Homology and cohomology reduction algorithms and applications.

Paweł Dłotko
Institute of Computer Science, Jagiellonian University,
Kraków, Poland.
What do people think about it?

"Topology! The stratosphere of human thought! In the twenty-fourth century it might possibly be of use to someone..."

The First Circle, A. Solzhenitsyn (found by Vin de Silva and Robert Ghrist)
Phenomena in many theories (far from topology) has a topological (co-homological) nature:

- Maxwell equations in electromagnetism,
- Coverage problem and hole detection in sensor network theory,
- Pattern and image recognition,
- Phase transition in physic,
- Hard ”regions” of NPC-problems in computability theory,
- ...
Introduction

How homology is computed in Cracow – the reduction algorithms.
How homology is computed in Cracow – the reduction algorithms.

- Pure mathematics – to compute (co)homology Smith Normal Form alg. suffices – requires only elementary matrix operations – easy and well established algorithm.
- Hypercubical complexity.
- In low dimensions 2 (Betti numbers + gens.) and 3 (Betti numbers when the input is suitable – Delfinado-Edelsbrunner, 1995) can be computed much faster.
- Higher up one can rely only on slow SNF computations.
How homology is computed in Cracow – the reduction algorithms.

**Question:** How to overcome the problem with subpercubical complexity?
How homology is computed in Cracow – the reduction algorithms.

Answer: Make its input as small as possible!
How homology is computed in Cracow – the reduction algorithms.

Idea! The same homology, different number of cells:

36 cells.

8 cells.
How homology is computed in Cracow – the reduction algorithms.

Elementary reductions by J. H. C. Whitehead:
How homology is computed in Cracow – the reduction algorithms.

Elementary reductions by J. H. C. Whitehead:
How homology is computed in Cracow – the reduction algorithms.

The main reduction algorithms used by us:

1. Elementary reductions (by J.H.C Whitehead),
2. Coreductions (by B. Batko and M. Mrozek),
3. Acyclic subspace method (by M. Mrozek, P. Pilarczyk, N. Żelazna),
The Coreduction homology algorithm.
The Coreduction homology algorithm.
The Coreduction homology algorithm.
The Coreduction homology algorithm.
The Coreduction homology algorithm.
Summary of elementary reductions & coreductions.

- In many cases elementary reductions + coreductions reduce the complex to the S-complex with zero boundary map.
- The reduction algorithms provide both Betti numbers and the homology generators.
- One can prove that in $\mathbb{R}^2$ do all the job.
- Numerical experiments show that in higher dimensions they also reduce a lot.
Acyclic subspace algorithm.

Elementary observations:

\[ H_n(X) = \begin{cases} 
  H_n(X, A) & n > 0, \\
  \mathbb{Z} \oplus H_n(X, A) & n = 0,
\end{cases} \]

when \( A \) is acyclic.
Acyclic subspace algorithm.
Acyclic subspace algorithm.

What’s left after acc. subspace being removed:
Acyclic subspace algorithm.

- No theorems saying how big percentage of the cells can be removed,
- In practical applications the reductions goes very deep.
Numerical experiments.

<table>
<thead>
<tr>
<th>Size</th>
<th>bbm</th>
<th>CR</th>
<th>AS</th>
</tr>
</thead>
<tbody>
<tr>
<td>28800</td>
<td>7.99</td>
<td>0.14</td>
<td>0.13</td>
</tr>
<tr>
<td>51200</td>
<td>15.2</td>
<td>0.23</td>
<td>0.20</td>
</tr>
<tr>
<td>115200</td>
<td>36.7</td>
<td>0.55</td>
<td>0.39</td>
</tr>
<tr>
<td>204800</td>
<td>65.0</td>
<td>0.97</td>
<td>0.63</td>
</tr>
<tr>
<td>320000</td>
<td>104.</td>
<td>1.56</td>
<td>0.96</td>
</tr>
<tr>
<td>Torus (dim=2, emb=3)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Size</th>
<th>bbm</th>
<th>CR</th>
<th>AS</th>
</tr>
</thead>
<tbody>
<tr>
<td>74341</td>
<td>16.7</td>
<td>0.44</td>
<td>0.27</td>
</tr>
<tr>
<td>132321</td>
<td>36.8</td>
<td>0.80</td>
<td>0.39</td>
</tr>
<tr>
<td>206901</td>
<td>71.5</td>
<td>1.27</td>
<td>0.61</td>
</tr>
<tr>
<td>298081</td>
<td>105.</td>
<td>1.80</td>
<td>0.84</td>
</tr>
<tr>
<td>405861</td>
<td>163.</td>
<td>2.52</td>
<td>1.23</td>
</tr>
<tr>
<td>Bing’s House (dim=2, emb=3)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Environment: 3.6 GHz Pentium 4, 2GB memory, Windows XP

