Engineering With Computers manuscript No.
(will be inserted by the editor)

Sequential Decision-Making Approach for Quadrangular Mesh Generation

Amaury Johnen · Damien Ernst · Christophe Geuzaine

Received: date / Accepted: date

Abstract A new indirect quadrangular mesh generation algorithm which relies on sequential decision-making techniques to search for optimal triangle recombinations is presented. In contrast to the state-of-art Blossom-quad algorithm, this new algorithm is a good candidate for addressing the 3D problem of recombining tetrahedra into hexahedra.

Keywords Finite element mesh generation · quadrangular meshes · recombination · sequential decision-making

1 Introduction

Finite element methods are numerical techniques largely used for solving physical problems governed by partial differential equations (PDEs). Those methods require the domain of interest to be discretized into a mesh, *i.e.* a set of discrete subdomains called elements [21]. In 2D, those elements are triangles or quadrilaterals.

Triangular meshes are largely used in finite element simulations because of the existence of fast, robust and automatic techniques to generate high-quality elements, even with size constraints. However, quadrilateral meshes are sometimes considered as superior to triangular meshes [5]. Nonlinear mechanics in particular have element models that only work with quadrangular meshes; in fluid dynamic simulations, quads are also often sought after to discretize boundary layers.

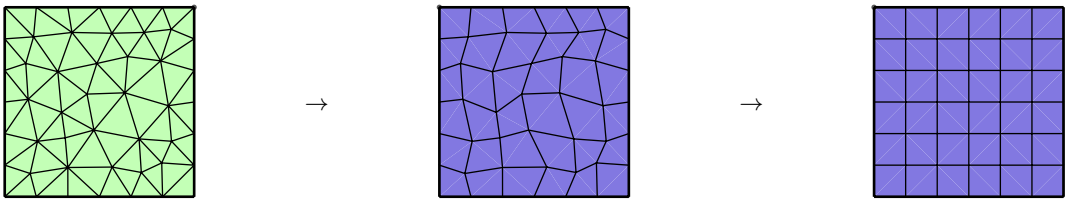
Automatic generation of quad meshes is not an easy problem. For a long time, existing techniques either were complex or produced poor-quality elements (quadrangles with very acute or obtuse angles and/or quadrangles that are poorly aligned with desired directions)—often in areas of the domain where good meshes were critically needed.

At present, there are essentially two approaches for automatically generating quad meshes with size constraints: direct methods and indirect methods. With direct methods, quadrilaterals are constructed at once, either by using advancing front techniques [2], using regular grid-based methods (quadtrees) [6] or partitioning the domain [10]. Indirect methods, on the other hand, rely on an initial triangular mesh (Fig. 1a) and apply merging techniques to recombine the triangles of the initial mesh into quadrangles (Fig. 1b) [11,3]. Other more sophisticated indirect methods use a mix of advancing front and recombination techniques [20,16].

The performance of such indirect methods crucially depends on the technique used to generate the initial triangular mesh. Indeed, triangular mesh algorithms usually aim at producing close to equilateral triangles, which is not optimal for recombination [17]. One recent original approach relies on Delaunay-frontal algorithms in L^∞-norm to generate close to right-angled triangles, facilitating the construction of high quality quadrangulations [12,17].

The main shortcoming of classical recombination techniques is the difficulty of generating globally high quality meshes [20,13]. Greedy recombination of triangles into quads (that merges triangles leading to the best quad first, then continues with the remaining triangles) always leads to orphaned triangles which have to be subdivided into poor-quality quads. Other techniques using advancing fronts produce poor-quality elements at meeting fronts. Blossom-quad proposed an elegant solution by using a minimum-cost perfect match-

Amaury Johnen · Damien Ernst · Christophe Geuzaine
University of Liège,
Department of Electrical Engineering and Computer Science
E-mail: {a.johnen, dernst, cgeuzaine}@ulg.ac.be



a b c

Fig. 1 The triangles can be recombined in order to get quadrangles. After that, a smoothing operation can be applied in order to relocate nodes in better places (b → c).

ing algorithm [18]. Blossom-quad guarantees that if a triangle-to-quad recombination exists, it will generate a full quad mesh that is optimal according to a quality functional. However, the best implementation of this algorithm is Blossom V [9], which runs in $O(n_\triangle^2)$ where $n_\triangle$ is the number of triangles. Moreover, Blossom-quad does not allow for any other "action" than the merging of two triangles into a quad, and it cannot be extended to 3D meshes (to recombine tetrahedra into hexahedra).

