On the complexity of Submap Isomorphism and Maximum Common Submap Problems

Christine Solnon[a,b], Guillaume Damiand[a,b], Colin de la Higuera[c], Jean-Christophe Janodet[d]

[a] *Université de Lyon, CNRS*
[b] *INSA-Lyon, LIRIS, UMR5205, F-69621, France*
[c] *LINA, UMR CNRS 6241, Université de Nantes, France*
[d] *IBISC, Université d'Evry, France*

Abstract

Generalized maps describe the subdivision of objects in cells and are widely used to model 2D and 3D images. In this context, several pattern recognition tasks involve solving submap isomorphism problems (to decide if a map is included in another map) or, more generally, computing maximum common submaps (to measure the distance between two maps). Recently, we have described a polynomial-time algorithm for solving the submap isomorphism problem when the pattern map is connected. In this paper, we show that submap isomorphism is $\mathcal{NP}$-complete when the pattern map is not connected. Then, we characterize the inherent difficulty of submap isomorphism with respect to the number of connected components. We show that it is Fixed-Parameter Tractable (FPT) and we give an FPT algorithm for submap isomorphism. We experimentally compare this algorithm with a state-of-the-art subgraph isomorphism algorithm for searching for patterns in an image and we show that it is both more accurate and more efficient. Finally, we study the complexity of the maximum common submap problem, and we show that it is $\mathcal{NP}$-hard even though we restrict the problem to the search of common connected submaps.

Keywords:

Email addresses: christine.solnon@liris.cnrs.fr (Christine Solnon), guillaume.damiand@liris.cnrs.fr (Guillaume Damiand), cdlh@univ-nantes.fr (Colin de la Higuera), Jean-Christophe.Janodet@ibisc.univ-evry.fr (Jean-Christophe Janodet)

Generalized Map, Submap Isomorphism, Maximum Common Submap, Complexity, Plane Graph

1. Motivations

To model the topology of the segmentation of an image in regions, one may use Region Adjacency Graphs (RAGs) [1, 2]: An RAG associates a node with every region and an edge with every pair of adjacent regions. However an RAG does not fully describe the topology of a partition in regions because it does not represent multi-adjacency relations nor the order of adjacent regions when turning around a given region. Thus two partitions having different topologies may be described by isomorphic RAGs, as illustrated in Fig. 1.

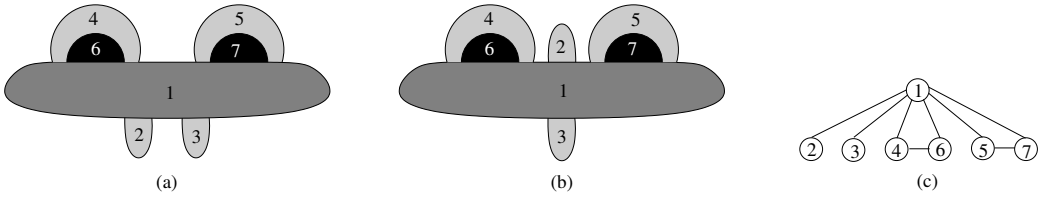


Figure 1: Segmented images (a) and (b) have different topologies but their RAGs are isomorphic to (c). (We don't represent the infinite region in the RAG.)

Dual graphs [3] extend RAGs and fully describe the topology of a segmented image by pairs $(G, \overline{G})$ of dual graphs such that each face of G corresponds to a vertex of $\overline{G}$, and vice-versa. This representation is able to encode any subdivision of the 2D topological space [4]. However, encoding 3D topological spaces with dual graphs is a difficult problem. Combinatorial maps and generalized maps [5] are more general data structures which fully model the topology of nD objects subdivided in cells (*e.g.*, 0D vertices, 1D edges, 2D faces, 3D volumes) by means of incidence and adjacency relationships between these cells. In particular, these models are very well suited for scene modeling [6], and for 2D and 3D image segmentation [7].

In [8], we have defined a basic tool for comparing combinatorial maps, *i.e.*, submap isomorphism which involves deciding if a copy of a pattern map may be found in a target map. This problem is central to many pattern recognition tasks as it may be used to decide whether a part of a segmented image is homeomorphic to another segmented image. In [8], we have proposed an efficient polynomial-time algorithm for solving this problem provided that

the pattern map is connected. This algorithm allows us to very quickly find all occurrences of a pattern, and we have illustrated its interest and feasibility for searching patterns in 2D and 3D segmented images, in the same way as any child would aim to do when he searches Wally in Martin Handford's books. However, a strong precondition of this algorithm is that the pattern map must be connected, and we have shown in [9] that submap isomorphism becomes $\mathcal{NP}$-complete when the pattern map is not connected. This is a strong impediment since combinatorial maps modeling image segmentations are not always connected: When some regions are completely enclosed in a region, the corresponding combinatorial map is not connected.

In this paper, we study further the tractability of submap isomorphism and we characterize the inherent difficulty of submap isomorphism with respect to the number of connected components. More precisely, we show that it is Fixed-Parameter Tractable (FPT) and we describe an FPT-algorithm which is exponential only in the number of connected components of the pattern map while being polynomial in the sizes of the two maps. We draw a series of experiments that aim at comparing this algorithm with a state-of-the-art subgraph isomorphism algorithm for searching for patterns in a segmented image. We show that our FPT-algorithm dramatically reduces the running time needed to find patterns and improves the relevance of retrieved patterns.

Submap isomorphism is a decision problem which cannot be used to measure the similarity of two maps as soon as there is no inclusion relation between them. In order to deal with this issue, we have introduced in [10] a distance measure which is based on the size of a largest common submap, in a similar way as a graph distance measure is defined by means of the size of a largest common subgraph in [11]. Many pattern recognition tasks such as, for example, mining, classification, or clustering, rely on the computation of distances so that it is important to have efficient algorithms for computing maximum common submaps. In order to better understand these questions, it is worth studying the theoretical complexity of this problem: If it is obviously $\mathcal{NP}$-hard in the general case[1], its theoretical complexity be-

[1]We may decide if a map M is isomorphic to a submap of another map M' by searching for the maximum common submap of M and M' and checking whether it is isomorphic to M. As submap isomorphism is $\mathcal{NP}$-complete when M is not connected, searching for a maximum common submap is $\mathcal{NP}$-hard if we do not restrict the search to connected submaps.

comes less obvious when we restrict the problem to the search of maximum common *connected* submaps. Indeed, the fact that there exists a polynomial-time algorithm for solving connected submap isomorphism lets us hope that the basic principle of this algorithm may be extended to efficiently solve the maximum common connected submap problem. In this paper, we prove that this problem is actually $\mathcal{NP}$-hard.

Contributions and outline of the paper. In Section 2 we recall definitions related to generalized maps, and in Section 3 we recall definitions and results related to computational complexity. In Section 4, we study the theoretical complexity of submap isomorphism, and we give an FPT algorithm for this problem. In Section 5, we experimentally compare this algorithm with a subgraph isomorphism algorithm for searching patterns in an image and we show that it is both more accurate and more efficient. In Section 6, we study the theoretical complexity of maximum common submap.

The new contributions of this paper with respect to [9] mainly are:

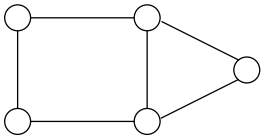
- A proof that submap isomorphism is Fixed-Parameter Tractable (FPT) and the description of an FPT-algorithm for this problem;
- An experimental evaluation of the FPT-agorithm for searching for patterns in a segmented image;
- A proof that the maximum common submap problem is $\mathcal{NP}$-hard, even when the common submap is required to be connected.

2. Recalls and basic definitions on generalized maps (nG-maps)

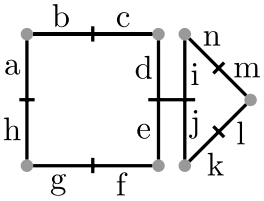
In this work we consider generalized maps, which are more general than combinatorial maps, and we refer the reader to [5] for more details on generalized maps.

