



Cuboid Partitioning for Parallel Matrix Multiplication on Heterogeneous Platforms

Olivier Beaumont, Lionel Eyraud-Dubois, Thomas Lambert

► To cite this version:

Olivier Beaumont, Lionel Eyraud-Dubois, Thomas Lambert. Cuboid Partitioning for Parallel Matrix Multiplication on Heterogeneous Platforms. 22nd International Conference on Parallel and Distributed Computing, Aug 2016, Grenoble, France. 10.1007/978-3-319-43659-3_13 . hal-01269881v2

HAL Id: hal-01269881

<https://inria.hal.science/hal-01269881v2>

Submitted on 12 May 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Cuboid Partitioning for Parallel Matrix Multiplication on Heterogeneous Platforms

Olivier Beaumont, Lionel Eyraud-Dubois, and Thomas Lambert

¹ Inria, University of Bordeaux, France

² LaBRI, University of Bordeaux, France

Abstract The problem of partitioning a square into zones of prescribed areas arises when partitioning matrices for dense linear algebra kernels onto a set of heterogeneous processors, and several approximation algorithms have been proposed for that problem. In this paper, we address the natural generalization of this problem in dimension 3: partition a cuboid in a set of zones of prescribed volumes (which represent the amount of computations to perform), while minimizing the surface of the boundaries between zones (which represent the data transfers involved). This problem naturally arises in the context of matrix multiplication, and can be seen as a heterogeneous generalization of 2.5D approaches that have been proposed in this context. The contributions of this paper are twofold. We prove the NP-completeness of the general problem, and we propose a $\frac{5}{6^{2/3}} \simeq 1.51$ -approximation algorithm for cube-partitioning. This is the first known approximation result for this 3D partitioning problem.

1 Introduction

In the case of homogeneous resources, the problem of partitioning data for Linear Algebra kernels in order to both balance the load throughout the computation and to minimize communications is well understood. 2D block-cyclic distributions, for instance, have been introduced in Scalapack [11] in order to achieve this goal. More recently, the problem has received a lot of attention in Communication Avoiding algorithms design (see [15,1] and [21,3] for Matrix Multiplication specifically). In this context, the goal is to partition the set of elementary computations to be performed into a minimal number of zones, each zone being able to be processed in local memory (*i.e.* both input, intermediate and output data). This corresponds to maximizing the volume of computations that can be processed with a given amount of memory.

In this paper, we concentrate on Matrix Multiplication algorithms and more specifically on Matrix Multiplication algorithms that involve N^3 elementary operations of type $C_{i,j} \leftarrow C_{i,j} + A_{i,k}B_{k,j}$, *i.e.* we ignore variants such as Strassen or Coppersmith-Winograd. Note that throughout the paper, we will assume that matrices are partitioned into blocks, whose size is chosen so as to be well adapted to all types of resources (typically CPUs and GPUs). On the other hand, we consider a fully heterogeneous platform, where all nodes may have different

processing capacities and we address the most general problem, where several partially aggregated copies of C can reside simultaneously in memory, such as in 2.5D algorithms [21]. In this context, the problem consists in partitioning the computational domain (the cube of N^3 points) into sub-domains allocated to the different resources. In order to balance the load between the processing units, each unit should receive a volume of computations proportional to its processing speed and the overall amount of communications, that corresponds to the overall boundary area between the zones should be minimized.

Many algorithms [16,6,9,13,18,14] have been proposed in the context of dense matrix multiplication based on Canon's-like algorithm, that corresponds to the 2D version of the problem, *i.e.* how to partition a matrix into zones of fixed area while minimizing the overall length of the boundaries. On the other hand, to the best of our knowledge, this paper is the first to consider the complexity of the 3D version of the algorithm, to prove the NP-Completeness of the underlying decision problem and to propose an approximation algorithm for it.

Related Works

The 2D version of this optimization problem has been first introduced by Lastovetsky and Kalinov in [16]. In [6], it has been proven that the problem is NP-Complete, and a first algorithm with bounded approximation ratio (1.75) has been proposed. This algorithm has been improved along two directions. On the one hand, Lastovetsky *et al.* have proposed to relax the assumption stating that the zones allocated to the processors should be single rectangles and have proposed optimal algorithms, but limited to 2 heterogeneous processors [9] and more recently to 3 heterogeneous processors [13]. On the other hand, recursive partitioning algorithms have recently been proposed where at each step, the set of processors is split into two parts. Sophisticated proof techniques enabled Nagamochi and Abe [18] to improve the approximation ratio down to 1.25. Recently, Fügenschuh *et al.* [14] improved this result to 1.15, but under the assumption that if we consider processors in decreasing order of their processing speeds, there is no abrupt change in the performance between 2 successive processors. Unfortunately, such an abrupt decrease typically happens when considering nodes consisting of CPUs and GPUs, such that Fügenschuh's algorithm is limited to the case of relatively homogeneous platforms. In [8], an algorithm based on the idea of non rectangular partitioning proposed by Lastovetsky and extended to any number of processors by adapting the recursive partitioning algorithm proposed by Nagamochi has been proposed. It achieves an approximation ratio of $\frac{2}{\sqrt{3}} \simeq 1.15$ and does not require any specific assumption on the relative speed of resources, so that it can be used in the case of nodes consisting of both regular cores and accelerators.

Besides a single heterogeneous node, this partitioning problem has been adapted to distributed hierarchical and highly heterogeneous platforms in [12], where the partitioning is applied at two levels (intra-node and inter-node), based on sophisticated performance models. The same partitioning has also been extended to finite-difference time-domain (FDTD) methods to obtain numerical

solutions of Maxwell's equations in [20]. More dynamic settings have also been considered in [17]. Recently, in order to cope with resource heterogeneity and the difficulty to build optimal schedules, the use of dynamic runtime schedulers have been proposed, such as StarPU [2], StarSs [19], or PaRSEC [10]. In these systems, at runtime, the scheduler takes the scheduling and allocation decisions based on the set of ready tasks (tasks whose all data and control dependencies have been resolved), on the availability of the resources (estimated using expecting processing and communication times), and on the actual location of input data. The comparison between static scheduling strategies and runtime scheduling strategies has been considered in [7], where the analysis of the behavior of static, dynamic, and hybrid strategies highlights the benefits of introducing more static knowledge and allocation decisions in runtime libraries.

Paper Outline

The paper is organized as follows. In Section 2, we formally present the partitioning problem and the notations that will be used throughout the paper. In Section 3, the complexity of the associated decision problem in the 3D case is established and a $\frac{5}{6^{2/3}} \simeq 1.51$ -approximation algorithm for cube-partitioning is proposed in Section 4. Conclusions and perspectives are given in Section 5.