In this paper we propose a new indirect method for generating quads. This method can in principle work with any initial triangular mesh, and allows for the use of a great variety of topological and geometrical operations to generate high quality quadrangulations. The time complexity is $O(n_\triangle)$ and there are *a priori* no difficulties to extend it to the 3D problem. The conceptual basis of the method relies on looking at the recombination problem as an optimal sequential decision-making problem.

The paper is organized as follows. After stating the problem in Section 2, we formulate it as a sequential decision-making problem in Section 3. In Sections 4 and 5, we present two ways of solving the sequential decision-making problem by using look-ahead trees. Finally, the results are presented in Section 6 and we conclude in Section 7.

2 Problem Statement

The goal of the mesh generation process in which the recombination algorithm takes part is, starting from a triangular mesh, to create a mesh made of quadrilaterals that has controlled element sizes and shapes. We assume that the initial triangular mesh fulfills the size requirements. In this paper, we only consider the topological operation of recombining two triangles into one quadrangle and no other topological (swap, collapse) or geometrical (smoothing) operations. This algorithm does not alter the size of edges, so the aim of the recombinations is to produce quadrangles with the best shape quality.

There exist many different shape quality definitions for quadrangles in the literature. The method we propose is not specific to a particular definition, and we will not enter into the debate about which shape quality measure should be used. In this paper, we will use the definition given in [3]. Consider a quadrilateral element Q and its four internal angles $\alpha_k,\ k = 1, 2, 3, 4$ (Fig. 2). The shape quality $\eta(Q)$ of Q is defined as:

$$\eta(Q) = \max\left(0,\ 1 - \frac{2}{\pi}\max_k\left(\left|\frac{\pi}{2} - \alpha_k\right|\right)\right). \tag{1}$$

This quality measure is equal to 1 if the element is a rectangle and 0 if one of those angles is either ≤ 0 or $\geq \pi$.

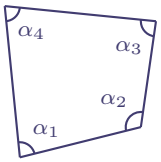


Fig. 2 Quadrilateral with its four internal angles.

We define the global quality q of a mesh $\mathcal{M}$ as the sum of shape quality of every quadrangle:

$$q(\mathcal{M}) = \sum_{Q_i \in \mathcal{Q}} \eta(Q_i), \tag{2}$$

where $\mathcal{Q}$ is the set of quadrangles of $\mathcal{M}$. With this definition, the initial triangular mesh has a global quality of zero.

The problem that we want to solve is the following: from an initial triangular mesh $\mathcal{M}_0$, find, by recombining all triangles into quadrangles, a mesh $\mathcal{M}^*$ that maximizes q, *i.e.* $q(\mathcal{M}^*) \geq q(\mathcal{M}), \forall \mathcal{M}$. Note that we only consider initial triangular meshes that can be fully recombined into quads.

The dual graph of the mesh is the graph $G(V, E)$ for which every vertex v_j is a triangle T_j and every edge e_{ij} represents the adjacency of two neighbour triangles T_i and T_j. It has been shown that the problem of recombining triangles is connected to the problem of

finding a matching in the dual graph [18], i.e., a subset of edges that do not share any vertices. Let us consider a mesh of a closed surface with zero genus, for which every triangle has three neighbours. This is among the worst cases in terms of the ratio between the number of possible recombinations over the number of triangles. The dual graph of such a mesh is a cubic planar graph, *i.e.* all vertices have degree three and it can be drawn on a plane in such a way that its edges do not intersect. In this case it has also been proven that the number of perfect matchings N (matchings which do not leave any unmatched vertex in the graph) grows exponentially with the number of vertices in that graph [7]:

$$3\,\varphi^{n_\triangle/72} \leq N \leq 2^{n_\triangle},$$

where φ is the golden ratio ($\simeq 1.62$). As a result, the number of possible solutions also grows exponentially with the size of the mesh.

Among all the possible solutions, Blossom-quad [18] allows to find the one that maximizes q in polynomial time. However, as stated in the introduction, Blossom-quad cannot be extended to the 3D problem of recombining tetrahedra into hexahedra. Indeed, the formulation of the problem as a minimum-cost perfect matching works in 2D where a pair of triangles forms a quadrangle. In 3D, at least five tetrahedra are required to form a hexahedron. This motivates the formulation of the problem as a sequential decision-making problem.