Throughout this paper n is a positive integer corresponding to the dimension of the nG-maps, and N denotes the set of all integers ranging between 0 and n.

Definition 1. (nG-map) An n-dimensional generalized map (or nG-map) is defined by a tuple $M = (D, \alpha_0, \ldots, \alpha_n)$ such that (i) D is a finite set of



(a)



(c)

	a	b	c	d	e	f	g	h	i	j	k	l	m	n
α_0	h	c	b	e	d	g	f	a	j	i	l	k	n	m
α_1	b	a	d	c	f	e	h	g	n	k	j	m	l	i
α_2	a	b	c	i	j	f	g	h	d	e	k	l	m	n

(b)

Figure 2: (a) A plane graph. (b) The corresponding 2G-map. (c) Its graphical representation: Darts are represented by segments labeled with letters, consecutive darts separated with a little segment are 0-sewn (*e.g.*, $\alpha_0(b) = c$ and $\alpha_0(c) = b$), consecutive darts separated with a dot are 1-sewn (*e.g.*, $\alpha_1(a) = b$ and $\alpha_1(b) = a$), parallel adjacent darts are 2-sewn (*e.g.*, $\alpha_2(d) = i$ and $\alpha_2(i) = d$).

components called darts; (ii) $\forall i \in N$, α_i is an involution[2] on D; and (iii) $\forall i, j \in N$ such that $i + 2 \leq j$, $\alpha_i \circ \alpha_j$ is an involution.

2G-maps may be used to model plane graphs, *i.e.*, embeddings of planar graphs into planes. For example, Fig. 2 displays a plane graph and the corresponding 2G-map. We say that a dart d is i-sewn with a dart d' whenever $d = \alpha_i(d')$ and $d \neq d'$, whereas it is i-free whenever $d = \alpha_i(d)$. A seam is a tuple (d, i, d') such that d' is i-sewn to d. For example, $(a, 0, h)$ is a seam of the 2G-map displayed in Fig. 2 because $\alpha_0(a) = h$.

Definition 2. (seams of a set of darts in an nG-map) Let $M = (D, \alpha_0, \ldots, \alpha_n)$ be an nG-map and $E \subseteq D$ be a set of darts. The set of seams associated with E in M is: $seams_M(E) = \{(d, i, \alpha_i(d)) | d \in E, i \in N, \alpha_i(d) \in E, \alpha_i(d) \neq d\}$.

For example, in the 2G-map M displayed in Fig. 2, we have

$$seams_M(\{d, e, i\}) = \{(d, 0, e), (e, 0, d), (d, 2, i), (i, 2, d)\}$$

[2]An involution f on D is a bijective mapping from D to D such that $f = f^{-1}$.

Map isomorphism [5] allows us to decide the equivalence of two nG-maps.

Definition 3. (map isomorphism [5]) Two nG-maps $M = (D, \alpha_0, \ldots, \alpha_n)$ and $M' = (D', \alpha'_0, \ldots, \alpha'_n)$ are isomorphic if there exists a bijection $f : D \to D'$, such that $\forall d \in D, \forall i \in N, f(\alpha_i(d)) = \alpha'_i(f(d))$.

Induced submaps are defined in [12]: M is an induced submap of M' if M preserves all seams of M', i.e, for every couple of darts (d_1, d_2) of M, d_1 is i-sewn to d_2 in M' if and only if d_1 is i-sewn to d_2 in M.

Definition 4. (induced submap [12]) An nG-map $M' = (D', \alpha'_0, \ldots, \alpha'_n)$ is an induced submap of another nG-map $M = (D, \alpha_0, \ldots, \alpha_n)$ if $D' \subseteq D$ and $seams_{M'}(D') = seams_M(D')$.

Another submap relation is introduced in [10] and is called partial submap by analogy with existing work on graphs. Indeed, induced subgraphs are obtained by removing some nodes (and all their incident edges) whereas partial subgraphs are obtained by removing not only some nodes (and all their incident edges) but also some edges. In our nG-map context, partial submaps are obtained by removing not only some darts (and all their seams) but also some other seams.

Definition 5. (partial submap [10]) An nG-map $M' = (D', \alpha'_0, \ldots, \alpha'_n)$ is a partial submap of another nG-map $M = (D, \alpha_0, \ldots, \alpha_n)$ if $D' \subseteq D$ and $seams_{M'}(D') \subseteq seams_M(D')$.

An nG-map is connected if any pair of darts is connected with a path of sewn darts.

Definition 6. (connected nG-map) An nG-map $M = (D, \alpha_0, \ldots, \alpha_n)$ is *connected* if $\forall d, d' \in D$, there exists a sequence of darts $(d_1, \ldots, d_k)$ such that $d_1 = d$, $d_k = d'$ and $\forall i \in \{1, \ldots, k-1\}$, $\exists j_i \in N, d_{i+1} = \alpha_{j_i}(d_i)$.

A connected component of an nG-map $M = (D, \alpha_0, \ldots, \alpha_n)$ is an induced submap M' of M which is connected and maximal, *i.e.*, such that there does not exist another induced submap $M'' \neq M'$ of M which is connected and such that M' is an induced submap of M''.

2G-maps modeling segmented images are not necessarily connected, as illustrated in Fig. 3. Note that we may transform a non connected nG-map into a connected nG-map by splitting some darts belonging to different

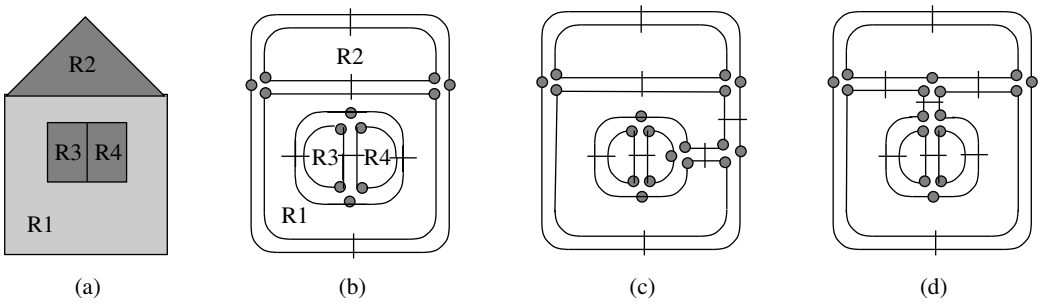


Figure 3: Example of segmented image (a) whose associated 2G-map (b) is not connected. Regions R_3 and R_4 are enclosed in region R_1 so that the 2G-map (b) contains 2 different connected components. The first one corresponds to the external boundaries of R_3 and R_4 and the internal boundary of R_1. The second one corresponds to the external boundaries of R_2 and R_1 and the internal boundary of the infinite region. (c) and (d) are two connected 2G-maps which are obtained from (b) by splitting some darts and sewing them. However, these two 2G-maps are not isomorphic.

connected components and sewing them together. However, the choice of the darts to be splitted is arbitrary so that two homeomorphic segmented images may be modeled with two non isomorphic connected nG-maps, as illustrated in Fig. 3.

3. Recalls on computational complexity

3.1. (Sub)Graph Isomorphism and Maximum Common Subgraph Problems

2G-maps may be used to model the topology of a plane graph. Hence, problems on nG-maps are at least as hard as corresponding problems on plane graphs in the sense that if we have an algorithm to solve a problem on nG-maps, then we may extend this algorithm to solve the corresponding problem on plane graphs in a straightforward way (and also on planar graphs which have unique planar embeddings such as, for example, 3-connected planar graphs).

In this section, we recall some complexity results for (sub)graph isomorphism and maximum common subgraph problems.

The complexity of graph isomorphism is still an open question. If it clearly belongs to $\mathcal{NP}$, no polynomial-time algorithm has been found for this problem, and it has not been shown to be $\mathcal{NP}$-complete. However, there exist special classes of graphs for which this problem is polynomial such as, for example, planar graphs [13], bounded valence graphs [14] and

ordered graphs [15]. For general graphs, many instances may be solved in polynomial time by exploiting local invariant properties such as node degrees [16, 17]. However, in some cases these local invariant properties are not strong enough to discriminate non isomorphic graphs.