- bbm - building boundary matrices
Numerical experiments.

<table>
<thead>
<tr>
<th>Size</th>
<th>bbm</th>
<th>CR</th>
<th>AS</th>
</tr>
</thead>
<tbody>
<tr>
<td>12522</td>
<td>14.</td>
<td>0.26</td>
<td>1.7</td>
</tr>
<tr>
<td>22280</td>
<td>36.</td>
<td>0.51</td>
<td>3.0</td>
</tr>
<tr>
<td>34830</td>
<td>86.</td>
<td>0.90</td>
<td>4.8</td>
</tr>
<tr>
<td>50172</td>
<td>184.</td>
<td>1.4</td>
<td>7.0</td>
</tr>
<tr>
<td>68306</td>
<td>326.</td>
<td>2.1</td>
<td>10</td>
</tr>
<tr>
<td>Klein bottle (dim=2, emb=4)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Size</th>
<th>bbm</th>
<th>CR</th>
<th>AS</th>
</tr>
</thead>
<tbody>
<tr>
<td>937</td>
<td>0.56</td>
<td>0.01</td>
<td>0.30</td>
</tr>
<tr>
<td>3745</td>
<td>3.1</td>
<td>0.07</td>
<td>1.9</td>
</tr>
<tr>
<td>8425</td>
<td>8.6</td>
<td>0.15</td>
<td>4.1</td>
</tr>
<tr>
<td>14977</td>
<td>18.</td>
<td>0.30</td>
<td>5.4</td>
</tr>
<tr>
<td>23401</td>
<td>39.</td>
<td>0.48</td>
<td>8.7</td>
</tr>
<tr>
<td>Projective plane (dim=2, emb=4)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Environment: 2.2 GHz Pentium 4, 1GB memory, Linux

- bbm - building boundary matrices
The representation problem.

- Usually the data for computing is given a priori.
- But sometimes, we are generating it ourselves.
- Some representations are more suitable for one kind of applications, another for others,
- For instance, not worth to triangulate the data obtained from some images (cubes are more feasible)
The representation problem.

- Selecting a good representation often results in decreasing the number of cells by a few orders of magnitude.

Simplicial versus Čech representation
The bibliography.


- B. Batko, M. Mrozek *Coreduction homology algorithm*, Discrete and Computational Geometry, 41(2009), 96-118.


Applications of computational (co)homology.

- **Part 1:** Homology of nodal sets using homology computations for regular CW-complexes.

- **Part 2:** The coverage problem in the sensor network.
Applications of computational (co)homology.

- **Part 3:** Where Poincare meets Maxwell, Discrete Geometric approach to Maxwell’s equations.
Homology groups of regular CW-complexes – computations of homology of nodal domains.

(joint work with Tomasz Kaczynski, Marian Mrozek and Thomas Wanner.)
Homology groups of nodal domains.

- High importance of *nodal domains* in applied science.
- For instance, for the function $u : \mathbb{R} \times \mathbb{R} \ni (t, x) \rightarrow u(t, x) \in \mathbb{R}$ the nodal domains are $N^+(t) = \{ x : u(t, x) \geq 0 \}$ and $N^-(t) = \{ x : u(t, x) \leq 0 \}$.
- An example of the nodal domain of a trigonometric polynomial $u$ presented in the previous slide.
- Studying the evolution of Betti numbers of nodal domains helps to understand the dynamical system induced by $u$ (papers by K. Mishaikow, T. Wanner, S. Day, W. Kalies).
Homology groups of nodal domains.

