Today we will continue the topic of reducing 3SUM to other problems. We will focus on reductions from 3SUM-Convolution. Recall that the 3SUM-Convolution problem is, given an integer array $A$ of length $n$, are there $i, j$, $i \neq j$ so that $A[i] + A[j] = A[i + j]$?

Last time we showed that 3SUM and 3SUM-Convolution are subquadratically equivalent. Today we will reduce 3SUM-Convolution to two problems: Triangle Listing and Exact Triangle.

We use the hash functions from last time due to Dietzfelbinger [2]. Recall their definition and properties:

Suppose we have a word-RAM with $w$ bit words. Let $a$ be a random odd $w$ bit integer. Let $1 \leq s < w$, and consider the following hash family parameterized by $a$, $h_a : \{0, \ldots, 2^w - 1\} \mapsto \{0, \ldots, 2^s - 1\}$:

$$h_a(x) := (a \cdot x \mod 2^w) >> (w - s).$$

In other words, $h_a$ multiplies $x$ by $a$ and then keeps only the $s$ top-order bits.

These hash functions have the following nice properties.

- **Almost Linearity**: For all $x, y \in \{0, \ldots, 2^w - 1\}$, $h_a(x + y) \in h_a(x) + h_a(y) + \{0, 1\} \mod 2^s$.
- **Few False Positives**: For any $x, y, z \in \{0, \ldots, 2^w - 1\}$, with $x + y \neq z$,

$$Pr[h(z) \in h(x) + h(y) + \{0, 1\} \mod 2^s] \leq O(1/2^s).$$

- **Load Balancing**: If $n$ numbers are hashed into $R = 2^s$ buckets, then the expected number of elements mapped to buckets with more than $3n/R$ elements mapped to them is $O(R)$.

# 1 Triangle Listing is 3SUM-Hard

The *Triangle Listing* problem is as follows: Given a graph $G = (V, E)$ and an integer $t$, if $G$ has at least $t$ triangles, report $t$ of them, and otherwise, report all triangles of $G$.

Assume that the matrix multiplication exponent $\omega$ is 2. Then, an algorithm by Bjorklund, Pagh, Vassilevska W. and Zwick [1] can list $t$ triangles in an $m$ edge graph in time $\tilde{O}(m^{4/3} + mt^{1/3})$. For $t \leq O(m)$, this is $\tilde{O}(m^{4/3})$.

We will give a reduction, originally due to Patrascu [3] from 3SUM-Convolution to Triangle Listing where the graph $G$ has $m$ edges and the integer $t$ is $\Theta(m)$.

For any $R \geq \sqrt{n}$, the reduction runs in $O(nR + n^3/R^2)$ time and produces a graph with $O(R\sqrt{n})$ nodes and $m = O(nR)$ edges.

If one can list $O(n^2/R)$ triangles in the graph in $O(m^{4/3-\varepsilon})$ time for some $\varepsilon > 0$ and $R$ between $\sqrt{n}$ and $n$, then using this listing algorithm with the reduction from 3SUM-Convolution to Triangle Listing, 3SUM-Convolution would be in time

$$O(nR + n^3/R^2 + (nR)^{4/3-\varepsilon}).$$

If we set $R = n^{(1+\delta)/2}$ for some $\delta > 0$, a function of $\varepsilon$, then we get that $nR + n^3/R^2 \leq O(n^{2-\delta})$, and the running time for 3SUM-Convolution becomes asymptotically

$$(nR)^{4/3-\varepsilon} = n^{(3+\delta)(2/3-\varepsilon/2)} = n^{2+\delta(2/3-\varepsilon/2)-3\varepsilon/2}.$$

If we set $\delta(2/3 - \varepsilon/2) - 3\varepsilon/2 < 0$, i.e. $\delta < 9\varepsilon/(4 - 3\varepsilon)$, we'd get a subquadratic time algorithm for 3SUM-Convolution. Since we want $R < n$, we should also make sure that $\delta < 1$, so just set $\delta$ to be the minimum of $3\varepsilon/(4 - 3\varepsilon)$ and 0.9. Since $9\varepsilon/(4 - 3\varepsilon) > 0$, we can do this.

Let's present the reduction now.