Subgraph isomorphism is more general than graph isomorphism (in the sense that graph isomorphism may be reduced to subgraph isomorphism) and it has been shown to be $\mathcal{NP}$-complete, even in the case of planar graphs [18]. However, there exist subclasses of planar graphs for which this problem becomes polynomial such as, for example, trees [19], 2-connected outerplanar graphs [20] and outerplanar graphs under the block-and-bridge preserving constraint [21]. Moreover, several studies have aimed at circumventing the source of $\mathcal{NP}$-completeness by studying its parameterized complexity. In particular, Eppstein [22] has proposed a first FPT-algorithm for planar subgraph isomorphism with fixed size patterns. The time-complexity of this algorithm is linear in the number of vertices of the target graph, and exponential only in the number k of vertices of the pattern graph. Recently, Dorn [23] has improved the dependence on the size k of the pattern graph from $k^{\mathcal{O}(k)}$ to $2^{\mathcal{O}(k)}$.

The maximum common subgraph problem is more general than subgraph isomorphism and it is $\mathcal{NP}$-hard in the general case. However, for all classes of graphs for which subgraph isomorphism is polynomial, it is worth studying the complexity of the maximum common subgraph on this particular class of graphs. In particular, this problem can be solved in polynomial time if input graphs are trees or almost trees of bounded degree [24]. Most recently, polynomial-time algorithms have been introduced for outerplanar graphs of bounded degrees [25], and for outerplanar graphs under the block-and-bridge-preserving constraint [26].

3.2. Boolean Satisfiability

The boolean satisfiability problem (SAT) involves deciding if a boolean formula F over a set $X = \{x_1, \ldots, x_v\}$ of variables may be satisfied. Without loss of generality, we assume that F is in Conjunctive Normal Form (CNF), *i.e.*, $F = c_1 \wedge \ldots \wedge c_m$ is a conjunction of m clauses such that each clause $c_i = l_{i,1} \vee \ldots \vee l_{i,j_i}$ is a disjunction of j_i literals, and each literal $l_{i,k}$ is either a variable of X (*i.e.*, $l_{i,k} = x_j$) or the negation of a variable of X (*i.e.*, $l_{i,k} = \neg x_j$). We use the set membership notation to denote that a clause c_j occurs in F and that a variable x_i occurs in a clause c_j (either positively or

negatively). Let $\#X$ be the number of variables in X, $\#F$ the number of clauses in F, and $\#c_j$ the number of literals in a clause c_j.

SAT is an $\mathcal{NP}$-complete problem [27], and it remains $\mathcal{NP}$-complete even if every clause is composed of at most 3 literals, yielding the 3-SAT problem.

Some problems which are $\mathcal{NP}$-complete for general graphs may become polynomial when graphs are planar or plane so that it is worth studying their complexity in these special cases. To facilitate $\mathcal{NP}$-completeness proofs of problems dealing with planar graphs, Lichtenstein [28] has introduced Planar 3-SAT. Given a 3-SAT instance (X, F), he defines the graph $G_{(X,F)} = (V, E_1 \cup E_2)$ such that

- V associates a vertex with every variable $x_i \in X$ and every clause $c_j \in F$;
- E_1 associates an edge with every couple (x_i, c_j) such that c_j is a clause and x_i is a variable which occurs in c_j, *i.e.*, $E_1 = \{(x_i, c_j) \mid c_j \in F, x_i \in c_j\}$; we note $pos(x_i)$ (resp. $neg(x_i)$) the set of edges $(x_i, c_j) \in E_1$ such that x_i occurs positively (resp. negatively) in c_j, and $\#pos(x_i)$ (resp. $\#neg(x_i)$) the cardinality of $pos(x_i)$ (resp. $neg(x_i)$).
- E_2 defines a cycle over the set of vertices associated with variables, *i.e.*, $E_2 = \{(x_i, x_{i+1}) \mid i \in [1, \#X - 1]\} \cup \{(x_{\#X}, x_1)\}$.

Lichtenstein has defined Planar 3-SAT as 3-SAT restricted to instances (X, F) such that the graph $G_{(X,F)}$ is planar, and he has shown that Planar 3-SAT is $\mathcal{NP}$-complete.

In our generalized map context, we are not only interested in planar graphs, but also in plane graphs, *i.e.*, embeddings of planar graphs in planes. When the graph $G_{(X,F)}$ is embedded in a plane, edges of E_2 define a cycle which separates the plane into two parts: A first part inside the cycle, and a second part outside. We note $in(x_i)$ (resp. $out(x_i)$) the set of edges $(x_i, c_j) \in E_1$ such that c_j is inside (resp. outside) the cycle defined by E_2 edges. For example, Fig. 4 displays an instance (X, F) of Planar 3-SAT and an embedding of its associated graph $G_{(X,F)}$ in a plane. On this example, edges of x_3 which belong to the cycle are $in(x_3) = \{(x_3, c_2), (x_3, c_4)\}$.

A plane embedding of $G_{(X,F)}$ is said to be *separable* if, for every variable x_i, we have $\{pos(x_i), neg(x_i)\} = \{in(x_i), out(x_i)\}$. In other words, all edges of $pos(x_i)$ belong to the same part of the plane with respect to the cycle defined by E_2, and all edges of $neg(x_i)$ belong to the other part. For example, in the

$$
\begin{aligned}
X &= \{x_1, x_2, x_3, x_4, x_5\} \\
F &= (\neg x_1 \vee x_2 \vee \neg x_4) \\
&\wedge\ (x_1 \vee \neg x_2 \vee x_3) \\
&\wedge\ (x_2 \vee \neg x_3 \vee \neg x_4) \\
&\wedge\ (x_3 \vee x_4 \vee \neg x_5) \\
&\wedge\ (\neg x_1 \vee x_5)
\end{aligned}
$$

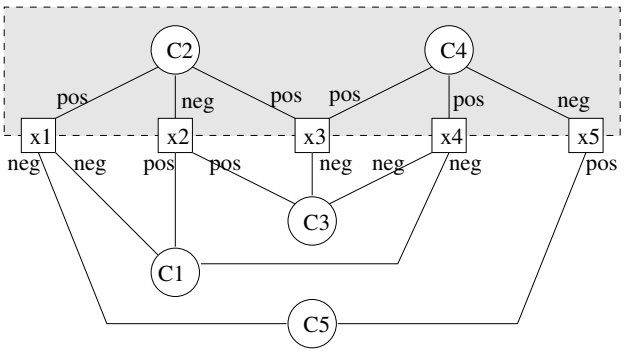


Figure 4: An instance (X, F) of Separable Planar 3-SAT and a separable embedding of its associated formula graph $G_{(X,F)}$ in a plane. Clauses correspond to circles, and variables to squares. Edges of E_2 are displayed in dotted lines and the part of the plane inside the cycle defined by E_2 edges is displayed in gray. For each variable x_i, edges of $pos(x_i)$ are labelled with *pos* and edges of $neg(x_i)$ are labelled with *neg*.

embedding displayed in Fig. 4, we have $in(x_3) = \{(x_3, c_2), (x_3, c_4)\} = pos(x_3)$ and $out(x_3) = \{(x_3, c_3)\} = neg(x_3)$, so that the separability constraint is satisfied for x_3. We can check that it is also satisfied for all other variables so that this plane embedding is separable.

Furthermore, Knuth and Raghunathan [29] have shown that any separable embedding may be drawn so that all variables are aligned from x_1 to $x_{\#X}$, and all clauses which are inside (resp. outside) the E_2 cycle are above (resp. below) the variable line, as displayed in Fig. 4.

Lemma 1 of [28] shows us that the Separable Planar 3-SAT problem defined below is $\mathcal{NP}$-complete.