- Among the reasons which make the topic interesting are the, so called, phase separation phenomena.
- Suppose some object consists of two materials the atoms of which are well mixed.
- Roughly, when the temperature of the object drops down rapidly from some level, the two materials separate.
- This results in local clustering of one material inside another.
- In consequence, the mechanical properties of the object get worse.
- Such a situation typically happens in pipes guiding steam in power stations (their lifetime is shorter then expected).
Approach used so far.

- Lots of subdivisions have to be done close to the boundary of the nodal domain.
- So far, after the nodal domain was fixed, the subdivisions were done to obtain cubical complex with the cubes having equal sizes...
- ... so the size of the complex grows drastically
Approach used so far.

- Even if one is able to remove most of the cubes with elementary reductions, the whole process is still very costly.

- **Idea!** Compute the homology of the non uniform size cubical grid:
How to compute homology of non uniform size cubical grid?

- CW- complex.
- The existing software designed for cubical, simplicial homology.
- No code available for general CW-complexes.
- To maintain the computations (SNF) the incidence numbers of cells are essential.
- In the standard theory the incidences are given either by Brouwer degree of some continuous function or by some relative homology of cells (triangulation of cells required!)
- Both approaches hardly feasible.
- Do we face the wall then?
Regular CW-complexes. Trade between computability and generality.

- Subclass of all CW-complexes for which computation of the incidence is easy exists (regular CW-complexes).
- This subclass was introduced by William Massey in "Basic course in Algebraic Topology".
- An n-cell $e^n$ is attached to the closed subset $K$ of the Hausdorff space if there exists a continuous map $f : B^n_1(0) \rightarrow e^n$, called the characteristic map, such that $f$ maps the open ball $B^n_1(0)$ homeomorphically onto $e^n$ and $f(\partial B^n_1(0)) \subset K$.
- A CW-complex is said to be regular if for each cell $e^n$, where $n = 1, \ldots, N$, there exists a characteristic map $f : B^n_1(0) \rightarrow \overline{e^n}$ which is a homeomorphism.
Regular CW-complexes. Trade between computability and generality.

- For the regular CW-complexes the incidence numbers have the following properties:
  
  (a) If $e_{\mu}^{n-1}$ is not a face of $e_{\lambda}^{n}$, then $\alpha_{\lambda\mu}^{n} = 0$.
  
  (b) If $e_{\mu}^{n-1}$ is a face of $e_{\lambda}^{n}$, then $\alpha_{\lambda\mu}^{n} = \pm 1$.
  
  (c) If $e_{\mu}^{0}$ and $e_{\nu}^{0}$ are the two vertices of the 1-cell $e_{\lambda}^{1}$, then
  \[ \alpha_{1\mu}^{1} + \alpha_{1\nu}^{1} = 0. \]
  
  (d) Let $e_{\lambda}^{n}$ and $e_{\rho}^{n-2}$ be two cells of $K$ such that $e_{\rho}^{n-2}$ is a face of $e_{\lambda}^{n}$. Furthermore, let $e_{\mu}^{n-1}$ and $e_{\nu}^{n-1}$ denote the unique $(n-1)$-cells $e_{\lambda}^{n-1}$ such that $e_{\rho}^{n-2} \subset e_{\lambda}^{n-1} \subset e_{\lambda}^{n}$. Then the identity
  \[ \alpha_{\lambda\mu}^{n} \alpha_{\mu\rho}^{n-1} + \alpha_{\lambda\nu}^{n} \alpha_{\nu\rho}^{n-1} = 0 \]
  holds.

- Using these properties it is possible to create a linear time algorithm which computes the incidences.
Regular CW-complexes. Trade between computability and generality.

- The family of regular CW-complexes is wide enough to cover all, known to us, practical applications.
- The nonuniform size cubical complexes are regular CW-complexes.
- Also in the mathematical modeling in electrical engineering the octahedron meshes are very popular.
- Soon the CAPD (Computer Assisted Proofs in Dynamics, capd.ii.uj.edu.pl) software will provide the functionality of computing the (co)homology groups of arbitrary regular CW-complexes.
Regular CW-complexes. Trade between computability and generality.

