Complexity Theory of Polynomial-Time Problems

Lecture 7: 3SUM II

Sebastian Krinninger
given sets $A, B, C$ of $n$ integers
are there $a \in A, b \in B, c \in C$ such that $a + b + c = 0$?

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

Conjecture: no $O(n^{2-\varepsilon})$ algorithm

$\rightarrow$ 3SUM-Hardness

Alternative algorithm: $O(|A| \cdot |B| + |C|)$
(store negated pairwise sums in hashmap)
Reminder: Hashing

Hash function $h: [U] \rightarrow [R]$

**Goal:** Distribute uniformly, avoid collisions, etc.
Magical hash functions

Desired properties for family of hash functions from $[U] \rightarrow [R]$ (i.e., for every $h$ chosen from family)

**Uniform difference:** $\Pr[h(x) - h(y) = i] = 1/R$

(for any $x, y \in [U]$ s.t. $x \neq y$ and $i \in [R]$)

**Balanced:** $|\{x \in S : h(x) = i\}| \leq 3n/R$

(for any set $S = \{x_1, ..., x_n\} \subseteq [U]$ and any $i \in [R]$)

**Linear:** $h(x) + h(y) = h(x + y) \pmod{R}$

(for any $x, y \in [U]$)

**But:** We do not know such a family…
Then a miracle occurs...
Almost magical hash functions

Desired properties for family of hash functions from \([U] \rightarrow [R]\) (i.e., for every \(h\) chosen from family)

**Uniform difference:** \(\Pr[h(x) - h(y) = i] = 1/R\)
(for any \(x, y \in [U]\) s.t. \(x \neq y\) and \(i \in [R]\))

**Almost balanced:** Expected number of elements from \(S\) hashed to heavy values is \(O(R)\), where value \(i \in [R]\) is heavy if \(|\{x \in S : h(x) = i\}| > 3n/R\)
(for any set \(S = \{x_1, ..., x_n\} \subseteq [U]\) and any \(i \in [R]\))

**Almost linear:** \(h(x) + h(y) \in h(x + y) + c_h + \{0,1\} \pmod{R}\)
(for any \(x, y \in [U]\) and some integer \(c_h\) depending only on \(h\))
Definition of hash function

Set \( r = km \) for some \( k \geq U/2 \) and \( U, R, r \) powers of 2

\[
\mathcal{H}_{U,R,r} = \{ h_{a,b} : [U] \to [R] \mid a \in [r] \text{ odd integer and } b \in [r] \}
\]

\[
h_{a,b}(x) = (ax + b \mod r) \div (r/R)
\]

**Thm:** Family \( \mathcal{H}_{U,R,r} \) is has the uniform difference property, is almost balanced and almost linear with \( c_{h_{a,b}} = (b - 1 \mod r) \div (r/R) \).

(Pairwise independence [Dietzfelbinger '96] implies uniform difference (easy to check) and almost balanced [Baran et al. '08]. Almost linear: easy to check.)

Rest of this lecture: \( h \) picked randomly from this family
Hashing down the universe

**Lem:** If 3SUM on universe of size $O(n^3)$ solvable in exp. time $O(n^{2-\epsilon})$, then 3SUM on arbitrary universe solvable in expect. time $O(n^{2-\epsilon})$.

Follows from [Baran et al. '08]

**Algorithm:**
Repeat until output:
- Pick hash function $h: [1 \ldots U] \to [1 \ldots 6n^3]$ at random
- $A' = \{ h(a) \mid a \in A \} \text{, } B' = \{ h(b) \mid b \in B \} \text{, } C' = \{ h(c) + c_h \mid c \in C \}$
- $A'' = \{ h(a) \mid a \in A \} \text{, } B'' = \{ h(b) \mid b \in B \} \text{, } C'' = \{ h(c) + c_h + 1 \mid c \in C \}$
- Solve two 3SUM instances $(A', B', C')$ and $(A'', B'', C'')$
- If algorithm reports no 3SUM witness: output ‘no 3SUM’
- Consider first reported 3SUM witness $x', y', z'$ for $(A', B', C')$:
  - If $h^{-1}(x'), h^{-1}(y'), h^{-1}(z' - c_h)$ contains witness $x, y, z$: output $x, y, z$
  - Consider first reported 3SUM witness $x'', y'', z''$ for $(A'', B'', C'')$:
    - If $h^{-1}(x''), h^{-1}(y''), h^{-1}(z'' - c_h - 1)$ contains witness $x, y, z$: output $x, y, z$

