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

Isabelle Sivignon and David Coeurjolly

Laboratoire LIRIS - Université Claude Bernard Lyon 1
Bâtiment Nautibus - 8, boulevard Niels Bohr
69622 Villeurbanne cedex, France
{isabelle.sivignon}{david.coeurjolly}@liris.cnrs.fr

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 k DPS ?) is NP-complete, and thus that the optimisation problem (finding the minimal 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.

1 Introduction

Digital objects are defined as 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 like digital lines [1] and digital planes [2, 3] originated a lot of works dealing with the decomposition of the contour of a digital object into digital linear primitives. Such a decomposition actually apprehends global geometrical properties of those objects. 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 (minimal number of parts) decomposition problem. In the 2D case, it has been shown that the minimal decomposition of a digital curve into digital line segments can be computed in linear time [4]. In the 3D case of surfaces, many decomposition algorithms have been proposed [5–8], offering comparisons on the number of faces recognised by different algorithms. Nevertheless, no optimality results exist, and no complexity study has been carried out. Related results have been recently proposed in [9] concerning the NP-completeness of the construction of an integer lattice polyhedron P with minimal number of convex facets such that $P \cap \mathbb{Z}^3$ corresponds to the input 3D digital object.

In computational geometry, the decomposition of a shape (*e.g.* a polygon) into a minimal 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 only 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, an old conjecture is that no time efficient solution exists to solve it. The remaining option is to consider approximation algorithms with or without heuristics.

Prior to a complexity study, the problem has to be formalised. In the sequel, we consider 6-connected sets of voxels which surface $\mathbb{S}$ is defined as the set of object voxels sharing a face with the background. The surface is a set of 18-connected voxels, and maximal digital naive planes [12, 13, 2, 3] are used for the decomposition. A digital plane segment (DPS for short) is maximal if no surface voxel may be added to it. In the following, we consider a sequential decomposition algorithm: given a voxel on the surface (called a *seed*), we construct the maximal digital naive plane segment adding iteratively voxels that are 18-connected to the DPS initialised with the seed. Then, a new seed is considered from the set of remaining voxels in $\mathbb{S}$. In this algorithm, both the propagation process during the DPS growing and the initialisation of seeds must be taken into account.

The optimisation problem we consider is defined as follows:

Min-DSD (Digital Surface Decomposition): Given a digital object surface $\mathbb{S}$, find the minimal decomposition of $\mathbb{S}$ into maximal digital naive plane segments using a sequential algorithm.

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

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

In this article, we prove that $k-$DSD is NP-complete whatever the propagation heuristic. Furthermore, the only requirement on the digital plane segments 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 AND, OR, NOT, variables 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 variables per clause, yielding the 3-SAT problem. An expression ϕ has the form:

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

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 of the 3-SAT problem to an instance of the k-DSD problem. The construction process, defining geometrical objects for variables, variable instances and clauses, is presented in Section 2, while the NP-completeness proof derived from this construction is given in Section 3.

2 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 steps process: after defining geometric elements for variables, instances of variables and clauses, we see how these basic components are organised and linked together in the 3D space.

2.1 General Considerations

All the basic elements we define further are composed of two parts:

- **idle part**: the surface of this part will be made of planes parallel to axis planes and only aims at defining a 6-connected object. The minimal number of DPS needed to cover the idle part will be fixed for each basic element, and any decomposition of this part into maximal DPS will exactly cover the same voxels, with no possible extension. This part is not used in the encoding of a 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 the 3-SAT elements.

For each basic element, we provide an illustration[1] of an example set of seeds (black voxels) which may be used in a decomposition algorithm. Moreover, those sets have the remarkable property that any two seeds cannot be covered by a single DPS.

The underlying basic idea for this construction is the following: the optimal decomposition chosen for a variable object generates a “signal” sent to clause objects through “wires” representing instances of variables. This kind of geometric construction of 3-SAT is a classic way to prove NP-completeness of geometric problems (see [14, 15] for instance). The construction of basic elements relies on several properties of digital planes structure that we set forth here (Figure 1):

Property 1. For the four configurations (a), (b), (b’) and (c) represented in Figure 1, one DPS cannot simultaneously cover the two light-coloured voxels and all the other ones. In the configuration represented in Figure 1(d), the three voxels cannot be covered by one DPS, but any two voxels can.

[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

Proof. The proofs of these properties are straight forward using either digital naive plane structural basic properties or their arithmetical definition [12, 6, 2].