Problem: Separable Planar 3-SAT
Instance: A triple (X, F, μ) where X is a set of boolean variables, F is a CNF formula over X and μ is a plane embedding of $G_{(X,F)}$ such that (i) the plane embedding μ is separable; (ii) every variable occurs in 2 or 3 clauses, and at least once positively and once negatively; and (iii) every clause contains 2 or 3 literals.
Question: Does there exist a truth assignment for X which satisfies F?

Note that the embedding μ is given as an input of the instance: Indeed, Lichtenstein [28] has proven that, given any planar 3-SAT instance (X, F), we can build in polynomial-time an equivalent SAT instance (X', F') and a separable plane embedding of $G_{(X',F')}$. Furthermore, the instance (X', F') satisfies constraints (ii) and (iii) of the definition of the problem.

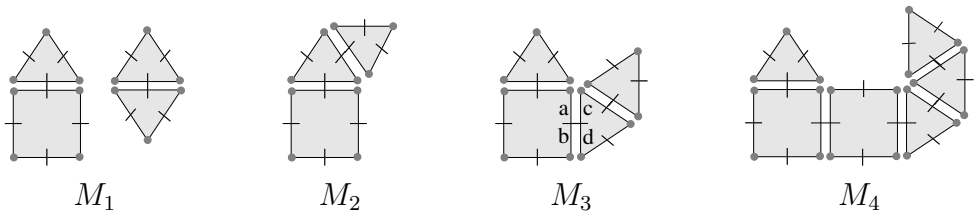


Figure 5: Submap isomorphism examples. M_1 is not isomorphic to a submap of M_2 (i.e., $M_1 \not\sqsubseteq^p M_2$ and $M_1 \not\sqsubseteq^i M_2$), though each connected component of M_1 is a submap of M_2. M_1 is isomorphic to a partial submap of M_3, but not to an induced one (i.e., $M_1 \sqsubseteq^p M_3$ and $M_1 \not\sqsubseteq^i M_3$), because the seams $(a, 2, c)$ and $(b, 2, d)$ of M_3 are not preserved in M_1. M_1 is isomorphic to an induced submap of M_4 and, therefore, it is also isomorphic to a partial submap of M_4 (i.e., $M_1 \sqsubseteq^p M_4$ and $M_1 \sqsubseteq^i M_4$).

4. Submap isomorphism

Submap isomorphism involves deciding if a pattern nG-map M is isomorphic to a submap of a target nG-map M', and it is formally defined below.

Problem: Partial (resp. induced) submap isomorphism
Instance: A couple (M, M') such that M and M' are nG-maps.
Question: Does there exist a partial (resp. induced) submap of M' which is isomorphic to M?

We note $M \sqsubseteq^p M'$ (resp. $M \sqsubseteq^i M'$) when the answer is yes. Note that $M \sqsubseteq^i M' \Rightarrow M \sqsubseteq^p M'$. Fig. 5 displays examples of submap isomorphisms.

The complexity of submap isomorphism depends on the connectedness of the pattern nG-map. For example, the nG-map M_1 of Fig. 5 is not connected, and is composed of two connected components, whereas the nG-maps M_2, M_3 and M_4 are connected. When the pattern nG-map M is connected, we can decide of submap isomorphism in polynomial time by using the algorithm described in [8]. When the pattern nG-map M is not connected, we may use this algorithm to search for all occurrences of each connected component of M in the target nG-map M'. Let us consider, for example, the nG-map M_1 of Fig. 5. Its left hand side component occurs once in M_2 and twice in M_3 and M_4, and its right hand side component occurs once in M_2 and M_3 and twice in M_4. To solve the submap isomorphism problem from these occurrence lists, we have to solve the following combinatorial problem: Can we select one occurrence in M' of each connected component of M so that the selected occurrences do not overlap in M'?

In Section 4.1, we prove that submap isomorphism becomes $\mathcal{NP}$-complete

when the pattern nG-map M is not required to be connected. In Section 4.2, we give an FPT-algorithm which proves that the tractability of this problem actually depends on the number of connected components in M.

4.1. Submap Isomorphism is $\mathcal{NP}$-complete

Theorem 1 claims that submap isomorphism is $\mathcal{NP}$-complete in the general case, when the pattern nG-map M is not required to be connected.

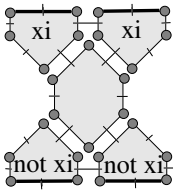
Theorem 1. *Partial (resp. induced) submap isomorphism is $\mathcal{NP}$-complete.*

The problem trivially belongs to $\mathcal{NP}$ since one can check that a given partial (resp. induced) submap of the target nG-map M' is isomorphic to the pattern nG-map M in polynomial time. We may use for example the polynomial time algorithm for map isomorphism of [8], which has been defined for non connected nG-maps.

To prove that the problem is $\mathcal{NP}$-complete, we show that Separable Planar 3-SAT may be reduced to it in polynomial time[3]. We first consider the induced case; the extension of the proof to the partial case is discussed at the end of this section. We consider an instance (X, F, μ) of Separable Planar 3-SAT and we show how to build an instance (M, M') of submap isomorphism such that $M \sqsubseteq^i M'$ iff the answer to (X, F, μ) is yes. We consider 2G-maps, so that $n = 2$.

Definition of the target 2G-map M'. Let us first describe the building blocks which compose M':

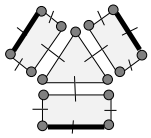
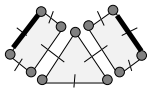
- For each variable x_i, M' contains a gadget which is composed of a central 12-dart face surrounded with four 10-dart faces such that two of these 10-dart faces are labelled with x_i and are adjacent, and the two other ones are labelled with $\neg x_i$ and are adjacent, as displayed below:



[3]We have given a different proof in [9], based on a reduction of Planar-4 3-SAT. The new proof is a bit simpler.

Darts displayed in bold are connecting darts which will be 2-sewn to define a connected 2G-map.

- For each clause c_j, M' contains a gadget which is composed of a central 6-dart face surrounded with $\#c_j$ 8-dart faces as displayed below:

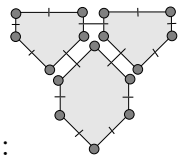
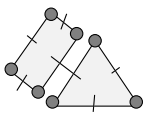
If $\#c_j = 3$:  If $\#c_j = 2$: 

Each 8-dart face is labelled with a literal of c_j, and the order of these labels when turning around the central 6-dart face is the same as the one in the embedding μ. Again, darts displayed in bold are connecting darts which will be 2-sewn to define a connected 2G-map.

These building blocks are positionned in the plane like in the separable embedding μ of $G_{(X,F)}$ in such a way that, for each variable x_i, the 10-dart faces labelled with x_i (resp. $\neg x_i$) are oriented towards the part of the cycle defined by E_2 edges which contains $pos(x_i)$ (resp. $neg(x_i)$) edges.

Then, these building blocks are 2-sewn in order to define a connected 2G-map: For each clause c_j, each connecting dart of the associated clause gadget is 2-sewn with a connecting dart of a 10-dart face which is labelled with a literal of c_j, according to the separable embedding μ of $G_{(X,F)}$. Note that in separable instances every variable occurs in 2 or 3 clauses, and at least once positively and once negatively, so no variable occurs more than twice positively or more than twice negatively. This ensures us that variable gadgets have enough 10-dart faces to connect clause gadgets with variable gadgets. Fig. 6 displays the 2G-map associated with the formula displayed in Fig. 4. This 2G-map contains only one connected component as the different gadgets which compose it have been 2-sewn.

