Minimum Decomposition of a Digital Surface into Digital Plane Segments is NP-Hard

Isabelle Sivignon, David Coeurjolly

Minimum Decomposition of a Digital Surface into Digital Plane Segments is NP-Hard

Isabelle Sivignon, David Coeurjolly

LIRIS, CNRS UMR-5205, Université Claude Bernard Lyon 1, 43 boulevard du 11 novembre 1918, F-69622 Villeurbanne, France

Abstract

This paper deals with the complexity of the decomposition of a digital surface into digital plane segments (DPS for short). We prove that the decision problem (does there exist a decomposition with less than λ DPS ?) is NP-complete, and thus that the optimisation problem (finding the minimum number of DPS) is NP-hard. The proof is based on a polynomial reduction of any instance of the well-known 3-SAT problem to an instance of the digital surface decomposition problem. A geometric model for the 3-SAT problem is proposed.

Key words: digital object, digital plane, decomposition, complexity

1 Introduction

Digital objects are defined as connected sets of grid points in $\mathbb{Z}^n$. Those objects carry redundant geometrical information due to their discrete structure: an object is represented as a set of elementary cells (called pixels in 2D, voxels in 3D). The definition of digital linear structures such as digital lines [1] and digital planes [2,3] originated a lot of works dealing with the decomposition of the surface of a digital object into digital linear primitives. Such a decomposition actually apprehends global geometrical properties of these objects and is the first step toward an efficient reversible polyhedrization process (see Figure 1) [4–6]. Many decomposition strategies may be designed and the number of parts computed by the algorithms may be a first criterion to compare the results. In this work, we focus on the complexity of the optimal (minimum number of parts) decomposition problem. In the 2D case, it has been shown that the minimum decomposition of a digital curve into digital line segments can be computed in linear time [7].

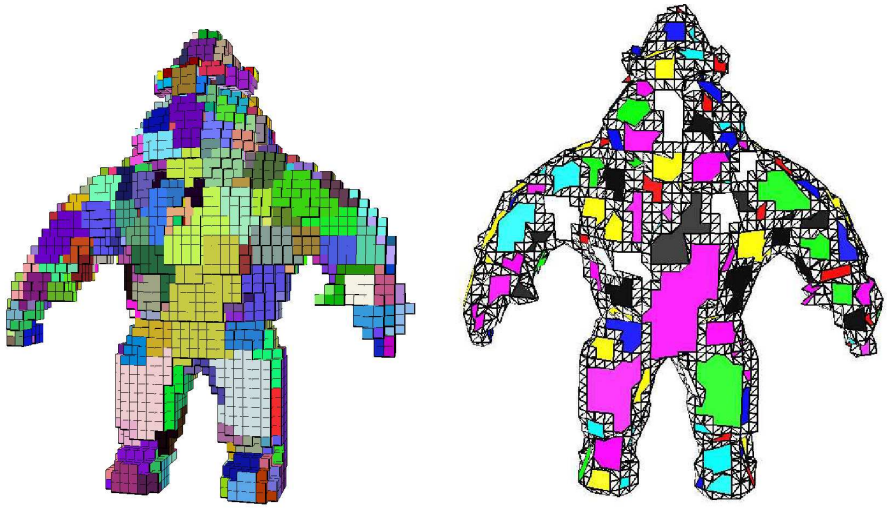


Fig. 1. The decomposition of a surface into DPS is the key step to obtain a reversible polyhedrization of the object.

In the 3D case, the boundaries of 3D objects define surfaces for which many decomposition algorithms have been proposed [8,9,5,6], offering comparisons on the number of digital plane segments recognised by different algorithms. Nevertheless, no optimality results exist, and no complexity study has been carried out.

In computational geometry, the decomposition of a shape (*e.g.* a polygon) into a minimum number of elements (*e.g.* convex polygons) usually leads to NP-complete problems [10]. A problem is in the NP class of algorithms if it can be solved in polynomial time by a non-deterministic machine [11]. As a corollary, the problem is in NP if a solution to the decision problem can be verified in polynomial time on a deterministic machine. A problem is said to be NP-complete if it is at least as difficult as any NP problem. In other words, if a problem is NP-complete, a conjecture is that no time efficient solution exists to solve it. The remaining option is to consider approximation algorithms with heuristics.

2 Problem statement

Prior to a complexity study, the problem has to be formalised. In the sequel, we consider 6-connected sets of voxels whose surface $\mathbb{S}$ is defined as the set of object voxels sharing a face with the background. Such a definition of the object surface may seem to be simple but many topological results can be derived [2]. The surface is a set of 18-connected voxels, and digital naive planes [12,13,2,3] are used for the decomposition. More particularly, we consider digital plane segments (*DPS* for short), which are 18-connected sets of voxels that

belong to the same digital naive plane. A decomposition of a surface into DPS consists in a labeling of all the surface voxels with a DPS tag. In the framework we consider, a voxel belongs to exactly one DPS of the decomposition. A DPS P is *maximal* on a given 18-connected surface if any surface voxel v 18-connected to P is either already labeled or such that $P \cup v$ is not a DPS.

Related results have been recently proposed in [14] concerning the NP-completeness of the construction of an integer lattice polyhedron P with minimum number of convex facets such that $P \cap \mathbb{Z}^3$ corresponds to the input 3D digital object. Even if the final objective of the DPS segmentation is to construct a polyhedral representation of the binary object (see Figure 1), we focus here on the segmentation step. Furthermore, the reduction presented in [14] is based on the NP-completeness of the decomposition of a polygon with holes into a minimum number of convex polygons (see below). In our framework, we do not have such a restriction.

In the following, we consider the decompositions resulting from a sequential decomposition algorithm, generically defined in Algorithm 1.

Algorithm 1 Sequential decomposition of a discrete surface $\mathbb{S}$ into DPS

1: choose a voxel on $\mathbb{S}$; this voxel is called a *seed*;
2: construct the maximal DPS iteratively adding voxels that are 18-connected to the DPS initialised with the seed and label these voxels;
3: choose an unlabeled voxel on $\mathbb{S}$ as a new seed and repeat from step 1 until all the voxels of $\mathbb{S}$ are labeled.

In this algorithm, both the propagation process during the DPS growing and the seeds initialisation may change the final resulting decomposition. We now have all the elements to define the optimisation problem we consider:

Min-DSD (Minimum Digital Surface Decomposition): Given a digital object surface $\mathbb{S}$, find a decomposition of $\mathbb{S}$ into a minimum number of maximal digital naive plane segments using Algorithm 1 .

In order to study the complexity of an optimisation problem, the related decision problem has to be considered:

λ-DSD: Given a digital object surface $\mathbb{S}$ and a number $\lambda \in \mathbb{N}^*$, does there exist a decomposition of $\mathbb{S}$ into λ maximal digital naive plane segments using Algorithm 1?

In this article, we prove that λ-DSD is NP-complete whatever the propagation heuristic. Furthermore, the only requirement on the DPS topology is connectivity.

To prove that a problem $\mathcal{P}$ is NP-complete, a classical scheme is to exhibit a polynomial reduction of any instance of a classical NP-complete problem,

denoted $\mathcal{P}_{NP}$ into an instance of $\mathcal{P}$. Then, we have to prove that a solution of $\mathcal{P}$ also leads to a solution of $\mathcal{P}_{NP}$. Since $\mathcal{P}_{NP}$ is known to be NP-complete, we could conclude that $\mathcal{P}$ is also NP-complete [11]. In the literature, the Boolean Satisfiability Problem (SAT) is a decision problem classically used in complexity theory since it was the first known NP-complete problem. An instance of SAT is a boolean expression written using only boolean operators AND, OR and NOT, literals (positive or negative instance of a boolean variable) and parentheses. The decision problem is: given an expression, is there an assignment of the variables such that the expression is TRUE ? The problem remains NP-complete even if the expression is written in conjunctive normal form with three literals per clause, yielding the 3-SAT problem. An expression ϕ has the form:

$$(x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_4 \vee x_5) \wedge (\neg x_6 \neg \vee x_3 \vee \neg x_5) \wedge \ldots , \quad (1)$$

where each x_i is a binary variable (and $\neg x_i$ its negation) that can appear several times in the expression.