Given an array $A$ which is an input to the 3SUM-Convolution problem, hash all elements $A(i)$ using a Dietzfelbinger hash function with range $\{0, \ldots, R-1\}$. Let $B(x) = \{i \mid h(A[i]) = x\}$ be the bucket of indices whose array values are hashed to $x$.

Consider every $A[i]$ for which $|B(A[i])| > 3n/R$ and for each such $A[i]$, try all $j \in [n]$ and check whether $A[i] + A[j] = A[i+j]$. If so, return YES. Otherwise, let's assume that for every $A[i]$, $|B(A[i])| < 3n/R$.

Recall that the number of $A[i]$ that have $|B(A[i])| > 3n/R$ is $O(R)$, so the above step takes $O(nR)$ time.

Now, consider some $i \in [n]$. We will search for some $\ell$ such that $A[i] + A[\ell] = A[i+\ell]$. Let $x = h(A[i])$, and we'll try all $y \in \{0, \ldots, R-1\}$ that are options for $h(A[\ell]) = y$.

We have fixed $i$ and $y$. By almost linearity, if $A[i] + A[\ell] = A[i+\ell]$, then $h(A[i+\ell]) \in h(A[i]) + h(A[\ell]) + \{0,1\} \mod R$. In other words, if $h(A[\ell]) = y$, then $\ell \in B(y)$ and either $i+\ell \in B(x+y \mod R)$ or $i + \ell \in B(x+y+1 \mod R)$.

Moreover, because of the few false positives condition, for any fixed $i$ and $y$, the expected number of $\ell$ for which $i+\ell \in B(x+y \mod R)$ or $i+\ell \in B(x+y+1 \mod R)$ and $A[i] + A[\ell] \neq A[i+\ell]$ (i.e. a false positive), is $O(1/R)$, thus over all $i, y$, the total expected number of such false $\ell$ is $O(n^2/R)$.

Define (for fixed $i, x, y$) $T = \{j - i \mid j \in B(x+y \mod R)\}$ and $T' = \{j - i \mid j \in B(x+y+1 \mod R)\}$.

Notice that the test $T \cap B(y) \neq \emptyset$ is equivalent to whether there exists an $\ell$ with $h(A[\ell]) = y \mod R$, and $h(A[i+\ell]) = x + y \mod R$. Similarly, the test $T' \cap B(y) \neq \emptyset$ is equivalent to whether there exists an $\ell$ with $h(A[\ell]) = y \mod R$, and $h(A[i+\ell]) = x+y+1 \mod R$.

By the discussion from before, any 'Yes' answer to one of these intersection tests is either a true 3SUM-Convolution solution, or one of the $O(n^2/R)$ false positives.

Consider the following "algorithm":

- For $i$ from 1 to $n$, let $x = h(A[i])$, and for all $y$ from 0 to $R-1$:
  - Define $T$ and $T'$ as above to be $T = \{j-i \mid j \in B(x+y \mod R)\}$ and $T' = \{j - i \mid j \in B(x+y+1 \mod R)\}$.
  - Run the intersection test $T \cap B(y) \neq \emptyset$ or $T' \cap B(y) \neq \emptyset$. If the test says yes,
    * then go through all $\ell \in B(y)$, testing if $A[i] + A[\ell] = A[i+\ell]$.

> **Exercise:** Suppose that we can run the intersection tests in $O(n^{2-\gamma})$ time for $\gamma > 0$ for some choice of $R = n^{0.5+\delta}$ for $\delta \in (0, 0.5)$. Show that the total running time of the above algorithm would be $O(n^{2-\gamma} + n^{1.5+\delta} + n^{2-2\delta})$, which is subquadratic.

Now we want to simulate the intersection tests. Recall that for fixed $i$, $x = h(A[i])$ and $y$, we want to check if there is some $\ell$ for which $\ell \in B(y)$ and $\ell + i \in B(x+y \mod R)$ (or $\ell + i \in B(x+y+1 \mod R)$). Let's handle the first test. The second is handled similarly, with a new call.