In the following, we refer to these configurations as Property 1(a), (b), (b'), (c) and (d).

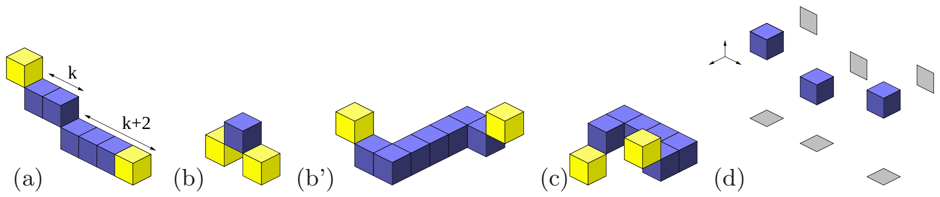


Fig. 1. Five configurations used for the reduction process

2.2 Variable and Clause Objects

An illustration of a variable element is given in Figure 2. Optimal decompositions are represented in Figure 2 (b) and (c): five DPS are used to cover the idle part of the object, and two more DPS are required to cover the remaining active part (upper part on the figure). Indeed, regardless of the idle part decomposition, the only two optimal decompositions for the active part (composed of the "bump" and the five voxels on each side of this bump) consists of two DPS, with two possible configurations (Figure 2 (b) and (c)): the "bump" voxels are either covered by the left or the right DPS (see Property 1(b)). Actually, any other decomposition is either not optimal or not composed of maximal DPS. We set that these two possibilities respectively encode true and false assignments of the variable. From these decomposition schemes and Property 1(b), seven seeds can be defined on each variable object.

Variable elements are linked to clauses thanks to wires that are connected on the area circled in red on Figure 2(a). The first part of these wires, described in details in Section 2.3, aims at generating a "signal" encoding the assignment of the variable. This signal is then "sent" to clause objects (see Section 2.3 for the transmission process). Figure 3 illustrates the signal generation: the number of voxels covered by the two active DPS differ by one according to the truth value of the variable - in the case of a true value, one more voxel is covered.

For a positive instance of variable, the wire is connected to the variable on the right-hand side of the variable, as depicted on Figure 3: the signal corresponding to the value of the variable is generated. For a negative instance of variable, the wire is connected on the left-hand side of the variable: in this case, the signal corresponding to the negated value of the variable is generated.

Finally, and to handle multiple instances of the same variable in a boolean expression, the length (along y axis) of the variable v depends on the maximum number of positive or negative instances of v in an expression, so that all the

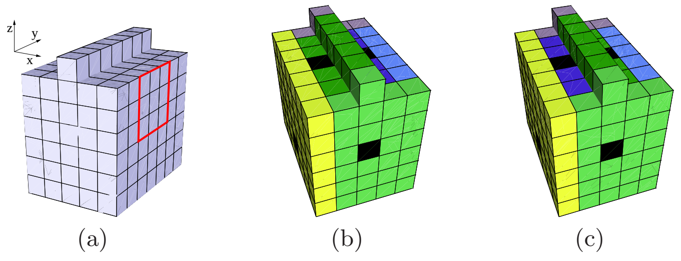


Fig. 2. Geometric discrete object encoding a variable: (a) general view with wire plugging area, (b) truth assignment, (c) false assignment

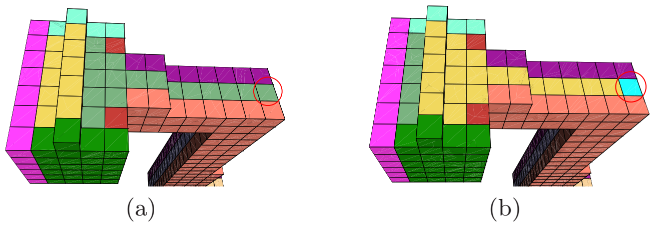


Fig. 3. Generation of a "signal" according to the variable assignment: when the variable is set to TRUE (a), the voxel circled in red is covered by a DPS of the variable, otherwise (b), this voxel cannot be covered by one of the two DPS of the variable (Property 1 (a))

connections can be made. Note that the length of the variable does not change the optimal number of DPS required for the decomposition.

A clause element is depicted in Figure 4. It is composed of a transversal rectangular parallelepiped on which three terminals are plugged. Since each clause has three literals (recall that 3-SAT is considered), each clause element has three incoming wires. The active part consists of the upper part of the object as depicted in Figure 4. Five idle DPS are required in order to decompose five out of the six faces of the rectangular parallelepiped. Idle planes related to the three terminals will be taken into account in the wire definition.

