On the encoding of large, high-dimensional and unorganized datasets

Ulderico Fugacci

30 January 2020
Motivation

Topological Data Analysis (TDA) and Persistent Homology (PH) allow for extracting the core topological information from large, high-dimensional and unorganized datasets.

E.g. point clouds, complex networks, (semi-)metric spaces
Topological Data Analysis (TDA) and Persistent Homology (PH) allow for extracting the core topological information from *large, high-dimensional* and *unorganized datasets*. E.g. *point clouds, complex networks, (semi-)metric spaces*. 
Motivation

Simplicial Complex:
A family $K$ of subsets (called *simplices*) of a finite set $V$
closed under the operation of taking subsets

$0-, 1-, 2-, 3-$ simplices
Motivation

**Simplicial Complex:**
A family $K$ of subsets (called *simplices*) of a finite set $V$ closed under the operation of taking subsets

**Issue:**

$V$ a point cloud with $|V| \geq 30$

Filtered complex $K$ associated with $V$ consists of more than a *billion* of simplices
Motivation

Simplicial Complex:
A family $K$ of subsets (called *simplices*) of a finite set $V$ closed under the operation of taking subsets.

Solution:
Development of *compact and efficient data structures* for encoding simplicial complexes.
Motivation

Outline:

• Which info to be stored?

• Data structures:
  • Simplex-based representations
  • Top-based representations
  • Operator-driven representations

• Issues and solutions in adopting top-based representations

• Comparisons
Motivation

Out of Scope:

• Data structures for specific classes of complexes
  • E.g., manifolds or complexes of low dimension
• Hierarchical and multi-resolution models
• Construction of a simplicial complex from a dataset
Data Structures for Simplicial Complexes

The entities which a simplicial complex consists of are:

- its simplices
  \[ K = K_0 \cup K_1 \cup \ldots \cup K_d \]

  where \( K_i \) is the collection of the \( i \)-simplices of \( K \)

- the topological relations
  \[ R_{i,j} \subseteq K_i \times K_j \]

  between the simplices of \( K \) encoding the (co-)boundary of each simplex
Data Structures for Simplicial Complexes

Data structure:

The *entities* which a simplicial complex consists of are:

- its *simplices*
  \[ K = K_0 \cup K_1 \cup \ldots \cup K_d \]
  where \( K_i \) is the collection of the i-simplices of \( K \)

- the *topological relations*
  \[ R_{i,j} \subseteq K_i \times K_j \]
  between the simplices of \( K \) encoding the (co-)boundary of each simplex

A *data structure* for \( K \) has to explicitly *store* a portion of the above information and to (efficiently) *retrieve* the remaining part
Data Structures for Simplicial Complexes

Topological Relations:

Given a $i$-simplex $\sigma$ and a $j$-simplex $\tau$ of $K$,

$$ (\sigma, \tau) \in R_{i,j} $$

$$ \begin{cases} 
\sigma \subseteq \tau & \text{for } i < j \\
| \sigma \cap \tau | = i \quad (\text{equivalently, } \sigma \cap \tau \in K_{i-1}) & \text{for } i = j \\
\tau \subseteq \sigma & \text{for } i > j 
\end{cases} $$

Example:

$$ (12, 123) \in R_{1,2} \quad (12, 24) \in R_{1,1} \quad (12, 1) \in R_{1,0} $$
Data Structures for Simplicial Complexes

**Topological Relations:**

Given an i-simplex \( \sigma \) and a j-simplex \( \tau \) of \( K \),

\[
(\sigma, \tau) \in R_{i,j}
\]

\[
\begin{align*}
\sigma &\subseteq \tau & \text{for } i < j \\
|\sigma \cap \tau| & = i \text{ (equivalently, } \sigma \cap \tau \in K_{i-1}) & \text{for } i = j \\
\tau &\subseteq \sigma & \text{for } i > j
\end{align*}
\]

**Example:**

\[
(12, 123) \in R_{1,2} \quad (12, 24) \in R_{1,1} \quad (12, 1) \in R_{1,0}
\]

An i-simplex \( \sigma \) is called a **top simplex** of \( K \) if \( R_{i,i+1} \) is empty.
Data Structures for Simplicial Complexes

Data structures:

- Store all the entities
- Store only the top-simplices

- **Simplex-based** representations
- **Top-based** representations
- **Operator-driven** representations
Data Structures for Simplicial Complexes

Data structures:

- **Store all the entities**
  - Incidence Graph

- **Operator-driven representations**

- **Simplex-based representations**

- **Top-based representations**

Store only the top-simplices
Data Structures for Simplicial Complexes

Data structures:

- Store all the entities
- Incidence Graph
- Simplex Tree

Efficiency

Compactness

- Simplex-based representations
- Top-based representations
- Operator-driven representations

Store only the top-simplices
Data Structures for Simplicial Complexes

Data structures:
- Store all the entities
- Incidence Graph
- Simplex Tree
- IA* Data Structure

Efficiency

Compactness

Simplex-based representations
Top-based representations
Operator-driven representations