- So far the code was implemented and tested for 2-dimensional nodal domains.
- In this case no algebraic computations are necessary to get homology, elementary reductions + coreductions do all the job.
- When compared with the coreduction algorithm based on regular size grid, a speed up of the order of magnitude ranging from two to three was observed.
- The CW complex is also highly efficient with respect to memory usage. Now we are able to get homology of examples which were far beyond the reach of the previous method due to lack of memory.
Speed up with respect to previously fastest algorithm and the average time of the algorithm.
The bibliography.


P.D, T. Wanner *Code for computing homology of planar nodal domains*, online soon...


The coverage problem in the sensor network.

Part 2,
Homology computations on the sensor network,

(joint work with Robert Ghrist, Mateusz Juda and Marian Mrozek.)
Formulation of the problem.

- The importance of the coverage problems in the sensor networks (security, mine field sweeping, ad hoc communication networks).
- Systems with stationary nodes in $\mathbb{R}^2$ with radially symmetric coverage domains.
- Verify if the given set $D$ such that $\partial D = F$ is covered by the disks of a given radius centered in the sensors.
How the problem was solved so far?

- Computational geometry approach (Delaunay or Voronoi diagrams).
  - Precise geometry of the domain,
  - known localization of the nodes,
  - then we have exact solution.
- Probabilistic methods.
  - Big enough number of the sensors,
  - uniformly distributed on the region
  - implies coverage of the region with high probability (but no guarantee).
- What we wish to have?
  - Deterministic Yes/No or Yes/No/Do_Not_Know algorithm
  - arbitrary geometry of the region,
  - no assumptions about the placement of the sensors.
Problem of computing the homology groups of a union of sets is classical (nerve of the cover).

Let \( \{X_1, \ldots, X_n\} \) be the points where sensors are placed.

Let \( r_{com} \) be the radius of communication of a sensor.

The Rips complex associated with the set \( X = \{X_1, \ldots, X_n\} \) is a simplicial complex whose simplices are subsets \( V \subset X \) such that for each \( X, Y \in V \), \( d(X, Y) \leq r_{com} \).
Each sensor has some radius of coverage $r_{cov}$.

Putting $r_{cov} \geq \sqrt{\frac{1}{3}} r_{com} \approx 0.577 r_{com}$ enables us to ”fill all the gaps”.

The radius of a cover.
Example of Rips complex (see V. Silva, R. Ghrist [3])
The homological criterion

- The coverage criterion based on the relative homology of the Rips complexes.

**Theorem (Silva, Ghrist, 2006)**

*If there exists a homology class $[\alpha] \in H_2(\mathcal{R}_{r_{com}}, \mathcal{F}_{r_{com}})$ such that $\partial \alpha \neq 0$, then the considered region is covered by the balls of a radius $r_{cov}$ centered in the given sensors.*

- To use this (partial) criterion we need to compute the second relative homology generator...
- check its boundary...
- When this criterion holds, some sensors can be turned off (power-saving mode).
Rips complexes in the coverage problem

- $\mathcal{R}_{r_{com}}$ denotes the Rips complex based on the set of distributed sensors and radius $r_{com}$.
- The fence complex $\mathcal{F}_{r_{com}}$ is a subcomplex of $\mathcal{R}_{r_{com}}$.
- A geometric realization of the fence complex is assumed to be a Jordan curve in the plane.
- The Silva-Ghrist criterion is used to check the coverage of the interior of this curve.
What we should do to check this criterion

- The sensors need to be distributed in the given set.
- The fence sensors have to know that they belong to the fence.
- Each sensor has to get to know all its neighbors (it suffices to construct the Rips complex).
- Construct the Rips complex.
- Compute relative homology of the complex.
- Last two things can be done in theory after gathering all the information from all the sensors into one computer...
- ...which is not easy and not effective for a real-sized complexes.
Idea

So maybe we can try to make these computations in a distributed way on the sensors themselves?

