# Persistence Algorithm Takes Cubic Time in the Worst Case

Dmitriy Morozov

February 21, 2005

Given a sequence of $N$ simplices, we consider the sequence of sets $K_i$ consisting of the first $i$ simplices, for $1 \leq i \leq N$. We call the sequence of $K_i$ a *filtration* if all the $K_i$ are simplicial complexes. In this note, we describe a filtration of a simplicial complex of $N$ simplices on which the algorithm PAIR-SIMPLICES of Edelsbrunner, Letscher and Zomorodian [1] performs $\Omega(N^3)$ operations. The existence of this filtration should be contrasted to the experimentally observed only slightly super-linear running time for filtrations that arise from applications.

We describe the space as well as the ordering on the simplices. Let $n = \lfloor (N+29)/7 \rfloor$, $v = \lfloor (n-1)/2 \rfloor$, and note that both $n$ and $v$ are in $\Omega(N)$. In our filtration, all vertices appear before all edges in the filtration, and all edges appear before all triangles. The indices that we assign to the simplices will be within their respective classes (e.g., edge labeled $n$ will appear before the triangle labeled 1). Some edges will receive a negative index, which is done for simplicity to indicate that they appear before the edges with positive labels (see Figure 2).

Figure 1 illustrates the construction of our space as well as the assignment of indices to the simplices. Starting with triangle $ABC$, we add $v$ vertices inside the triangle in the following manner: we place the first vertex $V_1$ near the middle of edge $AB$, the second vertex $V_2$ near the middle of $AV_1$, $V_3$ near the middle of $BV_2$, $V_4$ near $AV_2$, $V_5$ near $BV_3$, $V_6$ near $V_1V_2$, $V_7$ near $V_1V_3$, and so on, moving from both ends inwards at each stage. The edges joining $C$ with the $V_i$ are the first to appear in the filtration, each one merging the component containing $C$ with $V_i$. These edges are not important in our argument, so we do not label them. Edge $AB$ get index 1, and the remaining edges are assigned indices from the ends inwards similar to the vertices: $AV_1$ gets $n$, $BV_1$ gets $n-1$, $AV_2$ gets $n-2$, $BV_3$ gets $n-3$, $V_1V_2$ gets $n-4$, $V_1V_3$ gets $n-5$, and so on (see Figure 1). Similarly, the triangles are assigned indices from the ends inwards in stages: $ABV_1$ gets 1, $AV_1V_2$ gets 2, $BV_1V_3$ gets 3, $AV_2V_4$ gets 4, $BV_3V_5$ gets 5, and so on. We call these triangles the *base triangles*.

In addition, we place $n-1$ vertices above the plane of triangle $ABC$, one above each edge $AV_i$, $BV_j$, and $V_iV_j$, and join them to the vertices of those edges (Figure 1 depicts only two of the $n-1$ such vertices). One of the edges joining the vertex above the edge $k$ to its endpoints will merge the component containing the endpoint of the edge $k$ with the vertex above the plane. We do not label this edge. The other edge gets index $k-(n+1)$ which is negative. The triangle formed gets an index larger than $v$, so that the triangles not in the plane of $ABC$ appear last in the filtration. We call them *fin triangles*.

Consider what happens when algorithm PAIR-SIMPLICES processes the described filtration. There are two interesting parts to its execution. First, the base triangles 1 to $v$ are processed, the edges $n$ to $n-v$ build up lists of $\Omega(n)$ simplices each (see Figure 3). Second, when the fin triangles $v+1$ to $v+n$ are processed, the search for the corresponding edges goes through all the lists stored at these edges, merging lists of length $\Omega(n)$. As a result, for $\Omega(n)$ triangles we perform $\Omega(n)$ merges

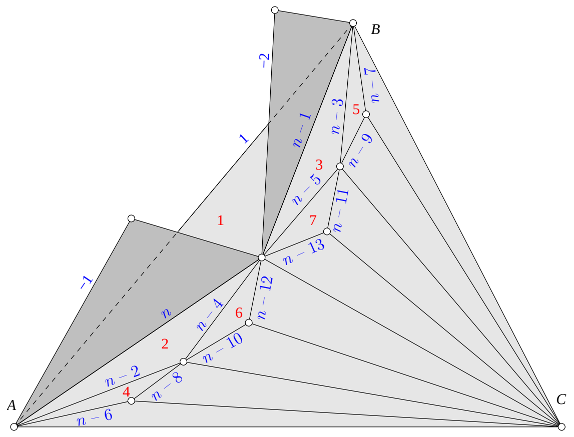


Figure 1: The underlying space of the filtration. The edge indices are blue and the triangle indices are red. Each edge with label larger than 1 has a triangle coming out of the plane of the triangle $ABC$ — only triangles above edges labeled $n$ and $n-1$ are shown.

