Noname manuscript No.
(will be inserted by the editor)

Distributed computation of coverage in sensor networks by homological methods

P. Dłotko · R. Ghrist · M. Juda · M. Mrozek

Received: date / Accepted: date

Abstract Recent work on algebraic-topological methods for verifying coverage in planar sensor networks relied exclusively on centralized computation: a limiting constraint for large networks. This paper presents a distributed algorithm for homology computation over a sensor network, for purposes of verifying coverage. The techniques involve reduction and coreduction of simplicial complexes, and are of independent interest. Verification of the ensuing algorithms is proved, and simulations detail the improved network efficiency and performance.

Keywords homology · sensor network · coverage · distributed computation

Mathematics Subject Classification (2000) 55-04 · 55N99 · 52B99

1 Introduction

Sensors — devices which return data tied to a location — are ubiquitous [13]. Although many sensors commonly used are stand-alone or *global* devices, there

The first, third and fourth author are partially supported by Polish MNiSW, Grant N201 037 31/3151 and N N201 419639. The first author is partially supported by Polish MNiSW Grant N N206 625439. The second author is supported by the ONR and by DARPA SToMP # HR0011-07-1-0002.

P. Dłotko & M. Juda
Institute of Computer Science, Jagiellonian University, Kraków, Poland
E-mail: Pawel.Dlotko@uj.edu.pl ; E-mail: Mateusz.Juda@ii.uj.edu.pl

R. Ghrist
Departments of Mathematics and Electrical & Systems Engineering, University of Pennsylvania, Philadelphia, PA
E-mail: ghrist@math.upenn.edu

M. Mrozek
Institute of Computer Science, Jagiellonian University, Kraków, Poland; and Division of Computational Mathematics, Graduate School of Business, Nowy Sącz, Poland
E-mail: Marian.Mrozek@ii.uj.edu.pl

is an increasing push to network multiple *local* sensors, thanks to progress in miniaturization and wireless communications. The problem of collating distributed pieces of sensor data over a communications network is a substantial engineering challenge for which mathematical tools of a wide variety are relevant.

One simple-to-state problem of direct relevance is that of COVERAGE, or how well a region is monitored by sensors. For concreteness, fix a domain $\mathcal{D} \subset \mathbb{R}^2$ and consider a finite collection $\mathcal{N}$ of sensors in $\mathcal{D}$. The nodes have two functions: they (1) sense a neighborhood of their locale in $\mathbb{R}^2$; and (2) they communicate with other nearby sensors. Both of these actions are assumed to be local in the sense that individual nodes cannot extract sensing data from or communicate data over all of $\mathcal{D}$. The problem of coverage, or more precisely, BLANKET COVERAGE, is the question of whether there are holes in the sensor network — are there any regions in $\mathcal{D}$ which are not sensed? Other important coverage problems include BARRIER COVERAGE, in which one wants to determine whether the region covered by a sensor network separates $\mathcal{D}$ or surrounds a critical region, and SWEEPING COVERAGE, the time-dependent coverage problem familiar to users of robotic vacuum sweepers.

Coverage problems in sensor networks have received extensive attention with a literature whose tools derive from computational geometry [22,31,32, 21], graph-theory [14], dynamical systems [7], and stochastic geometry [19]. Optimal coverage forms a distinct class of *art gallery* problems, with heavy representation in the computer science and complexity literature. Recently, several authors have turned to methods which are coordinate-free, inspired by problems involving non-localized ad hoc networks. In this class of coverage problems — where coordinates of sensor nodes cannot be assumed — available tools are less geometric. Among the techniques used in non-localized problems, algebraic topology has been recently seen to be both applicable and powerful [9,11,25,29,30,1]. Specifically, the tool used is SIMPLICIAL HOMOLOGY, an algebraic-topological construct that determines global features (e.g., holes) in a simplicial complex (e.g., generated by nodes and communication links) utilizing local connectivity data (e.g., communications protocols). Homology theory is outlined in Appendix A; homological coverage criteria are reviewed in §3.

The problem this paper solves is as follows: in all the initial work on homological criteria for sensor coverage [9], the computations were centralized. All sensor nodes were required to upload connectivity data to a central server for the crucial homology computation. Although there are no fundamental obstructions to a decentralized computation of homology [26], specific methods for doing so are nontrivial.

Our contributions are as follows:

1. We provide a provably correct algorithm for distributed computation of homological coverage criteria.
2. The algorithm can recover specific generators, for use in building minimal or power-reducing covers.

3. Most crucially, the algorithm computes homology in an arbitrary coefficient system. This allows for computations over finite fields, which avoids the roundoff errors present in $\mathbb{R}$-coefficients.
4. Simulations seem to indicate that for a unit-disc graph network of points in the plane, all complexes are completely reducible, indicating that homology computation is of linear algorithmic complexity.

The last point leads us to the following:

Conjecture 1 For a Vietoris-Rips complex of points $\mathcal{N}$ in the plane $\mathbb{R}^2$, the S-complex $\mathcal{K}^f$ remaining after applying Algorithm 8 is always boundaryless.

If this conjecture is true, there is no need for any additional computations to ascertain the homology. This would be beneficial as regards theoretical complexity bounds, since the reduction algorithm in a non-distributed setting and bounded amount of neighbors has complexity $O(N)$, where N is the number of simplices (see [23]). However, even if in some situations the conjecture fails, in practice the size of the remaining complex is very small: one node would easily compute the homology of the remaining complex.

Our work compares most closely to the recent approach of Jadbabaie and Tahbaz-Salehi [29]. They also derive a distributed algorithm for homology computation with the goal of satisfying the homological coverage criterion, determining the existence/location of holes, and generating optimal coverage. Their methods involve passing to homology with $\mathbb{R}$ coefficients and setting up the problem as a dynamical system — in essence, they use simplicial Laplacians and their connection to homology via Hodge theory [12] to solve a heat equation over the network. This can be done in a distributed manner, using message-passing and gossip algorithms [18]. The paper [29] is very creative in that they also apply compressive-sensing perspectives to the problem of provably and quickly determining optimal generators for homology classes.

There are a few detriments to their methods that the present paper complements. The reduction/coreduction scheme of this paper has several advantages.

1. It greatly reduces the communication complexity demanded by a dynamical systems approach as in [29]: waiting for a large network to asymptotically converge to a solution is nontrivial in time and in energy drain from communication.
2. It is applicable to arbitrary coefficient systems. The Hodge-theoretic approach of [29] cannot avoid the use of $\mathbb{R}$ coefficients and the ensuing round-off errors, which can accumulate to frustrate answers in settings where homology generators are not well-separated. Our approach works well with finite field coefficients (e.g., mod-2 arithmetic) or integer coefficients, avoiding roundoff altogether.

Moreover, the methods developed in this paper may be adapted to speed up general-purpose homology computations in the context of parallel architecture, in particular in multi-core processors and GPU's (work in progress). Finally, more flexible local coefficient systems (as in [15]) may be adaptable; if so, the techniques of this paper may be extended to give a distributed computation of the cohomology of simplicial sheaves, a problem whose relevance to networks is emerging [27,28,16].

We understand that an independent method for distributed homology computation is being investigated by Carlsson, de Silva, and Morozov, using the technique of ZIGZAG PERSISTENCE, as initiated in [4] and [5].

2 Sensor and network assumptions

For our applications of distributed homology computation to coverage problems, we operate under the following assumptions.

1. Sensors are modeled as a collection of nodes $\mathcal{N} \subset \mathbb{R}^2$.
2. Each sensor is assumed to have a unique identification which it broadcasts; certain neighbors detect the transmission and establish a communication link.
3. Communication links are symmetric, stable, and generate a well-defined communications graph $\mathcal{G}$ on $\mathcal{N}$.
4. Sensor coverage regions are correlated to communications: the convex hull of any subset of sensors $S \subset \mathcal{N}$ which pairwise communicate is contained in the union of coverage regions of S.
5. One fixes a cycle $\mathcal{C} \subset \mathcal{G}$ whose image in $\mathbb{R}^2$ is a simple closed curve bounding a simply connected domain $\mathcal{D} \subset \mathbb{R}^2$.

Under these assumptions, one wants to know whether $\mathcal{D}$ is contained in the coverage region of the network. These assumptions are chosen to be weak enough to be applicable in realistic systems; however, some important considerations — e.g., time-variability, node failure, false echoes, communications errors — are not modeled.

3 Homological coverage

This paper builds on a homological criterion for coverage in sensor networks described and explored in [9,11]. This section reviews that criterion, with details on homology theory placed in Appendix A. The reader for whom homology is foreign will want to make a brief excursion to Appendix A.

3.1 Simplicial complexes for networks

This paper uses as its basic data structure simplicial complexes based on a set of nodes (sensors) $\mathcal{N}$. A finite family X of nonempty finite subsets of $\mathcal{N}$ is an (abstract) SIMPLICIAL COMPLEX if for every $\sigma \in X$ and $\tau \subset \sigma$ we have $\tau \in X$. The elements $\sigma \in X$ are SIMPLICES whose DIMENSION equals the cardinality minus one: $\dim \sigma = |\sigma| - 1$. The 0-simplices of X are VERTICES and the 1-simplices are EDGES, as in graph theory. A FACE of a simplex σ in a simplicial complex X is a simplex $\tau \subset \sigma$ with $\dim \tau = \dim \sigma - 1$. For τ a face

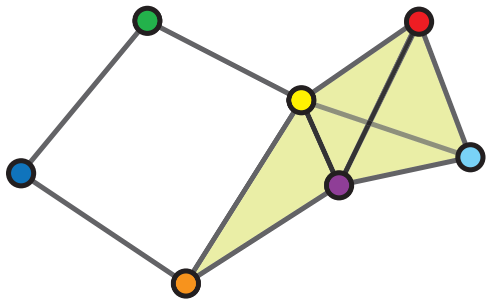

Fig. 1 The geometric realization of a simplicial complex.

of σ one says that σ is a COFACE of τ. Every simplicial complex has a natural geometric realization — also denoted X — as a topological space obtained by gluing disjoint copies of the standard k-simplex, Δ^k, along faces, as in Figure 1.

Simplicial complexes are natural to networks. Any communications network on nodes $\mathcal{N}$ has the structure of a simplicial complex with all simplices of dimension zero or one. In several applied contexts, higher dimensional simplicial complexes are natural data structures. Examples include the following. Assume a set of nodes $\mathcal{N}$ in Euclidean $\mathbb{R}^n$:

1. The size ϵ ČECH COMPLEX is the simplicial complex $\mathcal{C}_\epsilon$ on $\mathcal{N}$ whose k-simplices are generated by $k+1$ nodes about which the diameter ϵ balls have a mutually nonempty intersection.
2. The size ϵ VIETORIS-RIPS COMPLEX is the simplicial complex $\mathcal{R}_\epsilon$ on $\mathcal{N}$ whose k-simplices are generated by $k+1$ nodes about which the diameter ϵ balls have *pairwise* nonempty intersections. Namely, simplices are tuples of nodes with pairwise distance less that or equal to ϵ.
3. Given a network of edges based on $\mathcal{N}$, the FLAG COMPLEX of the network is the simplicial complex $\mathcal{F}$ whose k simplices are generated by pairwise-connected $k+1$-tuples of nodes in the network.

Thus, the Vietoris-Rips complex $\mathcal{R}_\epsilon$ is the flag complex of the size ϵ UNIT DISC GRAPH. This simple model of connectivity for a sensor or communications network is widely used [14,22,32] and widely disparaged [3,8,20]. Our results hold in the context of more general flag complexes associated to network communication graphs, whether they are unit disc graphs or not.

In this paper, we restrict attention to simplicial complexes. There is a broader notion of a CELL COMPLEX having as building blocks not merely simplices, but cubes or other polyhedral cells. These can sometimes be useful in modeling the geometry underlying a sensor network. The methods described in this paper extend to these as well but are not detailed explicitly.

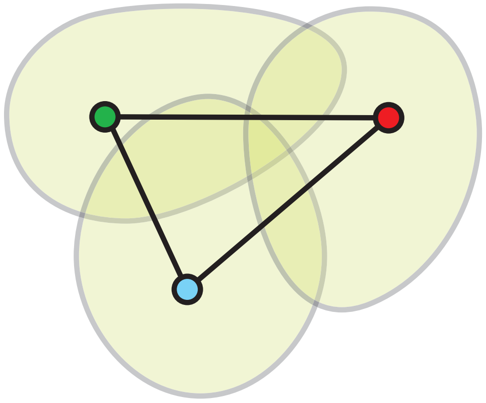

Fig. 2 Coverage regions are assumed to be correlated to communication distances: the convex hull of a triple of communicating nodes is assumed covered.