3 Formulation as a Sequential Decision-Making Problem

Recombinations can be seen as actions that are applied step by step to the mesh. This implies that the overall problem can be seen as a discrete-time system for which one seeks a sequence of actions a_t (the recombinations) that maximizes the sum of the rewards $r(a_t)$ defined as the shape quality of the created quadrangles Q_t, *i.e.* $r(a_t) = \eta(Q_t) \in [0, 1]$. The generic dynamics of this discrete system is given by equation $\mathcal{M}_{t+1} = f(\mathcal{M}_t, a_t)$, where the mesh $\mathcal{M}_t \in X$ is a state of the system and where $a_t \in A(\mathcal{M}_t)$. X is called the state space and $A(\mathcal{M}_t)$ is called the action space. We define $X_f \subset X$ as the set of final states (meshes for which no allowed recombination remains).

Definition 1 The "polder set" containing a triangle T, denoted $\mathcal{P}(T, \mathcal{M})$, is the set of triangles containing at least T for which the dual graph is a connected component of the dual graph of the whole mesh.

Definition 2 Recombination criterion: a recombination between a triangle T and one of its neighbours is forbidden if it separates the set $\mathcal{P}(T, \mathcal{M})$ into two odd-cardinality disjoint polder sets.

Remark: In practice, it is easy to understand that an action that leads to a mesh with isolated triangles (or leftover cavities that have an odd number of edges on their boundary) is highly suboptimal. Therefore in the following, we will exclude these actions by taking $A(\mathcal{M}_t)$ as being equal to the set of actions of $\mathcal{M}_t$ that do not violate the recombination criterion (see Definition 2). As a consequence, X is restricted to the set of all meshes obtained by recombining triangles of the initial mesh and for which there is no odd-cardinality *polder set*.

With this formulation, the optimal solution could be found by computing the tree for which every node corresponds to a state $\mathcal{M} \in X$ and every edge corresponds to a possible transition. The leaves of the tree would be the final states and among them we could find the optimal solution. But we showed in Section 2 that the size of X_f grows exponentially with the number of initial triangles, and so does the size of the tree. In the next section, we propose an algorithm for navigating into these large trees efficiently. We also relax the original problem statement to allow for non-fully recombined meshes, in which case we consider the percentage of recombinations with respect to the optimal

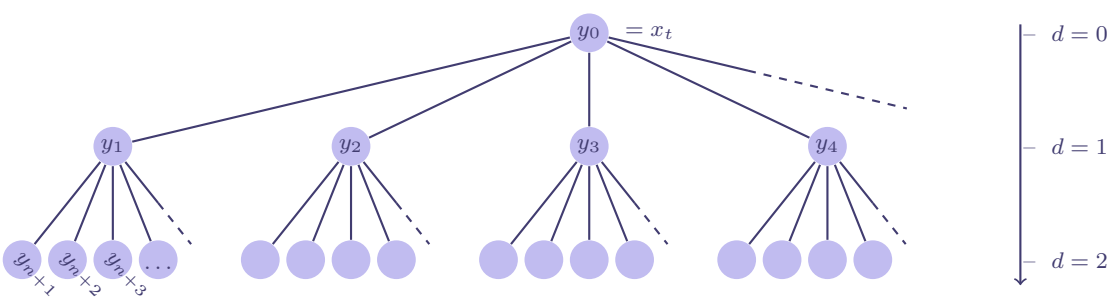


Fig. 3 Uniform look-ahead tree of horizon 2 where n is the number of nodes at depth 1 (*i.e.* the size of the action space $A(\mathcal{M}_t)$).

solution as a criterion to evaluate the performance of a the algorithm.

4 Uniform Look-ahead Tree

The uniform look-ahead tree (LT) can be seen as a computationally efficient way for exploring the tree made of all possible sequences of actions. A uniform LT $\mathcal{T}_h(\mathcal{M}_t)$ expanded from the mesh $\mathcal{M}_t$ with a horizon h is a tree for which the root y_0 is $\mathcal{M}_t$ and every node of depth $d \in \{0, \dots, h\}$ corresponds to a state that is reachable from $\mathcal{M}_t$ after d transitions (see Fig. 3). Let y_k^{leaf}, $k = 1, 2, \dots, N_{\text{leaf}}$ denote the leaf nodes, *i.e.* the nodes that are at depth h. At each leaf node y_k^{leaf} is associated the sequence of h actions $a_\tau^k, \tau = t, \dots, t+h$. We define the optimal sequence of actions as the one that corresponds to $\max_k \sum_{\tau=t}^{t+h} r(a_\tau^k)$. The action that is applied at time t, a_t, is then taken as the first action of the optimal sequence.