- Doing algebraic homology computations in the distributed way is not easy.
- Why not to use the reduction procedures, that can be implemented in a purely combinatorial way?
- In some cases the (co)reductions can reduce the complex up to the homology generators.
What kind of sensors do we need to maintain the parallel computations?

- Each sensor is identified by unique integer label and equipped with a processor and radio transmitter.
- Each sensor can read the labels of its neighbors lying not farther than $r_{com}$ (the communication range).
- The sensor can communicate with the sensors lying not farther then $r_{com}$.
- The sensor has the radius of detection $r_{cov}$ such that $r_{cov} \geq \sqrt{\frac{1}{3}} r_{com}$.
- The sensor does not have any metric information (neither coordinates, nor distance, nor direction to other sensors known).
What kind of sensors do we need to maintain the parallel computations?

- Each sensor can keep the list of simplices.
- Each simplex $S$ from the list can keep the following information:
  - Labels of all the sensors that belong to $S$,
  - labels of all the sensors that do not belong to $S$ and are in the communication range with all sensors that belong to $S$,
  - lists of faces and cofaces of the simplex.
- For the efficiency and simplicity we need to know exactly in which sensors each simplex from the Rips complex should be kept.
How the Rips complex can be effectively (without redundancy) represented in the sensor network?

- Each sensor has a unique integer label.
- Let \([X_1, \ldots, X_n]\) be the representation of the simplex \(S\) in the Rips complex with sensor’s labels.
- Convention – if the dimension of a simplex \(S\) is greater or equal 1, then the information about \(S\) is kept in the two sensors having the lowest labels among \([X_1, \ldots, X_n]\).
- 0-dimensional simplices are kept only in one sensor.
- Convention – all the decisions about the simplex \(S\) are taken by the sensor with the smallest label in \(S\).
- So for each simplex \(S\) all sensors that ”rule” the boundary of \(S\) have the information about \(S\).
Construction of the Rips complex by the sensors alone

- Once the sensors are distributed, they need to communicate to get to know the labels of all its neighbors.
- Moreover every sensor needs to know the labels of the neighbors of its neighbors.
- No more communication is needed to construct the Rips complex.
- The communication graph is a 1-skeleton of the Rips complex we are looking for.
- From the presented convention each 1-simplex is kept in the both of its endpoints.
Construction of the Rips complex by the sensors...

- Each simplex $S$ keeps the list of the sensors $\{X_1, \ldots X_n\}$ that are neighbors of each sensor involved in $S$.
- $S \cup X_i$ for $i \in \{1, \ldots n\}$ is a simplex in a Rips complex we are creating.
- We have to create the simplex $S \cup X_i$ in the right sensors to follow the given convention.
- The simplex is created in a sensor if the sensor is in charge of $S$ in the following two situations:
  - The label of $X_i$ is greater than any label of the sensor in $S$.
  - The label of $X_i$ is smaller than any label of the sensor in $S$.
- In this way all the simplices that belongs to the Rips complex will be created,
- each simplex will be created by at most two sensors, and only one of them will be in charge of the simplex.
- The presented procedure can be easily implemented on the parallel architecture like the sensor network.
Example.
Fence reduction procedure

- Silva-Ghrist’s criterion needs relative homology group generators $[\alpha] \in H_2(\mathcal{R}_{r_{\text{com}}}, \mathcal{F}_{r_{\text{com}}})$.
- Computation of relative homology is needed.
- The fence complex $\mathcal{F}_{r_{\text{com}}}$ is a closed subcomplex of the complex $\mathcal{R}_{r_{\text{com}}}$.
- In this case $H_2(\mathcal{R}_{r_{\text{com}}}, \mathcal{F}_{r_{\text{com}}})$ is isomorphic to $H_2(\mathcal{R}_{r_{\text{com}}} \setminus \mathcal{F}_{r_{\text{com}}})$.
- So we simply remove the 0 and 1 dimensional simplices that belong to the fence from the constructed Rips complex.
- This can be done by the sensors themselves if each sensor knows if it belongs to the fence or not (we assume they know that).
(Co)reduction algorithms

- The created Rips complex is usually a really big structure.
- The simplices of arbitrary dimension can appear in it.
- This is not a good input for the homology computation algorithm, we need to reduce the input as far as possible.
- To do so the combination of the elementary reduction and coreduction algorithm will be used.
(Co)reduction algorithms

- Elementary reduction – the retraction procedure:

- The (co)reduction algorithm is a kind of deal between a simplex and its (co)boundary.