2 General Context

Definition 1. *Let P be a connected polyhedron included in $[0, x] \times [0, y] \times [0, z]$. We define its covering cuboid as the smallest cuboid $Cu(P) = [x_1, x_2] \times [y_1, y_2] \times [z_1, z_2]$ that includes P . We define also its width $w(P)$ as $x_2 - x_1$, its height $h(P)$ as $y_2 - y_1$ and its length $l(P)$ as $z_2 - z_1$. Let us define $Hs(P) = h(P)l(P) + w(P)l(P) + h(P)w(P)$, $\rho(P) = \frac{\max(h(P), w(P), l(P))}{\min(h(P), w(P), l(P))}$ and $\rho'(P) = \frac{\max(h(P), w(P), l(P))}{\text{med}(h(P), w(P), l(P))}$. Finally we denote by $V(P)$ the volume of P .*

Problem 1 (Minimizing-Surface-Cuboid-Partition (MSCuboidP)). Given a set of n numbers $\{v_1, \dots, v_n\}$ such that $\sum v_k = xyz$, and the cuboid $Cu = [0, x] \times [0, y] \times [0, z]$, find for each v_k a polyhedron P_k of Cu such that $V(P_k) = v_k$ and $\bigcup P_k = Cu$ minimizing $\sum Hs(P_k)$.

A general lower bound for this problem has been established by Ballard et al. [4] and comes from Loomis-Whitney inequality. It simply states that a polyhedron P of volume $V(P)$ minimizes the surface of its covering cuboid if and only if it is shaped as a cube. This implies the following lower bound:

$$Hs(P) \geq 3V(P)^{\frac{2}{3}}. \quad (1)$$

3 NP-Completeness

We prove in this section the NP-completeness of the decision problem associated to MSCuboidP, MSCuboidP-DEC.

Problem 2 (MSCuboidP-DEC). Given a set of n given numbers $\{v_1, \dots, v_n\}$ such that $\sum v_k = xyz$, a cuboid $Cu = [0, x] \times [0, y] \times [0, z]$ and a number K , is there a set of n polyhedra P_k of Cu such that $V(P_k) = v_k$, $\bigcup P_k = Cu$ and $\sum Hs(P_k) \leq K$?

We start by reducing this problem to a more constrained variant named ACCuboidP, in which the goal is to partition the cuboid in cubes of specified side lengths.

Problem 3 (All-Cube-Cuboid-Partition (ACCuboidP)). Given a set of p given length $\{l_1, \dots, l_p\}$ such that $\sum l_k^3 = xyz$, and a cuboid $Cu = [0, x] \times [0, y] \times [0, z]$, is there a set of k cubes $C_k \in Cu$ such that $V(C_k) = l_k^3$ and $\bigcup C_k = Cu$?

Lemma 1. *ACCuboidP is NP-Complete.*

It is easy to check that ACCuboidP belongs to NP. We prove NP-hardness of ACCuboidP with a method inspired from the hardness proof of the equivalent 2D problem [5], by using a reduction from 2-PART-EQUAL, a variant of the well-known 2-PART problem. The NP-completeness of 2-PART-EQUAL can be proven by a reduction from 2-PART. Indeed, adding a constant C to every element of the instance of 2-PART and then adding n (where n is the size of the instance) elements of size C to the instance itself creates an instance of 2-PART-EQUAL that has a solution if and only if the original instance of 2-PART has one. Our proof consists in two steps: from an instance of 2-PART-EQUAL, we first derive another set of numbers b_i and prove that they can be partitioned into two equal sets if and only if the 2-PART-EQUAL instance has a solution. Then, we use the b_i numbers to build an instance of ACCuboidP for which the existence of a packing is equivalent to partitioning the b_i into two equal size sets.

Problem 4 (2-PART-EQUAL). Given a set of $2n$ integers $\{a_1, \dots, a_{2n}\}$, is there $I \subseteq [1, n]$ such that $|I| = n$ and

$$\sum_{i \in I} a_i = \sum_{i \notin I} a_i$$

First Reduction Let us now consider an instance of 2-PART-EQUAL, $\{a_1, \dots, a_{2n}\}$ and let us denote $2A = \sum_{1 \leq i \leq 2n} a_i$ and $M = 6n \times \max_i a_i$. Let us suppose, without loss of generality, that n is a multiple of 120 greater than 240 and let us define a new set $\{b_1, \dots, b_{2n}\}$ as

$$\begin{aligned} \forall i, b_i &= a_i + 3n \times \max_i a_i + D & \text{where } D &= \frac{60M - (A \bmod 60M)}{n} \\ &= a_i + \frac{M}{2} + D. \end{aligned}$$

In addition, let us set $k = \frac{n}{120} + \frac{A+nD}{60M}$ and $S = \frac{1}{2} \sum_{1 \leq i \leq 2n} b_i$. One can prove that k is an integer (since n is a multiple of 120) and that $S = 60k \times M$. In

addition, let us notice that for all i , $\frac{M}{2} < b_i$ (since $D \geq 0$ and $a_i > 0$) and $b_i \leq M$. Indeed, $D \leq \frac{60M}{n} \leq \frac{M}{4}$ and $a_i + \frac{M}{2} \leq M(\frac{1}{2} + \frac{1}{6n}) \leq \frac{4M}{6}$. Therefore $b_i \leq \frac{11M}{12} \leq M$.

Let us prove now that there exists a solution to our instance of 2-PART-EQUAL if and only if there exists a set $I \subset [1, n]$ such that $\sum_{i \in I} b_i = \sum_{i \notin I} b_i$. If there exists I such that $|I| = n$ and $\sum_{i \in I} a_i = \sum_{i \notin I} a_i$, then

$$\begin{aligned} \sum_{i \in I} b_i - \sum_{i \notin I} b_i &= \sum_{i \in I} a_i - \sum_{i \notin I} a_i + (|I| - |\bar{I}|)(\frac{M}{2} + D) \\ \sum_{i \notin I} a_i - \sum_{i \in I} a_i &= (|I| - |\bar{I}|)(\frac{M}{2} + D). \end{aligned}$$

Yet, $\sum_{i \notin I} a_i - \sum_{i \in I} a_i \leq 2n \times \max a_i = \frac{M}{3}$ and $\frac{M}{2} \leq \frac{M}{2} + D$. Therefore,

$$(|I| - |\bar{I}|)\frac{M}{2} \leq \frac{M}{3} \text{ and } (|I| - |\bar{I}|) \leq \frac{2}{3} < 1.$$

By symmetry, we obtain $|I| = |\bar{I}|$ and I is a solution to 2-PART-EQUAL.

Second Reduction In order to build the ACCuboidP instance that will be used in the reduction, we rely on a result from Walters [22] stating that it is possible to tile any cuboid with a number of cubes which is poly-logarithmic in the side lengths of the cuboid. We call the cubes in such a tiling Walters' cubes, and we denote by $WS(X, Y, Z)$ a (poly-logarithmic size) set of cubes tiling the cuboid $X \times Y \times Z$.

Let us consider the following instance of ACCuboidP:

- A cuboid of size $11M \times 15M \times S$ (with $S = 60k \times M$).
- $20k$ cubes of length $6M$.
- $24k$ cubes of length $5M$.
- $30k$ cubes of length $4M$.
- $20k$ cubes of length $3M$.
- $\forall i$, a cube of length b_i .
- $\forall i$, $WS(M - b_i, b_i, b_i)$ and $WS(M, M - b_i, b_i)$.

with M , k and the b_i 's defined from the a_i 's as in the first reduction described above. One can see that the reduction is polynomial, since the sizes of the Walters' cubes sets are poly-logarithmic functions of the b_i 's.

In the first part of the proof, we prove that if we can split the b_i items in two equal sets, then the above set of cubes can be packed into the cuboid.

