, find the max Tukey depth
- given 3 polygons in 2D, decide if their common intersection is empty
Exs of Integer-3SUM-Hard Problems

- zero-weight triangle in an edge-weighted graph: $O(n^{3-\delta})$ time? [Vassilevska–Williams (FOCS’10)]
- enumerate $t$ triangles in a graph: $O(t^{1/3}m^{1-\delta})$ time? [Pătraşcu (STOC’10); Kopelowitz–Pettie–Porat (SODA’16)]
- dynamic reachability & dynamic subgraph connectivity: $n^{o(1)}$ query & update time? [Pătraşcu (STOC’10)]
- dynamic max matching: similar [Abboud–Vassilevska (FOCS’14); Kopelowitz–Pettie–Porat (SODA’16)]
- local alignment: $O(n^{2-\delta})$ time? [Abboud–Vassilevska–Weimann (ICALP’14)]
- jumbled string indexing over $[\sigma]$: $n$ queries in $O(n^{2-1/(\sigma-1-\delta)})$ time? [Amir–C.–Lewenstein–Lewenstein (ICALP’14)]

(all via Integer Convolution 3SUM [Pătraşcu (STOC’10)]. . . )
Lower Bound Results

- Erickson [SODA’95]: 3SUM needs $\Omega(n^2)$ comparisons for 3-linear decision trees

$(k:SUM$ needs $\Omega(n^{\lceil k/2 \rceil})$ comps for $k$-linear decision trees)

(Ailon–Chazelle [STOC’04]: $k:SUM$ needs $n^{\Omega(k)}$ for $(k + O(1))$-linear decision trees)

(Erickson–Seidel [FOCS’93]: Erickson [SoCG’96]: 3COLLINEAR needs $\Omega(n^2)$ sidedness tests)
The Surprise...

- Grönlund–Pettie [FOCS’14]:

  3SUM can be solved in $\tilde{O}(n^{3/2})$ comparisons for 4-linear decision trees & has an $O(n^2 / \log^{\Omega(1)} n)$ time alg’m...
Subsequent Results

• Decision trees
  Grönlund–Pettie [FOCS’14] \( O(n^{3/2} \sqrt{\log n}) \)
  Gold–Sharir’15 \( O(n^{3/2}) \)
  Kane–Lovett–Moran [STOC’18] \( O(n \log^2 n) \)

• Alg’ms
  Grönlund–Pettie [FOCS’14] \( O^*(n^2/\log^{2/3} n) \) det.
  \( O^*(n^2/\log n) \) rand.
  Freund’15/Gold–Sharir’15 \( O^*(n^2/\log n) \) det.
  C. [SODA’18] \( O^*(n^2/\log^2 n) \) det.
1. Subquadratic Decision-Tree Upper Bounds

2. Slightly Subquadratic Alg’ms

3. Extensions to Other Problems
Fredman’s Trick

\[ a + b \leq a' + b' \]

\[\Updownarrow\]

\[ a - a' \leq b' - b \]

(Fredman [FOCS’75] used this to prove $\widetilde{O}(n^{5/2})$ decision tree upper bound for all-pairs shortest paths & (min, +)-matrix multiplication)

- Problem: Given sequences \(A, B\) of \(n\) real numbers, compute \(C[k] = \min_i (A[i] + B[k - i])\) for all \(k\).

- Preprocessing:
  - divide \(A\) into groups \(A_1, \ldots, A_{n/d}\) of size \(d\)
  - divide \(B\) into groups \(B_1, \ldots, B_{n/d}\) of size \(d\)
  - sort \(\bigcup_i (A_i - A_i) \cup \bigcup_j (B_j - B_j)\)

\[\Rightarrow O(n/d \cdot d^2 \cdot \log n) = \tilde{O}(dn)\] comps

(by Fredman’s trick, comparisons internal to each \(A_i + B_j\) are now free... )
**Warm-Up: (min, +)-Convolution**  

- To compute each $C[k]$:
  
  find min of $O(n/d)$ elements, in $O(n/d)$ comps

\[
\begin{align*}
B_{n/d} & \\
B_1 & \\
A_1 & \\
A_{n/d} &
\end{align*}
\]

\[
i + j = k
\]