The uniform LT has been applied to the mesh of a sphere that contains 2152 initial triangles (see Fig. 4). We compute the quality of the final state as a percentage of the quality obtained with Blossom-Quad [18]. For a horizon of 1, we obtain a quality of 87.6% in 0.16 seconds. For a horizon of 2, the quality is poorer with 85.1%, and the computation time is much greater with 225.6 seconds. Thus the result is worse in both quality and time. Note that on our computer (Macbook Pro Retina, Mid 2012) we have been unable to test the uniform LT for larger horizons.

One explanation for the lack of performance is the fact that with such small horizons the uniform LT algorithm behaves almost similarly to the greedy algorithm that recombines at each step the two triangles leading to the best quad. The fast increase in computation time is explained by the high branching factor, which is of the order of the number of remaining triangles.

In the next section, we define a different way to construct the LT that gives better performances.

5 Selective Look-ahead Tree

The tree made of all possible sequences of actions has intrinsically a large branching factor which hinders the performances of uniform LT techniques. In this section, we propose to apply another form of LT that works with a smaller branching factor allowing to exploit larger depths. We named it *selective look-ahead tree*.

Let us first introduce some definitions. For a given state y, $\mathcal{P}_k(y)$, $k = 1, \dots, 3$ is the set of all available triangles in y that take part in exactly k recombinations of $A(\mathcal{M}_t)$. We define $\mathcal{P}_{\min}(y)$ as the first non-empty

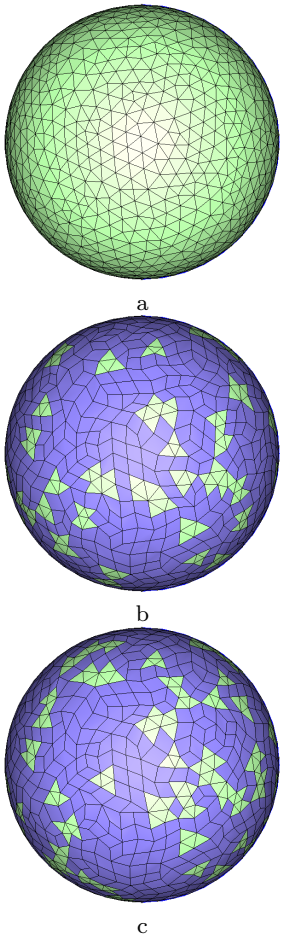


Fig. 4 The triangular mesh of a sphere (a) is recombined with a uniform LT of horizon 1 (b) and 2 (c). The initial mesh contains 2152 triangles. With horizon 1, 1850 triangles are recombined for a quality of 87.6%. With horizon 2, 1780 triangles are recombined for a quality of 85.1%.

set, *i.e.* equal to $\mathcal{P}_l(y)$ if $\mathcal{P}_{k<l}(y) = \emptyset$ and $\mathcal{P}_l(y) \neq \emptyset$, or equal to an empty set if $\mathcal{P}_k(y)$ are all empty. The construction of the selective LT is carried out as follows (see Fig. 5):

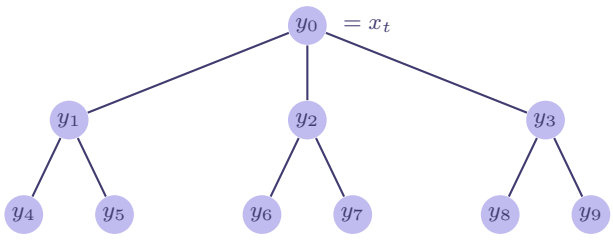


Fig. 5 Selective look-ahead tree of horizon 2. In this example, the triangle selected for y_0 has 3 possible recombinations, so the root has 3 branches. Triangles selected at depth 1 have all 2 possible recombinations.