Store only the top-simplices
Data Structures for Simplicial Complexes

- **Data structures:**
  - Store all the entities: Incidence Graph, Simplex Tree
  - Store only the top-simplices: IA* Data Structure, Stellar Tree

- **Representations:**
  - Simplex-based representations
  - Top-based representations
  - Operator-driven representations
Data Structures for Simplicial Complexes

Data structures:

- Store all the entities
- Incidence Graph
- Simplex Tree
- IA* Data Structure
- Stellar Tree
- Skeleton Blocker
- Store only the top-simplices

- **Simplex-based** representations
- **Top-based** representations
- **Operator-driven** representations
Simplex-based Representations

The simplicial complex $K$ is encoded via a directed graph $G = (N, A)$:

All the relations between simplices can be immediately retrieved

The representation \textit{size exponentially increases} with the complex dimension
Simplex-based Representations

The simplicial complex $K$ is encoded via a directed graph $G = (N, A)$:

$$N \leftrightarrow K$$

$$(\sigma, \tau) \in A \iff (\sigma, \tau) \in R_{i,i+1} \text{ and } I(\sigma) < I(\tau)$$

where $I(\sigma)$ denotes the maximum value taken by the vertices of $\sigma$ w.r.t. a total order on $K_0$

Graph is not uniquely determined but it depends on the chosen vertex order
Simplex-based Representations

The simplicial complex $K$ is encoded via a directed graph $G = (N, A)$:

- $N \leftrightarrow K$
- $(\sigma, \tau) \in A \leftrightarrow (\sigma, \tau) \in R_{i,i+1}$ and $I(\sigma) < I(\tau)$

where $I(\sigma)$ denotes the maximum value taken by the vertices of $\sigma$ w.r.t. a total order on $K_0$

Graph is *not uniquely determined* but it depends on the chosen vertex order
Top-based Representations

The simplicial complex $K$ is encoded via a directed graph $G = (N, A)$:

- $N \leftrightarrow K_0 \cup K_{top}$
- $(\sigma, \tau) \in A \leftrightarrow \begin{cases} \sigma \in K_{top} \text{ and } (\sigma, \tau) \in R_{i,0} \\ \sigma, \tau \in K_{top} \text{ and } (\sigma, \tau) \in R_{i,i} \\ \tau \in K_{top} \text{ and } (\sigma, \tau) \in (R_{0,i})^* \end{cases}$

- Compact: it explicitly stores just a fraction of the entities of a simplicial complex
- Not all the relations between simplices are immediately available
The simplicial complex $K$ is encoded by storing its 1-skeleton (i.e. the graph consisting of the 0- and the 1-simplices) and a map returning, for each 1-simplex $\sigma$, the blockers of $K$ containing $\sigma$, where:

- A simplex $\tau$ is a blocker if $\tau$ does not belong to $K$ but all its faces do.

*Designed for* flag complexes (e.g. VR complexes) and edge contraction

*Too specific: inefficient in any other task*
Top-based representations look like promising data structures for encoding a simplicial complex K.

But, how to…

1. Store information associated to each simplex of K (e.g. labels, gradient, …)?

2. Efficiently perform operators having explicitly stored a fraction of the entities of K?
Top-based representations look like promising data structures for encoding a simplicial complex $K$.

**But, how to…**

1. **Store information associated to each simplex of $K$ (e.g. labels, gradient, …)?**

   Attach information to the top simplices only

   $$
   \begin{array}{cccc}
   v_0 & v_1 & v_2 & v_0v_1v_2 \\
   0 & 0 & 1 & 1 \\
   \end{array}
   $$

2. **Efficiently perform operators having explicitly stored a fraction of the entities of $K$?**
Possible Issues in Top-based Representations

Top-based representations look like promising data structures for encoding a simplicial complex $K$.

But, how to…

1. **Store information associated to each simplex of $K$ (e.g. labels, gradient, …)?**

   Attach information to the top simplices only

2. **Efficiently perform operators having explicitly stored a fraction of the entities of $K$?**

   Re-define the algorithms performing the operators trying to extract the lowest possible amount of non-explicitly stored entities
## Comparisons

### Top-based vs Simplex-based

| Dataset      | $d$ | $|\Sigma_0| \quad |\Sigma_{top}| \quad |\Sigma|$ | Storage Cost |
|--------------|-----|-----------------|-----------------|-------|-------------|
|              |     |                 |                 |       | $IA^*$  | $IG$ | $ST$  |
| DTI-scan     | 3   | 0.9M            | 5.5M            | 24M   | 0.97    | 11.9 | 2.4   |
| VisMale      | 3   | 4.6M            | 26M             | 118M  | 4.7     | -    | 9.7   |
| Ackley4      | 4   | 1.5M            | 32M             | 204M  | 6.8     | -    | 12.8  |
| Amazon01     | 6   | 0.2M            | 0.4M            | 2.2M  | 0.12    | 1.6  | 0.3   |
| Amazon02     | 7   | 0.4M            | 1.0M            | 18.4M | 0.28    | 9.8  | 1.5   |
| Roadnet      | 3   | 1.9M            | 2.5M            | 4.8M  | 0.8     | 3.3  | 1.0   |
| Sphere-1.0   | 16  | 100             | 224             | 0.6M  | 0.003   | 0.9  | 0.04  |
| Sphere-1.2   | 21  | 100             | 285             | 26M   | 0.0032  | -    | 1.5   |
| Sphere-1.3   | 23  | 100             | 382             | 197M  | 0.0034  | -    | 11.01 |
Comparisons