3.2 The homological coverage criterion

Recall our standing assumption linking coverage to communication: any triple of nodes in pairwise communication has its convex hull in $\mathbb{R}^2$ contained in the coverage region, as in Figure 2. We emphasize that this neither assumes localization nor an unrealistic assumption about round balls, symmetry, or the like: this is a very flexible model. As stated, it is explicitly simplicial, and motivates the use of the flag complex $\mathcal{F}$ of the communications graph $\mathcal{G}$ to model the topology of the sensed region.

The second important assumption is that a cycle $\mathcal{C}$ in the network is chosen whose image in $\mathbb{R}^2$ is a simple closed curve bounding a polygonal domain $\mathcal{D}$. This cycle acts as a FENCE for the coverage problem. (Criteria for guaranteeing that a cycle in a non-localized network has simple image in the plane are simple to derive [9,6] — it suffices to have no 'shortcuts' between cycle nodes.) The following theorem gives a criterion for coverage based on (relative) homology of the flag complex $\mathcal{F}$ of the communications network modulo the fence cycle $\mathcal{C}$. It is a slight modification of Theorem 3.3 of [9].

Theorem 1 *(Homological coverage criterion [9]) Given a planar sensor network with $\mathcal{G}$, $\mathcal{C}$, and $\mathcal{D}$ as above, then all of $\mathcal{D}$ is completely covered by the sensors if, equivalently:*

1. $[\mathcal{C}] = 0 \in H_1(\mathcal{F})$.
2. *There exists* $[\zeta] \in H_2(\mathcal{F}, \mathcal{C})$ *with* $\partial\zeta = \mathcal{C}$.

In practice, the second form of the criterion is more useful, since, if an explicit relative cycle $\zeta \in Z_2(\mathcal{F}, \mathcal{C})$ is computed, then it provides a guarantee of coverage even when the nodes not implicated in ζ are removed (or 'powered down' for conservation reasons): see [9].

4 *S*-complexes and reduction algorithms.

This paper introduces a distributed homology computation that suffices to check the criterion of Theorem 1. To describe this algorithm, it is necessary to modify the relevant simplicial and chain complexes. Our approach is explicitly simplicial. Though the reductions and coreductions used to simplify a simplicial complex have geometric motivation, it is best to pass to an more algebraic framework in the form of a modification of the usual complexes used in basic homology theory. Recall from Appendix A that in homology theory, a simplicial complex (a fundamentally topological object) is replaced with a chain complex (a fundamentally algebraic device). Our technique is based on simplifying the simplicial and chain complexes simultaneously.

4.1 *S*-complexes

We review the concept of an S-COMPLEX introduced in [23] as a reformulation of a chain complex, convenient for algorithmic purposes. Let $\mathcal{K}$ be a finite set partitioned according to a grading $\mathcal{K} = (K_q)_{q\in\mathbb{Z}}$. For every element $\sigma \in \mathcal{K}$ there exists a unique $q \in \mathbb{Z}$ such that $\sigma \in K_q$. This is the DIMENSION of σ, denoted dim σ. Let R be a ring with unity and $R[\mathcal{K}]$ the graded free module over R generated by the graded set $\mathcal{K}$. We refer to the elements of $R[\mathcal{K}]$ as CHAINS and to the generators as ELEMENTARY CHAINS. Given two chains $c := \sum_{\sigma\in\mathcal{K}} c_\sigma \sigma$ and $d := \sum_{\sigma\in\mathcal{K}} d_\sigma \sigma$ we define their inner product via

$$\langle c, d\rangle := \sum_{\sigma\in\mathcal{K}} c_\sigma d_\sigma .$$

Let $\kappa : \mathcal{K} \times \mathcal{K} \to R$ be a map satisfying

$$\kappa(\sigma, \tau) \neq 0 \Rightarrow \text{dim } \sigma = \text{dim } \tau + 1.$$

We say that $(\mathcal{K}, \kappa)$ is an S-COMPLEX if $(R[\mathcal{K}], \partial^\kappa)$ with

$$\partial^\kappa : R[\mathcal{K}] \to R[\mathcal{K}]$$

defined on generators $\sigma \in \mathcal{K}$ by

$$\partial^\kappa(\sigma) := \sum_{\tau\in\mathcal{K}} \kappa(\sigma, \tau)\tau$$

is a free chain complex with basis $\mathcal{K}$.

For example, every simplicial complex $\mathcal{K}$ has a natural gradation $(K_q)_{q\in\mathbb{Z}}$, where K_q consists of simplices of dimension q. Assume an ordering of the vertices of $\mathcal{K}$ is given and every simplex σ in K_q is coded as $[v_0, v_1, \ldots, v_q]$, where $v_0, v_1, \ldots, v_q$ are listed according to the prescribed ordering. By putting

$$\kappa(\sigma, \tau) := \begin{cases} (-1)^i & \text{if } \sigma = [v_0, v_1, \ldots, v_{i-1}, v_i, v_{i+1}, \ldots, v_q] \\ & \text{and } \tau = [v_0, v_1, \ldots, v_{i-1}, v_{i+1}, \ldots, v_q] \\ 0 & \text{otherwise.} \end{cases}$$

we obtain an S-complex $(\mathcal{K},\kappa)$. An S-complex generalizes simplicial complexes.

The homology of an S-complex $(\mathcal{K},\kappa)$, $H(\mathcal{K})$, is defined as the homology of the chain complex $(R[\mathcal{K}],\partial^\kappa)$. In the sequel the value $\kappa(\sigma,\tau)$ will be referred to as the COINCIDENCE INDEX of σ and τ and the map κ as the COINCIDENCE MAP.

We say that $(\mathcal{K},\kappa)$ is BOUNDARYLESS if $\partial^\kappa = 0$. Note that if $(\mathcal{K},\kappa)$ is boundaryless, then there is no need to compute its homology, because $H(\mathcal{K}) = R[\mathcal{K}]$. Note: a boundaryless complex is highly desirable from the point of view of computation of homology, as no further computation is necessary!

For any $\mathcal{A}\subset\mathcal{K}$ we define the BOUNDARY SET and the COBOUNDARY SET of $\mathcal{A}$ in the S-complex $\mathcal{K}$ as follows:

$$\begin{aligned}\operatorname{bd}_{\mathcal{K}}(\mathcal{A}) &:= \{\,\tau\in\mathcal{K} \mid \ \kappa(\sigma,\tau)\neq 0 \text{ for some } \sigma\in\mathcal{A}\,\},\\ \operatorname{cbd}_{\mathcal{K}}(\mathcal{A}) &:= \{\,\sigma\in\mathcal{K} \mid \ \kappa(\sigma,\tau)\neq 0 \text{ for some } \tau\in\mathcal{A}\,\}.\end{aligned}$$

When $\mathcal{A}=\{\sigma\}$ we simplify the above notation by writing $\operatorname{bd}_{\mathcal{K}}\sigma$ and $\operatorname{cbd}_{\mathcal{K}}\sigma$.

We say that $\mathcal{K}'\subset\mathcal{K}$ is

1. CLOSED in $\mathcal{K}$ if $\operatorname{bd}_{\mathcal{K}}\mathcal{K}'\subset\mathcal{K}'$;
2. OPEN in $\mathcal{K}$ if $\mathcal{K}\setminus\mathcal{K}'$ is closed in $\mathcal{K}$.

The following proposition is straightforward.

Proposition 1 *The family of open subsets as well as the family of closed subsets of an S-complex $\mathcal{K}$ are closed under the operations of finite union and finite intersection.*

We say that $\mathcal{K}'\subset\mathcal{K}$ is a SUBCOMPLEX of the S-complex $\mathcal{K}$ if $(\mathcal{K}',\kappa')$ with $\kappa' := \kappa|_{\mathcal{K}'\times\mathcal{K}'}$ is an S-complex.

Theorem 2 *[23, Theorem 3.1] Let $(\mathcal{K},\kappa)$ be an S-complex and let $\mathcal{K}'\subset\mathcal{K}$ be a subset. If for all $\sigma,\tilde\sigma\in\mathcal{K}'$ and $\tau\in\mathcal{K}$*

$$\tau\in\operatorname{bd}_{\mathcal{K}}\sigma,\ \tilde\sigma\in\operatorname{bd}_{\mathcal{K}}\tau \implies \tau\in\mathcal{K}',$$

then $\mathcal{K}'$ is a subcomplex.

A subset $\mathcal{K}'\subset\mathcal{K}$ satisfying the assumptions of Theorem 2 is called REGULAR. Note that for a regular subset $\mathcal{K}'$ of an S-complex $\mathcal{K}$ and a subset $\mathcal{A}\subset\mathcal{K}'$ it makes sense to consider the boundary and coboundary of $\mathcal{A}$ in $\mathcal{K}'$. The following proposition is straightforward.

Proposition 2 *Assume $\mathcal{K}'$ is a subcomplex of an S-complex $\mathcal{K}$ and $\mathcal{A}\subset\mathcal{K}'$. Then*

$$\begin{aligned}\operatorname{bd}_{\mathcal{K}'}(\mathcal{A}) &= \operatorname{bd}_{\mathcal{K}}(\mathcal{A})\cap\mathcal{K}',\\ \operatorname{cbd}_{\mathcal{K}'}(\mathcal{A}) &= \operatorname{cbd}_{\mathcal{K}}(\mathcal{A})\cap\mathcal{K}'.\end{aligned}$$

From Proposition 2 we easily obtain the following proposition.

Proposition 3 *If $\mathcal{K}'$ is a regular subset of an S-complex $\mathcal{K}$ and $\mathcal{K}''$ is a regular subset of $\mathcal{K}'$, then $\mathcal{K}''$ is a regular subset of $\mathcal{K}$.*

Theorem 3 *[23, Theorem 3.2] If $\mathcal{K}'\subset\mathcal{K}$ is closed in $\mathcal{K}$, then $\mathcal{K}'$ and $\mathcal{K}\setminus\mathcal{K}'$ are regular.*

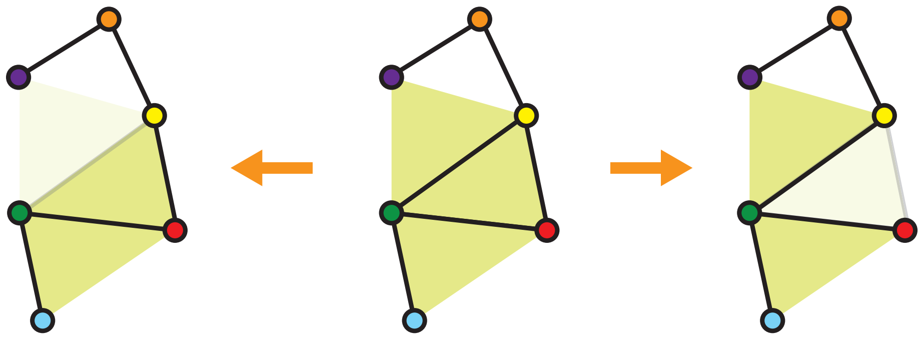
Fig. 3 Simplification of a complex [center] can be performed via elementary coreduction [left] and reduction [right] pairs.

4.2 Reduction pairs and reduction sequences.

The parallelized reduction process of an S-complex $\mathcal{K}$ consists of removing certain pairs of elements of $\mathcal{K}$. We first introduce some notation concerning such pairs. Given a pair $\alpha = (\tau, \sigma) \in \mathcal{K}^2$, we refer to the doubleton $\{\tau, \sigma\}$ as the SUPPORT of the pair α and denote it $|\alpha|$. We extend the concept and notation of support to sequences $\beta = (\beta_1, \beta_2, \dots, \beta_n) \in (\mathcal{K}^2)^n$ of pairs by

$$|\beta| := \bigcup_{i=1}^{n} |\beta_i|.$$

We now explain the concepts of pairs crucial in the reduction process. A pair $\alpha = (\tau, \sigma) \in \mathcal{K}^2$ is a REDUCTION PAIR if $\kappa(\sigma, \tau)$ is invertible in R. It is called an ELEMENTARY REDUCTION PAIR IN $\mathcal{K}$ if $\operatorname{cbd}_{\mathcal{K}}\{\tau\} = \{\sigma\}$. It is called an ELEMENTARY COREDUCTION PAIR IN $\mathcal{K}$ if $\operatorname{bd}_{\mathcal{K}}\{\sigma\} = \{\tau\}$. We will use the term S-REDUCTION PAIR IN $\mathcal{K}$ to denote a pair which is either an elementary reduction pair or an elementary coreduction pair in $\mathcal{K}$. Whenever the S-complex $\mathcal{K}$ is clear from the context, we will refer to an S-reduction pair in $\mathcal{K}$ simply as an S-reduction pair. S-reduction pairs are algebraic generalizations of pairs of simplices related by boundaries or coboundaries: see Figure 3.

