# Preface

This volume contains the papers presented at Ural-PDC 2018: 4th Ural Workshop on Parallel, Distributed, and Cloud Computing for Young Scientists (`https://ural-pdc.org/2018/`) held on November 15, 2018 in Yekaterinburg, Russia.

Ural-PDC is jointly organized by the Krasovskii Institute of Mathematics and Mechanics and the Ural Federal University. The aim of the workshop is to build the community of young researchers who work on modern problems in parallel, distributed, cloud computing, and to provide an academic forum that fosters them to share their ideas and results.

The key topics of interest are high-performance computing, distributed computing, cloud computing, Big Data processing, parallel computing education, and various applications of parallel and distributed computing. We have received 20 submissions, each of which has been reviewed by at least two program committee members. The committee decided to accept 14 regualar papers and an invited paper on online encyclopaedia of numerical algorithms.

Last but not least, we are grateful to our program and organizing committees, who made the workshop possible.

December 12, 2018                                           Dmitry Ustalov
Yekaterinburg, Russia                                Andrey Sozykin
     Timofey Epanchintsev
     Mikhail Chernoskutov

# Table of Contents

## Program Committee

| | |
|---|---|
| Alexander Bersenev | Krasovskii Institute of Mathematics and Mechanics / Ural Federal University |
| Mikhail Chernoskutov | Krasovskii Institute of Mathematics and Mechanics / Ural Federal University |
| Pavel Drobintsev | Peter the Great St.Petersburg Polytechnic University |
| Timofey Epanchintsev | Krasovskii Institute of Mathematics and Mechanics / Ural Federal University |
| Aleksandr Igumnov | Krasovskii Institute of Mathematics and Mechanics / Ural Federal University |
| Svyatoslav Khamzin | Ural Federal University / Institute of Immunology and Physiology |
| Evgheniy Kuklin | Krasovskii Institute of Mathematics and Mechanics / Ural Federal University |
| Alina Latipova | South Ural State University |
| Dmitry Nikitenko | Lomonosov Moscow State University |
| Gleb Radchenko | South Ural State University |
| Yaroslav Salii | Krasovskii Institute of Mathematics and Mechanics / Ural Federal University |
| Alexander Semenov | JSC NICEVT |
| Vladislav Shchapov | Institute of Continuous Media Mechanics |
| Andrey Sozykin | Krasovskii Institute of Mathematics and Mechanics / Ural Federal University |
| Konstantin Ushenin | Ural Federal University |
| Dmitry Ustalov | University of Mannheim |
| Vladimir V. Voevodin | Research Computing Center of Lomonosov Moscow State University |
| Vladimir Zverev | Krasovskii Institute of Mathematics and Mechanics / Ural Federal University |

## Invited Reviewers

| | |
|---|---|
| Alexander Agarkov | JSC NICEVT |
| Ameer B. A. Alaasam | South Ural State University |
| Timur Ismagilov | JSC NICEVT |
| Alexei Podkorytov | South Ural State University |

# Organizing Committee

| | |
|---|---|
| Mikhail Chernoskutov | Krasovskii Institute of Mathematics and Mechanics / Ural Federal University |
| Dmitry Ustalov | University of Mannheim |
| Alexander Bersenev | Krasovskii Institute of Mathematics and Mechanics / Ural Federal University |
| Timofey Epanchintsev | Krasovskii Institute of Mathematics and Mechanics / Ural Federal University |
| Alexey Grigoriev | Krasovskii Institute of Mathematics and Mechanics |
| Anna Kotegova | Ural Federal University |
| Evgeniy Kuklin | Krasovskii Institute of Mathematics and Mechanics / Ural Federal University |
| Andrey Sozykin | Krasovskii Institute of Mathematics and Mechanics / Ural Federal University |

# Parallel Partitioning Without Branching of Inner Boundaries for Arbitrary Domain[*]

Ilyas Kadyrov[1,2], Sergey Kopysov[1,2], and Alexander Novikov[1]