Definition of the pattern 2G-map M. M is composed of $\#X$ occurrences of the variable gadget and $\#F$ occurrences of the clause gadget, where variable and clause gadgets are displayed below:

Variable gadget:  Clause gadget: 

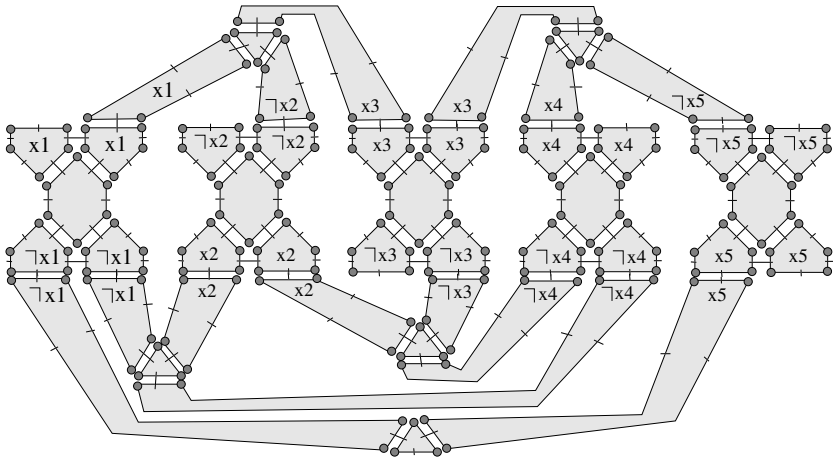


Figure 6: 2G-map M' associated with the SAT instance displayed in Fig. 4. Note that this 2G-map contains holes (corresponding to white parts in the figure): each dart d adjacent to these holes is 2-free so that $\alpha_2(d) = d$.

For example, the 2G-map M associated with the formula displayed in Fig. 4 contains 10 connected components: 5 occurrences of the variable gadget and 5 occurrences of the clause gadget.

Proof of $(M \sqsubseteq^i M') \Rightarrow (\exists$ *truth assignment of* X *which satisfies* $F)$. Let us first assume that there exists an induced submap M'' of M' which is isomorphic to M, and let us show that there exists a truth assignment of X which satisfies F.

If M'' is isomorphic to M then, according to Def. 3, there exists a bijection f which matches darts of M'' with darts of M and which preserves all seams. By extension, we say that f matches faces of M'' with faces of M. As we consider induced submap isomorphism, two faces of M which belong to two different connected components cannot be matched by f with faces which are 2-sewn in M'' (according to Def. 3). Fig. 7 displays an example of such a solution for the instance (M, M') of the induced submap isomorphism problem associated with the instance (X, F) displayed in Fig. 4.

M contains $\#F$ clause gadgets, and each of these clause gadgets has a 6-dart face adjacent to a 8-dart face. These faces can only be matched with faces which belong to clause gadgets in M' as 6-dart faces in M' only come from clause gadgets. As there are $\#F$ clause gadgets in M, each clause gadget in M' is matched with a different clause gadget in M. For the same reasons, each variable gadget in M is matched with a different variable gadget

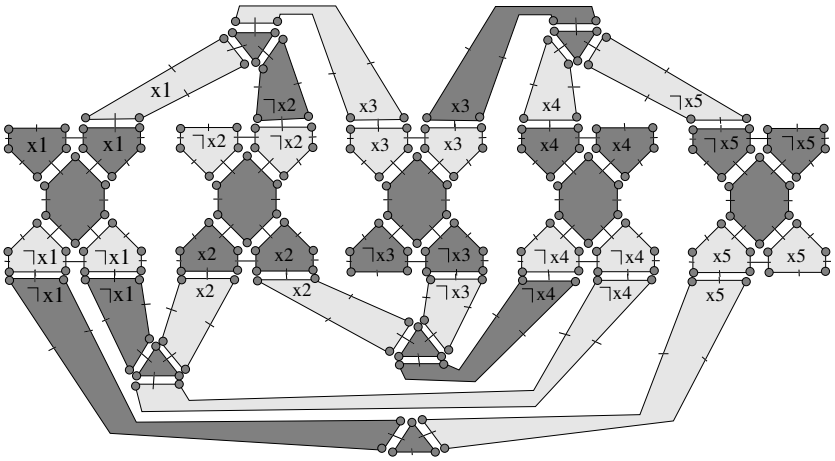


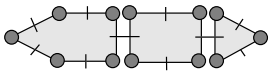
Figure 7: Solution of the induced submap isomorphism instance (M, M') associated with the Separable Planar 3-SAT instance displayed in Fig. 4. The induced submap of M' which is isomorphic to M is displayed in dark gray. Note that two different components of this submap cannot be 2-sewn in M' as we consider induced submap isomorphism.

in M'. For each variable gadget, the label of 10-dart faces in M' which are not matched with 10-dart faces of M gives the truth assignment for the corresponding variable. For each clause gadget, the label of the 8-dart face of M' which is matched with a 8-dart face of M corresponds to a literal which satisfies the clause. As two faces of M which belong to two different connected components cannot be matched by f with faces which are 2-sewn in M'', we ensure that when a 8-dart face of a clause gadget is matched, then its 2-sewn 10-dart face is not matched, *i.e.*, when a clause is satisfied by a literal, then no other clause can be satisfied by the negation of this literal so that the truth assignment deduced from the variable gadget matching actually satisfies all clauses of F. For example, the truth assignment corresponding to the solution displayed in Fig. 7 is $\{\neg x_1, \neg x_2, x_3, \neg x_4, x_5\}$.

Proof of ($\exists$ *truth assignment of* X *which satisfies* F) $\Rightarrow$ ($M \sqsubseteq^i M'$). Let us assume that there exists a truth assignment of X which satisfies F and let us show that there exists an induced submap M'' of M' which is isomorphic to M, i.e., that there exists a dart matching which preserves all seams of M. For each variable gadget in M associated with a variable x_i, we match the darts of the 12-dart face with the darts of the 12-dart face of the variable gadget associated with x_i in M' and we match the darts of the two 10-dart faces with the darts of the two 10-dart faces which are labeled with the negation of the

truth value of x_i. For each clause gadget in M associated with a clause c_j, we match the darts of the 6-dart face with the darts of the 6-dart face of the clause gadget associated with c_j in M' and we match the darts of the 8-dart face with the darts of one of the three 8-dart faces: We choose an 8-dart face which is labeled with a literal which is satisfied by the truth assignment (this 8-dart face cannot be 2-sewn with a matched 10-dart face).

Proof for the partial case. Let us now consider the partial case: We consider an instance (X, F, μ) of Separable Planar 3-SAT and we show how to build an instance (M, M') such that $M \sqsubseteq^p M'$ iff the answer to (X, F, μ) is yes. The proof is similar to the induced case. The difference between the induced and the partial cases is that, when considering induced submap isomorphism, two faces which belong to two different components in M cannot be matched with faces of M' which are 2-sewn whereas, when considering partial submap isomorphism, two faces which belong to two different components in M may be matched with faces of M' which are 2-sewn. Therefore, we modify the clause gadget C so that the 8-dart face is adjacent to a 6-dart face, on one side, and to a 10-dart face on the opposite side, as displayed below:



These 10-dart faces can only be matched with 10-dart faces of M'.The label of the matched 10-dart face corresponds to the literal which satisfies the clause. Fig. 8 displays an example of solution for partial submap isomorphism.

4.2. FPT-algorithm for Submap Isomorphism

The fact that submap isomorphism is $\mathcal{NP}$-complete implies that there does not exist an algorithm which can solve all instances of this problem in polynomial time, unless $\mathcal{P} = \mathcal{NP}$. However, we show in this section that the practical tractability of instances of this problem actually depends on the number of different connected components of the pattern nG-map M.

Let us first define the parameterized submap isomorphism problem:

Problem: Parameterized partial (resp. induced) submap isomorphism
Instance: A couple (M, M') such that M and M' are nG-maps.
Question: Does there exist a partial (resp. induced) submap of M' which is isomorphic to M?
Parameter: The number k of connected components in M

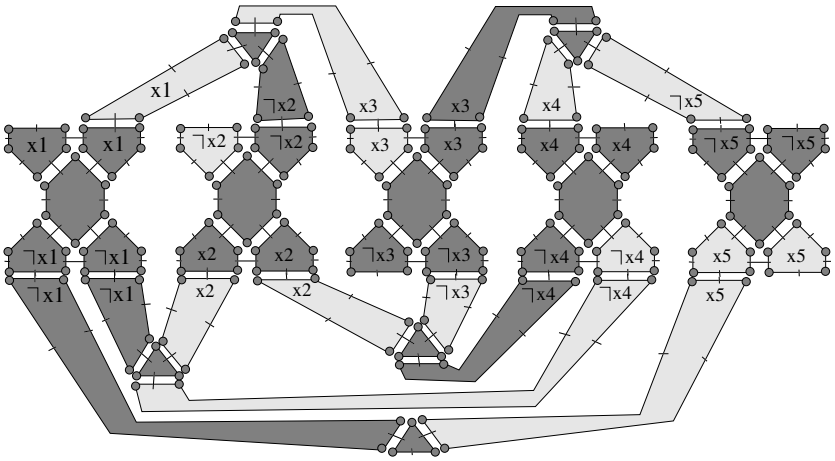


Figure 8: Solution of the partial submap isomorphism instance (M, M') associated with the Separable Planar 3-SAT instance displayed in Fig. 4. The partial submap of M' which is isomorphic to M is displayed in dark gray.

