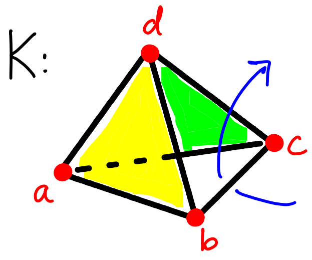
MATH 529 – Lecture 20 (03/21/2024)

Today:
* Example for incremental algorithm
* UNION-FIND data structure
* persistence algorithm

Example

K: 

filtration: a, b, c, d, ab, ac, ad, bc, bd, cd, abd, acd $(m=12)$

There is one connected component, and one hole.

So we expect $\beta_0=1, \beta_1=1$.

→ There are two triangles missing from the tetrahedron, but there is only one hole, as indicated by the arrow '↺'. Also, adding one more triangle gives a disc!

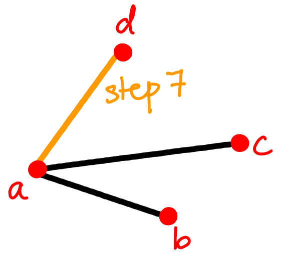
Steps of the algorithm $(\beta_0=\beta_1=0, \beta_2=0$ to start$)$ → we have no positive triangles. So β_2 remains at zero.

1–4: a, b, c, d: β_0 goes $1\to2\to3\to4$ (all are positive simplices)

5: ab: β_0--; $(\beta_0=3)$ (ab is a negative simplex)

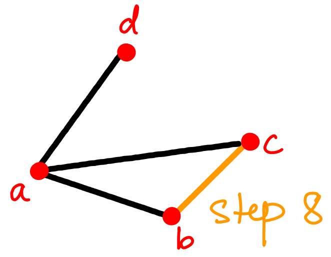
6: ac: β_0--; $(\beta_0=2)$ (ac is a negative simplex)

7: ad: β_0--; $(\beta_0=1)$ (ad is a negative simplex)



8: bc: β_1++; $(\beta_1=1)$ (bc is a positive 1-simplex)

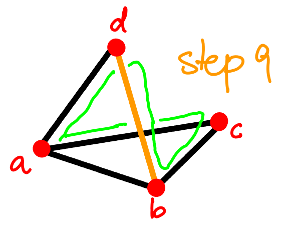
bc is part of exactly one 1-cycle: abca



9. bd: β_1++; $(\beta_1=2)$ (bd is positive)

bd is part of at least one 1-cycle: abda, acbda.

We will study this aspect later. For now, note that bd is positive.



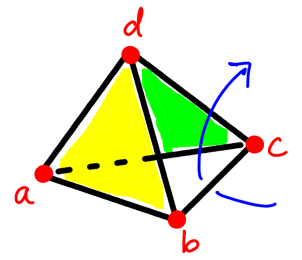
10. cd: β_1++; $(\beta_1=3)$ (cd is positive) → as with bd, cd is part of several 1-cycles

We now have $\beta_0=1, \beta_1=3$.

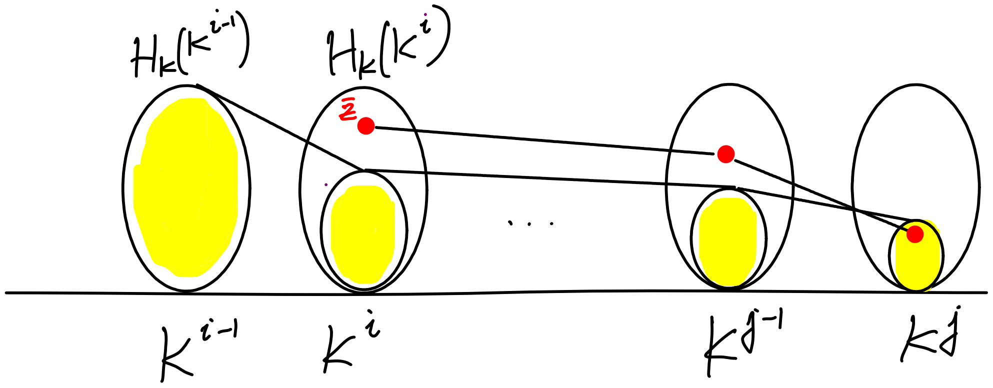
11. abd; β_1 – – ; ($\beta_1=2$) (abd is negative)

12. acd; β_1 – – ; ($\beta_1=1$) (acd is negative)

Finally, we get $\beta_0=1$, $\beta_1=1$, as expected.



Here is a visualization of how a homology class evolves:



The class $[z]$ is created at K^i, as σ^i comes in, and is destroyed as it enters K^j, as σ^j comes in.

We can maintain connected components and classify (0- and) 1-simplices efficiently as positive or negative using the UNION-FIND data structure. This data structure maintains (and updates) a collection of connected components with a label or ID. — think of each component as a set, There are two operations one can perform on the collection, which are UNION and FIND.