- But from the construction of the Rips complex we know, that the sensors that have the information about a simplex $S$ also have the information about its boundary.

- Once the technicalities are fixed the sequence of elementary and co-reductions works perfectly.
How far can we reduce the complex? Do we need algebraic computations?

Conjecture

*For the relative homology of a Rips complex built on the basis of the planar point set the sequence of elementary and co – reductions reduce the complex up to its homology generators.*

- By the generator in the reduced complex we mean the simplex without boundary elements.
- The idea behind this – even if the dimensions of simplexes in the Rips complex are high, the topology of it almost ”planar” (in planar case the reductions can do all the job) ...
- this is what we see from the numerical experiments.
- Even if the Conjecture is not true, the complex that remains after (co)reductions is very small...
- and can be gathered by one of the sensors and the homology can be computed.
Can this work? Can we prove the algorithm is correct?

- In the papers dealing with elementary and co-reductions only proofs of the correctness of the sequential algorithms are provided.
- We are using a parallel version of these algorithms.
- The parallel execution of the presented algorithm is equivalent to certain execution of the sequential algorithm.
- We demonstrate that the maps induced in homology by the parallel execution are isomorphic to the maps induced by the sequential algorithm...
- ... so both parallel (co)reductions and pulling back the generator are correct algorithms.
The bibliography for this section.


- CHOMP, [http://chomp.rutgers.edu](http://chomp.rutgers.edu).
Where Maxwell meets Poincare.

Part 3,
Computational homology in electromagnetism, where Poincare meets Maxwell.

(joint work with Ruben Specogna and Francesco Trevisan.)
Modelling of real-world objects

- Tetrahedral mesh used in modelling real-world objects.
- Discretization of the Maxwell’s equations used, computed values constant for each simplex in the mesh.
- Maxwell’s equations can be formulated by using only (co)chains and (co)boundary operator [2], [5].
- In such approach from very beginning theory works on the discrete level. No approximation of continuous mathematical operators needed.
- $T - \Omega$ formulation uses this approach. However it requires some prior knowledge about the topology of the mesh.
What do electrical engineers need the (co)homology for?

- Two regions present in the meshes:
  - Conducting region, $D_c$.
  - Non-conducting (air) region, $D_a$.
- Assumption: $D_a \cup D_c$ is acyclic.
- Extra topological information needed to enforce the Ampere's law in $D_a$.
- Ampere's law in the classical physics—integrated magnetic field around a closed loop is related to the electric current passing through any surface the boundary of which is the loop.
- Discrete level:
  - Cycles in $D_a$ instead of the closed loops.
  - Chains having the cycle as a boundary instead of the surfaces.
  - The magnetomotive force (this is what we are looking for) $d$ is a cochain.
  - The integral of a magnetic field over some chain $c$ is the evaluation $\langle d, c \rangle$. 
Non-trivial topology appears

- If $D_a$ – topologically trivial, then each cycle is a boundary. Electric current passing through the boundary is zero (the current can be driven only by elements of $D_c$).
- It is getting more interesting when $D_a$ is homologically non-trivial (nonzero $\beta_1$).
- If $c_1$ in $D_a$ is not a boundary in $D_a$, then each chain $S$ in $D_a \cup D_c$ such that $\partial S = c_1$ needs to ”intersect” $D_c$.
- The information about cycles that are not boundaries in $D_a$ is needed. The basis of such cycles is precisely the basis of the homology group $H_1(D_a)$.
Topologically nontrivial $D_a$
What is the topological information we need, though? How the magnetomotive force should be defined?

- Standard approach (used so far in the electrical engineering) — cochain called *thick cut* has to be found to make the $T - \Omega$ formulation computations in the homologically nontrivial regions possible.

