TOPOPROCESSOR: An Efficient Computational Topology Toolbox for *h*-Oriented Eddy Current Formulations

Paweł Dłotko[1], Bernard Kapidani[2], and Ruben Specogna[2]

[1]DataShape, Inria Saclay, Palaiseau 91120, France
[2]Polytechnic Department of Engineering and Architecture, Università di Udine, 33100 Udine, Italy

When solving eddy-current problems containing topologically non-trivial conductors with formulations using the magnetic scalar potential in the insulators, cohomology generators are necessary to obtain a well-defined problem. The Dłotko–Specogna (DS) algorithm is a simple and efficient tool to compute the lazy generators of the first cohomology group of the insulator that can be used in such potential design. This paper introduces an upgrade in the DS algorithm that speeds up the execution for very complicated geometries. Moreover, this paper provides, for the first time, a detailed comparison of computational resources needed for the topological pre-processing by our toolbox and by the tool to compute a standard cohomology basis available in the mesh generator GMSH. In addition, we make our implementation of DS algorithm available for download on request for non-profit use as a TOPOPROCESSOR package.

Index Terms— **Cohomology, computer aided software engineering, cuts, eddy currents, magnetic scalar potential, mathematical software.**

I. INTRODUCTION

WHEN solving an eddy current problem with the magnetic scalar potential in geometries containing topologically nontrivial conductors, representatives of first cohomology group generators of the insulator are the only objects that make the problem well defined [1]–[5]. Before this fact was recognized, for more than two decades, the computational electromagnetics community provided various—in general incorrect—ways of defining and computing *cuts* for that purpose. In the last years, the effort in promoting the use of cohomology theory among engineers and numerical analysts brought up new approaches to compute cohomology generators. Notably, general algebraic methods based on the reduction of the input complex followed by the computation of the Smith normal form (SNF) of the boundary matrix have been introduced in the electromagnetic context in [6], and have been efficiently implemented in the mesh generator GMSH, [7]. Concurrently, a radically novel class of physics-inspired algorithms as the Dłotko–Specogna (DS), tailored specifically for electromagnetic problems, has been introduced in [8] and [9].

The DS algorithm produces a so-called *lazy* cohomology basis [8], [9]: it gives a collection of cocycles that generate the cohomology group, but contains additional, dependent cocycles. The size of the lazy basis provided by the DS algorithm is no more than twice the size of a standard cohomology basis. With moderate effort, one can produce a standard cohomology basis [8] given a lazy one. However, it has been verified that this technique produces the same results up to the solver tolerance, while it does not provide any speedup in the solution of the electromagnetic problem. A formal and detailed proof of the correctness of the DS algorithm and its validation inside a FEM-like software may be found in [8].

Manuscript received November 16, 2016; accepted January 24, 2017. Date of publication January 31, 2017; date of current version May 26, 2017. Corresponding author: R. Specogna (e-mail: ruben.specogna@uniud.it).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMAG.2017.2661480

The original DS algorithm internally uses the *extended spanning tree technique* (ESTT) [10], which is a general version of the Webb–Forghani (WF) iterative algorithm [11] (recalled in detail in Section II-A.) to obtain a field whose curl is assigned. In very rare cases, the WF algorithm does not terminate [12] and therefore, the ESTT uses a different, slower technique. To solve this issue, in this paper we introduce an improvement to treat the case when the WF algorithm does not terminate, therefore producing a faster solution while maintaining full robustness.

The improved DS algorithm is implemented in the TOPOPROCESSOR toolbox presented in this paper. Given the mesh of the computational domain (i.e., the union of conductors and insulators), TOPOPROCESSOR computes the lazy generators of the first cohomology group of the insulating region with the modified DS algorithm. We assume, as always happens in eddy current computations, that the computational domain is topologically trivial.

This paper is structured as follows. In Section II, we survey the DS algorithm and explain how we modify the ESTT to render it faster when it fails to terminate by using the WF algorithm only. Section III presents the numerical results on three industrial-sized benchmarks, considered not addressable just a few years ago. In particular, for the first time, we compare the computational resources that GMSH and TOPOPROCESSOR need to perform the topological pre-processing. Finally, in Section IV, the conclusions are drawn.

II. NOVEL DS ALGORITHM

The material presented in this paper strongly relies on concepts from algebraic topology that due to the limited space cannot be reproduced here. Please consult [13] for a formal introduction or [2], [5], [8] for an informal one.