In the following we define a polynomial reduction of any instance ϕ of the 3-SAT problem to an instance of the λ-DSD problem. This reduction consists in defining a discrete object surface $\mathbb{S}(\phi)$ and a value $\lambda(\phi)$ such that the expression ϕ is satisfiable if and only if $\mathbb{S}(\phi)$ can be decomposed into at most $\lambda(\phi)$ DPS. The construction process, defining geometrical objects for variables, literals (instance of a variable in the boolean expression) and clauses, is presented in Section 3, while the NP-completeness proof derived from this construction is given in Section 4.

3 A Geometric Model for 3-SAT

Given a 3-SAT expression ϕ, we show how to construct a geometric discrete object. This construction is a two step process: after defining geometric objects for variables, literals and clauses, we see how these basic components are organised and connected together in the 3D space.

3.1 General Considerations

Any instance of the 3-SAT problem can be represented by a bipartite graph as depicted in Figure 2. The reduction from 3-SAT to λ-DSD we propose involves the construction of a digital geometric embedding of any graph of 3-SAT. Three geometric objects (called *gadgets*) must be defined to represent the nodes of the graph (variables and clauses in the boolean expression) and

the edges of the graph (literals in the boolean expression). We use the term "gadget" to name these objects, following the classical vocabulary of NP-completeness proofs for geometric problems:

Definition 1 *We call v-gadgets, c-gadgets and l-gadgets the digital objects encoding respectively variables, clauses and literals of a 3-SAT expression.*

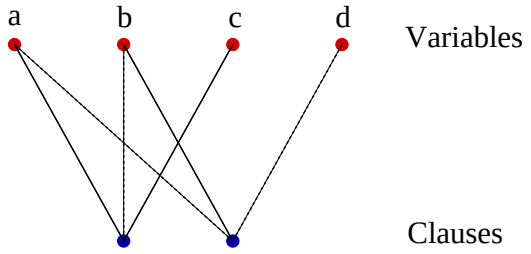


Fig. 2. Graph representation of the 3-SAT boolean expression $(\neg b \vee a \vee c) \wedge (\neg a \vee \neg d \vee b)$: positive literals are represented with black edges and negative ones with dotted edges.

A key point of the reduction from 3-SAT to λ-DSD is that the number of DPS needed to decompose the object surface has to be perfectly defined. To do so, we define the notion of *incompatible sets* which enables an exact counting of the DPS:

Definition 2 *Given the surface of discrete object, two sets of surface voxels S and S' are said to be incompatible if for all x in S and for all y in S', x and y cannot be part of the same DPS.*

n sets of voxels are said to be incompatible if they are pairwise incompatible. Thus, if n incompatible sets can be defined on a given surface, at least n DPS are required to decompose the surface.

Moreover, we set up a common scheme for the construction of all the gadgets, which are composed of two main parts:

- **idle part**: the surface of this part is made of planes parallel to the axis planes and only aims at defining a 6-connected object. The minimum number of DPS needed to cover the idle part is fixed for each gadget, and thus does not play any role in the optimisation of the total number of DPS needed to decompose the whole surface. This part is not used in the encoding of the 3-SAT expression;
- **active part**: this part consists of the remaining voxels after the decomposition of the idle part. It takes advantage of digital planes properties to geometrically encode a 3-SAT expression.

For each gadget, we provide an illustration [1] of incompatible sets that can be defined on the surface. Moreover, we also give illustrations of active and idle parts of the surface. These figures aim at helping the reader in the understanding of the proof. By construction, a DPS cannot cover both active and idle voxels. A DPS which contains active (resp. idle) voxels is called active DPS (resp. idle DPS). Surface voxels that are neither active nor idle are neutral and can be labeled by any type of DPS. They do not play any specific part in the decomposition of the surface.

The underlying basic idea for this construction is the following: the decomposition of a v-gadget generates a "signal" sent to c-gadgets through l-gadgets that represent literals. For each c-gadget, the minimum decomposition is such that at least one of the three incoming l-gadgets carries a "true" signal. This kind of geometric construction of 3-SAT is a classic way to prove NP-completeness of geometric problems (see [15,16] for instance).

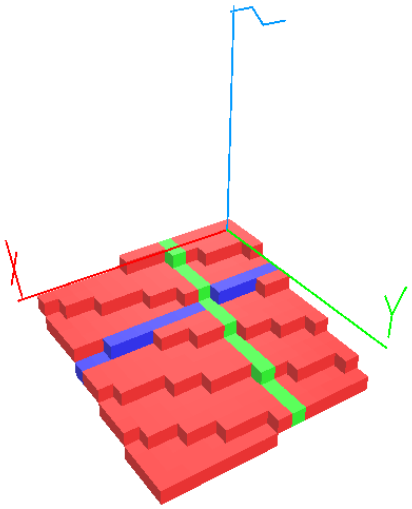


Fig. 3. Illustration of a DPS$(7, 17, 57, 0)$. Each intersection between such a DPS and a plane parallel to the axis grid is a digital straight line.

In the following, we consider digital naive plane segments $DPS(a, b, c, \mu)$ defined as connected sets of voxels satisfying (see Figure 3):

$$0 \leq ax + by + cz + \mu < \max(|a|, |b|, |c|) \tag{2}$$

with a, b, c and $\mu \in \mathbb{Z}$ [12,9,2]. We also introduce the notation $r(P) = ax + by + cz + \mu$ for a point $P(x, y, z)$. This analytical definition of DPS is of

[1] Most illustrations of this paper are originally colour artworks. To make the understanding of the paper easier from B&W printings, colour images are available on `http://liris.cnrs.fr/isabelle.sivignon/SatDSD.html`

help to prove some structural properties of DPS we use in the NP-completeness proof. These properties are set forth here for DPS with $0 \leq a \leq b < c$, but simple permutations can be done to generalize them (Figure 4):

Proposition 3

- *Consider two voxels P and Q and a 3D 18-connected curve $\mathcal{C}$ linking P and Q. We consider the four following configurations (see Figure 4 (a)-(d) for illustrations):*
  (a) $P(x, y, z)$, $Q(x + 2k + 2, y, z - 2)$ and $\mathcal{C} = \{(x + i, y, z - 1) \mid 1 \leq i \leq k\} \cup \{(x + i, y, z - 2) \mid k + 1 \leq i \leq 2k + 1\}$;
  (b) $P(x, y, z)$, $Q(x + 2, y, z)$ and $\mathcal{C} = \{(x + 1, y, z + 1)\}$;
  (c) $P(x_1, y_1, z)$, $Q(x_2, y_2, z)$ and $\mathcal{C} = \{(x_1 + i, y_1, z - 1) \mid 1 \leq i \leq k\} \cup \{(x_1 + k, y_1 + i, z - 1) | 1 \leq i \leq y_2 - y_1\} \cup \{(x, y_2, z - 1) | x + k \leq x < x_2\}$;
  (d) $P(x, y, z)$, $Q(x + k, y, z + 2)$ and $\mathcal{C} = \{(x, y + 1, z + 1)\} \cup \{(x + i, y + 2, z + 1) | 0 \leq k\} \cup \{(x + k, y + 1, z + 1)\}$.