The active part of a clause can be entirely covered by a single DPS, except one out of the three terminal extremities (see Figure 4(b)): indeed, following Property 1(d), the three terminal extremities cannot be covered by a single DPS whereas any couple of terminals can be entirely covered by a single DPS. To sum up, if one terminal extremity can be covered by a DPS of the wires active part, then only one DPS is required to cover the clause active part. Otherwise, two DPS are necessary. 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; one DPS is enough to cover the whole active part of a clause if and only if one terminal extremity is covered by another DPS. From the optimal decomposition, six seeds can be defined on the surface of each clause object. The wires linking variable elements to a clause are plugged on the areas circled in red on Figure 4(a), one on each terminal extremity.

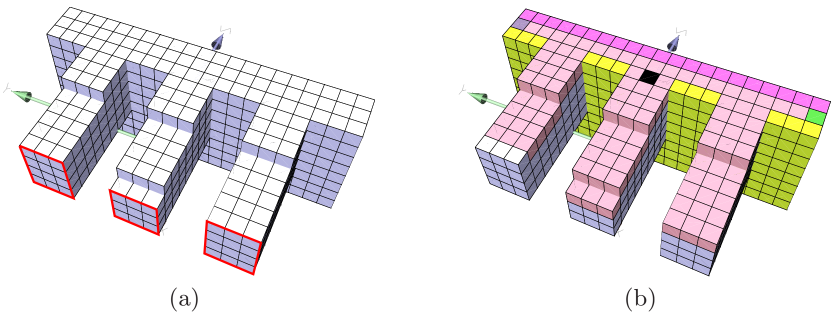


Fig. 4. Geometric discrete object encoding a clause: (a) general view, (b) optimal decomposition

To end with the variable and clause geometric objects, Figure 5 illustrates how these objects are put together in the 3D space when many objects are involved in a boolean expression: variables and clauses are lined up on two axis parallel to the y axis. The definition of wires connecting variables and clauses relies on this spatial construction.

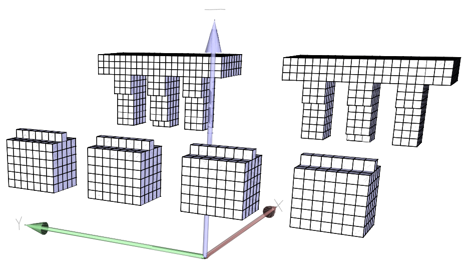


Fig. 5. Positions of (four) variables and (two) clauses objects in the 3D space

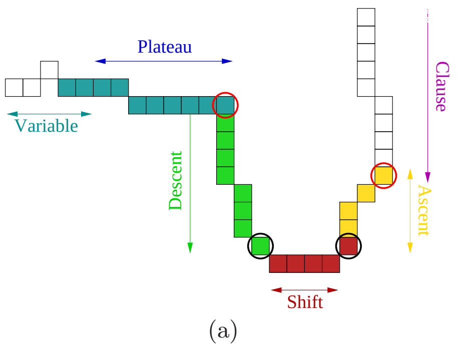
2.3 Linking Variables and Clauses

Variable elements are connected to clause elements through wires, that represent variable instances: if a variable v appears in a clause c, a wire links the geometric elements of v and c. Those wires aim at "transmitting" the truth value of a variable to the clause it belongs to. Before defining the geometric construction of wires, we describe the transmission process.

Figure 6 illustrates how the truth value of a variable is transmitted to a clause through a wire. This Figure represents a vertical cut of the active part of a variable, a wire and a clause terminal. Figure 6(a) illustrates the propagation of a true value while Figure 6(b) shows how a false value is transmitted to a clause. From the construction we propose, the vertical cut of a variable-wire-clause connection can be thought of as a 2D digital curve that we decompose in digital straight segments, using their properties.

We call "transmission voxels" the two voxels circled in red in Figure 6 (intermediate transmission voxels are circled in black). We consider an optimal decomposition of the surface into DPS. The left transmission voxel actually corresponds to the generation of the signal encoding the truth value of the variable (see Figure 3 for a 3D representation of a variable and a "plateau"). Using Property 1(a) and (b'), if the left transmission voxel is covered by a variable DPS, then the right transmission voxel (which is at the same time an extremity of a clause terminal) is covered by a wire DPS. On the contrary, if the left transmission voxel is not covered by a variable DPS, then the right transmission voxel is not covered by a wire 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 the first and last voxel of the descent.