[1] Udmurt State University, Izhevsk, Russia
**slasheeck@gmail.com**
[2] Institute of Mechanics, Udmurt Federal Research Center UB RAS, Izhevsk, Russia

**Abstract.** In this paper we consider an approach to the parallel partitioning of a arbitrary domain into connected subdomains without branching of internal boundaries. The Reeb graph is a simplified representation of the topology of the required domain. Algorithms are shown in this paper is a modification of the presented earlier algorithm of the formation of the plane Reeb graph. The algorithm for constructing the volume Reeb graph allows us to determine the internal topology of the domain. A new algorithm of the formation of the partitioning into subdomains without branching of internal boundaries is presented.

**Keywords:** Volume Reeb graph · Reeb graph · Unstructured mesh · Multiply connected domain · Topology definition · Mesh decomposition without branching

## 1 Reeb Graph Constructed by Surface Triangulation

A Reeb graph is a topological data structure which is able to capture the topology of shapes. Considering a scalar-valued function defined on the domain of interest, the level sets of this function may consist of several components, which evolve while sweeping through the function values. The Reeb graph encodes the evolution of these level set components, which are sometimes referred to as contours in the literature, by collapsing each contour to a point. Normal Reeb graph is a 2-dimentional graph, based and constructed on surface mesh. Volume Reeb graph is an array of subgraphs, that perform 3-dimentional graph.

Define a tree of connections between the critical nodes $v^c$ and construct the Reeb graph $R(V, E)$ [1] with edges $e(v_i, v_j) \in E$ for the surface triangulation $T$ in $\mathbb{R}^3$, consisting of 2-simplexes (Algorithm 1). We will use the Morse level function $f$ to determine the critical nodes $v^c$ and introduce the parameter $c(v^c)$ corresponding to the number of edges outgoing from it and taking the value: 1 - one node ; 2 - two nodes; 3 - branch; 0 is the auxiliary nodes $v^a$, which with the nearest critical $v^c$ forms curvilinear arcs between the critical nodes in the branching case [2].

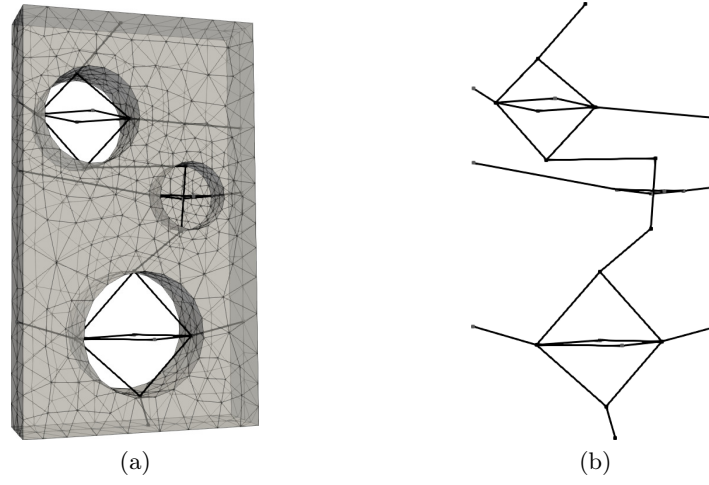(a)                                              (b)

**Fig. 1.** The mesh and volume Reeb graph: (a) mesh (b) volume Reeb graph with critical and auxilary nodes

---

**Algorithm 1:** Formation of the Reeb graph

---

**Data:** $T$ — triangulation, consisting of 2-simplexes $\sigma \in T$;
$f$ —level Morse function; $n_t$ — number of available threads
**Result:** Reeb graph $R_i(V, E)$