First, we make the root y_0 to be the current state $\mathcal{M}_t$. Then we randomly select a triangle T_0 that has

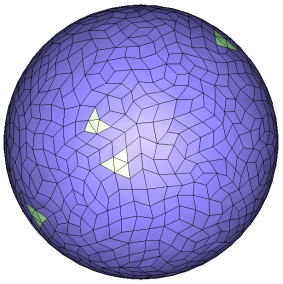


Fig. 6 The triangular mesh of the sphere (Fig. 4) is recombined with the selective LT of horizon 3. On the 2152 initial triangles, 2120 are recombined for a quality of 93.8%.

	Horizon	Quality	% Recombinations	Time [s]
Uniform	1	87.6	86	0.16
	2	85.1	82.7	225.6
Selective	1	91.8	98.2	0.037
	2	92.8	98.3	0.084
	3	94	98.7	0.146

Table 1 The uniform and selective LT are compared on the test case of the sphere. Results for the selective LT are presented as the median of 100 runs. The quality is presented as a percentage of the quality obtained with Blossom-Quad. A mesh obtained with the selective LT of horizon 3 is showed at Figure 6.

the minimum of recombinations, *i.e.* we select T_0 in $\mathcal{P}_{\min}(y_0)$, and we create a child for every possible recombination of T_0. For every child node y_i, we compute $\mathcal{N}_\triangle(y_i)$, the set of triangles that can be recombined and are neighbours of the quadrangles created in the sequence from y_0 to y_i. There are then two situations: either $\mathcal{N}_\triangle(y_i)$ is empty and T_i is taken in $\mathcal{P}_{\min}(y_0)$ (like the root node) or $\mathcal{N}_\triangle(y_i)$ is not empty. For the latter, we compute the first non-empty set of $\mathcal{P}_k(y) \cap \mathcal{N}_\triangle(y_i), k = 1, 2$ and randomly select T_i in it. Then, we branch on every possible recombination of T_i.

The rationale behind this is that if the optimal solution corresponds to a full recombination, those selective LT, deployed with the maximum possible depths, would also contain an optimal sequence of actions while still having a much smaller branching factor (bounded by 3 for the root and by 2 for the children). Note that if the optimal solution does not correspond to a fully recombined mesh, it may be reasonable to assume that those new LT may still contain a sequence of actions which is not far from the optimal one.

6 Results

6.1 Mesh of the Sphere

We applied the selective LT algorithm to the mesh of the sphere described in Section 4. The results are presented at Table 1. We can see that the selective LT shows better performances than the uniform LT on the three criterions: quality, number of recombinations and computation time. On the other hand, both quality and number of recombinations are increased from horizon 1 to 3.

6.2 Borouchaki Mesh

We present the results of the selective LT algorithm obtained for a test case proposed by Borouchaki and Frey [3]. The domain is a unit square with a circular hole of radius 0.15 centred at (0.75, 0.75) with a non-uniform mesh size field (see Fig. 7). The initial triangular mesh has been generated with the delquad algorithm [17] and contains 34,562 triangles (Fig. 7a).

In order to make a comparison with a reference mesh, the triangular mesh has been recombined with the Blossom-quad algorithm using the same quality criterion and without making any additional topological or geometrical optimizations. This is the optimal solution of our search and we take the obtained quality of 10,451.61 as reference. Quality results presented below are all given as percentages of this reference value.

We applied our algorithm for every horizon between 1 and 10. Since triangles are taken randomly, we have, for each horizon, run our algorithm 128 times. Quality, number of recombined elements and computation time are presented as "box and whiskers" graphs in Figure 8. The first two graphs show that both the quality and the number of recombined elements increase significantly when the horizon increases from 1 to 6. In the third graph, high-horizon values are shown to lead to an average branching factor of $10^{0.49/4} = 1.33$. The asymptotically linear behaviour suggests that the recombination process has a computational complexity of $n_\triangle \alpha^h$, where α is the branching factor.