- To compute each \(C[k]\):
  
  find min of \(O(n/d)\) elements, in \(O(n/d)\) comps

- total # comps \(\widetilde{O}(dn + n \cdot n/d)\)

- set \(d = \sqrt{n} \Rightarrow \widetilde{O}(n^{3/2})\) comps
**Problem:** Given sequences $A$, $B$ of $n$ real numbers, compute $C[k] = \text{median of } \{A[i] + B[k - i]\}_{i \in [k]}$

• To compute each $C[k]$: find median in union of $O(n/d)$ sorted lists of size $d$ in $O((n/d) \log d)$ comps [Frederickson–Johnson’80s]

• total # comps $\widetilde{O}(dn + n \cdot n/d)$

• set $d = \sqrt{n} \Rightarrow \widetilde{O}(n^{3/2})$ comps
Convolution 3SUM [Grønlund–Pettie’14]

- **Problem:** Given sequences $A, B, C$ of $n$ real numbers, decide $\exists i, k$ with $C[k] = A[i] + B[k - i]$
Convolution 3SUM [Grønlund–Pettie’14]

- To search for each $C[k]$:

  - search in $O(n/d)$ sorted lists of size $d$
  - in $O((n/d) \log d)$ comps (by binary searches)
Convolution 3SUM [Grønlund–Pettie’14]

- To search for each $C[k]$: search in $O(n/d)$ sorted lists of size $d$ in $O((n/d) \log d)$ comps (by binary searches)
- total # comps $\tilde{O}(dn + n \cdot n/d)$
- set $d = \sqrt{n}$ $\Rightarrow$ $\tilde{O}(n^{3/2})$ comps
Finally: 3SUM [Grønlund–Pettie’14]

• first sort $A$ & $B$

• rest is basically the same!
Finally: 3SUM [Grønlund–Pettie’14]

- To search for each $c \in C$:
  - search in $O(n/d)$ sorted lists of size $d^2$
  - in $O((n/d) \log d)$ comps (by binary searches)
Finally: 3SUM [Grønlund–Pettie’14]

• To search for each $c \in C$:
  
  search in $O(n/d)$ sorted lists of size $d^2$
  in $O((n/d) \log d)$ comps (by binary searches)

• total # comps = $\widetilde{O}(dn + n \cdot n/d)$

• set $d = \sqrt{n} \Rightarrow \widetilde{O}(n^{3/2})$ comps
Rest of Talk

1. Subquadratic Decision-Tree Upper Bounds

2. Slightly Subquadratic Alg’ms

3. Extensions to Other Problems
3SUM Alg’m 0

- build decision tree for small input size $m$
- divide $A, B, C$ into groups of size $m$
  $\Rightarrow O(n/m)^2$ subproblems of size $O(m)$
- can solve each subproblem in $\widetilde{O}(m^{3/2})$ time
- total time $\widetilde{O}((n/m)^2 \cdot m^{3/2} + 2\widetilde{O}(m^{3/2}))$
- set $m \approx \log^{2/3} n \Rightarrow O^*(n^2 / \log^{1/3} n)$ time
3SUM Alg’m 1 [C.’18]

- **Key Idea:** think geometrically... in $d$ dimensions!
  (inspired by C. [STOC’07] on $(\min, +)$-matrix multiplication)

- map each group $A_i$ to a point $(A_i[1], \ldots, A_i[d])$

- map each group $B_j$ to $O(d^4)$ hyperplanes
  $$\{(x_1, \ldots, x_d) \in \mathbb{R}^d : x_u + B_j[v] = x_{u'} + B_j[v']\}$$
  over $u, v, u', v' \in [d]$

  (comparisons internal to $A_i + B_j$ are resolved if we know the location of $A_i$’s point w.r.t. $B_j$’s hyperplanes...)
Cutting Lemma
[Clarkson–Shor’89, Chazelle–Friedman’90]

- Given \(N\) hyperplanes in \(\mathbb{R}^d\), can cut \(\mathbb{R}^d\) into \(d^{O(d)} r^d\) cells s.t. each cell intersects \(O(N/r)\) hyperplanes.
3SUM Alg’m 1 [C.’18]