- For example in case of simple torus the *thick cut* $d$ is defined in such a way that for each cycle $c$ that ”goes n times around the branch of a torus” we have $| < d, c > | = n$. 
GSTT, the standard algorithm used by the engineers to construct thick cut

- To obtain thick cut it is enough to enforce the evaluation of the cochain on all the representatives of homology generators, and enforce zero evaluation on all boundaries (once this is done, all other cycles are spanned "automatically").
- There exists a standard algorithm to compute the thick cut (representatives of homology generators needed at the input).
- So called belted tree is constructed at the beginning by the algorithm.
- $h_1, \ldots, h_{\beta_1(D_a)}$ – representatives of the $H_1(D_a)$ basis.
- A belted tree $T$ in $D_a$ – a maximal spanning tree $T'$ with the set of edges $\{E_i\}_{i=1}^{\beta_1(D_a)}$ such that only cycle in $T' \cup E_i$ is a support of $h_i$. 
When a belted tree \( T = T' \cup \{ E_i \}_{i=1}^{\beta_1(D_a)} \) is given the following algorithm is called for each \( i \in \{1, \ldots \beta_1(D_a)\} \).

For every edge \( E \in T' \) put \( < c_i, E > = 0 \) and \( < c_i, E_j > = \delta_{ij} \) for \( j \in \{1, \ldots \beta_1(D_a)\} \).

Let \( L \) be the list of all 2-simplexes in \( D_a \).

\[ \text{while } L \text{ is not empty} \]

\[ \text{Take } T \in L \]

\[ \text{if the cochain } c_i \text{ is defined for all edges in } \partial T, \text{ then if } < c_i, \partial T > \neq 0, \text{ return FAILURE, else } L := L \setminus T. \]

\[ \text{if the cochain } c_i \text{ is not defined for a single edge } E \text{ in the boundary of } T, \text{ then define the value of } < c_i, E > \text{ to have } < c_i, \partial T > = 0 \text{ and } L := L \setminus T. \]
How does the *GSTT* work?
The belted tree
The initial value of a cochain
Which 2-simplexes can be removed from the list $L$ in the first step of the algorithm?
Which 2-simplexes can be removed from the list $L$ in the second step of the algorithm?
Output
Output...
Combining of the GSTT algorithm with the computational homology.

- The basis of $H_1(D_a)$ is needed to run the presented algorithm.
- So far – computed by pure Smith Normal Form or chosen ”by hand” (only toy examples could have been computed).
- GSTT algorithm using the homology generators provided by the CAPD software was implemented by P.D., Ruben Specogna and Francesco Trevisian in [2].
- Fast reduction methods from CAPD open the field of computations for far bigger meshes than it was possible before.
- However the algorithm leaves some open questions.
  - Does the algorithm always converge?
  - Can it return FAILURE for some input?
  - What is the output from the mathematical point?
Let’s try to answer the question ”What is the output of the algorithm?”

We assume that the while loop has terminated and no FAILURE was returned by the algorithm.
A few simple lemmas

1. If $D_a \subset \mathbb{R}^3$ is compact and locally contractible then $H_i(D_a, \mathbb{Z})$ is 0 for $i \geq 3$ and torsion–free for $i = 0, i = 1$ and $i = 2$.

2. In a simplicial complex $D_a$, the evaluation of 1–cocycle $c \in Z^1(D_a)$ on a trivial 1–cycle $d \in B_1(D_a)$ is zero.

3. A 1–cochain $c$ is a 1–cocycle if and only if for every 2–simplex $S$, $\langle c, \partial S \rangle = 0$.

4. Conclusion– to define a 1-cocycle $c$, it is enough to impose the zero evaluation of $c$ in the boundary of all 2-simplexes. This is exactly what takes place in the GSTT algorithm. So the GSTT algorithm returns a cochain.
A few more simple lemmas...

- For two 1–cycles $d_1$ and $d_2$, which differ by a boundary, and a 1–cocycle $c$ we have $\langle c, d_1 \rangle = \langle c, d_2 \rangle$.

- Once the evaluation of a 1–cocycle $c \in C^1(D_a)$ is known over cycles $\{h_i\}_{i=1}^{\beta_1(D_a)}$ representing the basis of $H_1(D_a)$, the value of the evaluation of the cocycle $c$ over any 1–cycle is determined.