The computation time (around 1.6s at horizon 1 and 3.4s at horizon 2) can be compared to the naive greedy recombination algorithm, which simply recombines triangles by selecting pairs to be recombined in decreasing order of quality of the resulting quads. Without additional constraint, the greedy approach takes 0.98 seconds to make 15,472 recombinations. Since the remaining triangles are mostly isolated, the quality cannot be compared. The same greedy approach, with the additional constraint of definition 2 (which is equivalent to the uniform LT of Section 4 with a horizon equal to 1), is computed in 1.32 seconds and leads to a quality of 88.5% with 86% of recombinations. This confirms the usefulness of the sequential decision-making approach, especially in view of its future applicability to 3D prob-

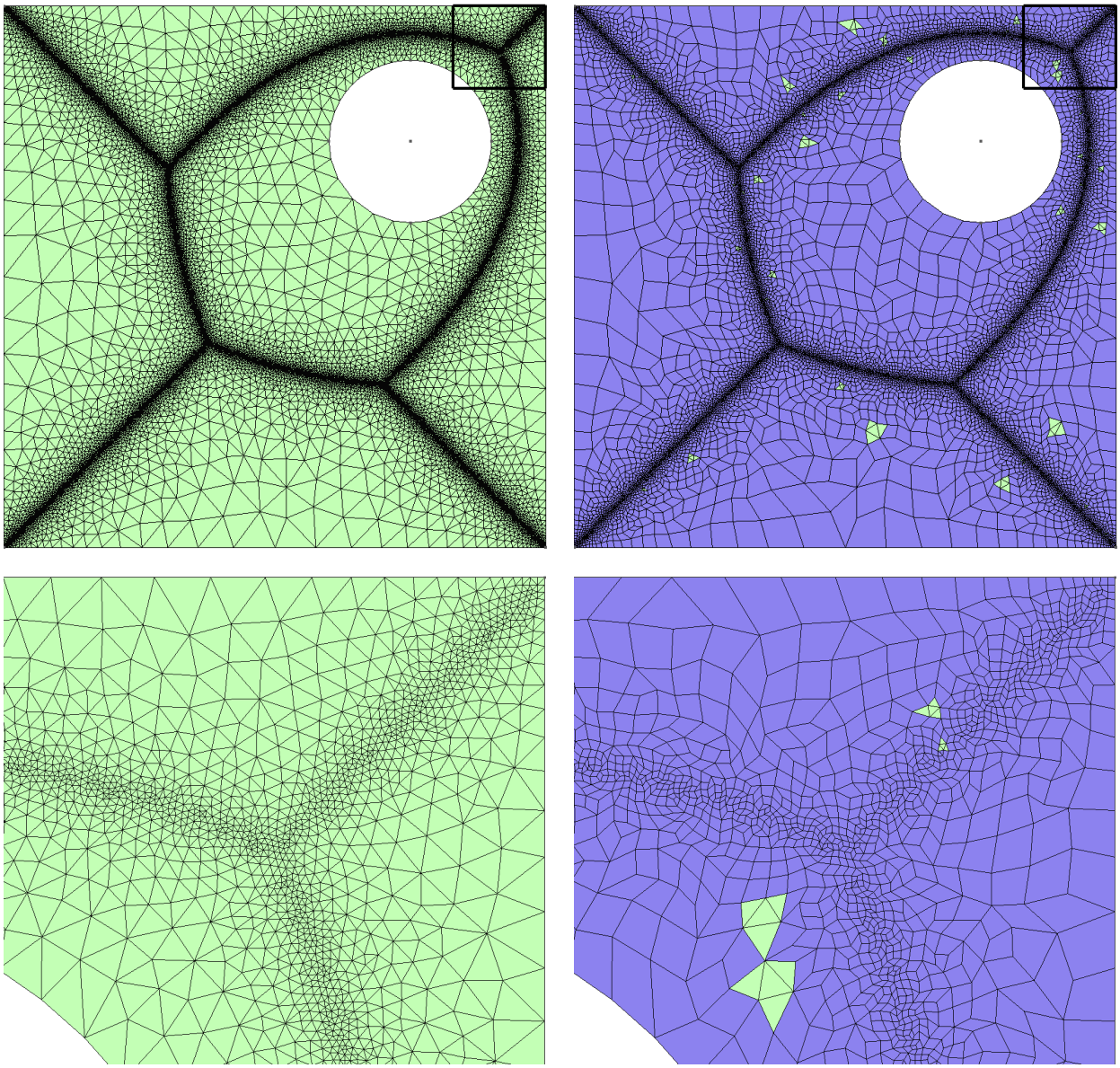


Fig. 7 Borouchaki test: initial mesh and a typical solution of our algorithm with a horizon of 5 (top, global mesh, bottom zoom of the top right corner). The solution has a quality of 93.17% and 98.8% of elements are recombined.

lems as well as of its natural handling of additional topological and geometrical actions.