  Then, one DPS cannot simultaneously cover all the voxels of $\mathcal{C}$, P and Q. But there exist DPS that contain P and $\mathcal{C}$ or Q and $\mathcal{C}$.
- *(e) Consider three voxels $P(x_1, y, z)$, $Q(x_2, y, z-1)$ and $R(x_3, y, z)$ such that $x_1 < x_2 < x_3$ (see Figure 4(e)). Then P, Q and R cannot be labeled by one DPS, but any pair can.*
- *(f) Consider three voxels $P(x, y, z)$, $Q(x+1, y, z)$ and $R(x+1, y, z+1)$ (see Figure 4(f)). Then P, Q and R cannot be labeled by one DPS.*

These six properties remain true for any permutation of x, y and z coordinates.

PROOF. The proofs of these properties directly ensue from structural or arithmetical properties of digital nave planes. For configurations (a) to (d), there exist DPS that contain P and $\mathcal{C}$ or Q and $\mathcal{C}$. Thus we only focus on the non-existence of a DPS covering simultaneously cover all the voxels of $\mathcal{C}$, P and Q. We provide either adequate references when the proofs are straightforward or extensive proofs for more complicated configurations.

(a) In this configuration, the main point is that $P \cup Q \cup \mathcal{C}$ contains a step (i.e. connected set of voxels with fixed y and z, in this example) of length k and a step of length $k + 2$. It is a well known property that digital planes only contain steps of length k and $k + 1$ [12,9,2]. Thus a DPS containing P and $\mathcal{C}$ cannot contain Q and conversely.

(b) Consider the projection of this set of voxels on $(0xz)$. Then, if the set of voxels were part of a DPS, then its projection would be part of a digital straight segment [12,2]. But two chain codes that differ by 2 define this projection, which proves that it cannot be a digital straight segment [1].

(c) Let us denote $P(x_1, y_1, z)$, $P'(x_1 + 1, y_1, z - 1)$, $Q'(x_2 - 1, y_2, z - 1)$ and $Q(x_2, y_2, z)$. Suppose that there exist a DPS $\mathcal{P}(a, b, c\mu)$ containing P, P',

Q and Q'. We have $-c < r(P') - r(P) = -c + a < c$, which implies $a > O$ and $-c < r(Q) - r(Q') = c + a < c$, which implies $a < 0$, and leads to a contradiction.

(d) Suppose that there exist a DPS $\mathcal{P}(a, b, c\mu)$ containing both P and Q. Then, on the x-axis, $\mathcal{P}$ contains a step of length $l \leq k - 1$ at height $z + 1$ between P and Q . However, by definition, the set $\mathcal{C}$ contains a step of length $k + 1$. We conclude using the same argument as in (a).

(e) Again, any pair of points can be labeled by one DPS. The proof that there does not exist a DPS containing all three points is easy using similar arguments as in (c).

(f) The DPS we are considering in the paper are naive DPS, that are by definition functional along one axis (z in the case $0 \leq a \leq b < c$). Thus, Q and R cannot belong to a common DPS.

In the following, we refer to these configurations as Proposition 3(a), (b), (c), (d), (e) and (f).

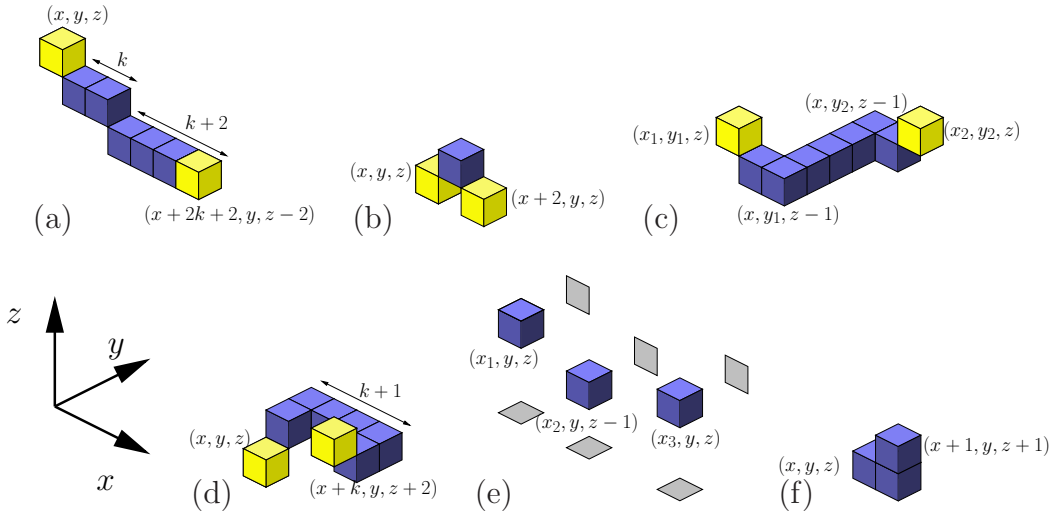


Fig. 4. Six configurations used for the reduction process

3.2 Encoding Variables: v-gadgets

In this section, we provide a constructive description of a v-gadget in order to give an hint on the underlying idea behind the reduction we propose. The definitions of c-gadgets and l-gadgets will be shorter since the principle is basically the same. In the following, we denote $l_a(gadget)$ the length of a given gadget along the axis $a \in \{x, y, z\}$.

Formally, the v-gadget representing a variable is the set of voxels defined by (up to translations):

$$\{(x,y,z)|0 \leq x < 5, 0 \leq y < l_y(v\text{-gadget}), 0 \leq z < 6\} \cup$$
$$\{(x,y,z)|x = 2, 0 \leq y < l_y(v\text{-gadget}), z = 6\}$$

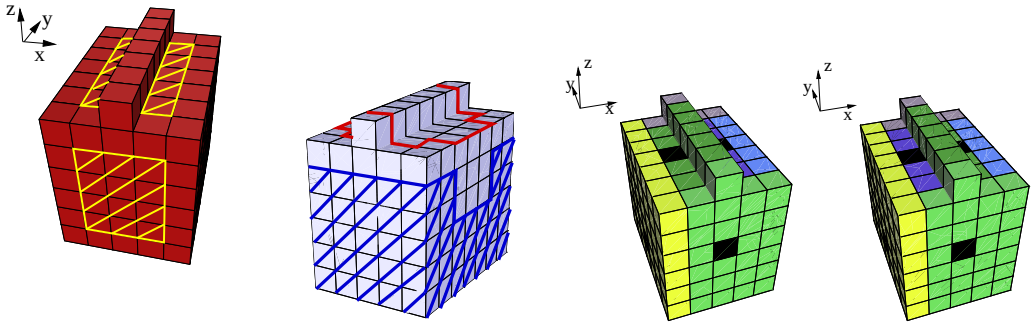
Thus, the lengths $l_x(v\text{-gadget})$ and $l_z(v\text{-gadget})$ are constant whereas $l_y(v\text{-gadget})$ depends on the number of literals of the variable (see below). The set of voxels $\{(x,y,z)|x = 2, 0 \leq y < l_y(v\text{-gadget}), z = 6\}$ is referred as the "bump" in the following.