**No false negatives:** If $x + y = z$, then $h(x) + h(y) \in h(z) + c_h + \{0, 1\}$
Running Time

We need to bound:
- Number of iterations \( O(1) \)
- Number of candidate witnesses \( O(1) \)

Then: number of calls to 3SUM algorithm: \( O(1) \)

Number of iterations:
Triple \( x, y, z \) gives false positive if \( x + y \neq z \) and one of
\[
h(x) + h(y) = h(z) + c_h \quad \text{or} \quad h(x) + h(y) = h(z) + c_h + 1
\]

**Linearity:** \( h(x) + h(y) = h(x + y) + c_h \) or \( h(x) + h(y) = h(x + y) + c_h + 1 \)

Thus, probability that fixed \( x, y, z \) (with \( x + y \neq z \)) gives false positive is:
\[
\Pr[h(x + y) - h(z) \in \{-1, 0, 1\}] \leq \frac{3}{6n^3} = \frac{1}{2n^3} \quad \text{(uniform difference)}
\]

Overall probability of false positive: \( \leq n^3 \cdot \frac{1}{2n^3} = \frac{1}{2} \)

In expectation: 2 iterations until no false positive \quad \text{(waiting time bound)}

(If no false positive, then algorithm certainly stops)
Running Time

We need to bound:
- Number of iterations \( O(1) \)
- Number of candidate witnesses \( O(1) \)

Then: number of calls to 3SUM algorithm: \( O(1) \)

**Number of candidate witnesses:**
Fix 3SUM witness \( x', y', z' \) of instance \((A', B', C')\)
Let \( x^* \in h^{-1}(x') \)
For every \( x \neq x^* \): \( \Pr[h(x) = h(x^*)] = \frac{1}{6n^3} \) \( \text{(uniform difference)} \)

\[
E[|h^{-1}(x')|] \leq 1 + \frac{n}{4n^3} \leq 2
\]
Similarly: \( E[|h^{-1}(y')|] \leq 2 \), \( E[|h^{-1}(z')|] \leq 2 \)

\[
E[|h^{-1}(x') \cup h^{-1}(y') \cup h^{-1}(z')|] \leq O(1) \quad \text{(linearity of expectation)}
\]
In expectation, algorithm manually checks constant number of candidate witnesses per iteration
**Convolution 3SUM**

Given array $A[1 \ldots n]$ of integers

are there $i, j$ such that $A[i] + A[j] = A[i + j]$?

<table>
<thead>
<tr>
<th></th>
<th>$x$</th>
<th>$y$</th>
<th>$x + y$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$i$</td>
<td>$j$</td>
<td>$i + j$</td>
<td></td>
</tr>
</tbody>
</table>

*trivial* algorithm: $O(n^2)$

**Thm:** There is no $O(n^{2-\epsilon})$ algorithm for Convolution 3SUM unless the 3SUM Conjecture fails.

[Pătraşcu 2010]

Stepping stone towards hardness of other “structured” problems
Reduction from 3SUM

Given set $X \subseteq [U]$ of integers
are there $x, y, z \in X$ such that $x + y = z$?

Preprocessing: Check if there is a solution $2x = z$ $O(n \log n)$

Pick random hash function $h: [U] \rightarrow [R]$ (almost linear, etc.)
For this proof: assume $h$ is almost balanced and linear ($magically$…)

\[
\begin{array}{ccc}
3n/R \\
\vdots \\
1
\end{array}
\begin{array}{ccc}
1 \\
2 \\
\vdots
\end{array}
R
\]

In expectation: $O(R)$ elements in buckets with load $> 3n/R$ ($almost \ bal.$)
For each such $x$: check for 3SUM triple involving $x$ $O(Rn)$ (in exp.)
Convolution 3SUM instance

Number elements in each bucket from 0 to $\frac{3n}{R} - 1$
Iterate over all triples $i, j, k \in [\frac{3n}{R}]$

For every bucket $t$:
- Put $i$-th element to $A[8t + 1]$
- Put $j$-th element to $A[8t + 3]$
- Put $k$-th element to $A[8t + 4]$
Set all other array entries to $\infty$ (sufficiently large number)