Let us assume that the computational domain is topologically trivial and covered with a simplicial complex $\mathcal{K} = \mathcal{K}_a \cup \mathcal{K}_c$, where $\mathcal{K}_a$ and $\mathcal{K}_c$ are subcomplexes representing the insulating and conducting regions, respectively.

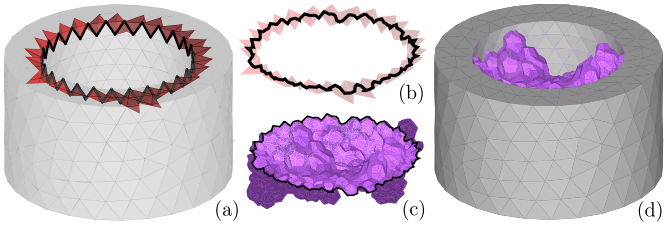

Fig. 1. (a) Thick edges represent the support of a cohomology generator of the boundary of $\mathcal{K}_c$, a solid 2-torus. The dark triangles represent the support of the thinned current. (b) Dual edges which are dual to thinned current faces form a 1-cycle $\tilde{c}$ on the dual complex. (c) This dual 1-cycle $\tilde{c}$ is the boundary of a 2-chain on the dual complex. (d) Such 2-chain, restricted to $\mathcal{K}_a$, is the dual of the cohomology generator of $\mathcal{K}_a$.

The DS algorithm finds the lazy generators of the first cohomology group $H^1(\mathcal{K}_a)$ in the following way.

1) First, the discrete surface $\mathcal{S} = \mathcal{K}_c \cap \mathcal{K}_a$ [see Fig. 1(a)] is extracted and its first cohomology group generators are computed with a combinatorial algorithm, which first constructs two interdigitated spanning trees on the primal and dual edges of $\mathcal{S}$ and then extracts dual cycles closed by edges that do not belong to either of the trees. The worst case time complexity of this algorithm is linear. See [8] for additional details.
2) *Thinned currents* are found by pre-multiplying the generators of $\mathcal{S}$ by the incidence matrix $\mathbf{C}_c$ between face and edge pairs [8] restricted to $\mathcal{K}_c$. The support of one of the thinned currents of a toric conductor is represented in Fig. 1(a) by dark faces.
3) Finally, a vectorialized version of the ESTT algorithm [10] is run on the whole complex $\mathcal{K}$ for all thinned currents at once. The ESTT algorithm is a general version of the WF iterative algorithm [11] (described in detail in the following) to obtain a discrete field whose discrete curl is assigned (in our case to the curl is the thinned current). Its typical complexity is linear (obtained for all tested practical problems), even though there is no theoretical proof yet that its worst case complexity is better than cubical. The output of the ESTT restricted to $\mathcal{K}_a$ form the required lazy cohomology generators [Fig. 1(d)].

The dual of a thinned current forms a 1-cycle $\tilde{c}$ on the dual complex [see the thick edges in Fig. 1(b)]. The ESTT is thus computing a (possibly self-intersecting) discrete surface on the dual complex having $\tilde{c}$ as boundary [see Fig. 1(c)].

A. Proposed Improvement of the ESTT Algorithm

The WF algorithm in [11] first finds a spanning tree in the domain, typically with linear-time breadth-first search (BFS), and sets the values on the tree edges to zero. The values on the cotree edges are found by considering only the faces whose boundary has the value set for all but one edges. The value of the remaining edge is then set by enforcing the discrete Ampére's law on that face. The circulation is enforced to be zero on all faces except the ones in the support of the thinned currents, where the circulation is determined by the thinned current value on those faces.

If the WF algorithm sets the value of all edges in $\mathcal{K}$, we have the lazy cohomology generators in linear time worst case complexity. However, it has been shown in [12] that this algorithm may get stuck before having set all edge values (even in extremely simple examples). Luckily, it works in the great majority of practical cases—hence, if terminating, exhibits a linear complexity—but one should be prepared in case the algorithm does not terminate.

When the WF algorithm gets stuck instead, one way to get out of the stall is by determining the value of a not-yet-set edge with another technique and later eventually continue with the standard iterations of the WF algorithm. Various ways to do so have already been proposed in the literature.

1) Continue the ESTT algorithm [10] by assuming that a value of a not-yet-set edge E is set to a (unknown) value v_E. The values of the edges that are determined thereafter will be linear functions of v_E. At the end, after solving a small system of equations, we obtain all v_E values and, consequently, find the cocycle.
2) Use a computationally prohibitive technique based on solving the remaining integer system explicitly with a linear solver.