Top-based vs Simplex-based

- **PROB.5D (607 MB)**: IA* = 1.98, Simplex tree = 11.7, Stellar tree = 1.1
- **PROB.7D (7.9 GB)**: IA* = 2, Simplex tree = 19.62, Stellar tree = 1.01
- **PROB.40D (2.6 GB)**: IA* = 2, Simplex tree = 5.54, Stellar tree = 1
- **VISMALE7D (134 MB)**: IA* = 2.13, Simplex tree = 73.88, Stellar tree = 1.06
- **FOOT10D (2.1 GB)**: IA* = 2, Simplex tree = 180, Stellar tree = 1.05
- **LUCY34D (2.0 GB)**: IA* = 2.05, Simplex tree = 4,925.9, Stellar tree = 1.05

Legend:
- IA*: IA-based
- Simplex tree: Simplex-based tree
- Stellar tree: Stellar-based tree
### Comparisons

**Top-based vs Operator-driven**

<table>
<thead>
<tr>
<th>Data</th>
<th>$\omega$</th>
<th>Contr. Edges</th>
<th>Timings</th>
<th>Memory Peak</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CHICAGO</strong></td>
<td>weak</td>
<td>9.15h</td>
<td>2.27m</td>
<td>9.19h</td>
</tr>
<tr>
<td></td>
<td>top</td>
<td>6.38K</td>
<td>0.01s</td>
<td>0.02s</td>
</tr>
<tr>
<td></td>
<td>Skel.</td>
<td>0.00s</td>
<td>0.15s</td>
<td>0.15s</td>
</tr>
<tr>
<td></td>
<td>weak</td>
<td>out-of-memory</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>top</td>
<td>7.99K</td>
<td>0.04s</td>
<td>0.06s</td>
</tr>
<tr>
<td></td>
<td>Skel.</td>
<td>0.00s</td>
<td>0.71s</td>
<td>0.71s</td>
</tr>
<tr>
<td><strong>ATHENS</strong></td>
<td>weak</td>
<td>out-of-memory</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>top</td>
<td>27.9K</td>
<td>0.08s</td>
<td>0.11s</td>
</tr>
<tr>
<td></td>
<td>Skel.</td>
<td>0.00s</td>
<td>0.74s</td>
<td>0.75s</td>
</tr>
<tr>
<td></td>
<td>weak</td>
<td>out-of-memory</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>top</td>
<td>31.2K</td>
<td>0.40s</td>
<td>0.49s</td>
</tr>
<tr>
<td></td>
<td>Skel.</td>
<td>0.01s</td>
<td>7.73s</td>
<td>7.74s</td>
</tr>
<tr>
<td><strong>VISAILE</strong></td>
<td>weak</td>
<td>34.3m</td>
<td>1.28m</td>
<td>40.4m</td>
</tr>
<tr>
<td></td>
<td>top</td>
<td>4.23M</td>
<td>4.34m</td>
<td>0.89m</td>
</tr>
<tr>
<td></td>
<td>Skel.</td>
<td>0.76m</td>
<td>3.34h</td>
<td>3.35h</td>
</tr>
<tr>
<td><strong>FOOT</strong></td>
<td>weak</td>
<td>killed after 25 hours</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>top</td>
<td>4.69M</td>
<td>2.89h</td>
<td>26.0m</td>
</tr>
<tr>
<td></td>
<td>Skel.</td>
<td>killed after 25 hours</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>LUCY</strong></td>
<td>weak</td>
<td>killed after 25 hours</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>top</td>
<td>14.0M</td>
<td>11.9m</td>
<td>14.8m</td>
</tr>
<tr>
<td></td>
<td>Skel.</td>
<td>23.19s</td>
<td>14.6h</td>
<td>14.6h</td>
</tr>
</tbody>
</table>
In Summary

We have briefly overviewed the **most common data structures** proposed in the literature.

**Future Directions:**

- **Express** the most frequently adopted operators *in terms of top simplices*

- Face the bottleneck concerning the **construction of a simplicial complex from a dataset** (maybe proposing an approximated construction?)

- Investigate with your help the connections between simplicial complexes (and the data structures to store them) and **itemsets in association rule learning**
Thank you!
Main References:

- **Simplex-based Representations:**
  - Boissonnat, Maria. *The Simplex Tree: an Efficient Data Structure for General Simplicial Complexes.* In Algorithmica, 2014

- **Top-based Representations:**

- **Operator-driven Representations:**

- **Comparisons:**
  - Fugacci, Iuricich, De Floriani. *Computing Discrete Morse Complexes from Simplicial Complexes.* In Graphical Models, 2019