From Proposition 3(b) and (f), seven pairwise incompatible sets can be defined on the surface of this object (see Figure 5(a)): one for each side on the parallelepiped (except the upper side) plus two for the upper side which is divided in two by a central "bump". The decomposition of this surface into DPS requires exactly seven DPS, obtained choosing a seed per incompatible set. Depending on the order in which the seeds are considered, many minimum decompositions exist. Nevertheless, only two minimum decompositions have an influence on the total number of DPS required to decompose the surface of the whole object. Indeed, we define the idle part of the v-gadget as the five sides of the parallelepiped different from the upper side, and the active part as the remaining voxels after the decomposition of the idle part into DPS (see Figure 5(b)). There are only two minimum decompositions of the active part, and in the global construction, v-gadgets are linked to other gadgets of the construction such that these only two different decompositions of the v-gadgets surface act upon the minimization of the total number of DPS.

These two configurations are depicted in Figure 5(c) and (d): the "bump" voxels are either labeled by the left or the right DPS. Actually, any other decomposition is neither optimal nor composed of maximal DPS. Indeed, since our algorithm (see Algorithm 1) is sequential, the "bump" cannot be half-covered by both the left and the right DPS. We set that these two decompositions respectively encode true (Figure 5-(c)) and false assignments (Figure 5-(d)) of the variable.

v-gadgets are linked to c-gadgets thanks to wires (l-gadgets) that are connected as illustrated in Figure 6. Figure 6(a) is an illustration of the incompatible sets we can define on the surface of this object: note that the five incompatible sets of the parallelepiped sides are preserved, and that one of the two upper incompatible sets is extended along the l-gadget.

The first part of these l-gadgets, described in details in Section 3.4, aims at generating a "signal" encoding the assignment of the variable. This is where we take advantage of the two decompositions defined previously (see Figure 6) and of Proposition 3(a). Indeed, using the terms of Proposition 3(a), the bump contains the point P, the right-most voxel of the l-object (circled on Figure 6(b) and (c)) stands for point Q and the incompatible set in between contains the curve $\mathcal{C}$. In the case of a true assignment, since the bump is not

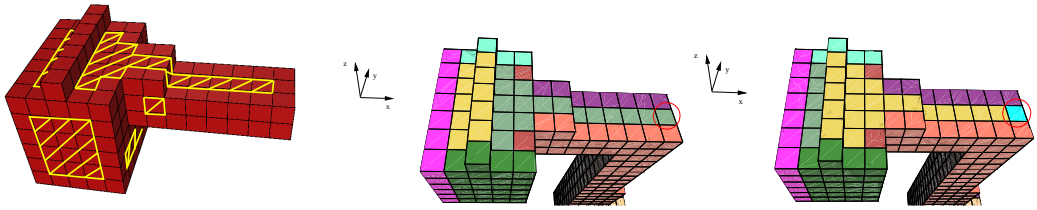


(a) (b) (c) (d)

Fig. 5. Illustration of a v-gadget: (a) general view with incompatible sets, (b) idle (dashed) and active (outlined) areas, (c) true assignment, (d) false assignment

labeled by the DPS on the right, the right-most voxel of the wire (circled on the figure) can be labeled by this DPS. On the contrary, in the case of a false assignment, with the same number of DPS, this voxel is not labeled.

To sum up, in this construction, with a fixed number of DPS, the true assignment of the variable labels on more voxel than the false assignment. This signal is then "sent" to clause objects (see Section 3.4.1 for the transmission process).



(a) (b) (c)

Fig. 6. Generation of a "signal" according to the variable assignment: (a) general view of the v- l-gadgets plugging area with incompatible sets, (b) when the variable is set to TRUE, the voxel circled in red is labeled by a DPS of the v-gadget, (c) otherwise, this voxel cannot be labeled by one DPS of the v-gadget. (Proposition 3 (a))

For a positive literal, the l-gadget is connected to the v-gadget on the right-hand side of the v-gadget, as depicted on Figure 6: the signal corresponding to the value of the variable is generated. For a negative literal, the l-gadget is connected on the left-hand side of the v-gadget: in this case, the signal corresponding to the negated value of the variable is generated (see Figure 7).

Finally, and to handle multiple instances of the same variable in a boolean expression, the length l_y(v-gadget) depends on the maximum number of positive or negative literals of a variable in the boolean expression, so that all the

connections can be made (see Figure 7). Note that the construction is such that the length of the v-gadget does not change the optimal number of DPS required for the decomposition (in particular, the side of the parallelepiped where the l-gadgets are plugged still contains only one incompatible set, and can be labeled with one DPS only).

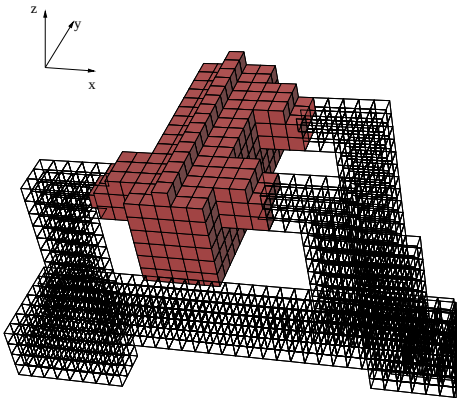


Fig. 7. Illustration of a variable with multiple instances, 2 positive literals and a negative one

3.3 Encoding Clauses: c-gadgets

A c-gadget is depicted in Figure 8. It is composed of a transversal rectangular parallelepiped of size $8 \times 21 \times 3$ on which three terminals are plugged. Since each clause has three literals (recall that 3-SAT is considered), each c-gadget has three incoming l-gadgets, which are plugged on these terminals in the global construction. Idle and active parts are depicted in Figure 8(a). Note that the idle part of the terminals is not examined here, and will be studied with l-objects (Section 3.4). Thanks to Proposition 3(f), six incompatible sets are defined on the surface of this object (see Figure 8(b)): five on the idle part, and one on the active part. The whole idle part can be labeled with five DPS (see Figure 8(c)).

The active part (see Figure 8(a)) of a clause is designed such that it can be entirely labeled by a single DPS, except one out of the three terminal extremities (see Figure 8(c)). First, note that the incompatible set included in the active part (actually defined by the whole active part without the terminal extremities) is composed of steps of length 5 or 4 along z axis and can be labeled by a DPS of parameters $(5, 0, -1, \mu)$ for instance. Next, the terminal extremities are denoted by P, Q and R following the terms of Proposition 3(d): all three voxels cannot be labeled by a single DPS whereas any couple can be entirely labeled by a single DPS. All in all, the active part plus two out of the three points P, Q and R (but not three) can be labeled by a single DPS.

The l-gadgets are plugged onto the c-gadgets terminal extremities such that the last voxels of a l-object are the terminal extremities. This plugging enables the transmission of the signal carried by l-objects (see Section 3.4). Basically, if one terminal extremity can be labeled by a DPS of the l-gadget, then only one DPS is required to cover the clause active part. Otherwise, two DPS are necessary.

In a nutshell, the link between a boolean clause and the geometric object we propose can be drawn up as follows: a boolean clause is true if and only if at least one literal is true; the active part of a c-gadget can be labeled by one DPS if and only if at least one of the three terminal extremities is labeled by a l-gadget DPS.