Let us first consider, for each i , the cube of length b_i and the two associated Walters' cubes sets. Figure 1 shows how they can be packed into a cuboid of size $M \times M \times b_i$, where the cuboid of size $(M - b_i) \times b_i \times b_i$ and the cuboid of size $(M - b_i) \times b_i \times M$ are tiled with the cubes from $WS(M - b_i, b_i, b_i)$ and

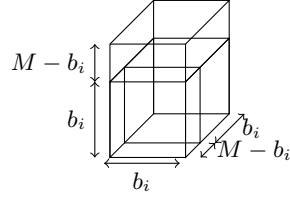


Figure 1: Tiling of a $b_i \times M \times M$ cuboid.

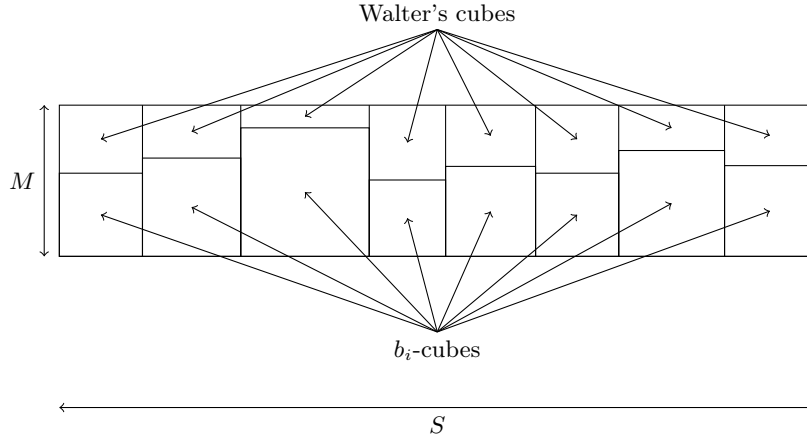


Figure 2

$WS(M, M - b_i, b_i)$ respectively. Stacking up such cuboids on top of one another, we can build two $M \times M \times S$ cuboids from the two sets I and \bar{I} , see Figure 2.

Figure 3 shows how to tile a $11M \times 15M$ rectangle with the corresponding squares, where both $M \times M$ squares represent a slice of the $M \times M \times S$ cuboids presented above. This arrangement can be repeated for a total length of S , since

$$10k \times 6M = 12k \times 5M = 15k \times 4M = 20k \times 3M = 60k \times M = S.$$

Hence, this provides a tiling of the whole $11M \times 15M \times S$ cuboid.

For the second part of the proof, we need to prove if the cuboid can be tiled with the set of cubes, then a partition of the b_i values in two equal size sets exists. We start by proving that in any valid tiling of the cuboid, the $11M \times 15M$ rectangle can only be tiled as shown on Figure 3 (or under the same pattern but with an horizontal symmetry).

Let us note that, except the b_i -cubes and the Walters' cubes, all cubes have length that are multiple of M . Therefore one can see that the resulting projections of the b_i -cubes and the Walters' cubes on the $11M \times 15M$ rectangle can be seen as several $M \times M$ squares.

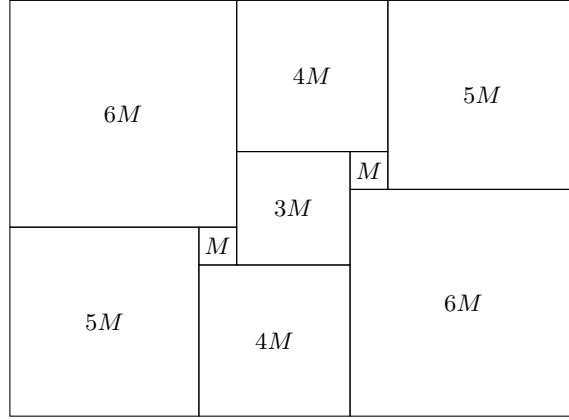


Figure 3: Tiling of a $11M \times 15M$ rectangle.

Let us consider any valid tiling of the cuboid, and analyze the disposition of the $3M$ -cubes, b_i -cubes and Walters' cubes. Their total volume is $27M^3 \times 20k + 120M^3 \times k = 660M^3 \times k$. On average over all the S slices of the cuboid, this represents a surface of $\frac{660M^3 \times k}{S} = 11M^2$. We now prove that it is actually impossible to tile this $11M \times 15M$ face with less than $11M^2$ surface coming from these cubes.

Let s be the surface coming from these cubes divided by M^2 , $s \in [0, 10]$. Let p_6 , respectively p_5 and p_4 , the number of $6M$ -cubes, respectively $5M$ -cubes and $4M$ -cubes, used to tile the $11M \times 15M$ -face. We have computed the possible values such that $(15 \times 11 - s)M^2 = (36p_6 + 25p_5 + 16p_4)M^2$, they can be found in Table 1. We now consider each case one by one.

Case	s	p_6	p_5	p_4
(1)	0	3	1	2
(2)	1	0	4	4
(3)	1	1	0	8
(4)	2	2	3	1
(5)	3	0	2	7
(6)	4	1	5	0
(7)	2	2	1	4

Case	s	p_6	p_5	p_4
	5			
(8)	6	1	3	3
(9)	7	3	2	0
(10)	8	0	5	2
(11)	1	1	6	
(12)	9	3	0	3
(13)	10	0	3	5

Table 1: Possible values of p_6 , p_5 and p_4 as a function of s .

Case (1) $s = 0$, $p_6 = 3$, $p_5 = 1$, $p_4 = 2$: 11 and 15 are odd, therefore there must be a least one fraction of an odd square in every line or column (we can see the rectangle as a grid 11×15). The odd cubes are the $5M$ -cubes, the $3M$ -cubes and the ones we can produces with b_i -cubes and Walters' cubes. Since we have only one $5M$ -cube, the total length of odd squares is less than 15, so we cannot

have one fraction of an odd square in every column. Therefore the tiling in this case is not possible.

Case (2) $s = 1$, $p_6 = 0$, $p_5 = 4$, $p_4 = 4$: As $5 \times 4 > 15$, we cannot have more than three 5-squares in a line and therefore, with four 5-squares, we have to be in one of the dispositions shown in Figure 4(a) and Figure 4(b). In both cases, there exist at least two columns where two 5-squares are superposed (denoted d_i in the figures). However, the only way to complete these columns to a height of 11 is to use a square of size 1, and we have only one M -square since $s = 1$. Therefore the tiling in this case is not possible.

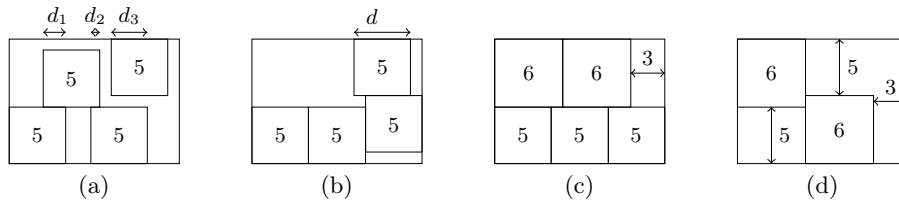


Figure 4: Tiling in cases (2), (4) and (9).