- In the GSTT algorithm the evaluation of a cocycle is fixed over the $H_1(D_a)$ basis. As a result it is fixed on all the cycles in the complex.
The Universal Coefficient Theorem for cohomology.

Theorem (See [1], Th. 3.2)

If a simplicial complex $D_a$ has (integer) homology groups $H_n(D_a)$, then the cohomology groups $H^n(D_a; G)$ are determined by split exact sequences

$$0 \rightarrow \text{Ext}(H_{n-1}(D_a), G) \rightarrow H^n(D_a, G) \xrightarrow{h} \text{Hom}(H_n(D_a), G) \rightarrow 0.$$ 

- Theorem will be used for $n = 1$ and $G = \mathbb{Z}$.
- I do not want to define the $\text{Ext}$ functor, it is enough to know, that $\text{Ext}(H_{n-1}(D_a), \mathbb{Z}) = 0$ if $H_{n-1}(D_a)$ is a free group. It is so in case of $n = 1$.
- So from the exactness of the sequence we have the isomorphisms: $h : H^1(D_a, \mathbb{Z}) \rightarrow \text{Hom}(H_1(D_a), \mathbb{Z})$. 
How is the map $h$ defined?

- $h : H^1(D_a, \mathbb{Z}) \rightarrow \text{Hom}(H_1(D_a), \mathbb{Z})$.
- For a class $[\psi] \in H^1(D_a, \mathbb{Z})$ we have $0 = \langle \delta \psi, z \rangle = \langle \psi, \partial z \rangle$, since $\psi$ is a cocycle. Consequently $\psi|_{B_1(D_a)} = 0$.
- Let us define the restriction $\psi_0 = \psi|_{Z_1(D_a)}$. Since $\psi_0|_{B_1(D_a)} = 0$, $\psi_0 \in \text{Hom}(H_1(D_a), \mathbb{Z})$. $\psi_0$ is well defined, since the value of the $\psi$ is the same for all representatives of the same $H_1(D_a)$ class.
- We define $h([\psi]) = \psi_0 \in \text{Hom}(H_1(D_a), \mathbb{Z})$. 
What is the output after all?

The output of the *GSTT* algorithm is a the basis of $H^1(D_a)$ group.

- $\{h_i\}_{i=1}^{\beta_1(D_a)}$ – representation of $H_1(D_a)$ basis (the same as the one at the input of the algorithm).
- Take $\phi_i \in Hom(H_1(D_a), \mathbb{Z})$ such that $\phi_i([h_j]) = \delta_{ij}$.
- Since $H_1(D_a)$ – free group, $\{\phi_i\}_{i=1}^{\beta_1(D_a)}$ is a basis of $Hom(H_1(D_a), \mathbb{Z})$.
- Since $h$ is an isomorphism, $\{h^{-1}(\phi_i)\}_{i=1}^{\beta_1(D_a)}$ forms the basis of $H^1(D_a, \mathbb{Z})$.
- The cocycles $\{c_i\}_{i=1}^{\beta_1(D_a)}$ obtained by the *GSTT* algorithm are in the same cohomology class as the cocycles $\{h^{-1}(\phi_i)\}_{i=1}^{\beta_1(D_a)}$.
- Consequently $\{c_i\}_{i=1}^{\beta_1(D_a)}$ forms a basis of $H^1(D_a, \mathbb{Z})$. 
The current work.

- The output of GSTT is the $H^1(D_a, \mathbb{Z})$ basis.
- In some cases GSTT return FAILURE,
- its performance is in fact equivalent to the performance of coreduction algorithm – but of course we know, that in general we cannot reduce everything,
Possibly applications of what we are doing:

- Rapid and cheap design of practical devices together with their optimization,
- Non-destructive testing (IP),
- induction heating,
- proximity sensors (IP),
- metal detectors (IP),
- hyperthermia cancer treatment (IP),
- ITER! (so far impossible to get the mesh).
The bibliography.

A. Hatcher, *Algebraic topology*.


The end.

Thank you for your attention!