Let $i = i_1\sqrt{n} + i_2$ for some $i_1, i_2 \in \{0, \ldots, \sqrt{n}\}$. Then the test is the same as:

> Is there some $\ell'$ (here $\ell' = \ell + i_2\sqrt{n}$) such that $\ell' - i_2\sqrt{n} \in B(y)$ and $\ell' + i_1 \in B((x+y) \mod R)$?

We will now create a tripartite graph with partitions $A, B, C$. $A$ contains nodes of the form $(z, i_1) \in \{0, \ldots, R-1\} \times [\sqrt{n}]$, $B$ contains nodes of the form $(y, i_2) \in \{0, \ldots, R-1\} \times [\sqrt{n}]$ and $C$ contains nodes of the form $\ell' \in [2n]$.

Let us describe the edges. For nodes $(z, i_1) \in A$ and $\ell' \in C$, we create an edge $((z, i_1), \ell')$ if and only if $\ell' + i_1 \in B(z)$ (here $z$ represents $x + y$). For nodes $(y, i_2) \in B$ and $\ell' \in C$, we add an edge $((y, i_2), \ell')$ if and only if $\ell' - i_2\sqrt{n} \in B(y)$. The number of such edges is $O(R\sqrt{n}) \times O(n/R) = O(n^{1.5})$ since the buckets are all small.

Finally, we add edges from $(z, i_1) \in A$ to $(y, i_2) \in B$ whenever $x = h(A[i_2\sqrt{n} + i_1])$ and $z = x + y$; the number of such edges is $nR$.

**Exercise:** Convince yourself that any triangle in the above graph corresponds to a positive answer to an intersection query from our algorithm from before.

The above graph has $O(n+R\sqrt{n})$ vertices and $m = O(n^{1.5}+nR)$ edges. How many triangles do we need to list? Well, we know that the total expected number of false positives is $F = O(n^2/R)$ and each triangle in the graph we created is either a true 3SUM-Convolution solution or a false positive. Thus, we only need to list $F+1 \leq O(n^2/R)$ triangles to make sure that if there is a true solution, then at least one true solution will be listed as a triangle.

The final algorithm looks like this:

(1) Take care of all elements hashing to heavy buckets, in $O(nR)$ time; return YES if any of them participate in a 3SUM-Convolution solution. Remove them.
(2) Now, form the $O(nR)$ edge graph $G$ described above.
(3) List $F+1 \leq O(n^2/R)$ triangles in $G$: $\{[(z_q, i_{1q}), (y_q, i_{2q}), \ell'_q]\}_{q=1,\ldots,F+1}$.
(4) For each triangle $[(z_q, i_{1q}), (y_q, i_{2q}), \ell'_q]$: Let $i = i_{1q} + i_{2q}\sqrt{n}$. Go through every $j \in B(y_q)$ and if $A[i] + A[j] = A[i+j]$, return YES.
(5) Return NO.

# 2 Exact Triangle.

The Exact Triangle problem is as follows. One is given a graph $G = (V, E)$ with integer edge weights $w(\cdot,\cdot)$ and one needs to determine if there are $a, b, c \in V$ with $(a, b), (b, c), (a, c) \in E$ such that $w(a, b) + w(b, c) + w(a, c) = 0$.

As previously, we can always assume that the input graph is tripartite with partitions $A$,$B$,$C$ and we are looking for $a \in A, b \in B, c \in C$.

We will reduce 3SUM-Convolution on an array of length $n$ in $O(n^{1.5})$ time to $\sqrt{n}$ instances of Exact Triangle in tripartite graphs with $O(\sqrt{n})$ nodes each.

Suppose we have this reduction and assume that Exact Triangle on $N$ node graphs can be solved in $O(N^{3-\varepsilon})$ time for some $\varepsilon > 0$. Then 3SUM-Convolution on $n$ length arrays can be solved in asymptotic time

$$n^{1.5} + \sqrt{n} \cdot (\sqrt{n})^{3-\varepsilon} = n^{1.5} + n^{2-\varepsilon/2},$$

which is subquadratic and would refute the 3SUM Hypothesis. Thus, under the 3SUM Hypothesis, Exact Triangle requires $N^{3-o(1)}$ time in $N$ node graphs.

We now give the reduction.

Suppose we are given an instance of 3SUM-Convolution, an array $X$ of length $n$, $[X[0], \ldots, X[n-1]]$.