Case (3) $s = 1$, $p_6 = 2$, $p_5 = 0$, $p_4 = 8$: By a parity argument similar to the one used in Case (1), we can prove the impossibility of the tiling.

Case (4) $s = 2$, $p_6 = 2$, $p_5 = 3$, $p_4 = 1$: It is easy to see that the 6-squares and the 5-squares can only be tiled in a way similar to Figure 4(c). Therefore there is no room to place the 4-square and the tiling is impossible in this case.

Case (5) $s = 3$, $p_6 = 0$, $p_5 = 2$, $p_4 = 7$: By a parity argument similar to the one used in Case (1), we can prove the impossibility of the tiling.

Case (6) $s = 4$, $p_6 = 1$, $p_5 = 5$, $p_4 = 0$: By reasoning on the number of 5-squares in a similar fashion than in Case (2), we can prove the impossibility of the tiling.

Case (7) $s = 4$, $p_6 = 2$, $p_5 = 1$, $p_4 = 4$: By a parity argument similar to the one used in Case (1), we can prove the impossibility of the tiling.

Case (8) $s = 6$, $p_6 = 1$, $p_5 = 3$, $p_4 = 3$: Since $5 + 5 + 6 > 15$, we cannot have two 5-squares and one 6-square appearing on the same line. Then we have only two choices to tiles these squares. The first is shown on Figure 5(a). In this case there is a rectangle of size $9M \times 6M$ that can be proved impossible to tile with six M -square and three $4M$ -squares. Therefore we have to be in the case of Figure 5(c), where the dashed zone does not intersect the last $5M$ -square (otherwise we are in the case in Figure 5(b) that is strictly harder to tile than the case of Figure 5(a)). Therefore the dashed zone, a square of size $5M \times 5M$ has to be tiled with only fractions of $4M$ -squares and M -squares. In order to fill the gap of size $5M$ the only way to begin is the one shown in Figure 5(d). To complete the last column, we have to fill the $7M$ -gap, and $7 = 5 + 2 \times 1$ is the only possibility (there are only two M -squares remaining). This yields the

situation shown in Figure 5(e), and one can see that the tiling can not be finished with the remaining $4M$ -squares: the tiling is impossible in this case.

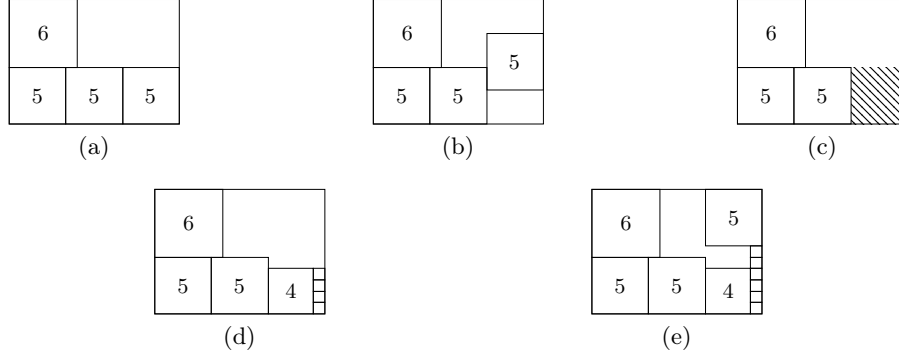


Figure 5: Tiling in case (8).

Case (9) $s = 7$, $p_6 = 3$, $p_5 = 2$, $p_4 = 0$: It is impossible to have three $6M$ -squares. Indeed $2 \times 6 > 11$ and then we cannot have two $6M$ -squares appearing on the same column. In the same time, $3 \times 6 > 15$ and then we cannot have three $6M$ -squares appearing on the same line (see Figure 4(d) to have a better visualization). Hence there is no room to store three $6M$ -squares, and the tiling in this case is not possible.

Case (10) $s = 8$, $p_6 = 0$, $p_5 = 5$, $p_4 = 2$: By reasoning on the number of 5-squares in a similar fashion than in Case (2), we can prove the impossibility of the tiling.

Case (11) $s = 8$, $p_6 = 1$, $p_5 = 1$, $p_4 = 2$: By a parity argument similar to the one used in Case (1), we can prove the impossibility of the tiling.

Case (12) $s = 9$, $p_6 = 3$, $p_5 = 0$, $p_4 = 3$: By a parity argument similar to the one used in Case (1), we can prove the impossibility of the tiling.

Case (13) $s = 10$, $p_6 = 0$, $p_5 = 3$, $p_4 = 5$: We consider two sub-cases: either we have one $3M$ -rectangle and one M -rectangle, or we have ten M -rectangles. In the first case we notice that with the available rectangles there are only two ways to achieve a exact length of $11M$: $1M + 5M + 5M$ and $3M + 4M + 4M$. Both options create gaps that cannot be filled with the remaining rectangles, see Figure 6(a). In the second case (ten M square and no $3M$ -square), there are more options to achieve a length of $11M$. The first one is to use two $4M$ -square and three M -square. This creates a $3M \times 4M$ gap, and there are too few M -squares to fill it, see Figure 6(b), therefore we cannot tile this way. The second option is to use two $5M$ -squares and one M -square. In this case we have to use four more M -squares to complete and then we have again a length $11M$ to achieve, that, with the remaining resources, can only be done either with two $4M$ -squares and three M -squares, and we have proven above that it is not feasible, or with a

$4M$ -square, a $5M$ -square and two M -squares, but in this case we cannot tile the resulting $2M \times 3M$ rectangle because we have too few M -squares, see Figure 6(c). Another option is to use a $4M$ -square and seven M -squares, but this results in a gap of length $7M$ to fill and one can see that is not possible, see Figure 6(d) (replacing the $5M$ -square of the figure by a $4M$ -square yields the same problem). Another possibility is to use a $5M$ -square and six M -squares, for which the only reasonable continuation is the case shown on Figure 6(e) and this creates a gap of length $6M$ which cannot be tiled with the remaining squares. The last option is to use a $5M$ -square, a $4M$ -square and two M -squares but the tiling is still impossible, as it is shown on Figure 6(f). Therefore, in any sub-case, the tiling is impossible in this case.

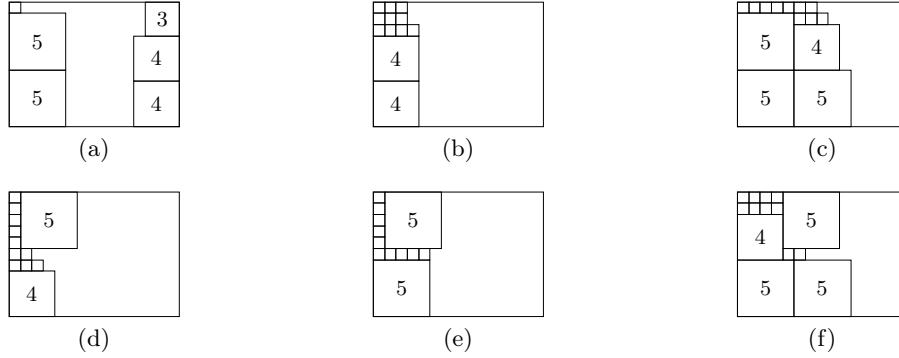


Figure 6: Tiling in case (13).