In this paper, we instead propose an alternative solution, inspired from [14], which is efficient and easy to implement.

1) Pick any edge E whose value is not yet set. Edge E closes a cycle c_E in the spanning tree. The cycle can be found by tracking two paths from the endpoints of E to the root of the BFS spanning tree.
2) From the Ampére's law and the fact that $\mathcal{K}$ is trivial, the value of E may be obtained by computing the linking number between c_E and the 1-cycles on the dual complex dual to thinned currents [see $\tilde{c}$ in Fig. 1(b)].

B. Advantages of the DS Algorithm

The DS algorithm, due to its localized and combinatorial nature, has various advantages over competing ones.

1) It is easy to implement in every in-house finite element-like software given that it is based on spanning trees and does not require matrix operations over integers.
2) It outperforms standard approaches based on reduction and SNF [6], [7] both in memory consumption and computation time. Its superiority is clearly visible on the benchmarks presented in Section III.
3) The DS algorithm by design performs a basis selection before the generators are constructed. From the DS algorithm, we get always two lazy generators per each active torus-shaped coil. By using the technique introduced in [8, Appendix], one may eliminate one of the two generators and thus find the one that generates the first cohomology group of the coil complement. This way we guarantee that the current flowing in each torus-shaped active coil is in one-to-one correspondence with a generator, which eases the enforcement of sources and the coupling with electric circuits. More details and an example concerning automatic basis selection are given in Section III-B.

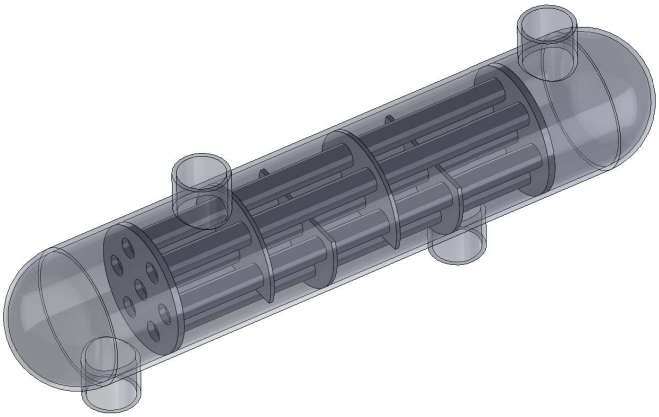

Fig. 2. Geometry of the shell and tube heat exchanger. The boundary of the heat exchanger is a combinatorial surface of genus 35.

TABLE I
HEAT EXCHANGER BENCHMARK

	Mesh 1	Mesh 2
Number of tetrahedra in $\mathcal{K}$	1 604 144	10 445 468
Number of tetrahedra in $\mathcal{K}_a$	1 233 030	8 820 579
Meshing time GMSH [s]	45.4	335
t_{GMSH} [7] [s]	152	9061
t_{TP} [s]	15.2	134
Speedup	$\times$ 10	$\times$ 68

III. NUMERICAL RESULTS

To demonstrate the potential of the DS algorithm, we perform the topological preprocessing required by the *h*-oriented eddy current formulations in three industrial test cases. In the following, t_{TP} denotes the total wall time (in seconds) needed by TOPOPROCESSOR, whereas t_{GMSH} represents the total time required by GMSH. All computations with the TOPOPROCESSOR are performed on a laptop with an Intel Core i7-3720QM processor clocked at 2.60 GHz and 16 GB of RAM. Due to memory consumption, for very big examples, GMSH requires a larger workstation with 256 GB of RAM (times in this case are represented inside brackets).

A. b1: Heat Exchanger

Maintenance of shell and tube heat exchangers is routinely performed with eddy current non-destructive testing. The first test comprises the complement of a heat exchanger (see Fig. 2) with respect to a box. The time required to perform the topological preprocessing with the two competing approaches on two meshes produced with GMSH is represented in Table I together with some information on the meshes.

B. b2: Magnetic Induction Tomography

MIT, also known as eddy current non-destructive testing, typically uses an array of coils that surrounds a conductive rod to be inspected (see Fig. 3). An inverse problem has to be solved to assess the presence of defects in the rod, which requires fast methods to solve the forward problem, i.e., computing eddy currents with a given distribution of conductivity inside the domain.