Note that a pair $\alpha \in \mathcal{K}^2$ which is not necessarily an S-reduction pair in $\mathcal{K}$ may be an S-reduction pair in a subcomplex of $\mathcal{K}$. We call such an S-reduction pair a POTENTIAL S-REDUCTION PAIR IN $\mathcal{K}$. Whenever potential S-reduction pairs exist, the reduction process may be self-feeding: removing some S-reduction pairs may give rise to new S-reduction pairs. On the other hand, if the supports of two S-reduction pairs in a given S-complex have non-empty intersection, performing one of these reductions kills the other reduction. But, if the supports are disjoint, both reductions may be performed. Moreover, they may be performed independently, hence, in parallel. We say that a collection $\mathcal{B} = \{\beta_1, \beta_2, \dots, \beta_n\}$ of potential S-reduction pairs in $\mathcal{K}$ is FREE if all elements of $\mathcal{B}$ are S-reduction pairs in the same subcomplex of $\mathcal{K}$ and any two different elements of $\mathcal{B}$ have disjoint supports.

Let $\mathcal{L}$ be a subcomplex of $\mathcal{K}$. We say that a sequence $\varphi = (\varphi_1, \varphi_2, \ldots, \varphi_n)$ of potential S-reduction pairs in $\mathcal{L}$ is a REDUCTION SEQUENCE in $\mathcal{L}$ if for each $j = 1, 2, \ldots, n$ the pair φ_j is an S-reduction pair in $\mathcal{L} \setminus \bigcup_{i=1}^{j-1} |\varphi_i|$. Note that a straightforward recursive argument shows that the definition in meaningful, i.e., $\mathcal{L} \setminus \bigcup_{i=1}^{j-1} |\varphi_i|$ is an S-complex. A reduction sequence is said to be FREE if the elements of the sequence form a free collection of S-reduction pairs. Two reduction sequences are said to be INTERCHANGEABLE if they differ only by a permutation. We use S-reduction pairs to selectively simplify an S-complex.

The salient points of the theory, developed in full in Appendix B, are as follows:

1. Each S-reduction pair α in a subcomplex $\mathcal{L}$ of $\mathcal{K}$ induces an isomorphism
$$\gamma_\alpha : H(\mathcal{L} \setminus |\alpha|) \to H(\mathcal{L}).$$
2. A reduction sequence φ in a subcomplex $\mathcal{L}$ of $\mathcal{K}$ yields an isomorphism $I_\varphi : H(\mathcal{L} \setminus |\varphi|) \to H(\mathcal{L})$
3. The isomporhpism I_φ is the composition of the isomorphisms γ_{φ_i}.

Moreover, the following is shown in Appendix B and used in the second half of this paper to prove correctness of our algorithms:

Theorem 4 *Any full ordering of a free collection of S-reduction pairs forms a reduction sequence. Two interchangeable reduction sequences φ, φ' give rise to the same isomorphism $I_\varphi = I_{\varphi'}$.*

A consequence of Theorem 4 which makes a distributed reduction process possible is the fact that the reductions in a free reduction sequence may be performed independently, so in parallel. This is because the order in which they are performed does not matter.

Example 1 Consider the set of nine sensors $\{A, B, C, D, E, F, G, H, J\}$ whose communication graph consists of eighteen edges (see Figure 4)

$$\begin{aligned}\{\ & AB, AE, AG, BC, BE, BF, CD, CE, CF, \\ & DF, DJ, EF, EG, EH, FH, FJ, GH, HJ\}.\end{aligned}$$

After removing the fence consisting of edges $\{AB, BC, DJ, HJ, GH, AG\}$, we obtain an S-complex composed of two vertices, eleven 1-simplices, eleven 2-simplices and one 3-simplex. A search for S-reduction pairs reveals elementary coreduction pairs (F, DF) and (E, CE). These form a free collection and may be removed in parallel. Continuing, we obtain the following reduction sequence, in which free collections are written in separate lines.

$$\begin{aligned}& (F, DF)\ (E, CE) \\ & (FJ, DFJ)\ (CF, CDF)\ (BE, BCE) \\ & (EF, CEF)\ (BF, BCF)\ (AE, ABE)\ (FH, FHJ) \\ & (BEF, BCEF)\ (EG, AEG)\ (EH, EFH).\end{aligned}$$

In the case of this reduction sequence the resulting S-complex is boundaryless. It consists only of the (open) 2-simplex EFH, which becomes the generator of the second homology group of the S-complex.

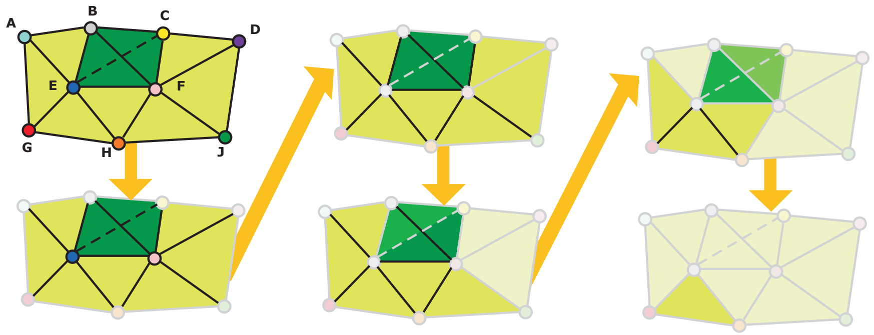

Fig. 4 From top left, to bottom right, following the arrows: a collection of sensors and the associated S-complex, the S-complex after removing the fence, the fence complex after removing the consecutive free collections of S-reduction pairs, the final reduced S-complex.

5 Construction of the Flag Complex

In this section we describe the distributed algorithms for the construction of the flag complex as a simplicial approximation of the sensor network. Henceforth we assume a fixed enumerative ordering of the sensor nodes $\mathcal{N}$, as per unique identification numbers. We denote the total order by $\lhd$. Moreover, we assume that all algorithms described in this section are executed separately on every node.

Following the conventions of object oriented programming, we write `s.alg()` to indicate that node s is requested to execute `alg()`. In every algorithm we denote the node running the algorithm by the word `this`. We drop the prefix "`this`" in `this.alg()`, assuming that by default the node supposed to execute the called algorithm is the node making the call. We assume that every node stores the set of its neighbors in variable `neighbors`.

To store simplex σ of the flag complex $\mathcal{F}$ associated with sensor network $\mathcal{N}$ we use a data structure `Simplex` with the following fields and methods:

- σ.`vert` - the set of the vertices (nodes) in σ,
- σ.`neighb` - the set of the neighbors of σ,
- σ.`faces()` - returns the set of the faces of σ,
- σ.`cofaces()` - returns the set of the cofaces of σ.

In course of running the reduction algorithm we also need some auxiliary fields. In particular, when a simplex is removed by the reduction algorithm, instead of deleting it from the data structure we only mark a flag indicating that the simplex is to be treated as deleted. We use another flag to lock a simplex when a node is negotiating its removal with its neighbors. Here is the list of all auxiliary fields:

- σ.`deleted` - a boolean variable marking that σ is deleted,
- σ.`locked` - a boolean variable marking that σ is locked.

In order to construct a simplex we use Algorithm 1. It returns a simplex with vertex set V and the neighbors set N.

Algorithm 1 newSimplex(V, N)

```
1. $\sigma :=$ new simplex object.
2. $\sigma.$vert $:= V$.
3. $\sigma.$neighb $:= N$.
4. $\sigma.$locked $:= \sigma.$deleted $:=$ false.
5. return $\sigma$.
```

Algorithm 2 createLocalFlagComplex

```
1. $\tau_0 :=$ newSimplex({ this }, neighbors).
2. Simp$[0] := \{\tau_0\}$.
3. $i := \max\{j \in \mathbb{N} \mid$ Simp$[j] \neq \emptyset\}$.
4. for each $\tau \in$ Simp$[i]$
   (a) next $\tau$ if $\min(\tau.$vert$) \neq$ this.
   (b) for each $t \in \tau.$neighb
        i. next $t$ if $\min(\tau.$vert$) \trianglelefteq t \trianglelefteq \max(\tau.$vert$)$.
       ii. $V := \tau.$vert $\cup \{t\}$,
      iii. $N := \tau.$neighb $\cap t.$neighbors,
       iv. $\sigma :=$ newSimplex$(V, N)$.
        v. Simp$[i+1] :=$ Simp$[i+1] \cup \{\sigma\}$.
5. if Simp$[i+1] \neq \emptyset$, then go to step 3.
```

Algorithm 2 is the algorithm for building the flag complex $\mathcal{F}$ of the network as expressed locally in terms of vertex-neighbor data. The algorithm is executed on each node. In the sequel, by `s.Simp` we mean the set of simplices created by node s in the course of running Algorithm 2. The subset of `s.Simp` consisting of simplices of dimension i is denoted by `s.Simp`$[i]$. We say that node `s` CONTROLS simplex σ if simplex σ is created in course of running Algorithm 2 on node `s`.

Theorem 5 *Assume Algorithm 2 has been initiated on each node. Then it terminates on each node. Moreover, once the algorithm has completed on all nodes, the following properties hold true.*

1. *For every node* `s` *and every simplex*

$$\sigma \in \texttt{s.Simp}$$

   variables $\sigma.$`vert` *and* $\sigma.$`neighb` *store respectively, as expected, the vertices and neighbors of simplex* σ.
2. *Every* $\sigma \in \mathcal{F}$ *is controlled by at least one and at most two nodes. It is controlled by exactly two nodes if and only if* $\dim \sigma > 0$. *Moreover, these two nodes are the two minimal elements of* $\sigma.$`vert`.

3. *The only simplices created by Algorithm 2 are the simplices in $\mathcal{F}$.*

Since every simplex σ is controlled by at least one and most two nodes, in the sequel we speak about the LOWER and the UPPER node controlling σ, assuming that the lower equals the upper if only one node controls σ. We omit the definitions of σ.`faces()` and σ.`cofaces()`. They may be computed by a node using `Simp` stored in the lower and in the upper node controlling σ.

Proof To see that the algorithm always terminates, first observe that the loop in line (4) always completes, because S is finite. The only other loop in the algorithm is formed by the conditional jump from line (5) to line (3). Note that whenever the jump takes place, the variable i is increased. However, the value of i may not exceed the number of nodes. Therefore, the jump in line (5) may be performed only a finite number of times and consequently the algorithm terminates.

The fact that variable σ.`vert` stores the vertices of σ is an immediate consequence of Algorithm 1. Let $\sigma \in$ `s.Simp` be a simplex and let d be its dimension. We will prove by induction on d that variable σ.`neighb` contains the neighbors of σ. If $d = 0$, the conclusion is obvious from the construction. Thus, assume that the conclusion holds for all simplices of dimension not exceeding d. The simplex σ is created from a simplex τ and its neighbor t at line (4.b.iii) of Algorithm 2. In particular, we see from the construction that

$$\sigma.\texttt{neighb} = \tau.\texttt{neighb} \cap \{u \in V \mid \{u, t\} \in E\}.$$

However, by the induction assumption

$$\tau.\texttt{neighb} = \{u \in V \mid \forall_{v \in \tau.\texttt{vert}} \{v, u\} \in E\}$$

and the conclusion follows from elementary set arithmetic. This proves property (1).

Let us now demonstrate property (2). First consider the case when $\dim \sigma = 0$. Observe that the only line of the algorithm which contains the construction of a zero dimensional simplex is line (1) of the algorithm. Moreover, it is straightforward to check that this line is executed exactly once in each node. Therefore, a zero dimensional simplex τ_0 is constructed by node `s` and only by node `s`. There remains to consider the case $\dim \sigma > 0$. Let

$$\sigma.\texttt{vert} = \{v_0, v_1, \ldots, v_n\}.$$

Observe that the only line of Algorithm 2 which contains a construction of a simplex of dimension greater than zero is line (4.b.iv). Let τ denote the simplex selected in line (4) just before the execution of line (4.b.iv). Since line (4.b.iv) is executed, the test in line (4.a) fails, which implies that

$$\min(\tau.\texttt{vert}) = \texttt{this}\,. \tag{1}$$

The execution of lines (4.b.ii) and (4.b.iii) implies that the difference

$$\sigma.\texttt{vert} \setminus \tau.\texttt{vert}$$

contains exactly one element t and, by the execution of line (4.b.i), we know that either $t \lhd \min(\tau.\mathtt{vert})$ or $t \rhd \max(\tau.\mathtt{vert})$. The first case happens when $t = v_0, \tau = \{ v_1, v_2, \dots, v_n \}$ and the other case when $t = v_n, \tau = \{ v_0, v_1, \dots, v_{n-1} \}$. Therefore, by (1), the node running the algorithm is either v_0 or v_1. It follows that if dim $\sigma > 0$, then σ is represented in the memory of the two minimal elements of $\sigma.\mathtt{vert}$.

To prove property (3) suppose by contrary, that there exists a simplex $\sigma \in \mathcal{F}$ which has not been created by the algorithm in any node. From line (2) of the algorithm it follows that dim $\sigma > 0$. Let σ be the simplex of minimal dimension with the described property. It follows that all the faces τ of σ have been created by the algorithm. From line (4) of the algorithm it follows, that σ must have been created by the algorithm, a contradiction.