$\left(\frac{3n}{R}\right)^3$ instances of Convolution 3SUM
Correctness

Assume $x + y = z$

Then $h(x) + h(y) = h(z) \pmod{R}$ \textit{(linearity)}

If $x = y$, triple found in preprocessing

If $x, y, \text{or } z$ hashed to heavy bucket: triple found in second step

Either $h(x) + h(y) = h(z)$ or $h(x) + h(y) = h(z) + R$

Duplicate array for Convolution 3SUM instance

\begin{align*}
\end{align*}

Thus, no false negatives. Also no false positives:

\textbf{Observation:} $A[i] + A[j] = A[i + j]$ only if $i = 8t_1 + 1$ and $j = 8t_2 + 3$

$(x + y = z \pmod{8})$ has unique solution over $\{1,3,4\}$ and $A[i] \neq A[j]$
Running Time

Assumption: Convolution 3SUM in time $O(n^{2-\varepsilon})$

Total expected running time: $O\left(n \log n + nR + \left(\frac{n}{R}\right)^3 n^{2-\varepsilon}\right)$

Set $R = n^{1-\varepsilon/4}$

Total time: $O(n^{2-\varepsilon/4})$

Contradicts 3SUM Conjecture
Set Disjointness Problem

1. **Preprocess** subsets \( \mathcal{A}, \mathcal{B} \subseteq U \) over universe \( U \)

2. Answer **queries**: Given \( A \in \mathcal{A}, B \in \mathcal{B} \), is \( A \cap B \neq \emptyset \)?

Repeated queries

(Static) data structure

Queries not known in advance

**Goal**: Lower bound on preprocessing and query time

**Offline Set Disjointness**: \( q \) queries known in advance (part of input)
Reduction to 3SUM [Kopelowitz et al]

**Thm:** Let $f(n)$ be such that 3SUM requires expected time $\Omega(n^2/f(n))$. For any constant $0 < \gamma < 1$, let ALG be an algorithm for offline Set Disjointness where $|\mathcal{A}| = |\mathcal{B}| = \Theta(n \log n)$, $|U| = \Theta(n^{2-2\gamma})$, each set in $\mathcal{A} \cup \mathcal{B}$ has at most $O(n^{1-\gamma})$ elements from $U$, and $q = \Theta(n^{1+\gamma} \log n)$. Then ALG requires expected time $\Omega(n^2/f(n))$.

**Cor:** Assuming the 3SUM conjecture, for any $0 < \gamma < 1$, any data structure for Set Disjointness has

$$t_p + \frac{1+\gamma}{2-\gamma} t_q = \Omega \left( \frac{2}{N^{2-\gamma-o(1)}} \right)$$

where $N$ is the sum of the set sizes, $t_p$ is the preprocessing time, and $t_q$ is the time per query.

(From Thm: $N = \Theta(n^{2-\gamma} \log n)$)

**Example:** Data structures with constant query time

Make $\gamma$ tend to 1, need $t_p = \Omega(N^{2-o(1)})$

Evidence that *trivial preprocessing algorithm is optimal (for constant query)*
3SUM version

Given set $X \subseteq [U]$ of integers
are there $x, y, z \in X$ such that $x - y = z$?

In the following proof we use a balanced, linear hash function with uniform difference property. *(magically…)*

This can be modified for almost balanced, almost linear hash function with uniform difference property.
Crucial insight

Higher order bits

\[ a \uparrow + b \uparrow = c \uparrow \]

Lower order bits

\[ a + b \downarrow = c \downarrow \]
Algorithm Overview

Set $R = n^\gamma$, $Q = \left(\frac{5n}{R}\right)^2$

Pick random hash functions $h: U \rightarrow [R]$ and $g_k: U \rightarrow [Q]$ for $k = 1$ to $10 \log n$

Initialize buckets $B[1], \ldots, B[R]$ s.t. $B[i] = \{x : h(x) = i\}$

For all $i \in [R], j \in [\sqrt{Q}]$, initialize buckets $B^\uparrow_k[i, j]$ and $B^\downarrow_k[i, j]$ s.t.

$B^\uparrow_k[i, j] = \{g_k(x) + j \cdot \sqrt{Q} \pmod{Q} \mid x \in B[i]\}$

$B^\downarrow_k[i, j] = \{g_k(x) - j \pmod{Q} \mid x \in B[i]\}$

Initialize $k$ set intersection problems with $B^\uparrow_k[i, j]$’s and $B^\downarrow_k[i, j]$’s

For every $z \in X$ and every $i = 1$ to $R$

Check if $B^\uparrow_k[i, g_k^\uparrow(z)]$ and $B^\downarrow_k[i - h(z) \pmod{R}, g_k^\downarrow(z)]$ intersect

If intersection for all $k$:

Search for $x \in B[i]$ and $y \in B[i - h(z) \pmod{R}]$ s.t.