This proves that at every slice of the main cuboid, the surface coming from the $3M$ -cubes, the b_i -cubes and the Walters' cubes is at least $11M^2$. Since this is exactly the average value, this implies that this surface is exactly $11M^2$ at each slice. We can now observe that this requires at least two M -squares in any slice and once again this is exactly the average value of M -squares. Therefore, in any slice, there are exactly two M -squares and one $3M$ -square. The argument used to create Table 1 now shows that the slice necessarily includes two $6M$ -squares, two $5M$ -squares and two $4M$ -squares, and one can prove that the only way to tile the $11M \times 15M$ rectangle with these squares is as shown on Figure 3.

We have built a pattern that must appear on each slice of the cuboid, and in which the b_i cubes have to be included into two separate parts of the tiling. Let us denote by I the indexes of the b_i cubes which appear in the leftmost $M \times M \times S$ cuboid. Since by construction, $b_i > \frac{M}{2}$ for all i , these cubes have to be arranged as depicted on Figure 2. This shows that $\sum_{i \in I} b_i \leq S$ and $\sum_{i \notin I} b_i \leq S$. Since the total sum is $2S$, this implies that both sums are in fact equal to S , and thus that there exists a solution to the original 2-PART-EQUAL instance.

Note that the pattern in Figure 3 can be horizontally reversed. Furthermore, both possible patterns can be present on the final tiling. However, even in this case, considering the b_i -cubes on the left side still yields a set I such that $\sum_{i \in I} b_i = \sum_{i \notin I} b_i = S$.

□

Theorem 1. *MSCuboidP-DEC is NP-complete.*

Proof. There is a reduction from ACCuboidP to MSCuboidP-DEC. Indeed, $ACCuboidP(\{l_1, \dots, l_p\}, [0, x] \times [0, y] \times [0, z])$ is true if and only if $MSCuboidP-DEC(\{l_1^3, \dots, l_p^3\}, [0, x] \times [0, y] \times [0, z], 3 \sum v_k^{2/3})$ is true (the bound in (1) is tight if and only if there exist a partitioning where only cubes are used). Yet, thanks to Lemma 1, ACCuboidP is NP-complete. Therefore MSCuboidP-DEC is NP-complete.

4 Approximation Algorithm

In this section, we present 3D-NRRP, an approximation algorithm for the case where the cuboid to partition is cubic, what corresponds to the multiplication of square matrices. It is inspired by the NRRP algorithm proposed in [8] and by Nagamochi *et al.* in [18].

4.1 Presentation and correctness of 3D-NRRP

3D-NRRP (see Algorithm 1) is based on the divide and conquer principle, and its analysis relies on the following invariant: at each step, the aspect ratio of the cuboid to be partitioned is smaller than 3. In what follows, we define the aspect ratio ρ of a cuboid as the ratio of the largest length by the smallest length. We also define the second aspect ratio ρ' as the ratio of the largest length to the median length.

At each step of the algorithm, the current cuboid (whose aspect ratio is smaller than 3) is split into two parts, and the same routine is recursively applied to each part. To ensure that the resulting parts have an aspect smaller than 3, the splitting is performed according to three modes. The first mode is the general case, in which the cuboid is partitioned in two disjoint cuboids by cutting along the largest length (Lines 7 to 18 in Algorithm 1, and Figure 7(a)). This is possible if there exists an index k such that $\sum_{i=1}^{k-1} v_i \geq \frac{v}{3\rho^2}$. Indeed, Lemmas 2 and 3 show that this condition is sufficient to prove the invariant for both parts. More specifically, Lemma 2 states that in that case, both resulting cuboids have a total volume greater than one third of the overall volume, and Lemma 3 states that the aspect ratio of both cuboids is smaller than 3, under the assumption that the previous one had also a ratio less than 3, what ensures the correctness of the algorithm.

Algorithm 1: 3D-NRRP

Input: A set of values $\{v_1, \dots, v_n\}$ sorted in non-decreasing order, a cuboid

$$Cu = [a, b] \times [c, d] \times [e, f] \text{ with } (b - a) \times (d - c) \times (f - e) = \sum_{i=1}^n v_i$$

Output: For each $1 \leq i \leq n$ a polyhedron P_i such that $\bigcup P_i = Cu$ and $V(P_i) = v_i$

```
1 if  $n = 1$  then
2   return  $Cu$ 
3 else
4    $v = \sum_{i=1}^n v_i$  ;
5    $w = b - a$  ;  $h = d - c$  ;  $l = f - e$  ;
6    $\rho_1 = \frac{\max(w, h, l)}{\min(w, h, l)}$  ;  $\rho_2 = \frac{\max(w, h, l)}{\min(w, h, l)}$  ;
7   if there exists  $k$  such that  $\sum_{i=1}^{k-1} v_i \geq \frac{v}{3\rho_2}$  then
8      $k =$  the smallest such index  $k$  ;
9      $v' = \sum_{i=1}^{k-1} v_i$  ;
10    if  $w = \max(w, h, l)$  then
11       $Cu_1 = [a, a + v'/(h * l)] \times [c, d] \times [e, f]$  ;
12       $Cu_2 = [a + v'/(h * l), b] \times [c, d] \times [e, f]$  ;
13    else
14      if  $h = \max(w, h, l)$  then
15         $Cu_1 = [a, b] \times [c, c + v'/(w * l)] \times [e, f]$  ;
16         $Cu_2 = [a, b] \times [c + v'/(w * l), d] \times [e, f]$  ;
17      else
18         $Cu_1 = [a, n] \times [c, d] \times [e, e + v'/(w * h)]$  ;
19         $Cu_2 = [a, b] \times [c, d] \times [e + v'/(w * h), f]$  ;
20    return  $\text{3D-NRRP}(\{v_1, \dots, v_{k-1}\}, Cu_1) + \text{3D-NRRP}(\{v_k, \dots, v_n\}, Cu_2)$  ;
21  else
22     $v' = \sum_{i=1}^{n-1} v_i$  and  $\alpha = v'/v$  ;
23    if  $\alpha \rho_1^2 \leq \rho_2$  then
24       $Cu_1 = [a, a + \sqrt[3]{v'}] \times [c, c + \sqrt[3]{v'}] \times [e, e + \sqrt[3]{v'}]$  ;
25    else
26      if  $w = \min(w, h, l)$  then
27         $Cu_1 = [a, b] \times [c, c + \sqrt{v'/w}] \times [e, e + \sqrt{v'/w}]$  ;
28      else
29        if  $h = \min(w, h, l)$  then
30           $Cu_1 = [a, a + \sqrt{v'/h}] \times [c, d] \times [e, e + \sqrt{v'/h}]$  ;
31        else
32           $Cu_1 = [a, a + \sqrt{v'/l}] \times [c, c + \sqrt{v'/l}] \times [e, f]$  ;
33    return  $\text{3D-NRRP}(\{v_1, \dots, v_{n-1}\}, Cu_1) + (Cu \setminus Cu_1)$ 
```

In the second mode, v_n is significantly larger than the other v_i 's. Splitting in two cuboids would result in the smallest cuboid having an aspect ratio larger than 3. Therefore, whenever possible (Lines 23 to 24 of Algorithm 1, and Figure 7(b)), 3D-NRRP shapes the smallest part as a cube included in the covering cuboid of the other part, which is made of one element only, namely v_n . Lemma 4 states that we can put a such cube in the current cuboid under the condition of Line 23 of Algorithm 1. When the condition is not met, the edge length of this cube would be greater than the smallest dimension of the cuboid; thus in that case we create a cuboid like in Figure 7(c) (Line 26-32 of the Algorithm 1) by setting one dimension to the smallest dimension of the current cuboid and making the remaining two dimensions equal. Lemma 5 states that in this case, such a partitioning is possible (the edges of this small cuboid are smaller than the ones of the containing cuboid) and its aspect ratio is below 3.