• apply Cutting Lemma to \( N = O(d^4 \cdot n/d) \) hyperplanes
• for each \((i, j)\):
  • **Case 1:** the hyperplanes of \( B_j \) do not intersect \( A_i \)'s cell
    - all other \( A_i \)'s in the same cell have “same” \( A_i + B_j \)
    - can pre-sort one \( A_i + B_j \) per cell per \( B_j \)
    \[ \Rightarrow \text{total time } d^{O(d)} r^d \cdot n \]
  • **Case 2:** some hyperplane of \( B_j \) intersects \( A_i \)'s cell
    - \# such \( B_j \)'s is \( d^{O(1)} n/r \) per \( A_i \)
    - can pre-sort \( A_i + B_j \) for each such \( A_i, B_j \)
    \[ \Rightarrow \text{total time } d^{O(1)} n/r \cdot n \]
3SUM Alg’m 1 [C.’18]

- To search for each $c \in C$:
  - binary-search in $O(n/d)$ sorted lists of size $d^2$
    in $O((n/d) \log d)$ time

- total time $\tilde{O}\left(d^{O(d)} r^d \cdot n + d^{O(1)} n/r \cdot n + n \cdot n/d\right)$

- set $r = d^{\Theta(1)} \Rightarrow \tilde{O}(d^{O(d)} n + n^2/d)$ time

- set $d \approx \log n / \log \log n \Rightarrow O^*(n^2 / \log n)$ time
3SUM Alg’m 2 [C.’18]

- **Key Idea**: Fredman’s trick + bit packing tricks

- **Lemma 1**: can do a batch of $Q$ internal comparisons, in $\tilde{O}(dn + Q/w)$ time on $w$-bit word RAM
  
  (Proof: to compare $A_i[u] + B_j[v]$ with $A_i[u'] + B_j[v']$, compare rank($A_i[u] - A_i[u']$) with rank($B_j[v'] - B_j[v]$)…)

- **Lemma 2**: can do a batch of $Q$ internal selection queries, in $\tilde{O}(dn + dQ/w)$ time on $w$-bit word RAM
  
  (Proof: simulate *parallel* alg’m for $k$-th smallest in $A_i + B_j$ with $O(\text{polylog } d)$ rounds of $O(d)$ comparisons, by Lemma 1…)
3SUM Alg’m 2 [C.’18]

• To search for each $c \in C$:
  
  binary-search in $O(n/d)$ lists of size $d^2$
  
  can simulate all binary searches with $O(\log d)$ calls to
  selection oracle from Lemma 2 with $Q = O(n \cdot n/d)$

• total time $\widetilde{O}(dn + n \cdot n/d + d \cdot n \cdot (n/d)/w)$

• set $d = \sqrt{n} \Rightarrow \widetilde{O}(n^{3/2} + n^2/w)$ time

• set $w \approx \log n \Rightarrow O^*(n^2/\log n)$ time again :-(

Final 3SUM Alg’m [C.’18]

Combine!

⇒ $O^*(\frac{n^2}{\log^2 n})$ time
1. Subquadratic Decision-Tree Upper Bounds

2. Slightly Subquadratic Alg’ms

3. Extensions to Other Problems
(median, +)-Convolution

similarly: $O^*(n^2 / \log^2 n)$ time alg’m [C.’18]
“Algebraic” 3SUM

- **Problem:** Given sets $A$, $B$, $C$ of $n$ numbers, & fixed-degree algebraic function $\varphi$, decide $\exists a \in A, b \in B, c \in C$ with $\varphi(a, b) = c$

- Fredman’s trick does not immediately extend!

- Barba–Cardinal–Iacono–Langerman–Ooms–Solomon [SoCG’17]: $\tilde{O}(n^{12/7})$ comparisons for algebraic decision trees
New Idea: think geometrically . . . in 2D!

map each group $B_j$ into $O(d^2)$ curves:

$$\{(x, x') : \varphi(x, B_j[v]) = \varphi(x', B_j[v'])\}$$

over $v, v' \in [d]$

map each group $A_i$ into $O(d^2)$ points:

$$(A_i[u], A_i[u']) \text{ over } u, u' \in [d]$$

(comparisons internal to $\varphi(A_i, B_j)$ are resolved if we know the location of $A_i$'s points w.r.t. $B_j$'s curves . . . )
Range Searching Lemma
[Agarwal, Chazelle, Matoušek,... '80s/'90s]