Since the left transmission voxel is covered by a variable DPS if and only if the variable instance is set to the value "true" (Property 1(a)), the clause terminal extremity is covered by a wire DPS if and only if the variable instance is set to the value "true".



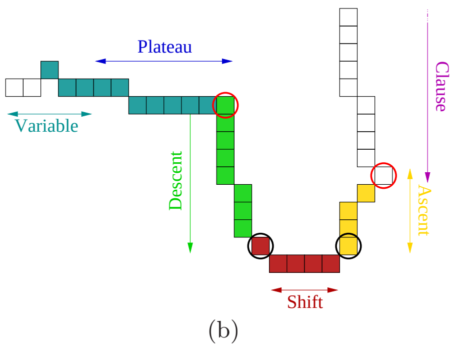


Fig. 6. Vertical cut illustration of the transmission of a truth assignment through a wire: (a) the value "true" is transmitted; (b) the value "false" is transmitted

Following the spatial arrangement of variable and clause elements (see Figure 5), and the rules defined for the connections of negative and positive instances of variables (Section 2.2), wires standing for positive instances are plugged on the variable side closest to clauses, while wires corresponding to negative instances are plugged on the opposite side. We see that in the case of a negative wire, a U-turn towards clause objects is required. As depicted in 2D in Figure 6, wires are basically composed of four parts, that are depicted in 3D in Figure 7(a) for a positive wire:

- a plateau (blue) generates the "signal" corresponding to the truth value of the variable;
- a descent (green) to a given level L: two distinct variable instances descend on two different levels to ensure an intersection free construction;
- a shift movement (red) on the level L to reach the clause position;
- an ascent (yellow) from the level L to the clause terminal extremity.

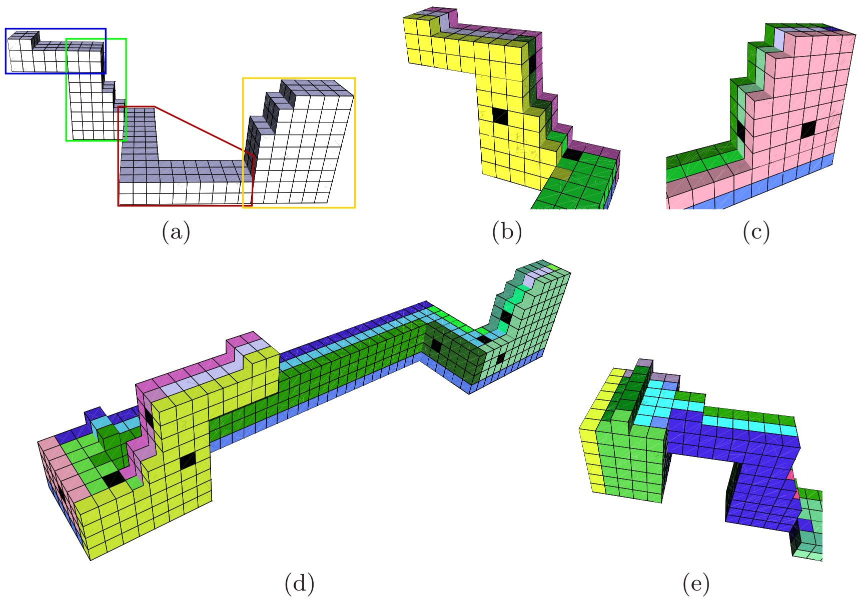


Fig. 7. Wire between a variable and a clause: (a) case of a positive variable instance, with zooms on (b) the descent and (c) ascent parts; (d) case of a negative variable instance

The optimal decomposition of the plateau is made of one DPS only, for the idle part (bottom of the plateau). Indeed, the sides are covered with descent idle DPS, and the upper part (active) is covered by a DPS coming from the variable decomposition. Figure 7(e) illustrates how a wire is connected to a variable.

Concerning the descent and ascent parts, optimal decompositions are depicted in Figure 7(b) and (c). Seven idle and one active DPS are required for both positive and negative wires. Moreover, 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. In addition to the function of this construction in the transmission process (see previous paragraph), the parameter k is used to ensure that every wire descend on a different level so that wires do not intersect.