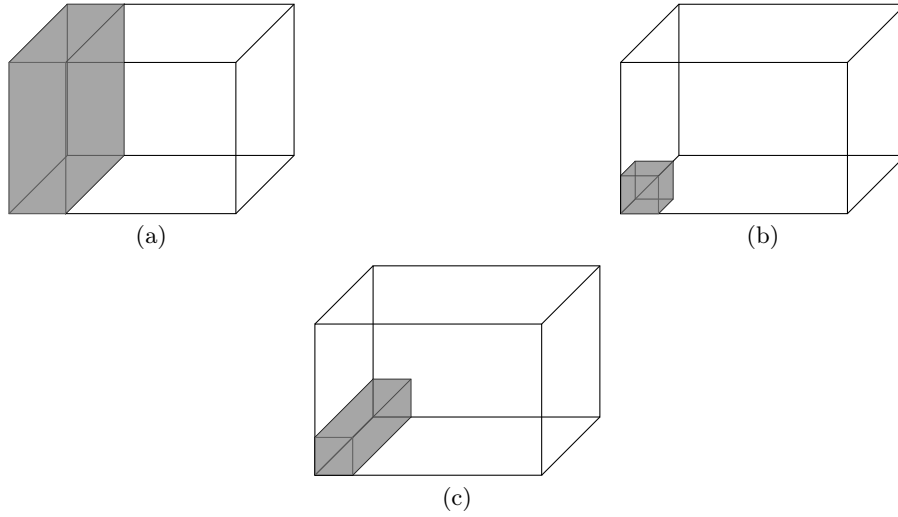


Figure 7: The three splitting modes of 3D-NRRP. In all cases, the gray polyhedra is attributed to $\{v_1, \dots, v_{k-1}\}$, the white one to $\{v_k, \dots, v_n\}$.

Lemma 2. *Let $\{v_1, \dots, v_n\}$ be a set of positive values sorted in non-decreasing order, $\rho \geq 1$, and $v = \sum_i v_i$. Let us assume that there exists an index k such that $\sum_{i=1}^{k-1} v_i \geq v/3\rho$, and let us consider the smallest such integer. Then $\sum_{i=k}^n v_i \geq v/3\rho$.*

Proof. By definition of k , $\sum_{i=1}^{k-2} v_i < v/3\rho$. Therefore, if we assume that $\sum_{i=k}^n v_i < v/3\rho$, we obtain $v_{k-1} = v - \sum_{i=1}^{k-2} v_i - \sum_{i=k}^n v_i \geq v/3\rho$ (since $\rho \geq 1$). Since $v_k \leq$

$\sum_{i=k}^n v_i < v/3\rho$, we have $v_{k-1} > v_k$ which is a contradiction with the fact that the v_i 's are sorted in non-decreasing order.

Lemma 3. *Let Cu be a cuboid of dimension $w \times h \times l$, with volume $V = hwl$, aspect ratio ρ , and second aspect ratio ρ' . Let us assume that Cu_1 and Cu_2 are obtained by cutting Cu along the largest length, with $V(Cu_1)$ and $V(Cu_2)$ not smaller than $\frac{V}{3\rho'}$. Then, $\rho(Cu_1) \leq \max(3, \rho)$ and $\rho(Cu_2) \leq \max(3, \rho)$.*

Proof. Let us suppose that $w \leq h \leq l$ without loss of generality. In this case, $w = w(Cu_1) = w(Cu_2)$, $h = h(Cu_1) = h(Cu_2)$, $\rho = l/w$ and $\rho' = l/h$. Let us denote $l_1 = l(Cu_1)$ and $l_2 = l(Cu_2)$ and let us consider cuboid Cu_1 . Then, there are 3 cases to consider:

- If $h \leq l_1$, then $\rho(Cu_1) = l_1/w \leq l/w = \rho$,
- If $w \leq l_1 \leq h$, then $\rho(Cu_1) = h/w \leq l/w \leq \rho$,
- If $l_1 \leq w \leq h$, by assumption $w \times h \times l_1 = V(Cu_1) \geq \frac{V}{3\rho'} = \frac{w \times h \times l}{3\rho'}$. Therefore $l_1 \geq l/3\rho'$. Then, $\rho(Cu_1) = h/l_1 \leq \frac{3h\rho'}{l} = 3$.

Thus, in all cases, $\rho(Cu_1) \leq \max(3, \rho)$. By symmetry, the same proof applies to Cu_2 .

Lemma 4. *Let Cu be a cuboid of dimension $w \times h \times l$, with volume $V = hwl$, aspect ratio ρ , and second aspect ratio ρ' . For any $\alpha > 0$ such that $\alpha\rho^2 \leq \rho'$, then $\sqrt[3]{\alpha V} \leq \min(w, h, l)$.*

Proof. Without loss of generality, we can suppose $w \leq h \leq l$ and therefore $l = \rho w$ and $l = \rho' h$ and finally $h = \frac{\rho}{\rho'} w$. Thus:

$$\sqrt[3]{\alpha V} = \sqrt[3]{\alpha whl} = \sqrt[3]{\alpha \frac{\rho^2}{\rho'} w} \leq w.$$

Lemma 5. *Let Cu be a cuboid of dimension $w \times h \times l$, with volume $V = hwl$, aspect ratio ρ , and second aspect ratio ρ' . Let α be such that $\frac{\rho'}{\rho^2} < \alpha < \frac{1}{3\rho'}$. Let us denote $\min_d = \min(w, h, l)$. Then $\sqrt{\alpha \frac{V}{\min_d}} \leq \min(w, h, l)$. In addition, if Cu' is a cuboid of dimensions $\min_d \times \sqrt{\alpha \frac{V}{\min_d}} \times \sqrt{\alpha \frac{V}{\min_d}}$ then $\rho(Cu') \leq \rho$.*

Proof. Without loss of generality we can suppose $\min_d = w \leq h \leq l$ and therefore $l = \rho w$ and $l = \rho' h$ and finally $h = \frac{\rho}{\rho'} w$. Thus:

$$\sqrt{\alpha \frac{V}{w}} = \sqrt{\alpha hl} = (\sqrt{\alpha \rho'}) h \leq \frac{h}{\sqrt{3}} \leq h.$$

Now, $\frac{\rho'}{\rho^2} < \alpha$ implies $\sqrt{\alpha \rho'} \geq \frac{\rho'}{\rho}$, and thus $\sqrt{\alpha \frac{V}{w}} = h \sqrt{\alpha \rho'} \geq \frac{h \rho'}{\rho} = w$. Then $\rho(Cu') = \frac{\sqrt{\alpha \frac{V}{w}}}{w}$, and the previous result yields $\rho(Cu') \leq \frac{h}{w} = \frac{\rho}{\rho'} \leq \rho$.