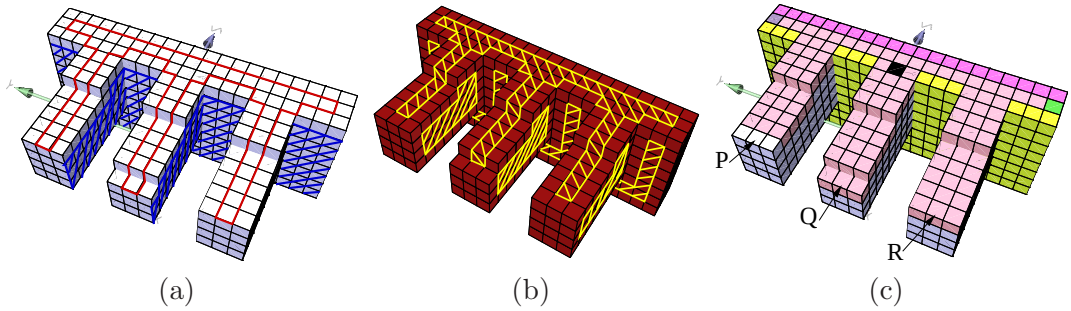


Fig. 8. Illustration of a c-gadget: (a) idle (dashed) and active (outlined) parts, (b) incompatible sets and (c) decomposition into DPS

To end with the geometrical objects encoding variables and clauses, Figure 9 illustrates how these objects are put together in the 3D space: v- and c-gadgets are lined up on two axis parallel to the y axis. The definition of l-gadgets connecting v-gadgets to c-gadgets relies on this spatial construction. In the following, we denote $dist_v$ the distance between two variables, $dist_c$ the distance between two clauses and $dist_{vc}$ the distance along the x-axis between variables and clauses These quantities are constant for the rest of the construction and do not depend on the boolean expression ϕ.

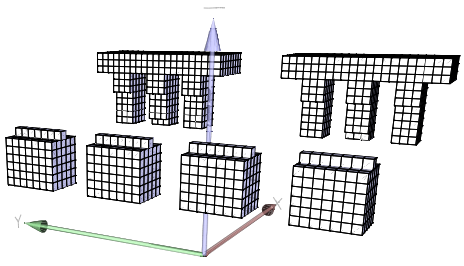


Fig. 9. Positions of four v-gadgets and two c-gadgets in the 3D space

v-gadgets are connected to c-gadgets through l-gadgets, that represent literals: if a variable appears in a clause, a l-gadget links the gadgets of this variable and this clause. These l-gadgets aim at "transmitting" the truth assignment of a variable to the clause it belongs to. Since positive and negative literals have to be considered, we define positive and negative l-gadgets. Before defining them, we describe the transmission process.

3.4.1 Transmission process

Figure 10 illustrates how the truth assignment of a variable is transmitted to a clause through a positive l-gadget. This figure represents a vertical cut of the active part of a v-gadget, a l-gadget and a c-gadget terminal. Figure 10(a) illustrates the propagation of a true value while Figure 10(b) shows how a false value is transmitted to a clause. From the construction we propose, the vertical cut of the connection of v-, l- and c-gadgets can be thought of as a 2D digital curve that we decompose into digital straight segments, using their properties.

We call "transmission voxels" the two voxels named A and D in Figure 10, intermediate transmission voxels are named B and C. We consider an optimal decomposition of the surface into DPS. The transmission voxel A actually corresponds to the generation of the signal encoding the truth assignement of the variable (see Figure 6 for a 3D representation of a v-gadget and a "plateau"). Using Proposition 3(a) and (c), if A is labeled by a v-gadget DPS, then D (which is at the same time an extremity of a clause terminal) is labeled by a l-gadget DPS. On the contrary, if A is not labeled by a v-gadget DPS, then D is not labeled by a l-gadget DPS. Note that for the plateau, descent and ascent parts, the relative length of the steps are the key point of this transmission process: for instance, a single DPS cannot cover both A and B (Proposition 3(a)).

Since the voxel A is labeled by a v-gadget DPS if and only if the literal value is "true" (see Section 3.2), the c-gadget terminal extremity, *i.e* the voxel D, is labeled by a l-gadget DPS if and only if the literal value is "true".

3.4.2 Geometric construction

Following the spatial arrangement of v- and c-gadgets (see Figure 9), and the rules defined for the connections of negative and positive literals (Section 3.2), positive l-gadgets are plugged on the v-gadgets side closest to c-gadgets, while negative l-gadgets are plugged on the opposite side. We see that in the case

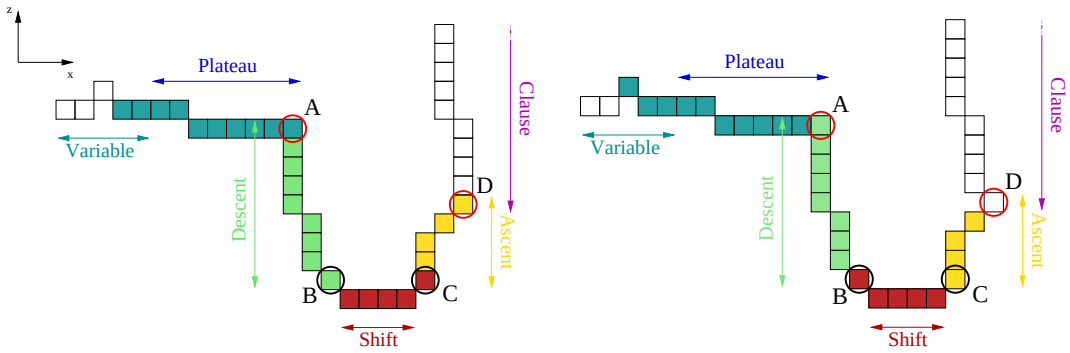


Fig. 10. Vertical cut illustration (active part only) of the transmission of a truth assignment through a l-gadget: (a) the value "true" is transmitted; (b) the value "false" is transmitted

of a negative l-gadget, a U-turn towards c-gadgets is required.

In order to ensure that the "signals" are correctly sent from v-gadgets to c-gadgets, the l-gadgets must not intersect each other. To do so, and as depicted in 2D in Figure 10, l-gadgets are basically composed of four parts, that are depicted in 3D in Figure 11(a) for a positive l-gadget:

- a plateau generates the "signal" corresponding to the truth assignement of the variable;
- a descent to a given level L: two distinct literals descend on two different levels to ensure an intersection free construction (note that the number of different levels is exactly the number of positive and negative l-gadgets);
- a shift movement on the level L to reach the c-gadget position;
- an ascent from the level L to the c-gadget terminal extremity.

First, incompatible sets can be defined on the surface as depicted in Figure 11(b) for a positive l-object and in Figure 11(e) for a negative one. These incompatible sets are defined using the configurations of Proposition 3:

- (f) is used to define the incompatible sets of the idle part;
- (a) is used to define the incompatible sets between A and B, and C and D respectively;
- (d) enables to define the incompatible set between B and B' for negative l-gadgets;
- (c) enables to define the incompatible sets between B and C for positive l-gadgets, and B' and C for negative ones.

All put together, 11 incompatible sets (eight for the idle part, three for the active part) are defined for each positive l-gadget and 17 (13 for the idle part, four for the active part) for each negative l-gadget (see Figure 11(b), (c) and (d)).