Theorem 2 shows us that this problem may be solved in polynomial time when the number k of connected components in the pattern nG-map is bounded by a fixed value.

Theorem 2. *Parameterized partial (resp. induced) submap isomorphism is fixed-parameter tractable.*

To prove this theorem, we describe in algorithm 1 a procedure for deciding if an nG-map M composed of k connected components is isomorphic to a submap of M', and we show that the time complexity of this procedure is exponential only in k while it is polynomial in n and in the number of darts of M and M'. We begin to consider the partial case; the proof is extended to the induced case at the end of this section.

Correctness of algorithm 1. Let us first show that Algorithm 1 returns true iff M is isomorphic to a partial submap of M'. It returns true iff the graph G has a clique of size k, *i.e.*, a subset $S \subseteq V$ such that $\forall((M'_x, i), (M'_y, j)) \in S \times S, (M'_x, i) \neq (M'_y, j) \Rightarrow ((M'_x, i), (M'_y, j)) \in E$. All vertices in V correspond to submaps of M' which are isomorphic to connected components of M, and two different vertices (M'_x, i) and (M'_y, j) are adjacent in G iff the submaps M'_x and M'_y of M' do not share darts and are isomorphic to two different connected components of M. Therefore, a clique of size k corresponds to k

Algorithm 1: solvePartialSubIsomorphism(M, M')

Input: two nG-maps M and M' such that M is composed of k connected components

Output: returns true iff M is isomorphic to a partial submap of M'

```
1  Decompose M into its k connected components denoted M_1, ..., M_k
2  Let V and E be two empty sets
3  for each connected component M_i of M do
4      for each partial submap M'_x of M' which is isomorphic to M_i do
5          add (M'_x, i) to V
6  for each (M'_x, i) ∈ V do
7      for each (M'_y, j) ∈ V such that i ≠ j do
8          Let D'_x and D'_y be the darts of M'_x and M'_y, respectively
9          if D'_x ∩ D'_y = ∅ then add ((M'_x, i), (M'_y, j)) to E;
10 if the graph G = (V, E) has a clique of size k then
11     return true
12 else
13     return false
```

vertices $(M'_1, i_1), \ldots, (M'_k, i_k)$ such that each M'_i is isomorphic to a different connected component M_i and all M'_i have different darts.

Complexity of algorithm 1. Let us note d, d', and d_i the number of darts of M, M', and M_i, respectively. As k is a fixed parameter, we consider here that it is a constant independent from the size of the instance which only depends on n, d and d'.

Line 1 may be done by a simple traversal of M in $\mathcal{O}(nd)$.

Lines 3 to 5 may be done in $\mathcal{O}(ndd')$ as $d_1 + \ldots + d_k = d$, and the complexity of searching for all partial submaps of M' which are isomorphic to M_i is $\mathcal{O}(nd_id')$ when using algorithm 3 of [8]: We modify line 5 of algorithm 3 of [8] by storing the initial dart of M' (instead of returning *true*) each time we find a subisomorphism function. Note that there are at most d' partial submaps of M' which are isomorphic to each connected component M_i (one for each initial dart of M'; see [8] for more details).

Lines 6 to 9 may be done in $\mathcal{O}(dd'^2)$. Indeed, the set V has $\mathcal{O}(d')$ elements, and submaps M'_x and M'_y have at most d darts. It is possible to check whether

$D'_x \cap D'_y$ is empty in $\mathcal{O}(d)$ by marking all the darts of D'_x after line 6, and verifying whether some dart of D'_y is marked.

Lines 10 to 13 may be done in $\mathcal{O}(d'^k)$ as the graph G has $\mathcal{O}(d')$ vertices and there are $\mathcal{O}(d'^k)$ subgraphs of G to check, and each subgraph has $\mathcal{O}(k^2)$ edges whose presence in G needs to be checked.

Thus, the overall time complexity of algorithm 1 is $\mathcal{O}(d'^k + ndd' + dd'^2)$.

Algorithm for the induced case. The algorithm for the induced case may be derived from algorithm 1 by searching for induced submaps instead of partial submaps (line 4), and by modifying the way the set of edges E is built (lines 6 to 9). Indeed, when considering induced submap isomorphism, two occurrences of two different connected components must not share darts and must not have darts which are sewn in M' but not in M. To ensure this, we simply have to mark all darts of D'_x as well as all darts which are sewn with D'_x (between lines 6 and 7).

5. Experimental evaluation for searching for patterns in images

In [8], we have introduced a polynomial-time algorithm for solving submap isomorphism when the pattern nG-map is connected, and we have shown how to use this algorithm for searching for a pattern into a database of images. However, in these experiments, we have restricted our attention to patterns which are modelled by connected nG-maps as the algorithm of [8] is defined for connected submaps only. In this section, we report a similar experiment and show how to use the FPT algorithm introduced in Section 4.2 to search for subimages which are modelled by non connected 2G-maps.

We have considered the segmented image displayed in Fig. 9, and we have used the algorithm of [30] to build a 2G-map from the segmented image. The basic idea of this algorithm is to build the 2G-map in an incremental way: Each pixel is considered iteratively and, for each pixel, a square face is created and sewn to the 2G-map. Then this square face is merged with adjacent faces that belong to the same region. Finally, consecutive edges separating the same two regions are merged so that two adjacent regions are separated by exactly one edge (see [30] for more details).

We have extracted 10 subimages from the segmented image. Each subimage is a connected rectangular subset of pixels. For each subimage, we have used the algorithm of [30] to build a 2G-map from the subimage. Then we have removed from this 2G-map every face corresponding to a region of the

Figure 9: Segmented image

subimage which is adjacent to the exterior of the subimage (to remove faces corresponding to regions which are not completely enclosed in the subimage). Table 1 describes these 2G-maps. These 2G-maps are not connected because some regions of the segmentation are completely enclosed in other regions, as illustrated in Fig. 3 (page 7).

We compare our FPT-algorithm with LAD, a state-of-the-art algorithm for solving subgraph isomorphism problems [31]. Given a 2G-map M, we build a multigraph $G = (V, E)$ such that V associates a vertex with every 0-cell of M and E associates an edge with every 1-cell of M. G is a multigraph because when a face has only two vertices i and j in its boundary, the edge (i, j) occurs more than once. For example, let us consider the 2G-map displayed in Fig. 10(b). Face 1 has only two vertices in its boundary so that the multiplicity of edge (a, g) in the graph displayed in Fig. 10(c) is equal to two.

To search for patterns in graphs, we consider the partial subgraph isomorphism problem. Indeed, the graph associated with a pattern 2G-map is not necessarily an induced subgraph of the graph associated with a target 2G-map, as illustrated in Fig. 10. Looking for partial subgraph isomor-

	t	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10
cc	79	2	2	3	3	4	4	5	5	6	6
D	30184	2240	3100	868	712	196	1858	1804	1856	758	3644
V	4978	229	298	108	82	31	366	199	191	170	720
E	7546	337	437	150	113	39	541	284	280	242	1060
F	2726	110	142	45	34	12	179	90	95	79	346

Table 1: Description of 2G-maps extracted from the image displayed in Fig. 9 (column t) and from 10 subimages of this image (columns p1 to p10). Each column successively displays the number of connected components (cc), darts (D), vertices (V), edges (E) and faces (F).

phisms allows us to find all submap isomorphisms. However, some subgraph isomorphisms may not correspond to submap isomorphisms as topological relationships are ignored. For example, the two graphs displayed in Fig. 10 (e) and (g) are isomorphic whereas the two 2G-maps from which they have been built (displayed in Fig. 10 (d) and (f)) are not isomorphic.