4.2 Approximation Ratio

This section is devoted to the proof of Theorem 2, which states that the 3D-NRRP achieves a $\frac{5}{6^{2/3}}$ approximation ratio ($\frac{5}{6^{2/3}} \simeq 1.51$).

Theorem 2. *3D-NRRP is a $\frac{5}{6^{2/3}}$ -approximation when the given cuboid is cubic.*

The sketch of the proof is as follows. First, we prove that if $\{P_1, \dots, P_n\}$ is an output of 3D-NRRP, then any output polyhedron P_i satisfies $\frac{Hs(P_i)}{3V(P_i)^{2/3}} \leq \frac{5}{6^{2/3}}$.

Remind that Equation (1) states that $Hs(P_i^*) \geq 3V(P_i^*)^{2/3}$ for any (optimal) solution $\{P_1^*, \dots, P_n^*\}$. By summing up these inequalities for all i , we get that $\sum_i Hs(P_i) \leq \frac{5}{6^{2/3}} \sum_i Hs(P_i^*)$ and obtain the approximation result claimed in Theorem 2.

The rest of the section is devoted to the proof that any polyhedron returned by 3D-NRRP satisfies the property stated above. One can see that there are only two situations in which 3D-NRRP returns a singleton: Line 2 and Line 33. In the first case, the returned zone is a cuboid with aspect ratio less than 3. In this case Lemma 7 provides the desired result, since $\frac{5}{3^{2/3}} \leq \frac{5}{6^{2/3}}$. We first start with a technical result in Lemma 6.

Lemma 6. *Let $f(x, y) = \frac{y+x(1+y)}{3(xy)^{2/3}}$. Then, with $x \in [1, y]$ and $y \in [1, 3]$, $f(x, y) \leq \frac{5}{3^{2/3}}$.*

Proof. One can show that $\frac{\partial f}{\partial x}(x, y) = \frac{2x-1-y}{9y^{1/3}x^{4/3}}$. Since $x \geq 0$ and $y \geq 0$, $\frac{\partial f}{\partial x}(x, y) \geq 0$ if and only if $x \leq \frac{1+y}{2}$. Hence, $x \mapsto f(x, y)$ is decreasing on $[1, \frac{1+y}{2}]$ and increasing in $[\frac{1+y}{2}, y]$, which implies $f(x, y) \leq \max(f(1, y), f(y, y))$.

$f(1, y) = \frac{y+2}{3^{2/3}y}$ and $f(y, y) = \frac{2y+1}{3y^{2/3}}$. From $y \geq 1$, one can obtain $(y+2)^{2/3}y \geq 2y+1$, and this implies $\frac{y+2}{3^{2/3}y} \geq \frac{2y+1}{3y^{2/3}}$. Let us denote $g(y) = \frac{y+2}{3^{2/3}y}$: the previous statements show that $f(x, y) \leq g(y)$. One can prove that $g'(y) = \frac{2(y-1)}{9y^{4/3}}$. Therefore g is increasing on $[1, \infty]$ and, in the considered case, $g(y) \leq g(3) = \frac{5}{3^{2/3}}$, which proves the desired result.

Lemma 7. *If P is a cuboid with $\rho(P) \leq 3$, then $\frac{Hs(P)}{3V(P)^{2/3}} \leq \frac{5}{3^{2/3}}$.*

Proof. We denote $\rho(P) = \rho$ and $V(P) = V$. We suppose that $w = w(P) \leq h = h(P) \leq l(P) = l$ without loss of generality. We denote $x = h/w$. In this case $l = \rho w$ and $V = whl = \rho x w^3$. Therefore

$$\frac{Hs(P)}{3V^{2/3}} = \frac{(x + \rho + x\rho)w^2}{3(\rho x w^3)^{2/3}} = \frac{\rho + x(1 + \rho)}{3(\rho x)^{2/3}} = f(x, \rho),$$

where f is as defined in Lemma 6, what ends the proof.

In the other case, 3D-NRRP returns a cuboid minus a cube (Line 22 of Algorithm 1, as described on Figure 7(b)). The bound on the volume of the cube is such that the conditions of Lemma 9 are fulfilled. As before, this relies on the technical Lemma 8.

Lemma 8. Let $f(x, y) = \frac{y+x(1+y)}{3(xy-\frac{x^2}{3})^{2/3}}$. Then, with $x \in [1, y]$ and $y \in [1, 3]$, $f(x, y) \leq \frac{5}{6^{2/3}}$.

Proof. One can prove that $\frac{\partial f}{\partial x}(x, y) = \frac{\frac{1+y}{3}x^2 + (\frac{7}{3}+y)yx - 2y^2}{9(yx-\frac{x^2}{3})^{5/3}}$. Then $\frac{\partial f}{\partial x}$ has the same sign than $P_y(x) = \frac{1+y}{3}x^2 + (\frac{7}{3}+y)yx - 2y^2$. Since P_y is a second order polynomial with $P_y(0) = -2y^2 < 0$ and $\lim_{x \rightarrow +\infty} p_y(x) = +\infty$, P_y is either positive on $[1, y]$, negative on $[1, y]$, or negative and then positive. Therefore $x \mapsto f(x, y)$ on $[1, y]$ is either increasing, decreasing or decreasing and then increasing. In any case, $f(x, y) \leq \max(f(1, y), f(y, y))$.

Let $g(y) = f(1, y) = \frac{1+2y}{3(y-1/3)^{2/3}}$. One can show that $g'(y) = \frac{2(y-2)}{9(y-1/3)^{5/3}}$, hence g is decreasing on $[1, 2]$ and increasing on $[2, 3]$. Therefore $g(y) \leq \max(g(1), g(3)) = \max((\frac{3}{2})^{2/3}, \frac{7}{4\sqrt[3]{3}}) = (\frac{3}{2})^{2/3}$.

Let $h(y) = f(y, y) = \frac{2+y}{\sqrt[3]{12y}}$. One can show that $h'(y) = \frac{2(y-1)}{\sqrt[3]{12y^2}}$, hence h is increasing on $[1, 3]$. Therefore $h(y) \leq h(3) = \frac{5}{6^{2/3}}$.

Putting it all together, we get $f(x, y) \leq \max((\frac{3}{2})^{2/3}, \frac{5}{6^{2/3}}) = \frac{5}{6^{2/3}}$.

Lemma 9. If $V(P) \geq (1 - \frac{1}{3\rho'(P)})V(Cu(P))$ and $\rho(P) \leq 3$, then $\frac{Hs(P)}{3V(P)^{2/3}} \leq \frac{5}{6^{2/3}}$.