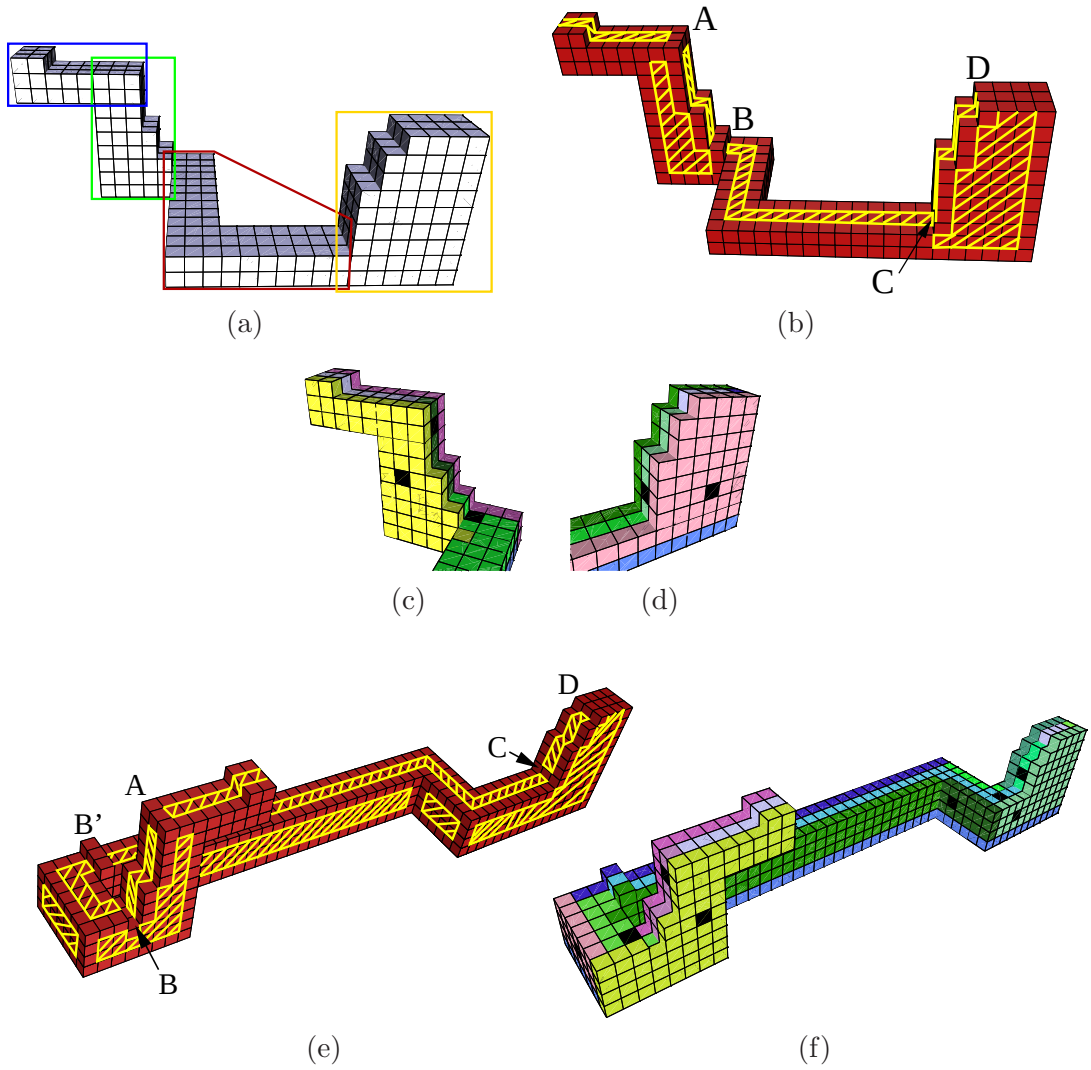


Fig. 11. l-gadget between a v- and c-gadget: (a) positive l-gadget with plateau, descent, shift and ascent part; (b) incompatible sets for a positive l-gadget; (c)-(d) (c) minimum decompositions of the descent and ascent parts; (e) incompatible sets for a negative l-gadget; (f) minimum decomposition for a negative l-object.

Next, we study one part after the other and simultaneously count the number of DPS required to decompose the surface of a l-object. We show that this number equals the number of incompatible sets.

The plateau has already been briefly presented in Section 3.2. In the total counting, its decomposition requires only one DPS for the idle part (bottom of the plateau): indeed, the sides are labeled with descent idle DPS, and the upper part (active) is labeled by a DPS coming from the v-gadget decomposition (see Section 3.2).

Concerning the descent and ascent, the active part is a three steps surface such that the first step is k voxels long ($k \geq 3$), the second one is $k-2$

voxels long and the third one is made of one voxel. Using Proposition 3(a), this configuration enables the propagation of the "signal" (see Section 3.4.1 and Figure 10, between A and B). Moreover, the parameter k is used to ensure that every l-gadget descend on a different level so that l-gadgets do not intersect. For an illustration of multiple l-gadgets in 3D, see Figure 13. The decomposition of the ascent and the descent requires (see Figure 11(c)-(d)):

- seven idle DPS: 2×3 for the sides of the descent and ascent part plus a common one for the bottom part;
- two active DPS to decompose the "steps" of the ascent and descent parts.

In the case of a positive l-gadget, the shift part is a flat surface and only one DPS is needed to cover it (the bottom part is labeled by the same DPS as for descent and ascent parts). Note that, at this point, the transmission of the "signal" is ensured thanks to Proposition 3(c) with the voxels denoted by B and C and the curve defined by the incompatible set linking these points (see Figure 11(b) and Figure 10).

The shift is trickier in order to ensure that the minimum number of DPS necessary to decompose its surface remains independent on the relative positions of the v- and c-gadgets it links. Indeed, suppose that the shift is flat as for positive l-objects. On one hand, if the c-gadget position along y axis is smaller than the position of the v-gadget, then one DPS cannot cover the voxel A and the active part of the shift. On the other hand, if the c-gadget position along y axis is greater than the position of the v-gadget, one DPS covers A and the active part of the shift. To elude this problem, we resort to two tricks: first, the level of the shift part is heightened by 2, and a "bump" pointed out by B' in Figure 11(e) is added: Proposition 3(d) used with the voxels B and B' and the incompatible set between them tells us that B and B' cannot be labeled by the same DPS on this surface. Proposition 3(c) is then used with the voxels B' and C to ensure that two DPS are required to decompose the active part of the shift, and that the transmission process still works: C is labeled by a shift DPS, if and only if B is labeled with a descent DPS. Altogether, seven DPS are required to decompose the surface of a negative l-gadget shift.

To sum up, 11 and 17 DPS are enough to decompose the surface of a positive l-object and a negative one respectively.

3.5 Summary of the Construction

To summarise the construction, we have proposed a polynomial reduction of any instance of the 3-SAT problem into an instance of the k-DSD problem. This reduction is based on the definition of v- and c-gadgets encoding variables

and clauses respectively. These objects are linked together through l-gadgets which pass the truth value of a variable on to clauses.

More precisely, for each geometrical object, we have defined incompatible sets such that we can exactly control the minimum number of DPS required to cover the overall objects. Indeed, using orientation properties of DPS (Proposition 3-(f)), we have defined incompatible sets for the so called idle part of the objects which do not interfere with the overall decomposition of the active parts. The active part is used to encode the truth assignments of the variables and to transmit them to clauses, with the help of arithmetical properties of DPS (Propositions 3-(a)-(d)). Moreover, note that the incompatible sets were defined independently for each gadget, but that they remain incompatible all together.

Finally, as illustrated in Figures 12 and 13, we have an embedding of any 3-SAT instance ϕ into a discrete object. We prove in the following section that there is a mapping between the decomposition of the object into λ digital planes and the assignment of the ϕ expression variables that satisfies ϕ.

4 NP-Completeness Proof

Let us consider a boolean 3-SAT expression ϕ, its corresponding digital object and $\mathbb{S}$ the digital surface of this object. We denote $|C|$, $|V|$, p and n the number of clauses, variables, positive literals and negative literals in ϕ respectively.

Proposition 4 *λ-DSD is in NP.*

PROOF. Given a digital surface $\mathbb{S}$ and a solution D, verifying that $|D| \leq \lambda$ and that it actually covers all the voxels of $\mathbb{S}$ is done in linear time in the number of voxels $\mathbb{S}$. Moreover, verifying that D is actually composed of DPS is also done in polynomial time since checking that a set of voxels is a DPS can be done in polynomial time [3].