Illustration of UNION-FIND Data Structure

The data structure maintains a set $\mathcal{S}$ of connected components.

Two operations (on members of $\mathcal{S}$)

$$\text{FIND}(v) = \begin{cases} \text{returns (the identity of) the set that } v \text{ belongs to, if it already belongs to one in } \mathcal{S} \\ \phi, \text{ otherwise.} \end{cases}$$

UNION$(\mathcal{U}, \mathcal{V})$: replaces $\mathcal{U}$ with $\mathcal{U} \cup \mathcal{V}$ (and deletes $\mathcal{V}$).

Steps in the algorithm for the Example ($\beta_0 = 0, \beta_1 = 0$ at start)

1-4: For $v = a, b, c, d$, FIND$(v) = \phi$, so we add $\{v\}$ as a new set to $\mathcal{S}$ each time, and do $\underline{\beta_0{+}{+}}$.

$\beta_0 = \beta_0 + 1$

Now $\mathcal{S} = \{\{a\}, \{b\}, \{c\}, \{d\}\}$, and $\beta_0 = 4$.

5-7: For $e = uv = ab, ac, ad$, do FIND(u), FIND(v)

```
If FIND(u) ≠ FIND(v)
    UNION(FIND(u), FIND(v)); β_0 - - ;
```

$$\mathcal{S} = \{\{\underline{\underline{a}}, b\}, \{c\}, \{d\}\} \longrightarrow \{\{\underline{\underline{a}}, b, c\}, \{d\}\} \longrightarrow \{\{\underline{\underline{a}}, b, c, d\}\}$$

a represents its set in each case

We have $\beta_0 = 1$ now.

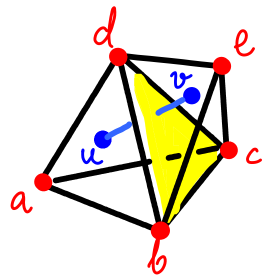
<u>8-10</u>: $uv = bc, bd, cd$.

In each case, FIND(u) = FIN(v) = $a \rightarrow$ set represented by a

so, $\beta_1{+}{+}$;

$\mathcal{S} = \{\{\underline{\underline{a}}, b, c, d\}\}$ $\beta_0 = 1, \beta_1 = 3$.

For triangles, we use duality: consider a vertex for each tetrahedron and an edge for each triangle:



u is dual to $\triangle$ abcd
v is dual to $\triangle$ bcde
uv is dual to $\triangle$bcd

We can traverse the filtration in the reverse order, and classify triangles as positive or negative. Recall that all tetrahedra are negative.

The notions of positive/negative will be flipped when one considers the corresponding dual edge or vertex.

The incremental algorithm gives

$$\beta_k^l = pos_k^l - neg_{k+1}^l \quad \text{where}$$

pos_k^l = # positive k-simplices in K^l and

neg_{k+1}^l = # negative $(k+1)$-simplices in K^l.

This result also holds for each subcomplex K^l in the filtration. We indicate this fact by adding the superscript l in each term.

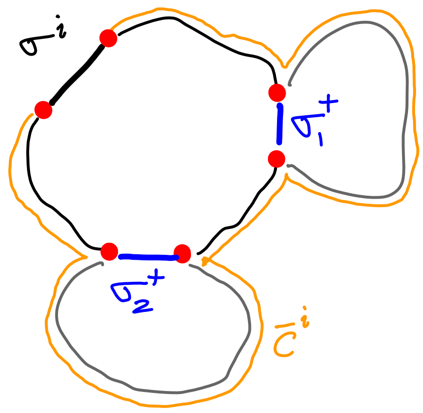
Persistence Algorithm (over $\mathbb{Z}_2$) (Edelsbrunner, Letscher, Zomorodian, 2002)

We can pair positive and negative simplices so that each pair represents a member in the homology class. The longer its "life", the more significant it is as a feature.

This original algorithm can be described as building on top of the incremental algorithm. But the matrix algorithm (SNF) can also be expanded to compute persistence, and the latter option generalizes to arbitrary dimensions.

We maintain a basis for H_k, which is empty to start with. For each positive k-simplex σ^i, we find a non-bounding k-cycle $\bar{c}^i$ that contains σ^i but no other positive k-simplex. $\bar{c}^i$ is the **canonical cycle** of σ^i.

We can show the canonical cycle always exists. Recall in the example, bd comes in at step 9, and is part of two cycles.



If we start with a cycle that has, say, two other positive k-simplices σ_1 and σ_2 apart from σ^i. Then we could add the canonical cycles corresponding to σ_1 and σ_2 to this cycle (addition is $\mathbb{Z}_2$), to get $\bar{c}^i$.

Now add $[\bar{c}^i]$ to the basis for H_k, i.e., the homology class of $\bar{c}^i$ is added as a new element of H_k. So, we represent $\bar{c}^i + B_k$ using $\bar{c}^i$, which in turn is represented using σ^i.