**1 while** $m < |V(T)|$ **do**
**2**     Partition nodes, according to coordinate $z(v)$, where $k = 1...n_t$
**3**     Computing values of Morse function $f(v_k, m)$, for nodes in thread $k$
**4**     $m \leftarrow m + 1$
**5** Form layers of Morse function $f$
**6 while** $\left( \sum_{j=1}^{m} |LS_j(\sigma)| < |T| \right)$ **do**
**7**     Each layer we will distribute between available threads $k = 0..n_t$
      **forall** $v_k$ **do**
**8**       $LS_m(\sigma) = \sum LS_m^k(\sigma) = \{\sigma \in Cl(LS(\alpha)) : \forall v \preceq \sigma, f(v_k) \geqslant \alpha\}$
**9**     $m = m + 1$
**10 foreach** $LS(\alpha_m)$ **do**
**11**     Define 2-simplexes $\sigma$, containing $v_i$ **foreach** $v_i$ **do**
**12**       $St(v_i) = \{\sigma \in T | v_i \preceq \sigma\}$ link «star»
**13**       **foreach** $v_h \in St$ **do**
**14**         Define link of 0-simplexes $v_h$
**15**         **if** $(Lk^+(v_h), Lk^-(v_h), Lk^\pm(v_h), Lk^\mp(v_h))$ **then**
**16**           $v_h^c = v_h$
**17**           $L(v_h^c) = l$, где $l = \{1, 2, 3, 4, 5\}$— link type of 0-simplexes

---

**18** Find auxiliary nodes
**19** **foreach** $v_i^c$ **do**
**20**      **if** $L(v_i^c) = 2 \wedge L(v_{i+1}^c) = 3$ **then**
**21**          Find nodes $v_i^a, v_{i+1}^a$ on surface line, between $v_i^c$ and $v_{i+1}^c$.
**22**          $v_{i+1}^c = v_i^a$ и $v_{i+2}^c = v_{i+1}^a$, $v^a \subseteq v^c$, $L(v_{i+1}^c) = 0 \wedge L(v_{i+1}^c) = 0$.
**23**          Renumbering nodes and suppose that $i \rightarrow i + 2$

**24** Form and compile Reeb graph $R_i(V, E)$

## 2   The Volume Reeb Graph

Define a tree of connections between the critical nodes $v^c$ and construct the Reeb graph $R(V, E)$ with the edges $e(v_i, v_j) \in E$ for the triangulation $T$ in $\mathbb{R}^3$, consisting of 3-simplexes [3].

We will use the Morse level function $f$ of different directions in some coordinate planes. To determine the critical nodes $v^c$ and introduce the parameter $c(v^c)$ corresponding to the number of edges outgoing from it, and taking the value: 1 - one node; 2 - two nodes; 3 - branch; 0 is an auxiliary point $v^a$, which with the nearest critical nodes $v^c$ form curvilinear arcs between the critical nodes in the branching case [4].

Note by 0 the simplex $v^s$, as the "auxiliary". In contrast to the previously implemented approach to the formation of the Reeb graph, this algorithm uses a volume triangulation of the domain $T$.

---

**Algorithm 2:** Computing volume Reeb graph

**Data:** $T$ — triangulation of 3-simplexes $\sigma \in T$; $n_t$ — number of available threads.
**Result:** Volume Reeb graph $R(V, E)$

**1** Compute graph by Algorithm 1 on result - 2d vertical Reeb graph $R_0(V, E)$
    **while** $m < |V(T)|$ **do**
**2**      **for** $k = 1$   $to$   $n_t$ **do**
**3**          For each thread we assign a separate subgraph.
**4**          **for** $i \leqslant n$ **do**
**5**              **if** $v^a - auxilary$ **then**
**6**                  Form layer $S_i = \{\forall v \subseteq T \,|\, v \preceq z(v^a)$ of 3-simplexes
                      $S_i$ — level layer for triangulation of 2-simplexes $T$
**7**              **foreach** $S_n$ **do**
**8**                  Compile Reeb graphs $R_{n+1}(V, E)$ from Algorithm 1,
                      separately for each $S_n$
                      Bypassing the Morse function $f$, one of the horizontal axes

**9**      Compile volume Reeb graph $R(V, E)$, where $R_{0..n}(V, E) \subseteq R(V, E)$
          $m = m + 1$

---

The volume Reeb graph $R(V,E)$ is represented as a set of subgraphs that are constructed from the mesh (or cells) layers in different directions of traversal of the Morse function.