In the similar way one can show that if $\sigma \notin \mathcal{F}$ then it cannot be created by the algorithm. Suppose by contrary, that such a σ has been created and σ is of minimal dimension among such simplices. Therefore, σ has been constructed in line (4.b.iv) of the algorithm from a simplex τ by adding a vertex $t \in \tau.\mathtt{vert}$. By the minimality assumption, $\tau \in \mathcal{F}$, and since $t \in \tau.\mathtt{vert}$, we get $\sigma \in \mathcal{F}$, a contradiction. □

The following Lemma is used in the Algorithm 6.

Lemma 1 *For all simplices σ created by nodes* `s` *and* `t` *each face $\tau \in \sigma.$*`faces()` *is created by node* `s` *or* `t`*.*

Proof By Theorem 5 we have $\mathtt{s}, \mathtt{t} \in \sigma.\mathtt{vert}$. For each face τ of the simplex σ, we have $\text{card}(\sigma.\mathtt{vert} \setminus \tau.\mathtt{vert}) = 1$, therefore $\mathtt{t} \in \tau.\mathtt{vert}$ or $\mathtt{s} \in \tau.\mathtt{vert}$. Since $\tau.\mathtt{vert} \subset \sigma.\mathtt{vert}$, we have $\min(\tau.\mathtt{vert}) \in \{\mathtt{s}, \mathtt{t}\}$ and by Theorem 5 simplex τ is created by node `s` or `t`. □

6 Reductions

The parallel reduction procedures used in the algorithm will now be presented.

6.1 Fence reduction

We assume that Algorithm 5 is executed on each node. Definitions of the auxiliary procedures `delete` and `canDelete` are presented as Algorithm 3 and Algorithm 4.

Lemma 2 *Algorithm 5 deletes all simplices in $\mathcal{C}$ and no other simplices.*

Proof A zero-dimensional simplex τ_0 is deleted at line (3) of the algorithm and line (1) guarantees that only 0-simplices in $\mathcal{C}$ can be deleted there. A 1-simplex of $\mathcal{C}$ is deleted at line (4.a) and (5.a) of Algorithm 5. Since at this stage the only deleted 0-simplices belong to $\mathcal{C}$, it is clear that at line (4.a) and (5.a) only 1-simplices from $\mathcal{C}$ can be deleted. From the definition of the fence as a simple

Algorithm 3 canDelete(σ)

```
1. return this = min($\sigma$.vert) and not $\sigma$.deleted
   and not $\sigma$.locked.
```

Algorithm 4 delete(σ)

```
1. if $\sigma \in$ Simp and not $\sigma$.deleted, then
   (a) $\sigma$.deleted := $\sigma$.locked := true.
   (b) for each node u controlling $\sigma$
       u.delete($\sigma$).
```

Algorithm 5 removeFence

```
1. if this $\notin F$ then exit.
2. $\tau_0$ := the unique simplex in Simp[0].
3. delete($\tau_0$).
4. for each s' $\in$ neighbors and for each $\sigma \in$
   s'.Simp[1] s.t.
   card{ $\tau \in \sigma$.faces() | not $\tau$.deleted } = 0
   and s'.canDelete($\sigma$)
   (a) s'.delete($\sigma$).
5. for each $\sigma \in$ Simp[1] s.t.
   card{ $\tau \in \sigma$.faces() | not $\tau$.deleted } = 0
   and canDelete($\sigma$)
   (a) delete($\sigma$).
```

cycle it is clear that in the points (4.a) and (5.a) all the 1-simplices in $\mathcal{C}$ are deleted. Since lines (3), (4.a) and (5.a) are the only lines where the reduction of a simplex takes place, no other simplices can be removed. □

Since $\mathcal{C}$ is a closed subset of $\mathcal{F}$ in the sense of definition in Section 4, it follows from Theorem 3 that $\mathcal{F} \setminus \mathcal{C}$, together with the usual boundary map, is an S-complex.

6.2 Distributed S-reductions

By Lemma 4 S-reduction pairs may be removed from an S-complex without changing the homology groups of the complex. We present now how the reduc-

Algorithm 6 elementaryCoreduction

```
1. if there exists a simplex $\sigma \in$ Simp s.t.
   card$\{ \tau \in \sigma$.faces() | not $\tau$.deleted$\} = 1$
   and canDelete($\sigma$), then proceed, otherwise
   return false.
2. $\sigma$.locked := true.
3. if there exists a simplex $\tau \in \sigma$.faces() s.t.
   canDelete($\tau$) then
   (a) delete($\tau$).
   (b) delete($\sigma$).
   (c) return true.
4. otherwise
   (a) $s'$ := the other node controlling $\sigma$.
   (b) $\tau$ := the unique simplex in $\sigma$.faces() s.t. not
       $\tau$.deleted.
   (c) if s'.canDelete($\tau$) (see Lemma 1) then
         i. s'.delete($\tau$).
        ii. delete($\sigma$).
       iii. return true.
   (d) else $\sigma$.locked := false.
5. return false.
```

tions may be performed in a distributed manner. We perform the reduction process in such a way that simplex σ may be reduced only by the lower node controlling it.

6.2.1 Elementary coreduction

On each node Algorithm 6 is executed to find and delete elementary coreduction pairs.

6.2.2 Elementary reduction

On each node Algorithm 7 is executed to find and delete an elementary reduction pair.

6.2.3 Parallel reduction algorithm.

Algorithm 8 is executed in a loop on every node. The algorithm terminates when no node in the network is able to find an S-reduction pair. The stop criterion of Algorithm 8 requires global information from the network. In consequence, the criterion cannot be implemented on the basis of purely local

Algorithm 7 elementaryReduction

```
1. if there exists a simplex $\tau \in$ Simp s.t.
   card$\{\, \sigma \in \tau.$cofaces() $\mid$ not $\sigma.$deleted$\,\} = 1$
   and canDelete($\tau$), then proceed, otherwise
   return false.
2. $\tau.$locked := true.
3. $\sigma$ := the unique simplex in $\tau.$cofaces() s.t. not
   $\sigma.$deleted.
4. if canDelete($\sigma$)
   (a) delete($\tau$).
   (b) delete($\sigma$).
   (c) return true.
5. else if not $\sigma.$locked.
   (a) $s'$ := the other node controlling $\sigma$.
   (b) if s'.canDelete($\sigma$) then
         i. s'.delete($\sigma$).
        ii. delete($\tau$).
       iii. return true.
   (c) else $\tau.$locked := false.
6. return false.
```

Algorithm 8 reductionAlgorithm

```
1. Run createLocalFlagComplex followed by
   removeFence.
2. Run the following code as long as there ex-
   ists a node in the whole sensor network which
   returns true from elementaryCoreduction or
   elementaryReduction algorithm.
   (a) Run the elementaryCoreduction algorithm as
       long as there exists a node that returns true.
   (b) Run the elementaryReduction algorithm as
       long as there exists a node that returns true.
```

information as in the case of the other algorithms presented in the paper. However, the stop criterion may be easily implemented via broadcasts.

7 Correctness

In this section we prove that the algorithm presented in the previous section correctly reduces the flag complex in the sense that the homology of the original flag complex considered as an S-complex and the homology of the reduced S-complex are the same.

Let $\mathcal{A}$ be the set of all S-reduction pairs reduced by Algorithm 8 in all nodes and let

$$\mathcal{A}^r := \{\, \alpha \in \mathcal{A} \mid \alpha \text{ is removed as a reduction pair} \,\},$$
$$\mathcal{A}^c := \{\, \alpha \in \mathcal{A} \mid \alpha \text{ is removed as a coreduction pair} \,\}.$$

For $\mathcal{A}' \subset \mathcal{A}$ put $|\mathcal{A}'| := \bigcup_{\alpha\in\mathcal{A}'} |\alpha|$. Let

$$\mathcal{K} := \mathcal{F} \setminus \mathcal{C}, \tag{2}$$
$$\mathcal{G}^r := |\mathcal{A}^r|, \tag{3}$$
$$\mathcal{G}^c := |\mathcal{A}^c|, \tag{4}$$
$$\mathcal{K}^f := \mathcal{K} \setminus \mathcal{G}^r \setminus \mathcal{G}^c. \tag{5}$$

In other words, $\mathcal{K}$ is the S complex resulting from removing the fence cycle from the flag complex of the sensor network, $\mathcal{G}^r$ is the subset of $\mathcal{K}$ consisting of generators removed from $\mathcal{K}$ in elementary reductions, $\mathcal{G}^c$ is the subset of $\mathcal{K}$ consisting of generators removed from $\mathcal{K}$ in elementary coreductions and $\mathcal{K}^f$ is the S-complex resulting from $\mathcal{K}$ after applying all S-reductions reduced by Algorithm 8 in all nodes.

Theorem 6 *Given a network communication graph with simple cycle $\mathcal{C}$, Algorithm 8 terminates in every node. When all copies of the algorithm complete in all nodes, the S-complex $\mathcal{K}^f$ satisfies*

$$H(\mathcal{F},\mathcal{C}) \cong H(\mathcal{K}^f).$$

Proof Since the number of nodes is finite, also the number of all possible reductions is finite. It is clear, that in Algorithm 6 and Algorithm 7 either an S-reduction pair is reduced, or `false` is returned. Consequently either some S-reduction pair is reduced in the sensor network, or according to point (2) of Algorithm 8 the algorithm terminates in the whole sensor network. Therefore the algorithm must terminate. By [23, Theorem 3.4]

$$H(\mathcal{F},\mathcal{C}) \cong H(\mathcal{K}).$$

We need to prove that

$$H(\mathcal{K}) \cong H(\mathcal{K}^f). \tag{6}$$

Observe that if $(\tau,\sigma) \in \mathcal{A}$ is a pair reduced as an elementary coreduction pair, then

$$\operatorname{bd}_{\mathcal{K}} \sigma \subset |\mathcal{A}|. \tag{7}$$

Indeed, if there were an element $\tilde{\tau} \in \operatorname{bd}_{\mathcal{K}} \sigma \setminus |\mathcal{A}|$, then (τ, σ) could never become an elementary coreduction pair. A similar argument shows that if $(\tau, \sigma) \in \mathcal{A}$ is a pair reduced as an elementary reduction pair, then

$$\operatorname{cbd}_{\mathcal{K}} \tau \subset |\mathcal{A}|. \tag{8}$$

For $\alpha = (\tau, \sigma) \in \mathcal{A}$ put

$$\mathcal{A}(\alpha) := \begin{cases} \{\, \bar{\alpha} \in \mathcal{A} \mid (\operatorname{bd}_{\mathcal{K}} \sigma \setminus \tau) \cap |\bar{\alpha}| \neq \emptyset \,\} & \text{if } \alpha \in \mathcal{A}^c, \\ \{\, \bar{\alpha} \in \mathcal{A} \mid (\operatorname{cbd}_{\mathcal{K}} \tau \setminus \sigma) \cap |\bar{\alpha}| \neq \emptyset \,\} & \text{if } \alpha \in \mathcal{A}^r. \end{cases}$$

In the course of running the reduction calls throughout the network a function $\lambda : \mathcal{A} \to \mathbb{N}$ is defined recursively as follows. The value $\lambda(\alpha)$ is 1 for $\alpha \in \mathcal{A}$ such that $\mathcal{A}(\alpha) = \emptyset$. Such pairs are S-reduction pairs in the original complex, form a free collection and may be reduced immediately. However, if $\mathcal{A}(\alpha) \neq \emptyset$, then the reduction may be performed only after all elements in $\mathcal{A}(\alpha)$ are already reduced. This, in particular, means that the value of λ is already assigned to all elements of $\mathcal{A}(\alpha)$. Therefore, we may put

$$\lambda(\alpha) := 1 + \max\{\, \lambda(\bar{\alpha}) \mid \bar{\alpha} \in \mathcal{A}(\alpha) \,\}.$$

Now, put

$$\begin{aligned} \mathcal{A}_n &:= \{\, \alpha \in \mathcal{A} \mid \lambda(\alpha) = n \,\}, \\ \mathcal{A}_n^r &:= \mathcal{A}^r \cap \mathcal{A}_n, \\ \mathcal{A}_n^c &:= \mathcal{A}^c \cap \mathcal{A}_n. \end{aligned}$$

and define

$$\mathcal{K}^0 := \mathcal{K}, \tag{9}$$

$$\mathcal{K}^n := \mathcal{K}^{n-1} \setminus |\mathcal{A}_n|. \tag{10}$$

We claim that for each $n \in \mathbb{N}$

(i) $\mathcal{A}_n$ is a free collection of S-reduction pairs in $\mathcal{K}^{n-1}$,
(ii) $\mathcal{K}^n$ is an S-complex.