The shift parts of positive and negative wires are different. In the case of a positive wire, only one DPS covers the active part (upper part). Five more idle planes are necessary for a negative wire, and two DPS are needed to cover the active part (see Figure 7(d)). The transmission of the truth value through the U-turn part of a negative wire is ensured thanks to the small "bump" on the shift part and Property 1(c).

All in all, optimal decomposition of wires leads to the definition of 11 seeds (eight for the idle part, three for the active part) for each positive wire and 17 (13 for the idle part, four for the active part) for each negative wire (see Figure 7(b), (c) and (d)).

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 variable and clause objects linked together through wires which pass the truth value of a variable on to clauses. Section 3 is dedicated to the proof of the relation between the two problems.

3 NP-Completeness Proof

Let us consider a boolean 3-SAT expression ϕ and the corresponding discrete object surface $\mathbb{S}$. We denote c, v, v_p and v_n the number of clauses, variables, positive instances and negatives instances of variables in ϕ respectively.

Proposition 1. *k-DSD is in NP.*

Proof. Given a digital surface $\mathbb{S}$ and solution D, verifying that $|D| \leq k$ and that it actually covers all the voxels of $\mathbb{S}$ can easily be done in linear time in the number of voxels $\mathbb{S}$. $\square$

Proposition 2. *The size of $\mathbb{S}$ is linear in the size of ϕ.*

Proof. The proof is straight forward considering the construction.

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

Lemma 1. *If the expression ϕ is satisfiable, then $\mathbb{S}$ admits a decomposition with k 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 k maximal DPS:

1. label all the voxels belonging to a construction DPS regardless of T: $5v + 5c + 8v_p + 13v_n$ DPS are used to cover the entire idle part of $\mathbb{S}$;

2. decompose each variable according to its truth assignment in T: this decompositions requires $2v$ DPS;
3. use $3v_p$ and $4v_n$ DPS to decompose the wires active parts, which may leave the tips of some wires (which are also the clause terminal extremities) uncovered;
4. since T satisfies ϕ, every clause has at least one incoming wire with a covered tip. Thus, every clause has at least one covered terminal extremity. Consequently, each clause active part can be covered with one single DPS.

All in all, $(5v+5c+8v_p+13v_n)+2v+3v_p+4v_n+c = 7v+6c+11v_p+17v_n = k$ DPS are used in this decomposition. $\square$

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

Lemma 2. *Consider a decomposition of $\mathbb{S}$ with k DPS. Then the decompositions of variable, positive wire, negative wire and clause objects are respectively covered by 7, 11, 17, 6 DPS.*

Proof. Consider a decomposition D of $\mathbb{S}$ with $|D| = k$. Suppose that there exist a variable object with a decomposition D_v such that $|D_v| > 7$. Extra DPS are either idle or active. Using more than five idle DPS has no effect on the number of DPS required to cover other variables, wires and clauses. Thus, $|D| = 7(v-1) + |D_v| + 6c + 11v_p + 17v_n > k$, which is a contradiction.

Now suppose that extra DPS are used for the active part. With these DPS, one can at best ensure that the "true" value is transmitted to every clause objects linked to this variable. Nevertheless, the number of DPS required to cover wires and clauses does not change, and we still have $|D| = 7(v-1)+|D_v|+6c+11v_p+17v_n > k$, which is a contradiction.

On the contrary, if less than seven DPS are used to cover a variable object, it is easy to check that some voxels will remain uncovered even if more DPS are used for wires or clause objects. Similar arguments can be used to show that positive and negative wires, and clause decompositions have to be composed of 11, 17 and 6 DPS. $\square$

Lemma 3. *If $\mathbb{S}$ admits a decomposition into k maximal DPS, then ϕ is satisfiable.*

Proof. Suppose that $\mathbb{S}$ admits a decomposition D with k DPS. Since $|D| = k$, from Lemma 2 the decomposition of every variable object is made of seven DPS. Variable objects 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 algorithm used). Thus, covering all variables requires $7v$ DPS. In the same way, using Lemma 2 covering wires uses $11v_p + 17v_n$ DPS. All in all, $k - 7v - 11v_p - 17v_n = 6c$ DPS remain for covering clause objects. The idle part of clause objects requires 5 DPS regardless of the rest of the decomposition. Thus c DPS remain to cover the clauses active parts. Since there are c clause objects, and c DPS remain, we know that the clause active parts are covered by one DPS only in D. This is possible if and only if every clause is satisfied, and thus ϕ is satisfied too. □

Theorem 1. *k-DSD is a NP-complete problem.*

Proof. The result is derived from Lemma 1 and 3.