We may search for partial subgraph isomorphisms without decomposing the pattern graph into connected components. However, when a same connected component occurs several times in the target graph, it may be more efficient to proceed in a similar way as what is done in Algo. 1, i.e., (i) decompose the pattern graph into its k connected components, (ii) search for all occurrences of each connected component in the target graph, (iii) build a graph which associates a vertex with every component occurrence and an edge with every pair of component occurrences which do not share edges, and (iv) search for all cliques of k vertices in this graph. Note that this algorithm is not FPT with respect to the number of connected components as searching for all occurrences of a connected component is an $\mathcal{NP}$-complete problem in the case of graphs, whereas it is polynomial in the case of nG-maps.

Table 2 compares our FPT submap algorithm with these two different subgraph isomorphism approaches. All algorithms have been implemented in C or C++. All our experiments were made on Intel(R) Core(TM) i7 CPU 930 2.80GHz. For each subimage, we have searched for all occurrences of this subimage in the target image.

Comparison of CPU times. We report the time performance (in seconds) in Table 2. Our FPT submap algorithm is very effective: It is able to find all occurrences of a pattern in .27 second on average for the 10 patterns. For all patterns but one (p7), most of the time is spent to find all occurrences of all

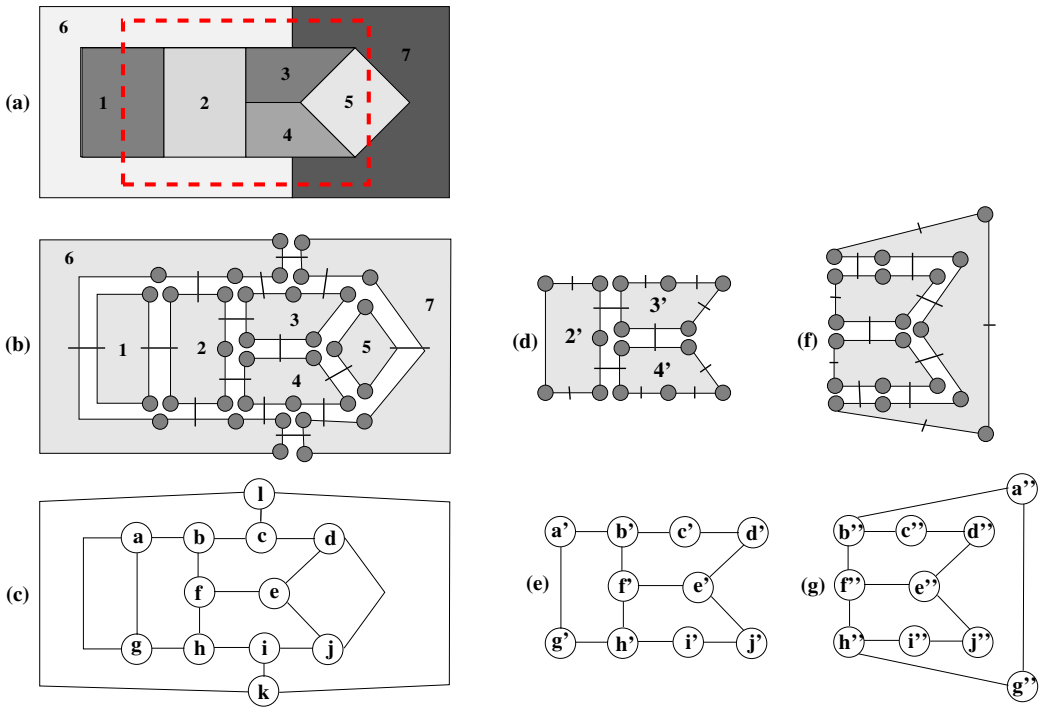


Figure 10: (a) Example of image segmented in 7 regions, with a subimage delimited by a red rectangular dotted line. (b) 2G-map M_t built from (a). (c) Multigraph G_t built from M_t. (d) 2G-map M_p built from the subimage of (a). (e) Multigraph G_p built from M_p. (f) A 2G-map M' which is not isomorphic to M_p. (g) Multigraph G' built from M'. G_p is a partial subgraph of G_t; it is not an induced subgraph as edge (d', j') does not belong to G_p and edge (a', g') occurs only once in G_p. G_p and G' are isomorphic whereas the 2G-maps from which they have been built (M_p and M') are not isomorphic.

connected components of the pattern in the target, whereas the cliques are very quickly found (in .01 second or less). Actually, the number of vertices in the graph in which we search for the cliques is usually far below the theoretical bound so that cliques are very quickly found and our algorithm scales well.

Subgraph isomorphism is much slower. When searching for each connected component separately, subgraph isomorphism is nearly 150 times as slow as submap isomorphism, on average. When searching all connected components together it is even slower and only five instances can be solved in less than one hour of CPU time. Of course, our submap algorithm does

	SMI				SGI1				SGI2
	$t1$	$t2$	v	s	$t1$	$t2$	v	s	t
p1	.14	+ .00	253	248	16.70	+ .00	536	2064	57.00
p2	.15	+ .00	2	1	1.74	+ .00	3	2	6.08
p3	.11	+ .00	3	1	1.48	+ .00	3	1	2.45
p4	.11	+ .01	280	18297	168.97	+ .23	1066	385616	>3600.00
p5	.17	+ .00	250	5376	32.21	+ .05	752	266016	>3600.00
p6	.25	+ .00	33	54	41.83	+ .00	57	716	478.34
p7	.22	+ .78	294	526864	59.97	+ 19.53	657	10096928	>3600.00
p8	.22	+ .01	152	28248	6.49	+ .12	327	950272	>3600.00
p9	.16	+ .00	46	264	4.61	+ .00	92	3968	5.11
p10	.32	+ .01	150	53568	53.37	+ .38	313	3999488	>3600.00
avg	.19	+ .08	146	63292	38.74	+ 2.03	381	1570507	>1854.90

Table 2: Experimental comparison of submap isomorphism (SMI), subgraph isomorphism when searching each connected component separately (SGI1), and subgraph isomorphism when searching all connected components together (SGI2). For SMI and SGI1, $t1$ = time for finding all occurrences of all connected components, $t2$ = time for searching for all cliques (corresponding to solutions), v = number of vertices of the graph in which cliques are searched, and s = number of solutions found. For SGI2, t = time for finding all solutions (>3600 for runs not completed within 1 hour). All times are in seconds.

not solve the same problem: It exploits the topology to search for each connected pattern in polynomial time. When ignoring this topology, the problem of searching for a connected pattern becomes NP-complete and is much more difficult to solve.

Comparison of the number of solutions. For two patterns (p2 and p3), submap isomorphism only has one solution, corresponding to the subimage that has been originally extracted in the target image. This comes from the fact that all connected components of these two patterns are rather large ones (the smallest connected component has 10 vertices). The eight other patterns (p1 and p4 to p10) contain at least one smaller connected component which occurs several times in the target image. For example, pattern p7 is composed of five connected components which have 4, 5, 6, 63, and 121 vertices, respectively. If the last two components only occur once in the target, the first three ones occur 34, 108 and 150 times in the target, respectively. In this case, the graph in which cliques are searched has $34+108+150+1+1 = 294$ vertices and this graph contains 526864 cliques of five vertices (among the $34 * 108 * 150 = 550800$ possible combinations of occurrences of the five con-

nected components). The time needed to enumerate these 526864 cliques is higher than the time needed to search occurrences of connected components (.78 compared to .22), but it is still reasonable.