We will prove both properties by induction on n. Since $\mathcal{A}_0 = \emptyset$ and $\mathcal{K}^0 = \mathcal{K}$, both (i) and (ii) are obvious for $n = 0$. Thus assume that the properties are satisfied for all $n < k$. To prove that property (i) holds for $n = k$ take $(\tau, \sigma) \in \mathcal{A}_k^c$. We get from Proposition 2 and (7)

$$\operatorname{bd}_{\mathcal{K}^{k-1}} \sigma = \operatorname{bd}_{\mathcal{K}} \sigma \cap \mathcal{K}^{k-1} \subset |\mathcal{A}| \cap \mathcal{K}^{k-1} \subset |\mathcal{A}_k|,$$

hence $\operatorname{bd}_{\mathcal{K}^{k-1}} \sigma = \tau$, i.e. (τ, σ) is an elementary coreduction pair in $\mathcal{K}^{k-1}$. Similarly, if $(\tau, \sigma) \in \mathcal{A}_k^r$, then

$$\operatorname{cbd}_{\mathcal{K}^{k-1}} \tau = \operatorname{cbd}_{\mathcal{K}} \tau \cap \mathcal{K}^{k-1} \subset |\mathcal{A}| \cap \mathcal{K}^{k-1} \subset |\mathcal{A}_n|.$$

Hence, $\operatorname{cbd}_{\mathcal{K}^{k-1}} \tau = \sigma$, i.e. (τ, σ) is an elementary reduction pair in $\mathcal{K}^{k-1}$. Thus (i) holds for $n = k$. To prove that also (ii) is satisfied: observe that for each

Algorithm 9 checkIfBoundaryless

```
1. for every $\sigma \in Simp$
   (a) if $\sigma.deleted == false$ then
       i. for every $\tau \in \sigma.faces()$
          if $\tau.deleted == false$ then return false;
2. return true;
```

$\alpha = (\tau, \sigma) \in \mathcal{A}_k^c$ the set $|\alpha|$ is closed in $\mathcal{K}^{k-1}$. Therefore, by Proposition 1 the set $|\mathcal{A}_k^c|$ is closed in $\mathcal{K}^{k-1}$. It follows from Theorem 2 and Theorem 3 that $\mathcal{K}^{k-1} \setminus |\mathcal{A}_k^c|$ is a regular subset of $\mathcal{K}^{k-1}$. We claim that for each $\alpha = (\tau, \sigma) \in \mathcal{A}_k^r$ the pair α is an elementary reduction pair in $\mathcal{K}^{k-1} \setminus |\mathcal{A}_k^c|$. Indeed,

$$\operatorname{cbd}_{\mathcal{K}^{k-1} \setminus |\mathcal{A}_k^c|} \tau = \operatorname{cbd}_{\mathcal{K}^{k-1}} \tau \setminus |\mathcal{A}_k^c| = \{\sigma\} \setminus |\mathcal{A}_k^c| = \{\sigma\},$$

because for each $\bar{\alpha} \in \mathcal{A}_k^c$ we have $\lambda(\bar{\alpha}) = k = \lambda(\alpha)$, hence $|\mathcal{A}_k^c| \cap \{\sigma\} = \emptyset$. It follows that for each $\alpha \in \mathcal{A}_k^r$ the set $|\alpha|$ is open in $\mathcal{K}^{k-1} \setminus |\mathcal{A}_k^c|$. Therefore, by Proposition 1 the set $|\mathcal{A}_k^r|$ is open in $\mathcal{K}^{k-1} \setminus |\mathcal{A}_k^c|$ and by Theorem 3

$$\mathcal{K}^k = \mathcal{K}^{k-1} \setminus |\mathcal{A}_k| = \mathcal{K}^{k-1} \setminus |\mathcal{A}_k^c| \setminus |\mathcal{A}_k^r|$$

is a regular subset of $\mathcal{K}^{k-1} \setminus |\mathcal{A}_k^c|$. Thus, by Proposition 3, $\mathcal{K}^k$ is a regular subset of $\mathcal{K}^{k-1}$, which proves (ii) for $n = k$.

Now, let φ_n be any sequence ordering the S-reduction pairs in $\mathcal{A}_n$. Since by (i) $\mathcal{A}_n$ is free, it follows from Theorem 4 that φ_n is a reduction sequence. Therefore, we have a well defined isomorphism $I_{\varphi_n} : H(\mathcal{K}^{n-1}) \to H(\mathcal{K}^n)$ and the composition of all the isomorphism I_{φ_n} gives the isomorphism required in (6). $\square$

To check if the S-complex resulting from the reduction process is boundaryless, one can use Algorithm 9. One easily verifies that if the resulting S-complex is boundaryless, then all the sensors return a value of true from this algorithm.

In case Conjecture 1 does not hold and the final complex is not boundaryless, we expect it to be very small so that we can easily transfer it to a single selected sensor via the communication channels available in the network. For this, we first create a spanning tree of the network using standard techniques for a distributed setting as in [2]. This may be achieved in $O(V^{1.6} + E)$, where V is a number of sensors, E is a number of edges in the flag complex. Then, we move the information along the spanning tree to its root and use the root sensor to compute the generators of the homology of the remaining complex, for instance using homology algorithm available in [33] or [34]. Finally, we use the spanning tree again to send the homology generators back to the respective simplices.

8 Verifying coverage

In this section we present an algorithm verifying the assumptions of Theorem 1. For the sake of simplicity, we assume that Conjecture 1 holds and that after applying Algorithm 8 there is left only one boundaryless 2-simplex ω. Let $\hat{\omega}$ denote the associated elementary chain which sends ω to one and everything else to zero. The nodes need to verify whether there exists a homology class $[c] \in H(\mathcal{F}, \mathcal{C})$ whose boundary is nonzero. By applying Algorithm 8 the nodes know the homology of $H(\mathcal{F}, \mathcal{C})$ but only via the isomorphic homology of $\mathcal{K}^f$, where $\mathcal{K}^f$ is given by (5). Therefore, it is necessary to find the isomorphic counterpart of $[\hat{\omega}]$ in $H(\mathcal{F}, \mathcal{C})$. To achieve this it is sufficient to apply formula (15) in Proposition 5 repetitively in proper order, for every S-reduction pair reduced by Algorithm 8. The respective algorithm is Algorithm 10. We make the following two assumptions concerning this algorithm:

- To store the resulting chain an extra field `coef` is added to the data structure `Simplex` and the field is initially set to `undefined`.
- The nodes remember not only simplices which were reduced but also whether they were reduced in an elementary reduction or coreduction.

In the sequel we use the natural pairing of the elements of $\mathcal{G}^r$ and $\mathcal{G}^c$ coming from S-reduction pairs in the form of a bijection

$$\mathcal{G}^r \cup \mathcal{G}^c \ni \sigma \mapsto \sigma^* \in \mathcal{G}^r \cup \mathcal{G}^c$$

which sends an element $\sigma \in \mathcal{G}^r \cup \mathcal{G}^c$ to its companion in the respective S-reduction pair.

The following proposition is straightforward.

Proposition 4 *If for some $\sigma \in \mathtt{Simp}[2]$ the value of $\sigma.\mathtt{coef}$ is not set after completing loop (1) of Algorithm 10, then $\sigma \in \mathcal{G}^c$ and* $\dim \sigma^* = 1$.

Lemma 3 *Assume that given an $n \in \mathbb{N}$ Algorithm 10 sets the value $\sigma.\mathtt{coef}$ for every $\sigma \in \mathtt{Simp}[2]$ such that $(\sigma^*, \sigma) \in \mathcal{G}^c$ and $\lambda(\sigma^*, \sigma) > n$. Then $\sigma.\mathtt{coef}$ is set for every $\sigma \in \mathtt{Simp}[2]$ such that $(\sigma^*, \sigma) \in \mathcal{G}^c$ and $\lambda(\sigma^*, \sigma) = n$.*

Proof Assume $\sigma_0 \in \mathtt{Simp}[2]$ is such that

$$(\sigma_0^*, \sigma_0) \in \mathcal{G}^c \text{ and } \lambda(\sigma_0^*, \sigma_0) = n.$$

Then $\tau_0 := \sigma_0^* \in L$. Let $\sigma \in \operatorname{cbd} \tau_0 \setminus \sigma_0$. If $\sigma \in \mathcal{G}^r \cup \mathcal{K}^f$ or $\sigma \in \mathcal{G}^c$ and $\dim \sigma^* \neq 1$, then by Proposition 4 $\sigma.\mathtt{coef}$ is set already in loop (1). Consider the other case when $\sigma \in \mathcal{G}^c$ and $\dim \sigma^* = 1$. Then $\operatorname{bd} \sigma \setminus \sigma^* \subset |\mathcal{A}(\sigma^*, \sigma)|$. Since $\tau_0 \in \operatorname{bd} \sigma$ and $\tau_0 \neq \sigma^*$, we get $(\sigma_0^*, \sigma_0) \in \mathcal{A}(\sigma^*, \sigma)$. Therefore $\lambda(\sigma^*, \sigma) > \lambda(\sigma_0^*, \sigma_0) = n$ and by the assumption $\sigma_0.\mathtt{coef}$ is set also in this case. It follows that when τ_0 is considered in line $(3.a)$, the condition in line $(3.a.i)$ is satisfied and consequently $\sigma.\mathtt{coef} = \tau_0^*.\mathtt{coef}$ is set in line $(3.a.i.C)$. □

Theorem 7 *Algorithm 10 terminates in all nodes. Moreover, the assumptions of Theorem 1 are satisfied if and only if at least one node exits the algorithm with the report "coverage verified".*

Algorithm 10 verifyCoverage

```
1. for each $\sigma \in \texttt{Simp}[2]$
   (a) if $\sigma \in \mathcal{K}^f$ set $\sigma.\texttt{coef} := 1$;
   (b) if $\sigma \in \mathcal{G}^r$ set $\sigma.\texttt{coef} := 0$;
   (c) if $\sigma \in \mathcal{G}^c$ and $\dim \sigma^* \neq 1$ set $\sigma.\texttt{coef} := 0$;
2. Let $L := \{\, \tau \in \texttt{Simp}[1] \cap \mathcal{G}^c \mid \dim \tau^* = 2 \,\}$;
3. while $(L \neq \emptyset)$
   (a) for each $\tau \in L$
         i. if $\sigma.\texttt{coef}$ is set for each $\sigma \in \operatorname{cbd} \tau \setminus \tau^*$
            A. $s := 0$;
            B. for each $\sigma \in \operatorname{cbd} \tau \setminus \tau^*$ do
                  $s \mathrel{+}= \sigma.\texttt{coef} / \kappa(\sigma, \tau)$;
            C. $\tau^*.\texttt{coef} := s$;
            D. $L := L \setminus \tau$;
4. for each $\tau \in \texttt{Simp}[1]$ such that $\tau$ is a fence edge
   (a) $s := 0$;
   (b) for each $\sigma \in \operatorname{cbd} \tau$ do $s \mathrel{+}= \sigma.\texttt{coef} * \kappa(\sigma, \tau)$;
   (c) if $s \neq 0$ report "coverage verified" and exit;
```

Proof First we will show that the algorithm sets $\sigma.\texttt{coef}$ for each $\sigma \in \texttt{Simp}[2]$. By Proposition 4 we need to consider only the case when $\sigma \in \mathcal{G}^c$ and $\dim \sigma^* = 1$. However, this case follows immediately from Lemma 3 by induction with respect to $n := N - \lambda(\sigma^*, \sigma)$, where

$$N := \max \{\, \lambda(\alpha) \mid \alpha \in \mathcal{A} \,\}.$$

Observe that whenever $\sigma.\texttt{coef}$ is set in line $(3.a.i.C)$, then σ^* is removed from L in line $(3.a.i.D)$. Therefore, the loop in line (3) completes and consequently also Algorithm 10 completes in every node.

It is now straightforward to verify that after completing loop (3), the fields $\sigma.\texttt{coef}$ for $\sigma \in \texttt{Simp}[2]$ store a chain $c_\omega \in \mathcal{K}$ whose homology is the image of $[\hat{\omega}]$ under the isomorphism established in Theorem 6.

Now, the variable s evaluated in loop (4) stores $\langle \partial c_\omega, \tau \rangle$ for some $1-$simplex τ of the fence. Therefore, if at least one node reports "coverage verified", then the assumptions of Theorem 1 are satisfied and the coverage is indeed archived. If the assumptions are not satisfied, then obviously no node can report "coverage verified". □

In case, when there is more then one boundaryless 2-simplex left at the end of the reduction process, the described algorithm may be called for each of the remaining 2-simplices and if at least one of the nodes reports "coverage verified" for at least one of the remaining 2-simplices, the assumptions of Theorem 1 are satisfied. The adaptation of Algorithm 10 to the case when

Conjecture 1 does not hold is only slightly more complicated. It requires finding the homology generators of $H(\mathcal{K}^f)$ by some algebraic means, for instance by applying the algorithm described in [1] and then starting Algorithm 10 with the chains representing the homology generators of $H(\mathcal{K}^f)$ instead of the elementary chain $\hat{\omega}$. Therefore, the extra assumptions from the beginning of the section are not restrictive.