- Given $N$ points and $N$ fixed-degree algebraic curves in 2D, can (implicitly) locate all the points w.r.t. the curves in $\tilde{O}(N^{4/3})$ time
“Algebraic” 3SUM

- **Preprocessing**: apply Range Searching Lemma to
  \[ N = O(d^2 \cdot n/d) \] points & curves
  \[ \Rightarrow \tilde{O}(N^{4/3}) = \tilde{O}(dn)^{4/3} \text{ comps} \]

- **Rest**: same!

- total # comps \( \tilde{O}\left((dn)^{4/3} + n \cdot n/d\right) \)

- set \( d = n^{2/7} \Rightarrow \tilde{O}(n^{12/7}) \text{ comps} \)

- similarly: \( O^*(n^2 / \log^2 n) \) time alg’m [C.’18]
Algebraic 3SUM in $s$ Dimensions

- **Problem:** Given sets $A$, $B$, $C$ of $n$ points in $\mathbb{R}^s$, & fixed-degree algebraic function $\varphi$, decide $\exists a \in A, b \in B, c \in C$ with
  $$\varphi(a, b, c) = 0$$

- **Application:** 3COLLINEAR (for $s = 2$)...

- Range Searching Lemma generalizes, with $O(N^{2-1/O(s)})$ comps

- rest is same, except... grouping trick doesn’t work :-(
Algebraic 3SUM in $s$ Dimensions: Partial Results

- subquadratic upper bound for algebraic decision trees & $O^*(n^2 / \log^2 n)$ time alg’m for
  - *convolution* version of algebraic 3SUM in const dimension $s$, e.g., “convolution 3COLLINEAR”
  - 3CONCURRENT for 3 sets of disjoint line segments in 2D
  - given 3 polygons in 2D, decide if their common intersection is empty
Main Open Questions

- **3COLLINEAR:** $O(n^{2-\varepsilon})$ decision tree bound??

- **3SUM:** $O(n^{2-\varepsilon})$ time??
  or better than $n^2 / \log^2 n$ time?

  $((\min, +)$-convolution has $n^2 / c^{\sqrt{\log n}}$ time alg’m by Williams [STOC’14] via the \textit{polynomial method}…

  but it doesn’t work for (median, $+$)-convolution)
Integer 3SUM?

- general integer case:
  \( O^*(n^2 / \log^2 n) \) time by Baran–Demaine–Pătraşcu’05

- easy case, when input is in \([n^{2-\epsilon}]\):
  \( O(n^{2-\epsilon}) \) time by FFT
Integer 3SUM?

- clustered integer case, when input can be covered by $n^{1-\varepsilon}$ intervals of length $n$:
  
  $O(n^{2-\Omega(\varepsilon)})$ time by C.–Lewenstein [STOC’15]

  $\Rightarrow$ bounded monotone integer $d$-dimensional 3SUM in $O(n^{2-1/(d+O(1))})$ time

  $\Rightarrow$ jumbled string indexing over $[\sigma]$ in $O(n^{2-1/(\sigma+O(1))})$ time

  $\Rightarrow$ bounded-difference integer (min, $+$)-convolution in $\widetilde{O}(n^{(9+\sqrt{177})/12}) = O(n^{1.86})$ rand. time

  - after preprocessing universe $U$ of size $n$, can solve 3SUM for any subset of $U$ in $\widetilde{O}(n^{13/7})$ time
Integer 3SUM?

- clustered integer case, when input can be covered by $n^{1-\varepsilon}$ intervals of length $n$: 
$O(n^{2-\Omega(\varepsilon)})$ time by C.–Lewenstein [STOC’15] 

(based on Balog–Szemerédi–Gowers (BSG) theorem from additive combinatorics: for any sets $A, B, C$ of size $N$,

if $|\{(a, b) \in A \times B : a + b \in C\}| = \Omega(\alpha N^2)$, then 
$\exists A' \subset A, B' \subset B$ both of size $\Omega(\alpha N)$ with 
$|A' + B'| = O((1/\alpha)^5 N))$