7 Conclusion

We designed a new algorithm to recombine triangular meshes into quadrangular meshes that relies on decision-making techniques. The main advantage of this new approach is that it can produce good quality quad meshes with size constraints in linear time. We showed that the use of selective look-ahead trees instead of uniform look-ahead trees greatly improves the computation time as well as the quality of the final mesh. We also showed that the solution is improved by increasing the horizon of the selective look-ahead trees. The complexity of the algorithm was found experimentally to be $n_{\triangle}(1+\alpha^h)$, where $\alpha \leq 2$. Unlike Blossom-quad, our algorithm does not guarantee that all triangles are recombined. However, the remaining triangles are grouped and can easily be replaced by quadrangles in post-processing using an adaptation of Bunin's algorithm [4].

Ongoing research focuses on three extensions. First, more advanced tree navigation techniques published in the machine learning literature are investigated [14,8]. Second, instead of considering topological and geometrical operations (such as edge swaps, collapses or node

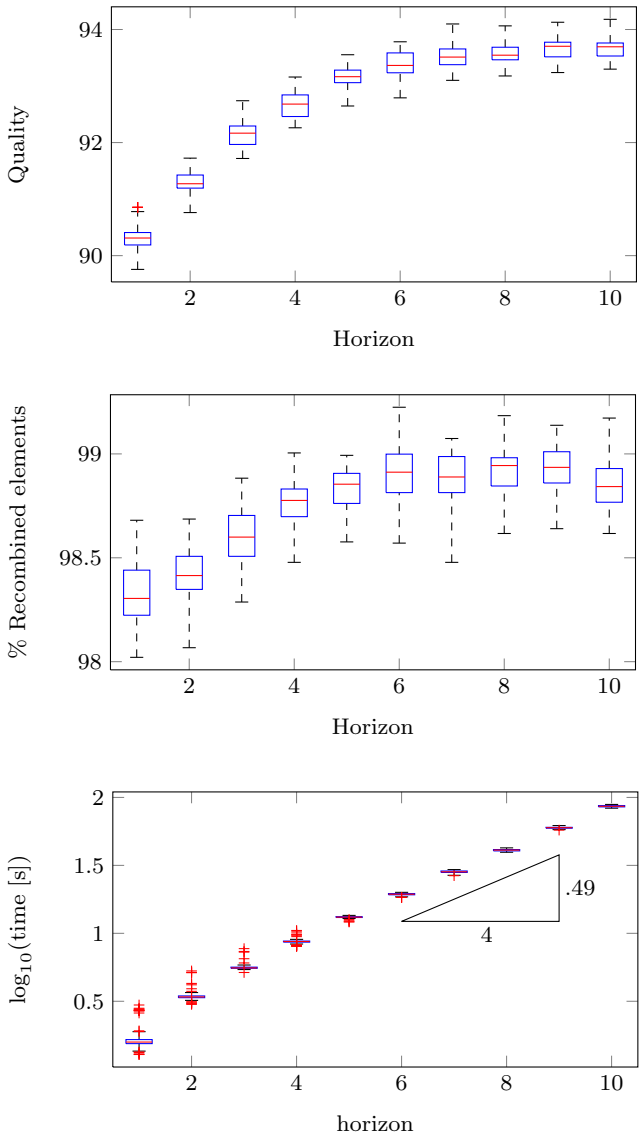


Fig. 8 Box and whiskers plots for the Borouchaki test, with medians as red lines, interquartile ranges (IQR) as blue boxes, and whiskers in black. Red crosses depict outliers that are 1.5×IQR above or below the IQR. Whiskers only extend to the most extreme data points not considered as outliers. 128 runs have been performed for every horizon from 1 to 10.
495.0pt
148.50151pt242.55272pt

relocation) as post-processing operations, they could be added directly as actions in the sequential decision-making process. Third, the algorithm could also be extended to recombinations of tetrahedra into hexahedra [1,19,15].

Acknowledgements This research project was funded in part by the Walloon Region under WIST 3 grant 1017074 (DOMHEX).