9 On complexity

The full complexity analysis of the presented algorithms is not possible; in this paper we consider neither a concrete communication model nor the synchronization methods needed to collate algorithm phases. However, even at the level of generality with which our analysis is performed, some basic remarks toward a more complete analysis are possible. The details of a full complexity analysis, given length and technicality, are to be published elsewhere.

We make the following simplifying assumptions:

1. Every sensor can send an integer to its neighbors without error and in one time unit.
2. A global synchronization method which may be used to synchronize in constant time the consecutive steps of the algorithms in the sensors is available.
3. The implementation of the algorithm is based on best performing data structures for finding simplices, queues for reduction candidates etc.

One can show that under these assumptions the algorithm constructing the flag complex (Algorithm 2) and the algorithm reducing it (Algorithm 8) may be implemented in such a way that the complexity for one sensor (compare network time defined in Section 10) is $O(n^2 + K * (n \log K + n^2))$, where:

1. n is the upper bound for the number of sensors in the neighborhood of a given sensor.
2. K is the upper bound for the number of simplices which have the given sensor as a vertex.

In other words, the complexity depends only on the local size of the network, and not on the size of the whole network as in the case of centralized computations. This justifies the utility of the presented algorithms in the context of large-scale sensor networks of bounded density. The analysis depends on the Conjecture 1, because the estimate does not include the potential postprocessing of the final S-complex which is not boundaryless, although we expect this cost to be negligible even if nonzero.

10 Simulations

The parallel reduction algorithms presented in the paper have been implemented in Java programming language [35]. The code executes the algorithms

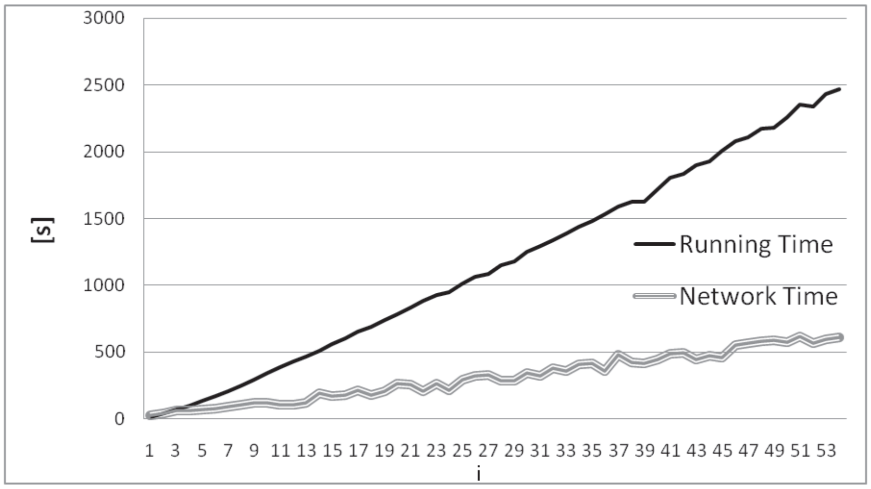

Fig. 5 Network Time and Running Time for the family L^{53} of networks.

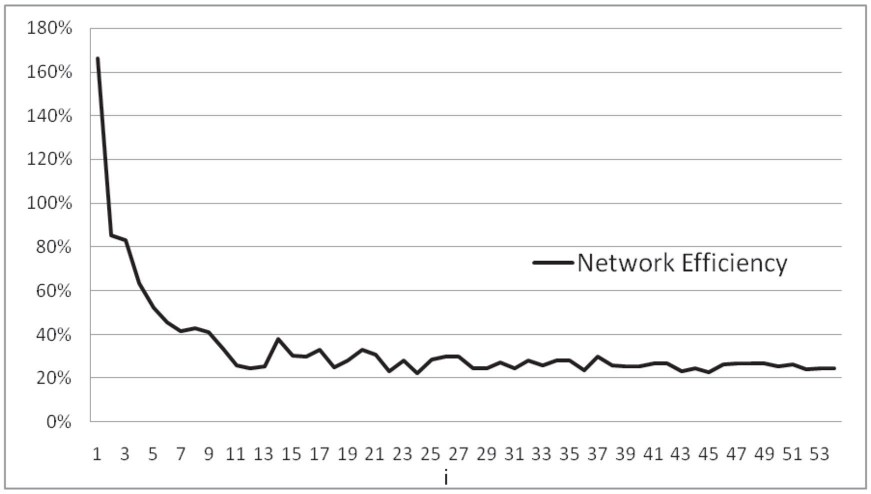

Fig. 6 Efficiency of the network for the family L^{53} of networks.

on each node in a separate thread, so that the simulations may be performed with only a few or even one processor unit. The real time of the network is also simulated.

In this section we present the experimental results showing the advantages of the distributed homology computation. In the presented series of experiments, the following configuration, referred to as the BASE TEST will be used. The base test is a small network of 23 nodes randomly placed on a 4×4 units rectangle with the communication radius fixed at 2 units. This base test will be copied and shifted and new fence cycles created. This process is performed to obtain a reasonably uniform distribution of the nodes on larger networks

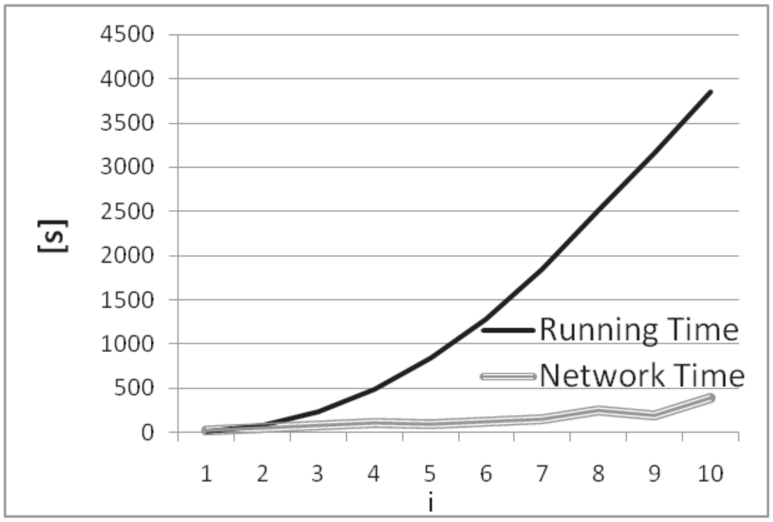

Fig. 7 Network Time and Running Time for the family S^{10} of networks.

and to avoid excessive local clustering which results in high dimensional flag complexes.

We first define a family L^n of networks on a sequence of 'linear' domains, $L^n = \{ L_i \}_{i=0}^{n}$, where L_i is defined recursively as follows:

1. L_0 is the base test case with a rectangular fence cycle $\mathcal{C}$ around the network; and
2. L_i is created from L_{i-1} by placing a new copy of the base test on the right side of L_{i-1} with a new rectangular fence cycle $\mathcal{C}$ around the network.

Another family $S^n = \{ S_i \}_{i=0}^{n}$ of 'square' networks is defined recursively as follows:

1. S_0 is the base test case with a rectangular fence cycle $\mathcal{C}$ around the network; and
2. S_i is created from S_{i-1} by placing new copies of the base test: i-copies on the right, i-copies on the bottom side, and one on the bottom-right. A new rectangular fence $\mathcal{C}$ around the network is created.

Let C_s denote the CPU time used by node `s` during the simulation and let R_s be the real time used by node `s`, i.e., the world time passing from the moment the node starts the computations until the moment it completes. We define the *network time* as $\max_{s \in S} R_s$ and the *running time* as $\sum_{s \in S} C_s$. In other words, the network time is the time required by the real network to do the computation. In the case of the simulations performed on a single machine with a single CPU, the running time is the CPU usage for the whole simulation. We define the network efficiency as the ratio

$$\texttt{Network Efficiency} := \frac{\texttt{Network Time}}{\texttt{Running Time}} \cdot 100\%.$$

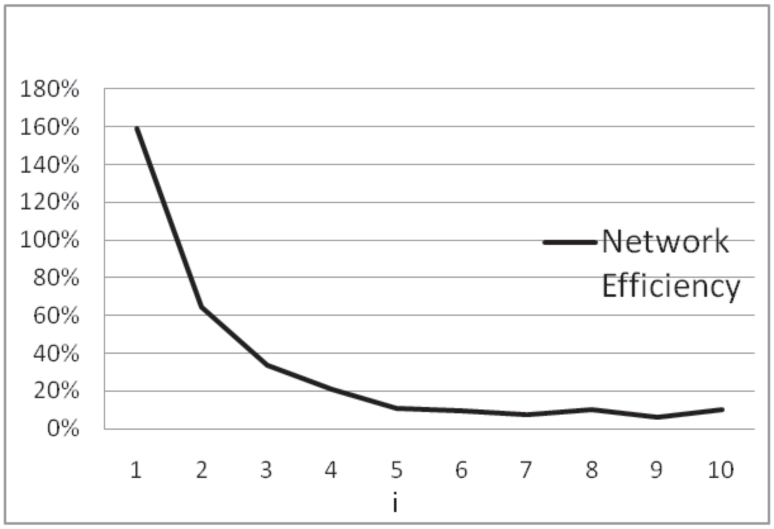

Fig. 8 Efficiency of the network for the family S^{10} of networks.

For the families L^{53} and S^{10} we ran simulations and measured network time, running time and network efficiency. Figure 5 presents the network and running time as the functions of the size of the network for the family L^{53}. The network efficiency for the same data is presented in Figure 6. Analogous results for the family S^{10} are in Figure 7 and Figure 8. The outcome of the experiments clearly indicates the advantage gained through distributed computation.

Appendix

A Primer on homology theory

This abbreviated introduction to homological tools assumes a knowledge of basics from algebra and topology. From algebra, the notions of rings (algebraic generalizations of $\mathbb{Z}$) and modules (algebraic generalizations of vector spaces) are assumed, as are homomorphisms (algebraic generalizations of linear transformations), kernels, and the like. From topology, only basic notions of simplicial complexes are needed.

A.1 Chain complexes

Homology counts objects with cancelation to provide a topological invariant in algebraic terms. The simplest version of homology is SIMPLICIAL HOMOLOGY of a simplicial complex. Fix a finite simplicial complex X. The building blocks of a rudimentary homology for X are as follows.

1. Fix a coefficient ring R.

2. Grade the simplices of X by dimension.
3. Define k-CHAINS C_k as the R-module with basis the k-simplices of X.
4. Consider the BOUNDARY MAPS — the linear transformations $\partial : C_k \to C_{k-1}$ which send a basis k-simplex to its boundary faces (as an abstract sum of basis $k-1$-simplices, each with coefficient ± 1 depending on parity).

The sequence of chains and boundary maps are assembled into a CHAIN COMPLEX — a sequence $\mathcal{C}$ of R-modules C_k and homomorphism $\partial_k : C_k \to C_{k-1}$ with $\partial_k \circ \partial_{k+1} = 0$ for all k. A chain complex is often written out as a diagram:

$$\cdots \xrightarrow{\partial_{n+1}} C_n \xrightarrow{\partial_n} C_{n-1} \xrightarrow{\partial_{n-1}} \cdots \xrightarrow{\partial_1} C_0 \xrightarrow{\partial_0} 0. \tag{11}$$

It is notationally simpler to denote the chain complex as a single object $\mathcal{C} = (C_*, \partial_*)$ and to write ∂ for the boundary operator acting on any chain of unspecified grading. Chain complexes need not be generated by simplices of a simplicial complex: they are decidedly algebraic devices. The key requirement for a chain complex is that the *boundary of a boundary is null*: $\partial_k \circ \partial_{k+1} = 0$ for all k.

A.2 Homology

Homology counts equivalence classes of certain chains with regards to the boundary maps. A CYCLE of $\mathcal{C}$ is a chain with empty boundary, i.e., an element of ker ∂. Homology is an equivalence relation on cycles of $\mathcal{C}$. Two cycles in $Z_k :=$ ker ∂_k are HOMOLOGOUS if they differ by an element of $B_k :=$ im ∂_{k+1}. The HOMOLOGY of $\mathcal{C}$ is the sequence of quotient modules $H_k(\mathcal{C})$, for $k \in \mathbb{N}$, given by:

$$H_k(\mathcal{C}) := Z_k / B_k = \text{ker } \partial_k / \text{ im } \partial_{k+1}. \tag{12}$$

Here, $Z_k :=$ ker ∂_k is the submodule of CYCLES in C_k, and $B_k :=$ im ∂_{k+1} is the submodule of BOUNDARIES. Elements of $H_k(C)$ are HOMOLOGY CLASSES. We write $H(\mathcal{C})$ to denote the full module of graded homologies $H_k(\mathcal{C})$. When $\mathcal{C}$ is the chain complex generated by a simplicial complex X, we write $H(X)$ for its homology: it is an invariant of the space X up to homotopy.