The general graph has the notation $R_0(V,E)$, the remaining subgraphs will have the notation $R_{n+1}(V,E)$, where $n_t$ is the number of inner layers.

## 3  Domain Partitioning by Volume Reeb Graph

Segmentation of subdomains into smaller fragments, which are blocks that represent split pants-decomposition. After obtaining the volume Reeb graph it is necessary to cut the domain $T$ along the auxiliary nodes of the graph. The obtained layers $S$, which represent some contour of 2 or 3 simplexes, have some upper bound.

For each layer, the Morse function $f$ with a different direction in the coordinate system is applied.

---

**Algorithm 3:** Partitioning of triangulation $T$ on basis of Reeb graph

**Data:** $T$ — triangulation of 3-simplexes $\sigma \in T$; $R(V,E)$ — Reeb graph
**Result:** Set of layers $\{LS_m(\sigma)\}$

1  We define auxiliary vertices $v^a = \{v^a \subseteq v^c \,|\, c(v^c) \equiv 0\}$
2  **while** $v_m^c < |V(R_0)|$ **do**
3     Distribute each layer for auxilary nodes for available threads $k = 0..n_t$
4     Looking for the nearest auxiliary nodes $v_m^c$ for holes
5     Abstractly draw the cutting planes with respect to 4 $v^c$ along two axes
6     Define the layers $LS_m^k(\sigma) = \sigma^- \in LS_m^k(\sigma), \sigma \notin LS_m^k(\sigma) \bigcup LS_{m-1}^k(\sigma)$
7     $m \leftarrow m + 1$

---

The resulting Reeb graph is associated with the main graph obtained in Algorithm 1 and is positioned relative to the auxiliary nodes[6]. Thus we obtain a volume connected graph which characterize the complete topology of the given domain.

An example of how the algorithm works is shown in the figure 2. On the figure 2 (a) presented partition of "frame" without branching of inner boundaries with only 1 secant axis. This decomposition cannot produce many subdomains without branching. On the figure 2 (b) decomposition produced in two axis on a large number of subdomains.

The topology of this domain and the graph characterizing it allow us to divide the domain into minimal independent parts.[5] The partitioning algorithm uses the Reeb graph for each layer, which makes it possible to implement a large number of algorithms.

## 4  Numerical Experiments

Computational experiments were carried out on three-dimensional unstructured meshes for multiply connected regions, differing in the topological type, surface geometry and the details of its description by the mesh itself.
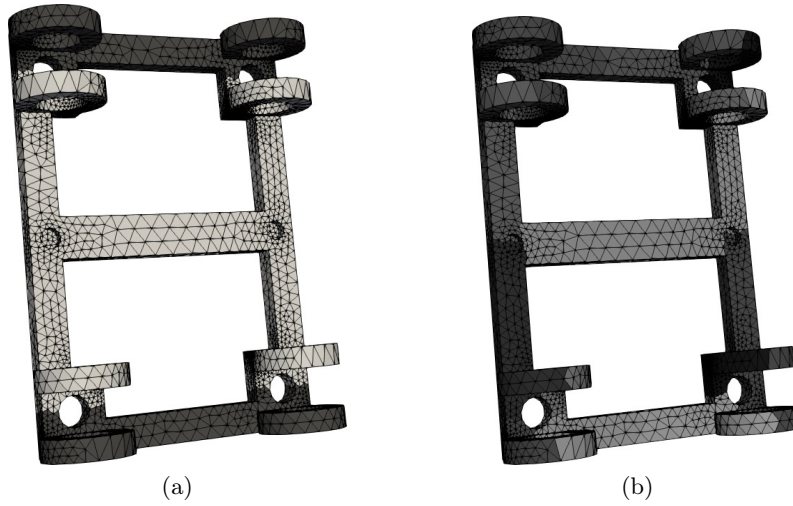
Fig. 2. Partition on basis of Volume Reeb graph: (a) without branching of inner boundaries; (b) with branching
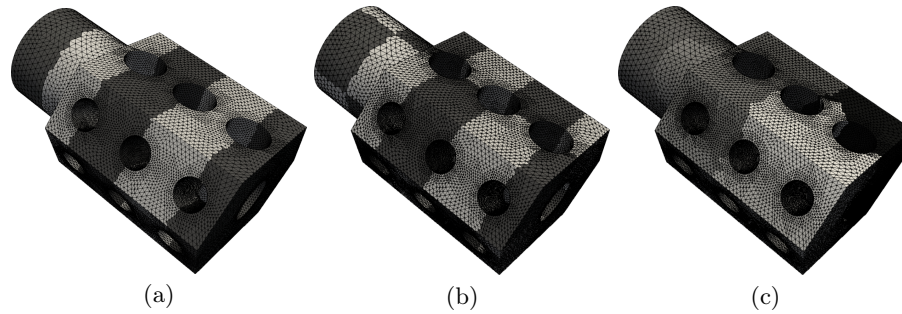