For each negative $(k+1)$-simplex σ^j, we find the corresponding positive k-simplex σ^i, and remove its class from H_k.

A general class is represented as $\bar{d} + B_k = \sum_g' (\bar{c}^g + B_k)$.

But $\sum_g' (\bar{c}^g + B_k) = \sum_g' \bar{c}^g + B_k$, so $\bar{d} \sim \sum_g' \bar{c}^g$.

Each $\bar{c}^g$ is represented by the positive k-simplex σ^g for $g < j$, that are not yet paired. This collection of all positive k-simplices in $[\bar{d}]$, written as $\Gamma = \Gamma(\bar{d})$, is uniquely determined by $\bar{d}$.

Note that $\bar{d}$ here is $\partial\sigma^j$ when considering the negative simplex σ_j.

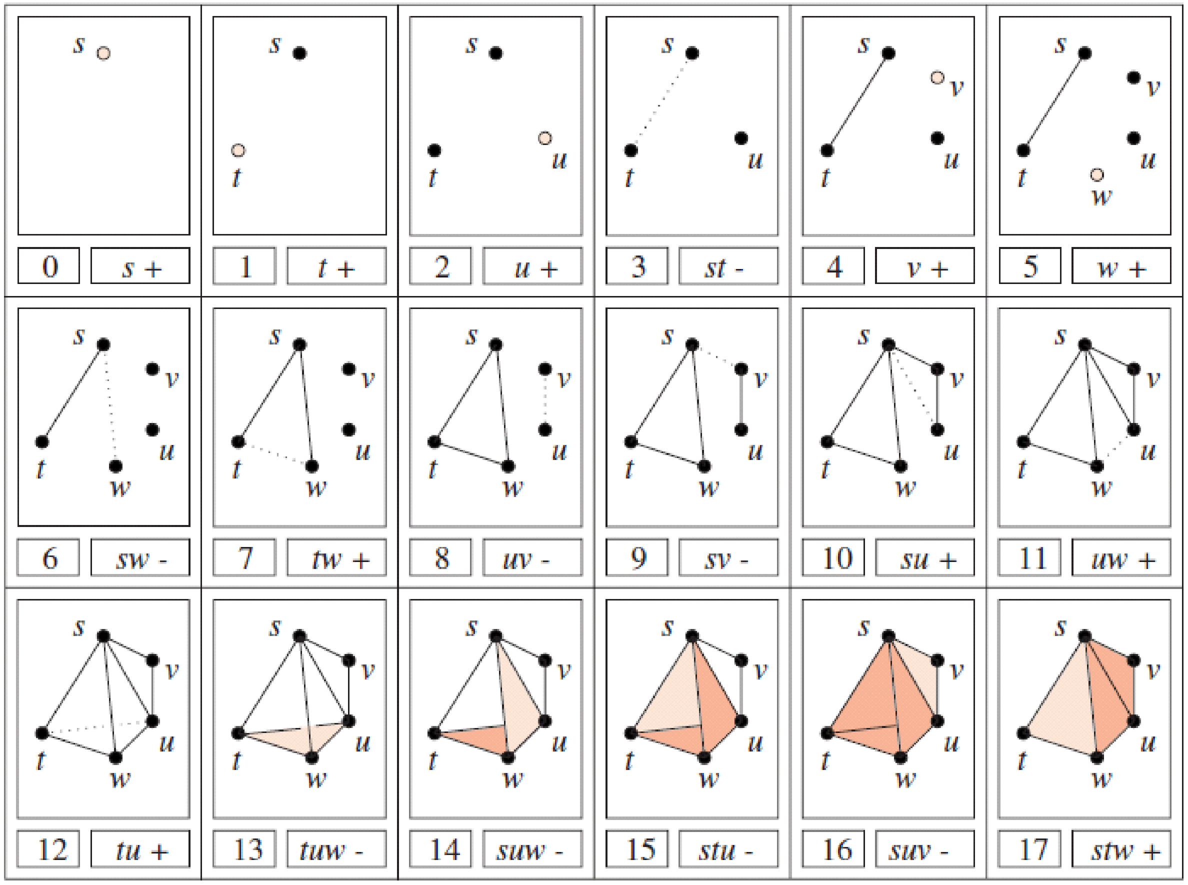
$\underline{\text{youngest}} \to$ youngest : comes in the latest.

Let σ^i = the $\underline{\text{youngest}}$ positive k-simplex in $\Gamma(\bar{d})$.

We record the pair (σ^i, σ^j), and its persistence as $j-i-1$.

We consider the example from the paper by Edelsbrunner, Letzcher, and Zomorodian (Topological Persistence and Simplification, 2002).

The final complex K is a hollow tetrahedron with a flap (i.e., it is a 2-complex). We get a filtration with 18 subcomplexes, corresponding to the 18-simplices, as shown in the next page.



We will go through all pairings in the next lecture...