A.3 Relative homology

Homology is defined for any chain complex $\mathcal{C} = (C_k, \partial_k)$. Although taking C_k to be a module generated by k-dimensional simplices is common, it is by no means exclusive. Many of the constructs of this paper rely on a modified chain complex giving rise to REDUCED HOMOLOGY relative to a subcomplex. Let $A \subset X$ be a (necessarily closed) subcomplex of a simplicial complex X. Then both X and A define simplicial chain complexes, with the result that $C_k(A) \subset C_k(X)$ is a submodule for all k. Thus, one can take the quotient module $C_k(X, A) := C_k(X)/C_k(A)$, so that a relative chain is an equivalence class of chains relative to simplices in A. The boundary map ∂_k extends naturally to

$\partial_k : C_k(X, A) \to C_{k-1}(X, A)$ in a manner that preserves the $\partial^2 = 0$ condition. Therefore, the RELATIVE HOMOLOGY $H_k(X, A)$ is well-defined and measures RELATIVE CYCLES (chains in X whose boundaries lie in A) modulo relative boundaries.

A.4 Functoriality

A CHAIN MAP is a map $\varphi : \mathcal{C} \to \mathcal{C}'$ between chain complexes that is a homomorphism on chains respecting the grading and commuting with the boundary maps. This is best expressed in the form of a COMMUTATIVE DIAGRAM:

$$\begin{tikzcd}
\cdots \arrow[r] \arrow[d, "\varphi"] & C_{n+1} \arrow[r, "\partial"'] \arrow[d, "\varphi"] & C_n \arrow[r, "\partial"'] \arrow[d, "\varphi"] & C_{n-1} \arrow[r] \arrow[d, "\varphi"] & \cdots \arrow[d, "\varphi"] \\
\cdots \arrow[r] & C'_{n+1} \arrow[r, "\partial'"'] & C'_n \arrow[r, "\partial'"'] & C'_{n-1} \arrow[r] & \cdots
\end{tikzcd} \tag{13}$$

Commutativity means that homomorphisms are path-independent in the diagram; e.g., $\varphi \circ \partial = \partial' \circ \varphi$. Chain maps are the analogues of continuous maps, given their respect for the boundary operators: neighbors are sent to neighbors.

A chain map φ induces homomorphisms on homology groups, written $H(\varphi) : H(\mathcal{C}) \to H(\mathcal{C}')$, sending $[\zeta] \in H_k(\mathcal{C})$ to $[\varphi(\zeta)] \in H_k(\mathcal{C}')$. The reader may check that this is a well-defined homomorphism. We denote by $H(\varphi)$ the full sequence of INDUCED HOMOMORPHISMS on homology. Homology is FUNCTORIAL, meaning that induced homomorphisms respect composition of chain maps. Specifically,

1. The identity chain map id induces isomorphisms $H(\text{id}) : H(\mathcal{C}) \to H(\mathcal{C})$.
2. Composable chain maps φ and ψ satisfy $H(\psi \circ \varphi) = H(\psi) \circ H(\varphi)$.

A.5 Exact sequences

Homology computations are greatly aided by a theoretical tool called an EXACT SEQUENCE. Any complex $=(A_k, \phi_k)$ of abelian groups and homomorphism is EXACT when its homology vanishes: ker $\phi_n =$ im ϕ_{n+1} for all n. An exact chain complex is the chain analogue of a nullhomologous space. Exact sequences are most often used to prove isomorphisms between various homologies by means of zeroing out terms in an exact sequence. For example, if some subsequence of an exact sequence read as:

$$\cdots \to 0 \to A \overset{\phi}{\to} B \to 0 \to \cdots ,$$

then it follows that $\phi : A \to B$ is an isomorphism.

B Proofs of homological S-reduction isomorphisms

The goal of the Appendix is to give a detailed proof of Theorem 4: namely, any full ordering of a free collection of S-reduction pairs forms a reduction sequence and two interchangeable reduction sequences give rise to the same isomorphism in homology.

In order to prove this, we need to introduce several terms and lemmas, after recalling related work. We say that a regular subset $\mathcal{T} \subset \mathcal{K}$ is a NULLSET of $\mathcal{K}$ if $\mathcal{T}$ is closed or open in $\mathcal{K}$ and $H(\mathcal{T}) = 0$.

Theorem 8 *[23, Theorem 4.1] Assume α is an S-reduction pair in a subcomplex $\mathcal{L}$ of an S-complex $\mathcal{K}$. If α is an elementary reduction pair then $|\alpha|$ is open in $\mathcal{L}$. If α is an elementary coreduction pair then $|\alpha|$ is closed in $\mathcal{L}$. Moreover, in both cases, $|\alpha|$ is a nullset.*

Theorem 9 *[23, Theorem 3.4] Let $\mathcal{K}$ and $\mathcal{K}'$ be S-complexes. Assume $\mathcal{K}' \subset \mathcal{K}$ is closed in $\mathcal{K}$. Let $\mathcal{K}'' := \mathcal{K} \setminus \mathcal{K}'$ and $\kappa' := \kappa|_{\mathcal{K}' \times \mathcal{K}'}$, $\kappa'' := \kappa|_{\mathcal{K}'' \times \mathcal{K}''}$. Then the inclusion ι and the projection π defined on generators by*

$$\iota : (R[\mathcal{K}'], \partial^{\kappa'}) \ni \sigma \mapsto \sigma \in (R[\mathcal{K}], \partial^{\kappa})$$

$$\pi : (R[\mathcal{K}], \partial^{\kappa}) \ni \sigma \mapsto \begin{cases} \sigma & \text{if } \sigma \in \mathcal{K}'' \\ 0 & \text{otherwise} \end{cases} \in (R[\mathcal{K}''], \partial^{\kappa''})$$

are chain maps. Moreover, we have the following short exact sequence

$$0 \to R[\mathcal{K}'] \xrightarrow{\iota} R[\mathcal{K}] \xrightarrow{\pi} R[\mathcal{K}''] \to 0$$

and the following long exact sequence of homology modules

$$\cdots \xrightarrow{\delta} H_q(\mathcal{K}') \xrightarrow{H(\iota)} H_q(\mathcal{K}) \xrightarrow{H(\pi)} H_q(\mathcal{K}'') \xrightarrow{\delta} \cdots$$

Lemma 4 *If α is an S-reduction pair in an S-complex $\mathcal{K}$, then there is an isomorphism $\gamma_{(\mathcal{K},\alpha)} : H(\mathcal{K} \setminus |\alpha|) \to H(\mathcal{K})$ given by*

$$\gamma_{(\mathcal{K},\alpha)} := \begin{cases} H(\iota_{(\mathcal{K},\alpha)}) & \text{if } \alpha \text{ is an elementary reduction pair,} \\ H(\pi_{(\mathcal{K},\alpha)})^{-1} & \text{if } \alpha \text{ is an elementary coreduction pair,} \end{cases} \tag{14}$$

where $\iota_{(\mathcal{K},\alpha)} : R[\mathcal{K} \setminus |\alpha|] \to R[\mathcal{K}]$ and $\pi_{(\mathcal{K},\alpha)} : R[\mathcal{K}] \to R[\mathcal{K} \setminus |\alpha|]$ are the maps discussed in Theorem 9.

Proof Consider first the case when $\alpha = (\tau, \sigma)$ is an elementary reduction pair. It follows from Theorem 8 that $|\alpha|$ is open in $\mathcal{K}$ and a nullset. Therefore $\mathcal{K} \setminus |\alpha|$ is closed in $\mathcal{K}$ and by Theorem 9 we have the following exact sequence

$$\cdots \xrightarrow{\delta} H_q(\mathcal{K} \setminus |\alpha|) \xrightarrow{H_q(\iota)} H_q(\mathcal{K}) \xrightarrow{H_q(\pi)} H_q(|\alpha|)) \xrightarrow{\delta} \cdots$$

Since $H(|\alpha|) = 0$, the map $H(\iota_{(\mathcal{K},\alpha)}) : H(\mathcal{K} \setminus |\alpha|) \to H(\mathcal{K})$ is an isomorphism.

Consider in turn the case when α is an elementary coreduction pair. By Theorem 8 the set $|\alpha|$ is closed in $\mathcal{K}$ and a nullset. Therefore, we have the following exact sequence

$$\cdots \xrightarrow{\delta} 0 \xrightarrow{H_q(\iota)} H_q(\mathcal{K}) \xrightarrow{H_q(\pi)} H_q(\mathcal{K} \setminus |\alpha|) \xrightarrow{\delta} 0 \cdots$$

and consequently $H(\pi_{(\mathcal{K},\alpha)}) : H(\mathcal{K}) \to H(\mathcal{K} \setminus |\alpha|)$ is an isomorphism. □

In the sequel, whenever the S-complex $\mathcal{K}$ is clear from the context we will write simply γ_α instead of $\gamma_{(\mathcal{K},\alpha)}$.

The common notation for the two (in principle different) isomorphisms is justified by the following result which is an immediate consequence of [24, Corollary 2.7].

Proposition 5 *The isomorphism $\gamma_{(\mathcal{K},\alpha)}$ is induced by the chain map $\eta_{(\mathcal{K},\alpha)} : C(\mathcal{K} \setminus |\alpha|) \to C(\mathcal{K})$ given by*

$$\eta_{(\mathcal{K},\alpha)}(c) = c - \frac{\langle \partial c, \tau \rangle}{\langle \partial \sigma, \tau \rangle} \sigma, \tag{15}$$

where $\alpha = (\tau, \sigma)$ and the boundary ∂ denotes the boundary map in $\mathcal{K}$.

In the sequel it will be convenient to reformulate the concept of a reduction sequence as a path in a graph. A STATE GRAPH of the S-complex $\mathcal{K}$ is a directed graph $G_{\mathcal{K}} = (V, E)$ whose set of vertices consists of regular subsets of $\mathcal{K}$ and a pair $(\mathcal{L}, \mathcal{L}')$ of vertices of $G_{\mathcal{K}}$ is an edge in $G_{\mathcal{K}}$ if there exist an S-reduction pair α in $\mathcal{L}$ such that $\mathcal{L}' = \mathcal{L} \setminus |\alpha|$. We then refer to α as the label of the edge.

The reduction process in the context of the state graph takes the form of a REDUCTION PATH, which is just any directed path in $G_{\mathcal{K}}$. It is straightforward to observe that any reduction path originating in a vertex $\mathcal{L} \in G_{\mathcal{K}}$ defines a reduction sequence in $\mathcal{L}$ consisting of the labels of the edges along the path. Similarly, any reduction sequence in a subcomplex $\mathcal{L}$ defines a reduction path originating in $\mathcal{L}$, so in the sequel, we will use the concepts of the reduction path and the reducton sequence interchangeably. Given a reduction path ψ in the state graph $G_{\mathcal{K}}$, for $i = 1, 2, \ldots, n+1$ we denote by $\mathcal{K}_i^\psi$ the ith S-complex (vertex) along the path.

Let $\mathcal{K}$ and $\mathcal{K}'$, where $\mathcal{K}' \subset \mathcal{K}$, be two nodes in the state graph for which there exists a reduction path ψ from $\mathcal{K}$ to $\mathcal{K}'$ of length n. The isomorphism

$$H(\psi) : H(\mathcal{K}') \to H(\mathcal{K})$$

is defined as the composition

$$H(\psi) := \gamma_{\psi_1} \circ \gamma_{\psi_2} \circ \ldots \circ \gamma_{\psi_n}$$

of isomorphisms γ_{ψ_i} induced by the S-reduction pairs along the path. It is called the homology of the reduction path. Since every reduction sequence φ in a regular subcomplex $\mathcal{L}$ of $\mathcal{K}$ defines a reduction path in the state graph

$G_{\mathcal{K}}$ originating from $\mathcal{L}$, we define $H(\varphi)$ as the homology of the associated reduction path.

Note that if $\mathcal{L}$ is a subcomplex of $\mathcal{K}$ and α is an S-reduction pair in $\mathcal{L}$ then it need not be an S-reduction pair in $\mathcal{K}$. However, the following proposition is a straightforward consequence of Proposition 2.

Proposition 6 *If $\mathcal{K}'$ is a regular subcomplex of an S-complex $\mathcal{K}$ and α is an S-reduction pair in $\mathcal{K}$ such that $|\alpha| \subset \mathcal{K}'$ then α is an S-reduction pair in $\mathcal{K}'$.*

The following lemma is needed to prove Theorem 4.