Fig. 3. The partitions: (a) without branching of inner boundaries; (b) partition with branching to 10 subdomains and (c) partition, made by METIS algorithm

Figure 3 presents us result of decomposition by volume Reeb graph and algorithm METIS. Decomposition produced by algorithm of volume Reeb graph on figure 3 (a), (b) uses topology of arbitary domain. Decomposition by the METIS algorithm partitions the domain based on cells connectivity, but does not limit the number of adjacent subdomains.

On figure 4 was hidden some subdomains to present the difference between (a) partition without branching of inner boundaries and (b) with branching.

At the moment, by parallelizing the main cycles and passes through 2 and 3 - simplexes of triangulation, the following estimates of computations are achieved. To measure estimate speed of algorithms in this paper we will use "speedup" $S = \frac{T_s}{T_{n_t}}$. The speedup is defined as the ratio of the serial runtime of the best
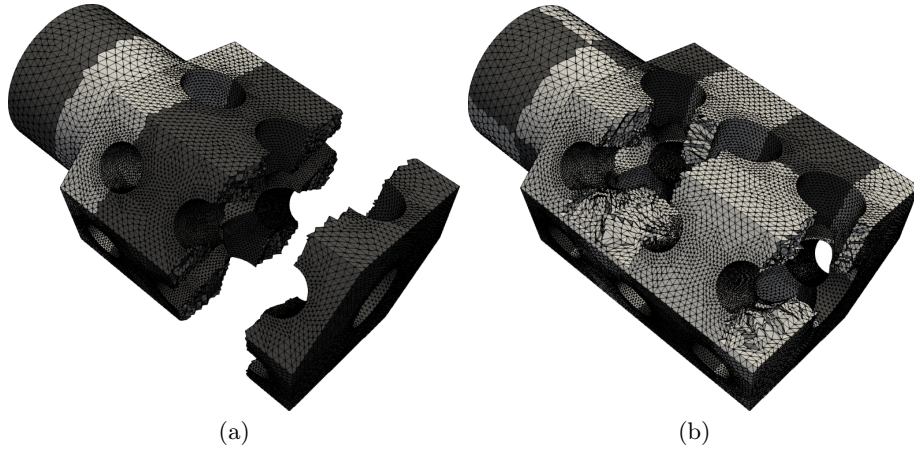
(a)                                                    (b)

**Fig. 4.** Hidden layers in partition of mesh: (a) without branching of inner boundaries; (b) with branching

sequential algorithm for solving a problem $T_s$ to the time taken by the parallel algorithm time $T_{n_t}$ to solve the same problem on $n_t$ processors, cores or threads.