Proposition 5 *The size of $\mathbb{S}$ is polynomial in the size of ϕ.*

PROOF. For a given 3-SAT expression ϕ, the discrete object we define is included in a bounded box the size of which depends on the size of ϕ. More precisely, we have the following upper bounds on the side length of the bounding box B:

- the size of B along the x axis is fixed and is equal to $l_x(c\text{-gadget}) + dist_{vc} + l_x(v\text{-gadget}) + l_x(\text{descent of negative } l\text{-gadgets})$, where $dist_{vc}$ is the distance

along x axis between v-gadgets and c-gadgets. All these values are independent of the size of ϕ. v-gadget

- the size of B along the z axis depends on the lowest level of the w-gadgets. If we number the clauses from 0 to $|C|-1$ and the literals of each clause from 0 to 2, we define the level of the literal number j in clause number i by $8+4(3*i+j)+4$. 8 is the minimum height we set for the descent part of the first l-gadget, the height increment between two successive l-gadgets is 4, and we set the vertical width of the shift part of the l-gadgets to 4. Consequently, the maximum height of l-gadgets is $12(|C|+1)$. Since l_z(v-gadget) and l_z(c-gadget) do not depend on the size of ϕ, the size of B along the z axis is linear in the number of clauses.
- the size of B along the y axis is the maximum of the two following values:
  - $|C| \times l_y(\text{c-gadget}) + (|C|-1)dist_c$, where $dist_c$ is the distance between two c-gadgets. l_y(c-gadget) and $dist_c$ do not depend on the size of ϕ, such that the sum is linear in the number of clauses $|C|$ of ϕ.
  - $|V| \times l_y(v\text{-gadget}) + (|V|-1)dist_v$, where $dist_v$ is the distance between two v-gadgets. $dist_v$ is a fixed value, independent on the size of ϕ. $l_y(v$-gadget) depends on the maximal number of occurrences of the variables. Indeed, the size of a v-gadget along y axis changes according to the number of l-gadgets that have to be plugged. However, the maximal number of occurrences of a variable is bounded by $3|C|$. Similarly, the number of variables is also bounded by $3|C|$. Thus, the sum depends on $|C|^2$.

All in all, the size of the bounded box of our construction is in $O(|C|^2)$.

We shall now prove that the construction is a reduction of 3-SAT to λ-DSD, *i.e.* that the expression ϕ is satisfiable if and only if $\mathbb{S}$ admits a decomposition with at most λ maximal DPS. We prove the two implications one after the other.

Lemma 6 *If the expression ϕ is satisfiable, then $\mathbb{S}$ admits a decomposition with λ maximal DPS.*

PROOF. Assume that ϕ is satisfiable under some truth assignment T. The following algorithm builds a decomposition of the surface of $\mathbb{S}$ into λ maximal DPS:

(1) label all the voxels belonging to an idle DPS regardless of T: $5|V|+5|C|+8p+13n$ DPS are used to cover the entire idle part of $\mathbb{S}$;
(2) decompose each v-gadget according to its truth assignment in T: these decompositions require $2|V|$ DPS;
(3) use $3p$ and $4n$ DPS to decompose the l-gadgets active parts, which may leave the tips of some l-gadgets (which are also the c-gadget terminal extremities) unlabeled;

(4) since T satisfies ϕ, every c-gadget has at least one incoming l-gadget with a labeled tip. Thus, every c-gadget has at least one labeled terminal extremity. Consequently, each c-gadget active part can be labeled with one single DPS.

All in all, $(5|V|+5|C|+8p+13n)+2|V|+3p+4n+|C| = 7|V|+6|C|+11p+17n$ DPS are used in this decomposition. In the following, we set $\lambda = 7|V|+6|C|+11p+17n$.

In order to prove the reverse implication, we need to show that there is only one way of decomposing $\mathbb{S}$ into λ DPS according to Algorithm 1. Next, we show that this unique solution leads to a satisfactory assignment of ϕ's variables.

Lemma 7 *Consider a decomposition of $\mathbb{S}$ with λ DPS using Algorithm 1. Then the decompositions of v-gadgets, positive and negative l-gadgets, and c-gadgets surfaces are respectively composed of 7, 11, 17 and 6 DPS.*

PROOF. The proof of this lemma is based on the incompatible sets defined on the surface. Indeed, since respectively 7, 11, 17 and 6 incompatible sets can be defined on the surface of v-gadgets, positive l-gadgets, negative l-gadgets and c-gadgets respectively, at least 7, 11, 17 and 6 DPS are required to decompose each gadget respectively. All in all, $7|V| + 6|C| + 11p + 17n = \lambda$ DPS are required. This means that if an extra DPS is used to decompose any gadget surface, then the total number of DPS used to decompose the surface is strictly greater than λ.

Lemma 8 *If $\mathbb{S}$ admits a decomposition into λ maximal DPS using Algorithm 1, then ϕ is satisfiable.*

PROOF. Suppose that $\mathbb{S}$ admits a decomposition D with λ DPS. Since $|D| = \lambda$, from Lemma 7 the decomposition of every v-gadget is made of seven DPS. v-gadgets can only be decomposed two ways into seven DPS, each of which encodes a truth assignment. This decomposition is made of 5 DPS for the idle part and 2 DPS for the active part (regardless of the sequential algorithm used). Thus, covering all v-gadgets requires $7|V|$ DPS. In the same way, using Lemma 7 covering l-gadgets uses $11p + 17n$ DPS. All in all, $\lambda - 7|V| - 11p - 17n = 6|C|$ DPS remain for covering c-gadgets. The idle part of c-gadgets requires 5 DPS regardless of the rest of the decomposition. Thus $|C|$ DPS remain to cover the c-gadgets active parts. Since there are $|C|$ c-gadgets, and $|C|$ remaining DPS, we know that the c-gadgets active parts are labeled by one DPS only in D. This is possible if and only if every clause in ϕ is satisfied, and thus ϕ is satisfied too.

Theorem 9 *λ-DSD is a NP-complete problem.*

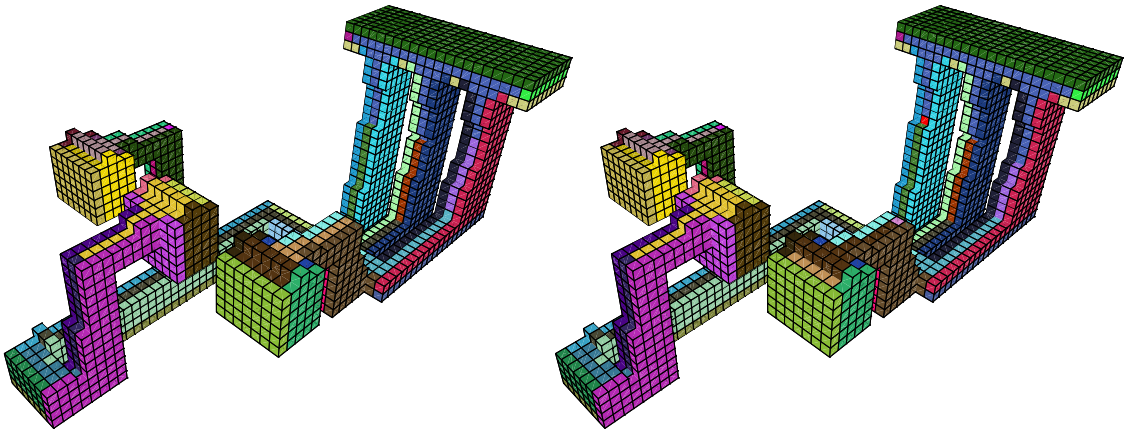
PROOF. The result is derived from Lemmas 6 and 8.