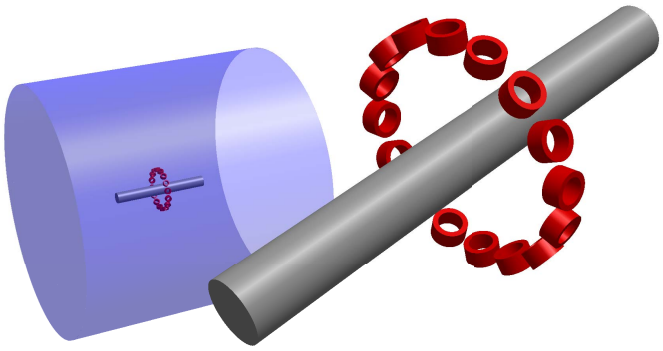

Fig. 3. Typical MIT Device. It Is Composed by 16 Toric Coils Which Surround the Conductive Rod Being Analyzed

TABLE II
MIT BENCHMARK

	Mesh
Number of tetrahedra in $\mathcal{K}$	7 976 328
Number of tetrahedra in $\mathcal{K}_a$	7 484 064
Meshing time NETGEN [s]	210
t_{GMSH} random basis [s]	(1967)
t_{GMSH} with basis selection [s]	(25 600)
t_{TP} [s]	77.8
Speedup	$\times$ 329

The classical way to enforce the current in a torus-shaped coil is to fix (by usual boundary conditions techniques) the degree of freedom corresponding to a cohomology generator (i.e., the *independent currents* in [8]). This is possible only if a cohomology generator is in one-to-one correspondence with a toric coil. This means that, in this application, not all cohomology basis are useful, and therefore, a basis selection has to be performed. A nice feature of the DS algorithm is that by design it produces a basis suitable for imposing current sources. This is due to the generators being computed from the boundary of each conductor independently.

On the contrary, there is no known mean to include natively this basis selection in algebraic methods as the ones used inside GMSH. A possible, but costly, way out is to perform computations separately for each coil, inspired from [15]. To be precise, in the jth computation just the jth coil is considered as conductor, whereas all others are considered as insulator. Table II contains the computational time of GMSH (both in case of a random first cohomology basis given by the software and with the described basis selection) and TOPOPROCESSOR.

C. b3: ITER-Like Nuclear Fusion Reactor

The last test considers the complement of the conductive structures of an ITER-like nuclear fusion device [see Fig. 4(a)] with respect to a box which represents the insulating region. The conductor is formed by gluing together 18 structures as the one in Fig. 4(b). Here, electromagnetic solvers play a fundamental role in the estimation of the electromagnetic forces produced by eddy currents that flow in the presence a strong magnetic field that is required for plasma confinement.

Fig. 4. (a) Geometry of the considered conductive structures of an ITER-like nuclear fusion reactor. The boundary of the conductive structures has two connected components. The first is of genus 1 and the other is of genus 1620. (b) Detail of the geometry of 1/18 of the conductive structures of the ITER-like nuclear fusion reactor.

TABLE III
NUCLEAR FUSION REACTOR BENCHMARK

	Mesh
Number of tetrahedra in $\mathcal{K}$	26 648 351
Number of tetrahedra in $\mathcal{K}_a$	13 225 740
Meshing time GMSH [s]	4684
t_{GMSH} [s]	(61 562)
t_{TP} [s]	493
Speedup	$\times$ 125

The number of mesh elements and the topological features render the topological pre-processing for this benchmark particularly challenging. In fact, we expect to extract 1621 generators of the first cohomology group of the insulating region, which corresponds to 3242 lazy generators. TOPOPROCESSOR has been able to compute all generators in less than 8 min of total computing time on the laptop, whereas GMSH terminated after more than 17 h on the workstation, see details on the mesh and on timings in Table III.

IV. DISCUSSION

The results show that the implementation of the combinatorial DS algorithm contained in the code TOPOPROCESSOR outperforms by more than two orders of magnitude the algebraic technique implemented inside GMSH [7] in case of large-sized industrial problems. A detailed time profiling of the various parts of the DS algorithm for all benchmarks is shown in Fig. 5.