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

4 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 also provided to compute the decomposition derived from those seeds.

Figure 8 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 8(a), the variable objects encode the assignment $(a = true,\ b = true,\ c = false)$, and the optimal decomposition is represented. In Figure 8(b), the variable objects 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.

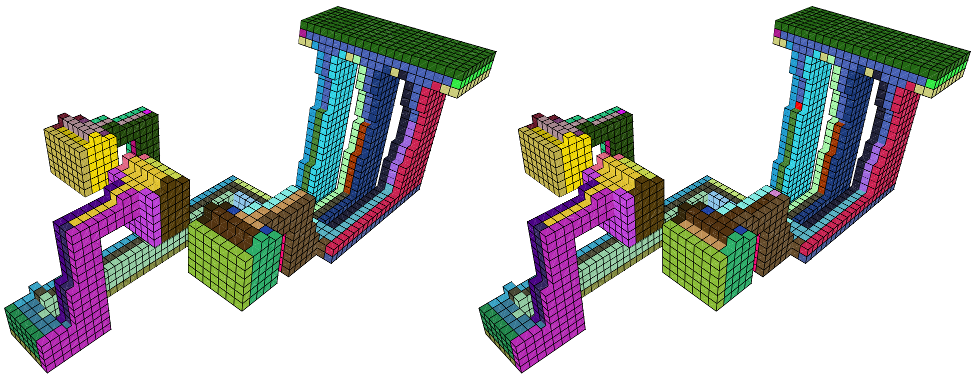

(a) (b)

Fig. 8. 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

5 Conclusion and Future Works

In this article, we have proved that the decomposition of a digital object surface into a minimal number of maximal DPS using a sequential algorithm is NP-complete. 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 ?

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. Is $k-$DSD still NP-complete, and thus Min-DSD still NP-hard for hole-free objects ?

References

1. Rosenfeld, A., Klette, R.: Digital straightness. In: Int. Workshop on Combinatorial Image Analysis. Volume 46 of Electronic Notes in Theoretical Computer Science., Elsevier Science Publishers (2001)
2. Klette, R., Rosenfeld, A.: Digital Geometry: Geometric Methods for Digital Picture Analysis. Series in Comp. Graph. and Geom. Modeling. Morgan Kaufmann (2004)
3. Brimkov, V., Coeurjolly, D., Klette, R.: Digital planarity - a review. Technical Report CITR-TR-142, CITR - Univerity of Auckland (2004)
4. Feschet, F., Tougne, L.: On the min dss problem of closed discrete curves. Discrete Applied Mathematics **151**(1-3) (2005) 138–153
5. Françon, J., Papier, L.: Polyhedrization of the boundary of a voxel object. In: 8th DGCI. Volume 1568 of LNCS., Springer-Verlag (1999) 425–434
6. Debled-Rennesson, I.: Etude et reconnaissance des droites et plans discrets. PhD thesis, Université Louis Pasteur (1995)
7. Sivignon, I., Dupont, F., Chassery, J.M.: Decomposition of a three-dimensional discrete object surface into discrete plane pieces. Algorithmica **38**(1) (2003) 25–43
8. Sivignon, I., Dupont, F., Chassery, J.M.: Reversible polygonalization of a 3D planar discrete curve: Application on discrete surfaces. In: 12th DGCI. (2005) 347–358
9. Brimkov, V.: Discrete volume polyhedrization is srongly NP-hard. Technical report, CITR-RR 179 (2006)
10. Goodman, J.E., O'Rourke, J., eds.: Handbook of Discrete and Computational Geometry. CRC Press (1997)
11. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms. 2nd edn. MIT Press, Cambridge, MA (2001)
12. Reveillès, J.P.: Géométrie discrète, calcul en nombres entiers et algorithmique. PhD thesis, Université Louis Pasteur - Strasbourg (1991)
13. Andres, E., Acharya, R., Sibata, C.: Discrete analytical hyperplanes. Graphical Models and Image Processing **59**(5) (1997) 302–309
14. Worman, C.: Decomposing polygons into diameter bounded components. In: Canadian Conference on Computational Geometry (CCCG'03). (2003) 103–106
15. Chazelle, B., Dobkin, D.P., Shouraboura, N., Tal, A.: Strategies for polyhedral surface decomposition: An experimental study. In: Symp. on Compututational Geometry. (1995) 297–305