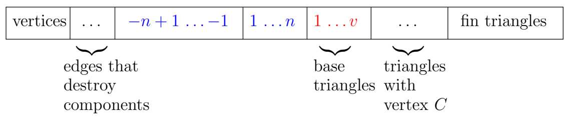
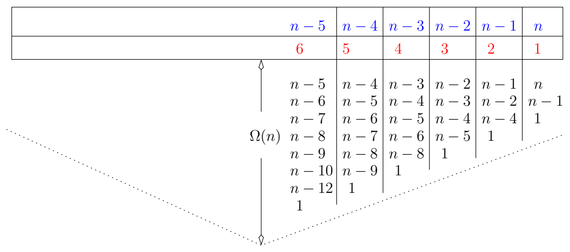


Figure 2: Simplex ordering



$n-5$
$n-4$
$n-3$
$n-2$
$n-1$
$n$
6
5
4
3
2
1
$\Omega(n)$
$n-5$ $n-6$ $n-7$ $n-8$ $n-9$ $n-10$ $n-12$ $1$
$n-4$ $n-5$ $n-6$ $n-7$ $n-8$ $n-9$ $1$
$n-3$ $n-4$ $n-5$ $n-6$ $n-8$ $1$
$n-2$ $n-3$ $n-4$ $n-5$ $1$
$n-1$ $n-2$ $n-4$ $1$
$n$ $n-1$ $1$

Figure 3: Processing the filtration: pairing of edges 1 through $n$ with the base triangles.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $n$ | $n-1$ | $n-2$ | $n-3$ | $n-4$ | $n-5$ | $n-6$ | $n-7$ | |
| $-1$ | $1$ | $n-4$ | $n-5$ | $n-6$ | $n-7$ | $n-8$ | $n-9$ | |
| | $-1$ | $-1$ | $1$ | $n-8$ | $n-9$ | $n-10$ | $n-11$ | $\dots$ |
| | | | $-1$ | $-1$ | $1$ | $n-12$ | $n-13$ | |
| | | | | | $-1$ | $-1$ | $1$ | |
| | | | | | | | $-1$ | |

Figure 4: Processing the filtration: intermediate lists during the search for the pairing of the fin triangle above edge $n$.

each of which takes time $\Omega(n)$. It follows that the total running time is $\Omega(N^3)$.

To see how this happens, let us consider what happens when we process the triangles one by one (see Figure 3). First, the base triangles are processed. Triangle 1 is paired with edge $n$, depositing its list of boundary edges ($n$, $n-1$, 1) with edge $n$. Triangle 2 with boundary edges ($n$, $n-2$, $n-4$) causes a collision at edge $n$, the two lists merge and the result ($n-1$, $n-2$, $n-4$, 1) gets deposited with edge $n-1$. Triangle 3 with boundary edges ($n-1$, $n-3$, $n-5$) causes a collision at edge $n-1$, the lists merge and the result ($n-2$, $n-3$, $n-4$, $n-5$, 1) is deposited with edge $n-2$. Triangle 4 with boundary edges ($n-2$, $n-6$, $n-8$) causes a collision at edge $n-2$, the lists merge and the result ($n-3$, $n-4$, $n-5$, $n-6$, $n-8$, 1) is deposited with edge $n-3$. Continuing to the last fin triangle builds up lists stored with edges 1 to $n$ and $\Omega(n)$ of those lists have length $\Omega(n)$.

Second, the fin triangles are processed. The triangle above edge $n$ has the labeled edges $n$ and $-1$ in its boundary. It eventually gets paired with edge $-1$, but before that happens, the search goes through each one of the edges $n$ through 1. To see this, note that the boundary edges cause a collision at edge $n$, the lists are merged to get ($n-1$, 1, $-1$), the new list causes a collision at edge $n-1$, and after merging we get ($n-2$, $n-4$, $-1$). The next collision is at edge $n-2$, after merging the list becomes ($n-3$, $n-5$, 1, $-1$), the collision at edge $n-3$ produces the list ($n-4$, $n-6$, $n-8$, $-1$), the collision at $n-4$ gives ($n-5$, $n-7$, $n-9$, 1, $-1$), and so on; see Figure 4. This process is repeated for each fin triangle: a similar merge pattern occurs, the only difference being that it starts at the base edge of the processed fin triangle.

Observe that some edges get cancelled when the lists are merged, but get reintroduced two merges later. The length of each intermediate list grows by one every two merges, and its length reaches $\Omega(n)$. This implies that the running time of the algorithm is cubic as claimed earlier: for each one of the $\Omega(n)$ fin triangles, $\Omega(n)$ lists of length $\Omega(n)$ are merged, therefore, the running time is $\Omega(N^3)$.

## References

[1] H. Edelsbrunner, D. Letscher, and A. Zomorodian. Topological persistence and simplification. *Discrete Comput. Geom.*, 28:511–533, 2002.