The computational time required for the topological pre-processing with TOPOPROCESSOR is always a fraction of the meshing time. On the contrary, when using GMSH, the topological pre-processing may become a serious bottleneck for the whole electromagnetic simulation. Tested on

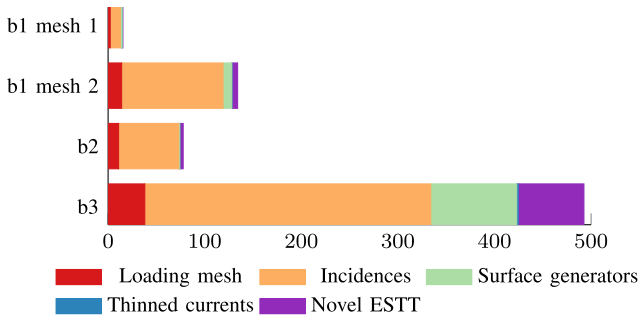

Fig. 5. Profiling of the computational time of TOPOPROCESSOR: all times are in seconds.

a computationally burdening practical application arising in fusion engineering and design, TOPOPROCESSOR was the only approach able to carry out the topological preprocessing on a laptop, suggesting the importance of the toolbox presented in this paper. The TOPOPROCESSOR executable is available, upon request, for non-profit use in the spirit of reproducible research (http:\\www.topoprocessor.com for more information).

REFERENCES

[1] P. R. Kotiuga, "Hodge decompositions and computational electromagnetics," Ph.D. dissertation, Dept. Elect. Eng., McGill Univ., Montreal, QC, Canada, 1984.
[2] P. W. Gross and P. R. Kotiuga, *Electromagnetic Theory and Computation: A Topological Approach*, vol. 48. Cambridge, U.K.: Cambridge Univ. Press, 2004.
[3] Z. Ren, "T-Ω formulation for eddy-current problems in multiply connected regions," *IEEE Trans. Magn.*, vol. 38, no. 2, pp. 557–560, Mar. 2002.
[4] R. Specogna, S. Suuriniemi, and F. Trevisan, "Geometric T-Ω approach to solve eddy-currents coupled to electric circuits," *Int. J. Numer. Meth. Eng.*, vol. 74, no. 1, pp. 101–115, 2008.
[5] P. Dłotko and R. Specogna, "Cohomology in 3D magneto-quasistatics modeling," *Commun. Comput. Phys.*, vol. 14, no. 1, pp. 48–76, 2013.
[6] P. Dłotko and R. Specogna, "Efficient cohomology computation for electromagnetic modeling," *Comput. Model. Eng. Sci.*, vol. 60, no. 3, pp. 247–278, 2010.
[7] M. Pellikka, S. Suuriniemi, L. Kettunen, and C. Geuzaine, "Homology and cohomology computation in finite element modeling," *SIAM J. Sci. Comput.*, vol. 35, no. 5, pp. 1195–1214, 2013.
[8] P. Dłotko and R. Specogna, "Physics inspired algorithms for (co)homology computations of three-dimensional combinatorial manifolds with boundary," *Comput. Phys. Commun.*, vol. 184, no. 10, pp. 2257–2266, Oct. 2013.
[9] P. Dłotko and R. Specogna, "Lazy cohomology generators: A breakthrough in (co)homology computations for CEM," *IEEE Trans. Magn.*, vol. 50, no. 2, Feb. 2014, Art. no. 7014204.
[10] P. Dłotko and R. Specogna, "Efficient generalized source field computation for h-oriented magnetostatic formulations," *Eur. Phys. J. Appl. Phys.*, vol. 53, no. 2, 2011, Art. no. 20801.
[11] J. P. Webb and B. Forghani, "A single scalar potential method for 3D magnetostatics using edge elements," *IEEE Trans. Magn.*, vol. 25, no. 5, pp. 4126–4128, Sep. 1989.
[12] P. Dłotko and R. Specogna, "Critical analysis of the spanning tree techniques," *SIAM J. Numer. Anal.*, vol. 48, no. 4, pp. 1601–1624, 2010.
[13] J. R. Munkres, *Elements of Algebraic Topology*, Cambridge, MA, USA: Perseus Books, 1984.
[14] A. A. Rodriguez, E. Bertolazzi, R. Ghiloni, and R. Specogna, "Efficient construction of 2-chains with a prescribed boundary," *SIAM J. Numer. Anal.* [Online]. Available: https://arxiv.org/abs/1409.5487v1
[15] P. Dłotko and R. Specogna, "A novel technique for cohomology computations in engineering practice," *Comput. Methods Appl. Mech. Eng.*, vol. 253, pp. 530–542, Jan. 2013.