Proof. Let us denote $\rho = \rho(P)$, $\rho' = \rho'(P)$, $V = V(P)$, and let us denote by w , h , l for the dimensions of $Cu(P)$. Let us suppose that $w \leq h \leq l$ without loss of generality. Then, $l = \rho w$ and $l = \rho' h$, and let us also denote $x = h/w = \rho/\rho'$. With such notations, we get $V(Cu(P)) = \rho x w^3$ and $Hs(P) = (\rho + x(1 + \rho))w^2$. Thus, the condition on $V(P)$ can be written as

$$V(P) \geq (1 - \frac{1}{3\rho'(P)})V(Cu(P)) = (1 - \frac{x}{3\rho})\rho x w^3 = (\rho x - \frac{x^2}{3})w^3,$$

$$\text{what leads to } \frac{Hs(P)}{3V(P)^{2/3}} \leq \frac{(\rho + x(1 + \rho))w^2}{3(\rho x - \frac{x^2}{3})^{2/3}w^2} = \frac{\rho + x(1 + \rho)}{3(\rho x - \frac{x^2}{3})^{2/3}} = f(x, \rho),$$

where f is as defined in Lemma 8, what provides the conclusion.

5 Conclusion

We introduce a model of the partitioning problem associated to the 2.5D matrix multiplication algorithm on heterogeneous resources, a problem of crucial importance in high performance computing. This corresponds to partitioning a cuboid into several polyhedra, each representing the volume of computations attributed to a resource. We provide two theoretical results: a proof of the NP-completeness for this problem, and an approximation result for 3D-NRRP, which generalizes the results obtained in the 2D case. This is the first known approximation result for

this problem, and it provides a strong guarantee ($\frac{5}{6^{2/3}} \simeq 1.51$). In addition, the computational time of the algorithm is extremely low, $O(n \log n)$, where n is the number of processors, what makes it perfectly suitable for practical use. This work opens several interesting perspectives for dimensions higher than 3, what corresponds to tensor products. It would also be interesting to combine this algorithm, whose general goal is to minimize the overall volume of communications while enforcing perfect load balancing between heterogeneous resources, with algorithms that explicitly take into account memory constraints at each node.

References

1. Anderson, M., Ballard, G., Demmel, J., Keutzer, K.: Communication-avoiding QR Decomposition for GPUs. In: Parallel & Distributed Processing Symposium (IPDPS), 2011 IEEE International. pp. 48–58. IEEE (2011)
2. Augonnet, C., Thibault, S., Namyst, R., Wacrenier, P.A.: StarPU: A Unified Platform for Task Scheduling on Heterogeneous Multicore Architectures. *Concurrency and Computation: Practice and Experience*, Special Issue: Euro-Par 2009 23, 187–198 (Feb 2011), <http://hal.inria.fr/inria-00550877>
3. Ballard, G., Demmel, J., Holtz, O., Lipshitz, B., Schwartz, O.: Communication-optimal Parallel Algorithm for Strassen’s Matrix Multiplication. In: Proceedings of the twenty-fourth annual ACM symposium on Parallelism in algorithms and architectures. pp. 193–204. ACM (2012)
4. Ballard, G., Demmel, J., Holtz, O., Schwartz, O.: Minimizing Communication in Linear Algebra. *SIAM Journal on Matrix Analysis and Applications* 32(3), 866–901 (Jul 2011), <http://arxiv.org/abs/0905.2485>, arXiv: 0905.2485
5. Beaumont, O., Boudet, V., Rastello, F., Robert, Y.: Matrix Multiplication on Heterogeneous Platforms. *IEEE Transactions on Parallel and Distributed Systems* 12(10), 1033–1051 (Oct 2001)
6. Beaumont, O., Boudet, V., Rastello, F., Robert, Y., et al.: Partitioning a Square into Rectangles: NP-completeness and Approximation Algorithms. *Algorithmica* 34(3), 217–239 (2002)
7. Beaumont, O., Eyraud-Dubois, L., Guermouche, A., Lambert, T.: Comparison of Static and Dynamic Resource Allocation Strategies for Matrix Multiplication. In: Proceedings of the 26th IEEE International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD), pp. 1–10. IEEE (2015)
8. Beaumont, O., Eyraud-Dubois, L., Lambert, T.: A New Approximation Algorithm for Matrix Partitioning in Presence of Strongly Heterogeneous Processors. In: 30th IEEE International Parallel and Distributed Processing Symposium (2016)
9. Becker, B., Lastovetsky, A.: Towards Data Partitioning for Parallel Computing on Three Interconnected Clusters. In: Parallel and Distributed Computing, 2007. ISPDC’07. Sixth International Symposium on. pp. 39–39. IEEE (2007)
10. Bosilca, G., Bouteiller, A., Danalis, A., Faverge, M., Hérault, T., Dongarra, J.: PaRSEC: A Programming Paradigm Exploiting Heterogeneity for Enhancing Scalability. *Computing in Science and Engineering* 15(6), 36–45 (Nov 2013)
11. Choi, J., Demmel, J., Dhillon, I., Dongarra, J., Ostrouchov, S., Petitet, A., Stanley, K., Walker, D., Whaley, R.C.: ScaLAPACK: A Portable Linear Algebra Library for Distributed Memory Computers: Design Issues and Performance. In: APCC in Physics, Chemistry and Engineering Science, pp. 95–106. Springer (1995)

12. Clarke, D., Ilic, A., Lastovetsky, A., Sousa, L.: Hierarchical Partitioning Algorithm for Scientific Computing on Highly Heterogeneous CPU + GPU Clusters. In: Euro-Par 2012 Parallel Processing, pp. 489–501. Springer (2012)
13. DeFlumere, A., Lastovetsky, A., Becker, B.: Optimal Data Partitioning Shape for Matrix Multiplication on Three Fully Connected Heterogeneous Processors. In: Euro-Par 2014: Parallel Processing Workshops. pp. 201–214. Springer (2014)
14. Fügenschuh, A., Junosza-Szaniawski, K., Lonc, Z.: Exact and Approximation Algorithms for a Soft Rectangle Packing Problem. *Optimization* 63(11), 1637–1663 (2014)
15. Hoemmen, M.: Communication-avoiding Krylov Subspace Methods. Ph.D. thesis, University of California, Berkeley (2010)
16. Kalinov, A., Lastovetsky, A.: Heterogeneous Distribution of Computations Solving Linear Algebra Problems on Networks of Heterogeneous Computers. *Journal of Parallel and Distributed Computing* 61(4), 520–535 (2001)
17. Mohamed, N., Al-Jaroodi, J., Jiang, H.: DDOps: Dual-Direction Operations for Load Balancing on Non-dedicated Heterogeneous Distributed Systems. *Cluster Computing* 17(2), 503–528 (2014)
18. Nagamochi, H., Abe, Y.: An Approximation Algorithm for Dissecting a Rectangle into Rectangles with Specified Areas. *Discrete Applied Mathematics* 155(4), 523 – 537 (2007)
19. Planas, J., Badia, R.M., Ayguadé, E., Labarta, J.: Hierarchical Task-based Programming with StarSs. *International Journal of High Performance Computing Applications* 23(3), 284–299 (2009)
20. Shams, R., Sadeghi, P.: On Optimization of Finite-difference Time-domain (FDTD) Computation on Heterogeneous CPU and GPU clusters. *Journal of Parallel and Distributed Computing* 71(4), 584–593 (2011)
21. Solomonik, E., Demmel, J.: Communication-optimal Parallel 2.5 D Matrix Multiplication and LU factorization Algorithms. In: Euro-Par 2011 Parallel Processing, pp. 90–109. Springer (2011)
22. Walters, M.: Rectangles as Sums of Squares. *Discrete Mathematics* 309(9), 2913 – 2921 (2009)