**Table 1.** Speedup by parallelizing of partitioning with Reeb graph

| Threads | 623552 nodes | 1123102 nodes | 7620096 nodes |
|---------|--------------|---------------|---------------|
| Intel Xeon E5-1650 Six-core 12-threads 16 GB RAM | | | |
| $T_s$, sec | 4103 | 6591 | 37532 |
| 4 (2 cores) | 2,78 | 2,84 | 2,90 |
| 6 | 3,21 | 3,25 | 3,29 |
| 12 | 5,28 | 5,32 | 5,35 |
| Intel i5-4460 Quad-core 4-threads 8 GB RAM | | | |
| $T_s$, s | 3705 | 5772 | 33718 |
| 2 | 1,78 | 1,81 | 1,82 |
| 4 | 3,69 | 3,65 | 3,51 |

The results of parallel algorithm speedup shown in Table 1 was computed only on main parts of algorithm. The main parts of algorithm is compute of volume Reeb graph and partition on subdomains. We didn't measured parts of reading and writing mesh from external memory.

Source codes produced in C++ and parallel part made by using OpenMP. The algorithms considered in this paper make it possible to determine the features of the topology of various domains. The parallel version of algorithm was evaluated in $O(K \log^2 K/n)$, where $n$ is the number of available threads and $K$ is a number of 0-simplexes of mesh. The estimation of the computational costs of the partitioning algorithm by the Reeb graph is $O(L \log^2 L)$, where $L$ is a sum

of all simplexes of mesh. Algorithms showed sufficiently accurate results, which indicates the reliability of their application.

The considered partitions, due to the elimination of branching of inner boundaries, reduce the number and simplify the structure of exchanges between computational processes in parallel algorithms of mesh methods.

The layer-by-layer ordering of mesh cells in the subdomains of the unstructured mesh excludes conflicts in computing in the shared memory of computational systems with a hybrid architecture.

Partitioning into a given number of subdomains can be provided by recursively dividing the obtained subdomains of an unstructured mesh and combining the layers of cells that provide independent access to the data.

## References

1. Korneev V., Langer U. 2004 Encyclopedia of Computational Mechanics pp 617-647
2. Kadyrov I. R., Kopysov S. P., Novikov A. K. 2018 Partitioning of Triangulated Multiply Connected Domain into Subdomains Without Branching of Inner Boudaries (Uchenye Zapiski Kazanskogo Universiteta. Seriya Fiziko-Matematicheskie Nauki) 160 [In print]
3. Kopysov S. P., Kuzmin I. M., Nedozhogin N. S., Novikov A. K. 2012 Parallel Algorithms for Constructing and Solving the Schur Complement on Graphics Accelerators (Uchenye Zapiski Kazanskogo Universiteta. Seriya Fiziko-Matematicheskie Nauki) 154 (3) pp 202-215.
4. Martynenko S. I. 2008 Formalization of Computations at Numerical Solution of Boundary Value Problems (Uchenye Zapiski Kazanskogo Universiteta. Seriya Fiziko-Matematicheskie Nauki) 150 (1) pp 76-90
5. Postnikov M. M. 1971 Introduction in Morse Theory (Moscow: Nauka) p 568
6. Zimovnov A. V., Mestetskiy L. M. 2016 On algorithm of curve-skeleton extraction for 3D model based on planar projections (Vestn. TvGU Seriya: Prikladnaya matematika) (3) pp 67-83
7. Hajij M., Dey T., Li X. 2016 Segmenting a surface mesh into pants using Morse theory (Graphical Models) 88 pp 12-21
8. Mestetskiy L. M. 2009 Continuous morphology of binary images: figures, skeletons and circulars (Moscow: Fizmatlit) p 288
9. Harvey W., Wang Y., Wenger R. 2010 A Randomized O. (M log M) Time Algorithm for Computing Reeb Graphs of Arbitary Simplical Complexes (USA: SoCG '10) p 267-276
10. Novikov A. K., Piminova N. K., Kopysov S. P., Sagdeeva Y. A. 2016 Layer-by-layer ordering in parallel finite composition on shared-memory multiprocessors (IOP Conf. Ser.: Mater. Sci. Eng.) 158 (1)