References

1. Baudouin, T.C., Remacle, J.F., Marchandise, E., Henrotte, F., Geuzaine, C.: A frontal approach to hex-dominant mesh generation. Advanced Modeling and Simulation in Engineering Sciences **1**(1), 1–30 (2014)
2. Blacker, T.D., Stephenson, M.B.: Paving: A new approach to automated quadrilateral mesh generation. International Journal for Numerical Methods in Engineering **32**(4), 811–847 (1991)
3. Borouchaki, H., Frey, P.J.: Adaptive triangular–quadrilateral mesh generation. International Journal for Numerical Methods in Engineering **41**(5), 915–934 (1998)
4. Bunin, G.: Non-local topological clean-up. In: P.P. Pébay (ed.) Proceedings of the 15th International Meshing Roundtable, pp. 3–20. Springer (2006)
5. D'Azevedo, E.: Are bilinear quadrilaterals better than linear triangles? SIAM Journal on Scientific Computing **22**(1), 198–217 (2000)
6. Frey, P.J., Marechal, L.: Fast adaptive quadtree mesh generation. In: Proceedings of the Seventh International Meshing Roundtable, pp. 211–224 (1998)
7. Jiménez, A., Kiwi, M.: Counting perfect matchings in the geometric dual. Electronic Notes in Discrete Mathematics **37**(0), 225–230 (2011)
8. Jung, T., Wehenkel, L., Ernst, D., Maes, F.: Optimized look-ahead tree policies: a bridge between look-ahead tree policies and direct policy search. International Journal of Adaptive Control and Signal Processing (2013). Available as preprint at: *http://onlinelibrary.wiley.com/doi/10.1002/acs.2387/full*
9. Kolmogorov, V.: Blossom V: a new implementation of a minimum cost perfect matching algorithm. Mathematical Programming Computation **1**(1), 43–67 (2009)
10. Kowalski, N., Ledoux, F., Frey, P.: A PDE based approach to multidomain partitioning and quadrilateral meshing, pp. 137–154. Springer (2013)
11. Lee, C.K., Lo, S.H.: A new scheme for the generation of a graded quadrilateral mesh. Computers & Structures **52**(5), 847–857 (1994)
12. Lévy, B., Liu, Y.: Lp centroidal voronoi tessellation and its applications. ACM Trans. Graph. **29**(4), 119:1–119:11 (2010)
13. Lo, S., Lee, C.: On using meshes of mixed element types in adaptive finite element analysis. Finite Elements in Analysis and Design **11**(4), 307–336 (1992)
14. Maes, F., St-Pierre, D., Ernst, D.: Monte carlo search algorithm discovery for single-player games. Computational Intelligence and AI in Games, IEEE Transactions on **5**(3), 201–213 (2013). DOI 10.1109/TCIAIG.2013.2239295
15. Meshkat, S., Talmor, D.: Generating a mixed mesh of hexahedra, pentahedra and tetrahedra from an underlying tetrahedral mesh. International Journal for Numerical Methods in Engineering **49**(1-2), 17–30 (2000)
16. Owen, S.J., Staten, M.L., Canann, S.A., Saigal, S.: Q-Morph: an indirect approach to advancing front quad meshing. International Journal for Numerical Methods in Engineering **44**(9), 1317–1340 (1999)
17. Remacle, J.F., Henrotte, F., Carrier-Baudouin, T., Bechet, E., Marchandise, E., Geuzaine, C., Mouton, T.: A frontal delaunay quad mesh generator using the lnorm. International Journal for Numerical Methods in Engineering **94**(5), 494–512 (2013)
18. Remacle, J.F., Lambrechts, J., Seny, B., Marchandise, E., Johnen, A., Geuzaine, C.: Blossom-quad: A non-uniform quadrilateral mesh generator using a minimum-cost perfect-matching algorithm. International Journal

for Numerical Methods in Engineering **89**(9), 1102–1119 (2012)
19. Yamakawa, S., Shimada, K.: Fully-automated hex-dominant mesh generation with directionality control via packing rectangular solid cells. International journal for numerical methods in engineering **57**(15), 2099–2129 (2003)
20. Zhu, J., Zienkiewicz, O., Hinton, E., Wu, J.: A new approach to the development of automatic quadrilateral mesh generation. International Journal for Numerical Methods in Engineering **32**(4), 849–866 (1991)
21. Zienkiewicz, O., Taylor, R.: The Finite Element Method - The Basis (Volume 1). Elsevier (2000)