This theorem proves that the decision problem associated to Min-DSD is NP-complete. Thus, according to the theory of complexity, Min-DSD is NP-hard.

5 Example

A software that generates a 3D object from a 3-SAT boolean expression is available on `http://liris.cnrs.fr/isabelle.sivignon/code.html`. This program also generates the seeds of the object, and a simple surface decomposition algorithm into maximal DPS is provided to compute the decomposition derived from those seeds.

Figure 12 is an illustration of the digital surface encoding the expression $\phi = (a \vee \neg b \vee c)$. The optimal decomposition into maximal DPS is composed of 49 idle DPS and 17 active DPS. In Figure 12(a), the v-gadgets encode the assignment $(a = true,\ b = true,\ c = false)$, and the optimal decomposition is represented. In Figure 12(b), the v-gadgets encode the assignment $(a = false,\ b = true,\ c = false)$: in this case, since ϕ is not satisfied, the optimal decomposition cannot be achieved, and an extra DPS (in red) is added.



(a) (b)

Fig. 12. Discrete object encoding the expression $\phi = (a \vee \neg b \vee c)$: (a) optimal decomposition corresponding to the satisfaction of ϕ; (b) ϕ is not satisfied and one more DPS is required to achieve a complete decomposition

Figure 13 illustrates a more complex example : $\phi = (a \vee \neg b \vee c) \wedge (a \vee d \vee b) \wedge (\neg d \vee \neg c \vee b)$. Note that there is no intersection between the l-gadgets.

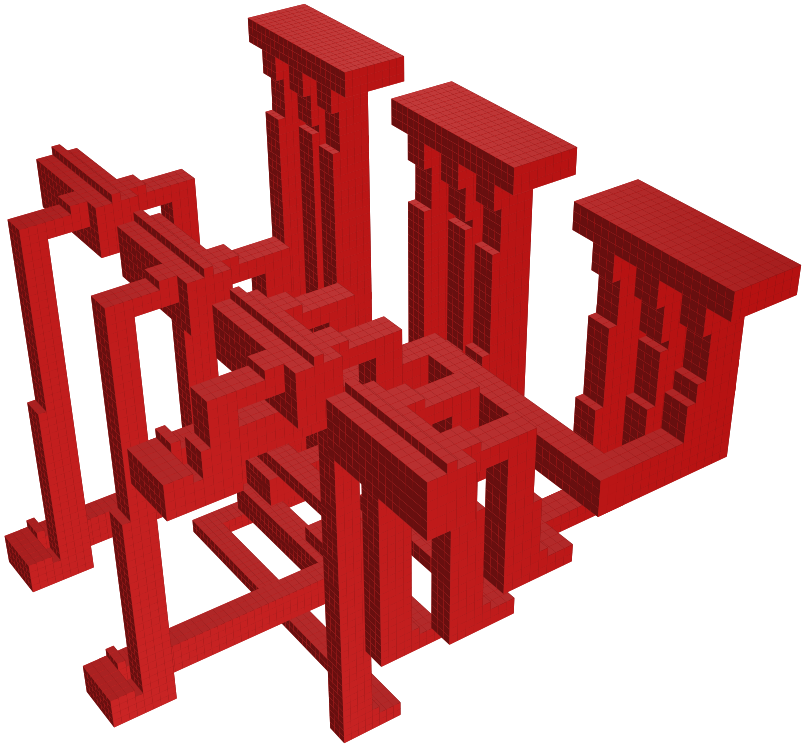


Fig. 13. A more complex example with four variables and three clauses.

6 Conclusion and Future Works

In this article, we have proved that the decomposition of a digital object surface into a minimum number of maximal DPS using a sequential algorithm is NP-complete. In our proof, we use octant orientation principles of DPS to handle object idle parts and DPS arithmetical properties for the active part of the objects. This theoretical result concludes an important open problem in the discrete geometry community: no efficient algorithms exist to solve the Min-DSD problem. A logical consequence of this answer is that only heuristics can be used.

Among possible heuristics, important theoretical future works exist: does there exist a polynomial-time approximation scheme for the Min-DSD problem ? More precisely, is there a polynomial in time approximation of Min-DSD that produces a solution that is within ϵ factor of the optimal solution ?

The theoretical result is based on a decomposition of a specific discrete object. Indeed, by construction of variables, clauses and links, the genus of the obtained binary object depends on the number of cycle in the 3-SAT instance. In applications, we usually deal with simpler objects and an important future

work concerns the following open question: Is $k-$DSD still NP-complete, and thus Min-DSD still NP-hard for simply connected objects ?

References

[1] R. Klette, A. Rosenfeld, Digital straightness - a review, Discrete Applied Mathematics 139 (1-3) (2004) 197–230.
URL http://dx.doi.org/10.1016/j.dam.2002.12.001

[2] R. Klette, A. Rosenfeld, Digital Geometry: Geometric Methods for Digital Picture Analysis, Series in Comp. Graph. and Geom. Modeling, Morgan Kaufmann, 2004.

[3] V. Brimkov, D. Coeurjolly, R. Klette, Digital planarity - a review, Discrete Applied Mathematics 155 (4) (2007) 468–495.

[4] R. Klette, H. J. Sun, Digital planar segment based polyhedrization for surface area estimation, in: IWVF, 2001, pp. 356–366.

[5] I. Sivignon, F. Dupont, J. M. Chassery, Decomposition of a three-dimensional discrete object surface into discrete plane pieces, Algorithmica 38 (1) (2003) 25–43.

[6] I. Sivignon, F. Dupont, J.-M. Chassery, Reversible polygonalization of a 3D planar discrete curve: Application on discrete surfaces, in: 12th DGCI, 2005, pp. 347–358.

[7] F. Feschet, L. Tougne, On the min dss problem of closed discrete curves, Discrete Applied Mathematics 151 (1-3) (2005) 138–153.

[8] J. Françon, L. Papier, Polyhedrization of the boundary of a voxel object, in: 8th DGCI, Vol. 1568 of LNCS, Springer-Verlag, 1999, pp. 425–434.

[9] I. Debled-Rennesson, Etude et reconnaissance des droites et plans discrets, Ph.D. thesis, Université Louis Pasteur (1995).

[10] J. E. Goodman, J. O'Rourke (Eds.), Handbook of Discrete and Computational Geometry, CRC Press, 1997.

[11] T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein, Introduction to Algorithms, 2nd Edition, MIT Press, Cambridge, MA, 2001.

[12] J.-P. Reveillès, Géométrie discrète, calcul en nombres entiers et algorithmique., Ph.D. thesis, Université Louis Pasteur - Strasbourg (1991).

[13] E. Andres, R. Acharya, C. Sibata, Discrete analytical hyperplanes, Graphical Models and Image Processing 59 (5) (1997) 302–309.

[14] V. Brimkov, Discrete volume polyhedrization: Complexity and bounds on performance, in: Computational Methodology of Objects Represented in Images: Fundamentals, Methods and Applications, Proc. of the International Symposium CompIMAGE'06, Taylor and Francis Publisher, Coimbra, Portugal, 2006.

[15] C. Worman, Decomposing polygons into diameter bounded components, in: Canadian Conference on Computational Geometry (CCCG'03), 2003, pp. 103–106.

[16] B. Chazelle, D. P. Dobkin, N. Shouraboura, A. Tal, Strategies for polyhedral surface decomposition: An experimental study, in: Symp. on Compututational Geometry, 1995, pp. 297–305.