For every index $i \in \{0, \ldots, \sqrt{n}-1\}$, we create a graph $G_i$ as follows. $G_i$ is tripartite with partitions $U_i, V_i, W_i$. $U_i$ contains a node $t$ for every $t \in \{0, \ldots, \sqrt{n}-1\}$, $V_i$ contains a node $q$ for every $q \in \{0, \ldots, 2\sqrt{n}-2\}$, and $W_i$ contains a node $s$ for every $s \in \{0, \ldots, \sqrt{n}-1\}$.

The edges of $G_i$ are as follows.

- For every $q \in W_i, t \in U_i$, if $q - t \in \{0, \ldots, \sqrt{n}-1\}$, we add an edge $(q, t)$ with weight $X[i\sqrt{n} + (q-t)]$.
- For every $s \in V_i, t \in U_i$, add an edge $X[s\sqrt{n} + t]$.
- For every $s \in V_i, q \in W_i$, add an edge $-X[(s+i)\sqrt{n} + q]$.

**Claim 1.** *$X$ has a 3SUM-Convolution solution if and only if for some $i \in \{0, \ldots, \sqrt{n}-1\}$, $G_i$ contains an Exact Triangle.*

*Proof.* Suppose that $X$ has a 3SUM-Convolution: some $k, j$, $k \neq j$ such that $X[k] + X[j] = X[k+j]$. Now, $k = i\sqrt{n} + \ell$ for some $i, \ell \in \{0, \ldots, \sqrt{n} - 1\}$, and $j = s\sqrt{n} + t$ for some $s, t \in \{0, \ldots, \sqrt{n} - 1\}$. Thus also, $k + j = (s+i)\sqrt{n} + (t + \ell)$.

Consider graph $G_i$. The nodes $t \in U_i, s \in V_i, (t+\ell) \in W_i$ in $G_i$ form a triangle (since $(t+\ell) - t \in \{0, \ldots, \sqrt{n} - 1\}$). The weight of this triangle is

$$X[s\sqrt{n} + t] + X[i\sqrt{n} + \ell] - X[(s+i)\sqrt{n} + (t+\ell)] = X[j] + X[k] - X[k+j] = 0.$$

Now, let us assume that for some $i$, $G_i$ contains an Exact Triangle ($q \in W_i, s \in V_i, t \in U_i$). The weight of the triangle is $X[s\sqrt{n} + t] + X[i\sqrt{n} + (q - t)] - X[(s+i)\sqrt{n} + q] = 0$.

Let $a = s\sqrt{n} + t, b = i\sqrt{n} + (q - t), c = (s+i)\sqrt{n} + q$. Notice that $c = a + b$. Also $X[a] + X[b] = X[c]$. We thus have a 3SUM Convolution solution. □

We also have a reduction from Negative Triangle to Exact Triangle:

**Theorem 2.1** (R. and V. Williams'2009)**.** *There is an $O(n^2 \log M)$ time reduction that given an $n$ node graph with edge weights in $\{-M, \ldots, M\}$ produces $O(\log M)$ instances of Exact Triangle on $n$ nodes each and with weights in $\{-O(M), \ldots, O(M)\}$. Thus if Exact Triangle in $n$ node graphs with polynomial edge weights is in $O(n^{3-\varepsilon})$ time for some $\varepsilon > 0$, then Negative Triangle with polynomial edge weights is in $O(n^{3-\varepsilon} \log n)$ time.*

Exact Triangle is thus a problem that requires $n^{3-o(1)}$ time under both the APSP and the 3SUM Hypotheses. So even if one of the hypothesis fails, Exact Triangle would still be hard as long as the other one is true.

# References


[1] Andreas Björklund, Rasmus Pagh, Virginia Vassilevska Williams, Uri Zwick: Listing Triangles. ICALP (1) 2014: 223–234.

[2] M. Dietzfelbinger. Universal hashing and k-wise independent random variables via integer arithmetic without primes. In Proc. 13th Symposium on Theoretical Aspects of Computer Science (STACS), pages 569–580, 1996.

[3] Mihai Patrascu: Towards polynomial lower bounds for dynamic problems. STOC 2010: 603–610.