Lemma 5 *If α_1, α_2 are two S-reduction pairs in $\mathcal{K}$ with disjoint supports, then the sequences (α_1, α_2) and (α_2, α_1) are reduction sequences in $\mathcal{K}$ and*

$$H(\alpha_1, \alpha_2) = \gamma_{\alpha_1} \circ \gamma_{\alpha_2} = \gamma_{\alpha_2} \circ \gamma_{\alpha_1} = H(\alpha_2, \alpha_1). \tag{16}$$

Proof For $i = 1, 2$ let $\mathcal{K}_i := \mathcal{K} \setminus |\alpha_i|$ and $\mathcal{K}_{12} := \mathcal{K} \setminus |\alpha_1| \setminus |\alpha_2|$. It follows from Proposition 6 that α_{3-i} is an S-reduction pair in $\mathcal{K}_i$. Therefore, we conclude from Theorems 8, 2 and 3 that $\mathcal{K}_1$, $\mathcal{K}_2$ and $\mathcal{K}_{12}$ are S-complexes.

If both α_1 and α_2 are elementary coreduction pairs in $\mathcal{K}$, then we have the commutative diagram of projections:

$$\begin{tikzcd} R[\mathcal{K}_2] \arrow[r, "\pi_4"] & R[\mathcal{K}_{12}] \\ R[\mathcal{K}] \arrow[u, "\pi_2"] \arrow[r, "\pi_1"] & R[\mathcal{K}_1] \arrow[u, "\pi_3"] \end{tikzcd} \tag{17}$$

Similarly, if both α_1 and α_2 are elementary reduction pairs in $\mathcal{K}$, then we have the commutative diagram of inclusions:

$$\begin{tikzcd} R[\mathcal{K}_2] \arrow[d, "i_2"] & R[\mathcal{K}_{12}] \arrow[l, "i_4"'] \arrow[d, "i_3"] \\ R[\mathcal{K}] & R[\mathcal{K}_1] \arrow[l, "i_1"'] \end{tikzcd}$$

and when α_1 is an elementary coredution pair and α_2 is an elementary reduction pair in $\mathcal{K}$, we have the following commutative diagram of inclusions and projections:

$$\begin{tikzcd} R[\mathcal{K}_1] & R[\mathcal{K}_{12}] \arrow[l, "i_1"'] \\ R[\mathcal{K}] \arrow[u, "\pi_1"] & R[\mathcal{K}_2] \arrow[l, "i_2"'] \arrow[u, "\pi_2"] \end{tikzcd}$$

Due to functoriality of homology each of these diagrams induces a commutative diagram in homology and due to the definition (14) they all reduce to (16). □

Lemma 6 *Assume ψ is a reduction path in the state graph $G_{\mathcal{K}}$ of length n. If the S-reduction pair ψ_n in $\mathcal{K}_n^{\psi}$ is also an S-reduction pair in $\mathcal{K}_1^{\psi}$, then $\psi' := (\psi_n, \psi_1, \psi_2, \dots \psi_{n-1})$ is a reduction path in $\mathcal{K}_1^{\psi}$ and $H(\psi') = H(\psi)$.*

Proof We proceed by induction in the length of ψ. If the length is one, the conclusion is obvious. Thus, fix $n > 1$ and assume the conclusion holds for reduction paths of length less than n. It follows from Proposition 6 that ψ_n is an S-reduction pair in $\mathcal{K}^{\psi}_{n-1}$. Hence, both ψ_{n-1} and ψ_n are S-reduction pairs in $\mathcal{K}^{\psi}_{n-1}$ with disjoint supports and, by Lemma 5, (ψ_n, ψ_{n-1}) is a reduction path in $\mathcal{K}^{\psi}_{n-1}$ and $\gamma_{\psi_n} \circ \gamma_{\psi_{n-1}} = \gamma_{\psi_{n-1}} \circ \gamma_{\psi_n}$. Again using Proposition 6, we see that ψ' is a reduction sequence in $\mathcal{K}^{\psi}_1$ and we get from the induction hypothesis that

$$\begin{aligned} H(\psi') &= (\gamma_{\psi_n} \circ \gamma_{\psi_1} \circ \gamma_{\psi_2} \circ \cdots \circ \gamma_{\psi_{n-2}}) \circ \gamma_{\psi_{n-1}} \\ &= (\gamma_{\psi_1} \circ \gamma_{\psi_2} \circ \cdots \circ \gamma_{\psi_{n-2}} \circ \gamma_{\psi_n}) \circ \gamma_{\psi_{n-1}} \\ &= (\gamma_{\psi_1} \circ \gamma_{\psi_2} \circ \cdots \circ \gamma_{\psi_{n-2}}) \circ (\gamma_{\psi_n} \circ \gamma_{\psi_{n-1}}) \\ &= (\gamma_{\psi_1} \circ \gamma_{\psi_2} \circ \cdots \circ \gamma_{\psi_{n-2}}) \circ (\gamma_{\psi_{n-1}} \circ \gamma_{\psi_n}) \\ &= H(\psi). \end{aligned}$$

□

We are now ready to give the proof of Theorem 4.

Proof (of Theorem 4) In order to prove the first statement of the theorem take a free collection of potential S-reduction pairs in a subcomplex $\mathcal{L}$ of $\mathcal{K}$ and fix some it ordering $(\varphi_1, \varphi_2, \ldots, \varphi_n)$. We need to show that for each $j = 1, 2, \ldots, n$ the pair φ_j is an S-reduction pair in $\mathcal{L} \setminus \bigcup_{i=1}^{j-1} |\varphi_i|$. However, this follows immediately from Proposition 6.

Assume in turn that ψ and φ are two interchangeable reduction paths from $\mathcal{K}$ to $\mathcal{K}'$. In order to prove that $H(\psi) = H(\varphi)$ we will proceed by induction in the common length of ψ and φ denoted by n. If $n = 1$, there is only one path from $\mathcal{K}$ to $\mathcal{K}'$, hence $\psi = \varphi$ and $H(\psi) = H(\varphi)$. Thus, assume the conclusion holds for paths of length less then n. Since ψ and φ differ only by a permutation, there exists a $p \leq n$ such that

$$\psi_1 = \varphi_p. \tag{18}$$

Therefore, φ_p is an S-reduction pair in $\mathcal{K}$ and by Lemma 6 the sequence $(\varphi_p, \varphi_1, \varphi_2, \ldots, \varphi_{p-1})$ is a reduction path in $\mathcal{K}$ and consequently

$$\varphi' = (\varphi_p, \varphi_1, \varphi_2, \ldots, \varphi_{p-1}, \varphi_{p+1}, \ldots \varphi_n)$$

is a reduction sequence in $\mathcal{K}$. Moreover, we get from Lemma 6 that

$$\begin{aligned} H(\varphi) &= H(\varphi_1, \ldots, \varphi_p) \circ H(\varphi_{p+1}, \ldots, \varphi_n) \\ &= H(\varphi_p, \varphi_1, \ldots, \varphi_{p-1}) \circ H(\varphi_{p+1}, \ldots, \varphi_n) \\ &= H(\varphi'). \end{aligned}$$

Thus, we get from (18) and the induction assumption

$$\begin{aligned} H(\psi) &= H(\psi_1) \circ H(\psi_2, \psi_3, \dots, \psi_n) \\ &= H(\varphi_p) \circ H(\varphi_1, \varphi_2, \dots, \varphi_{p-1}, \varphi_{p+1}, \dots \varphi_n) \\ &= H(\varphi') \\ &= H(\varphi). \end{aligned}$$

□

References

1. Z. Arai, K. Hayashi, and Y. Hiraoka. Mayer-Vietoris sequences and coverage problems in sensor networks, preprint 2009.
2. B. Awerbuch, R. Gallager, ”A new distributed algorithm to find breadth first search trees”, *IEEE Transactions on Information Theory*, Vol. 33 Issue: 3, pp. 315 - 322, 1987.
3. L. Barrière, P. Fraigniaud, and L. Narayanan, “Robust position-based routing in wireless ad hoc networks with unstable transmission ranges,” In *Proc. Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, 2001.
4. G. Carlsson and V. de Silva. Zigzag Persistence, in *Proc. Found. of Computational Mathematics*, Jan 2009.
5. G. Carlsson, V. de Silva, and D. Morozov. Zigzag Persistent Homology and Real-valued Functions, in *Proc. Symp. on Comput. Geometry.*, June 2009.
6. E. Chambers, V. de Silva, J. Erickson, and R. Ghrist. Rips complexes of planar point sets, *Discrete Computat. Geom.*, 44(1), 75-90, 2010.
7. J. Cortes, S. Martinez, T. Karatas, and F. Bullo. Coverage control for mobile sensing networks, *Proc. IEEE Int. Conf. Robot. Autom.*, Washington, DC, 2002, vol. 2, pp. 13271332.
8. M. Damian, S. Pandit, and S. Pemmaraju, “Local approximation schemes for topology control.” In *Proc. ACM Symp. on Prin. of Dist. Comput. (PODC)* 2006, 208–217.
9. V. de Silva and R. Ghrist. Coordinate-free coverage in sensor networks with controlled boundaries via homology, *Intl. J. Robotics Research* **25**(2006), 1205–1222.
10. V. de Silva and R. Ghrist. Coverage in sensor networks via persistent homology, *Alg. & Geom. Top.*, 7, (2007), 339–358.
11. V. de Silva and R. Ghrist. Homological sensor networks, *Notices Amer. Math. Soc.*, 54(1), 10-17, 2007.
12. B. Eckmann, Harmonische funktionen und randwertaufgaben einem komplex, *Commentarii Math. Helvetici*, vol. 17, pp. 240245, 1945.
13. D. Estrin, D. Culler, K. Pister, and G. Sukhatme. Connecting the physical world with pervasive networks, *IEEE Pervasive Computing*, 1(1), (2002), 59–69.
14. S. Fekete, A. Kröller, D. Pfisterer, and S. Fischer, “Deterministic boundary recongnition and topology extraction for large sensor networks,” in *Algorithmic Aspects of Large and Complex Networks*, 2006.
15. S. Gelfand and Y. Manin, *Methods of Homological Algebra*, 2nd ed., Springer-Verlag, 2003.
16. R. Ghrist and Y. Hiraoka. Applications of sheaf cohomology and exact sequences to network coding, preprint, 2011.
17. A. Hatcher, *Algebraic Topology*, Cambridge University Press, 2002.
18. D. Kempe, A. Dobra, and J. Gehrke, Computing aggregate information using gossip, in Proc. Foundations of Computer Science, Cambridge, MA, Oct. 2003.
19. H. Koskinen, “On the coverage of a random sensor network in a bounded domain,” in *Proceedings of 16th ITC Specialist Seminar*, pp. 11-18, 2004.
20. F. Kuhn, R. Wattenhofer, and A. Zollinger, “Ad-hoc networks beyond unit disk graphs,” *Wirel. Netw.* 14, 5 (2008), 715-729.
21. X.-Y. Li, P.-J. Wan, and O. Frieder, “Coverage in wireless ad-hoc sensor networks” IEEE Transaction on Computers, Vol. 52, No. 6, pp. 753-763, 2003.

22. S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M. Srivastava. Coverage problems in wireless ad-hoc sensor networks, *IEEE INFOCOM* (2001) 13801387.
23. M. Mrozek and B. Batko, Coreduction homology algorithm, *Discrete and Computational Geometry*, **41**(2009), 96–118.
24. M. Mrozek and T. Wanner, Coreduction homology algorithm for inclusions and persistent homology, *Computers and Mathematics with Applications*, accepted.
25. A. Muhammad and M. Egerstedt. Control using higher order Laplacians in network topologies, in *Proc. of the 17th International Symposium on Mathematical Theory of Networks and Systems*, (2006) 10241038.
26. A. Muhammad and A. Jadbabaie. Decentralized computation of homology groups in networks by gossip, in *Proc. of American Control Conference* (2007), 34383443.
27. M. Robinson, Inverse problems in geometric graphs using internal measurements, arXiv:1008.2933v1, August 2010.
28. M. Robinson, Asynchronous logic circuits and sheaf obstructions, arXiv:1008.2729v1, August 2010.
29. A. Tahbaz Salehi and A. Jadbabaie. Distributed coverage verification in sensor networks without location information. *IEEE Transactions on Automatic Control,* 55(8), August 2010.
30. A. Tahbaz Salehi and A. Jadbabaie. Distributed coverage verification in sensor networks without location information *IEEE Conference on Decision and Control*, December 2008.
31. F. Xue and P. R. Kumar, “The number of neighbors needed for connectivity of wireless networks,” *Wireless Networks*, pp. 169-181, vol. 10, no. 2, March 2004.
32. H. Zhang and J. Hou, ”Maintaining Coverage and Connectivity in Large Sensor Networks,” in *International Workshop on Theoretical and Algorithmic Aspects of Sensor, Ad hoc Wireless and Peer-to-Peer Networks*, Florida, Feb. 2004
33. The RedHom homology algorithms library :
http://redhom.ii.uj.edu.pl
34. Computational Homology Project:
http://chomp.rutgers.edu
35. Sensor Network Simulator:
http://redhom.ii.uj.edu.pl/sensors/