$x - y = z$ and output it if found

If nothing found: output ‘no 3SUM’

$g_k^\uparrow(z)$: higher order bits of $g_k(z)$

$g_k^\downarrow(z)$: higher order bits of $g_k(z)$
Correctness I

Algorithm verifies every triple before stopping
Need to show: if $x - y = z$, then algorithm finds it

Claim 1: If $x - y = z$, then $B[i] \cap (B[j] + z) \neq \emptyset$
where $i = h(x)$, $j = i - h(z) \pmod{R}$

Linear hash function: $h(x) - h(y) = h(x - y) = h(z) \pmod{R}$
Thus: $j = h(x) - h(z) = i - h(z) \pmod{R}$
\[ y \in B[j] \Rightarrow x = y + z \in B[j] + z \]
Correctness II

Claim 1: If \( x - y = z \), then \( B[i] \cap (B[j] + z) \neq \emptyset \)
where \( i = h(x), j = i - h(z) \) (mod \( R \))

Claim 2: If \( B[i] \cap B[j] + z \neq \emptyset \), then \( B^\uparrow[i, g_k(z)] \cap B^\downarrow[j, g_k(z)] \neq \emptyset \ \forall k \).

\[
\begin{align*}
B[i] \cap B[j] + z &\neq \emptyset \\
\downarrow & \\
g_k(B[i]) \cap g_k(B[j] + z) &\neq \emptyset \\
\downarrow & \\
g_k(B[i]) \cap (g_k(B[j]) + g_k(z)) &\neq \emptyset \\
\downarrow & \\
g_k(B[i]) \cap (g_k(B[j]) + g^\uparrow_k(z) + g^\downarrow_k(z)) &\neq \emptyset \\
\downarrow & \\
(g_k(B[i]) - g^\uparrow_k(z)) \cap (g_k(B[j]) + g^\downarrow_k(z)) &\neq \emptyset \\
\downarrow & \\
B^\uparrow[i, g_k(z)] \cap B^\downarrow[j, g_k(z)] &\neq \emptyset
\end{align*}
\]

Conclusion: If \( x - y = z \), then \( B^\uparrow[i, g_k(z)] \cap B^\downarrow[j, g_k(z)] \neq \emptyset \ \forall k \).
Running time

Set intersection instance:
- Number of sets: $O(R \sqrt{Qk}) = O(n \log n)$
- Number of elements in each set: $O(\sqrt{Q}) = O(n^{1-\gamma})$
- Size of universe: $O(Q) = O(n^{2-2\gamma})$
- Number of set intersection queries: $O(nRk) = O(n^{1+\gamma} \log n)$

Finding witnesses:
- If $B_k^\uparrow[i, g_k^\uparrow(z)]$ and $B_k^\downarrow[j, g_k^\downarrow(z)]$ intersect, try to find $x \in B[i], y \in B[j]$ s.t. $x - y = z$
- Time $O\left(\frac{n}{R}\right)$ per witness check
- But: pair $i, j$ could be false positive with no such $x \in B[i], y \in B[j]$
- Probability of false positive is small
- In expectation: $O(1)$ false positives (next slide)
- Total time: $O\left(\frac{n}{R} + (#\text{false positives}) \frac{n}{R}\right) = O\left(\frac{n}{R}\right)$
Bounding number of false positive

For a fixed $z$ and any pair $x, y \in U$ s.t. $x - y \neq z$:

$$\Pr[g_k(x) = g_k(y) + g_k(z)] = \Pr[g_k(x - y) = g_k(z)] = \frac{1}{Q}$$

(linear and uniform difference)

Remember: Every bucket has size $\leq \frac{3n}{R}$ (balanced)

Prob. of false positive in buckets $B[i]$ and $B[j]$ for hash function $g_k$:

$$\Pr[g_k(B[i]) = g_k(B[j]) + g_k(z)] \leq \left(\frac{3n}{R}\right)^2 \frac{1}{Q} = \frac{9}{25}$$

Prob. of false positive in buckets $B[i]$ and $B[j]$ for all hash functions $g_k$:

$$\leq \frac{1}{n^c}$$

In expectation: total number of false positives is a constant.