For all patterns but one (p3), subgraph isomorphism finds more solutions than submap isomorphism. For example, let us consider pattern p7: the connected component with 4 (resp. 5, 6, 63, and 121) vertices occurs 70 (resp. 236, 348, 1 and 2) times in the target. In this case, the graph in which cliques are searched has $70 + 236 + 348 + 1 + 2 = 657$ vertices and this graph contains more than ten million cliques of five vertices.

6. Maximum Common Submap

The maximum common submap problem has been defined in [32]. Given two nG-maps M and M', the goal is to find the largest nG-map which is isomorphic to submaps of M and M'. The size of an nG-map depends on the number of its darts and seams. In order to have a more generic definition, it is parameterized by two weights as follows.

Definition 7. (size of an nG-map [32]) Let $M = (D, \alpha_0, \ldots, \alpha_n)$ be an nG-map, and $(\omega_1, \omega_2) \in \mathbb{R}^+$ be positive weights. The size of M is:
$size_{(\omega_1,\omega_2)}(M) = \omega_1 \cdot |D| + \omega_2 \cdot |seams_M(D)|$

Maximum common submap is more general than submap isomorphism as we may solve submap isomorphism by searching for a maximum common submap. As submap isomorphism is $\mathcal{NP}$-complete when M is not connected, maximum common submap is $\mathcal{NP}$-hard when the common submap is not required to be connected.

Let us now consider the maximum common connected submap problem, where the common submap is required to be connected. As there exists a polynomial-time algorithm for solving submap isomorphism for connected nG-maps, we could try to extend the key ideas of this algorithm to search for maximum connected common submaps (we have actually been trying for a while...). In this section, we show that this is not possible, unless $\mathcal{P} = \mathcal{NP}$.

The maximum common connected submap problem is an optimization problem. As class $\mathcal{NP}$ is defined for decision problems only, we consider the decision problem which involves deciding if there exists a common connected submap whose size is greater than a given bound. Obviously, the maximum

common connected submap problem is at least as hard as this decision problem. The decision problem is formally defined below.

Problem: Partial (resp. induced) common connected submap
Instance: A tuple $(s, \omega_1, \omega_2, M, M')$ such that M and M' are nG-maps, $(\omega_1, \omega_2) \in \mathbb{R}^+$ are positive weights, and $s \in \mathbb{R}^+$ is a bound on the size of the common submap
Question: Does there exist a connected nG-map M'' such that $size_{(\omega_1,\omega_2)}(M'') \geq s$, $M'' \sqsubseteq^p M$ and $M'' \sqsubseteq^p M'$ (resp. $M'' \sqsubseteq^i M$ and $M'' \sqsubseteq^i M'$)?

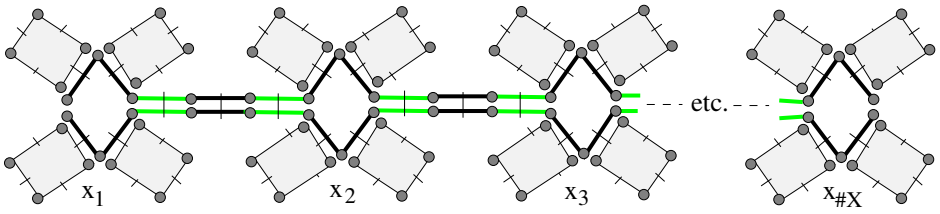
Theorem 3 claims that this problem is $\mathcal{NP}$-complete.

Theorem 3. *Partial (resp. induced) common connected submap is $\mathcal{NP}$-complete.*

The problem trivially belongs to $\mathcal{NP}$ since one can check that a given connected nG-map M'' has a size greater than s and is isomorphic to partial (resp. induced) submaps of M and M' in polynomial time.

To prove $\mathcal{NP}$-completeness, we exhibit a polynomial-time reduction of Separable Planar 3-SAT to the problem. We first consider the induced case; the extension of the proof to the partial case is discussed at the end of this section. We consider an instance (X, F, μ) of Separable Planar 3-SAT and we show how to build an instance $(s, \omega_1, \omega_2, M, M')$ of induced common connected submap such that the two instances have the same answers. We consider 2G-maps, so that $n = 2$. We set ω_1 to 1 and ω_2 to 0 so that the size of an nG-map is equal to its number of darts.

Definition of the 2G-map M. Let us first describe building blocks associated with variables. For each variable $x_i \in X$, M contains four 8-dart faces connected by two chains of darts, called connecting chains, as displayed below.



The two connecting chains are displayed in bold, and they connect all variables, starting from x_1, and ending on $x_{\#X}$. Some darts in the connecting chains are displayed in green. We call them choice-point darts.

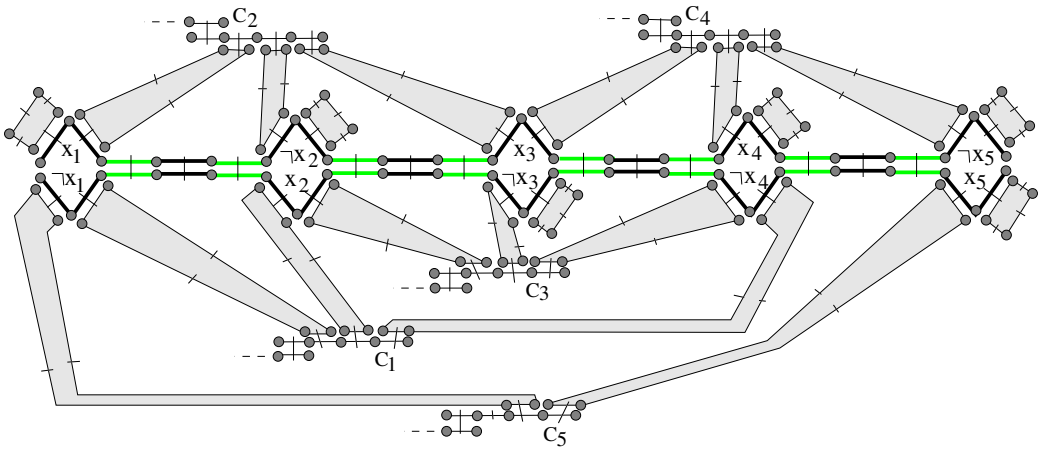
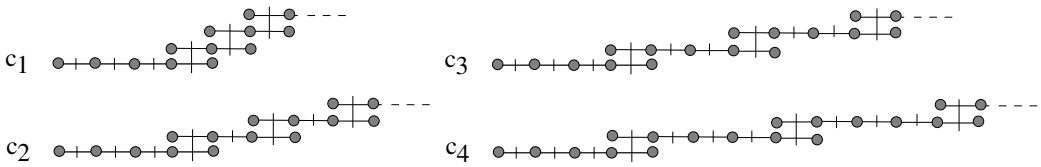


Figure 11: Map M associated with the Seperable Planar 3-SAT instance displayed in Fig. 4. Clause gadgets contain $D+1$ darts and are all different but we have only displayed the first 10 darts.

Let us recall that any separable embedding may be drawn so that all variables are aligned from x_1 to $x_{\#X}$, and all clauses which are inside (resp. outside) the E_2 cycle are above (resp. below) the variable line [29]. Without loss of generality, we assume in this reduction that the embedding μ is such that all variables are aligned. The upper part of the connecting chain corresponding to a variable x_i is labelled with x_i (resp. $\neg x_i$) if $pos(x_i) = in(x_i)$ (resp. $neg(x_i) = in(x_i)$) whereas the lower part is labelled with $\neg x_i$ (resp. x_i), as displayed in Fig. 11.

The whole variable gadget is composed of four 8-dart faces for each variable and two connecting chains of $(\#X - 1) * 10$ darts each. Therefore, it has $52 * \#X - 20$ darts. Let us define $D = 52 * \#X - 20$.

We now introduce the building blocks associated with clauses. For each clause c_j, M contains a gadget composed of $D+1$ darts: The first eight darts are aligned in a chain; the remaining darts are composed of 2-sewn chains which have $2j + 2$ darts as illustrated below for c_1, c_2, c_3 and c_4.

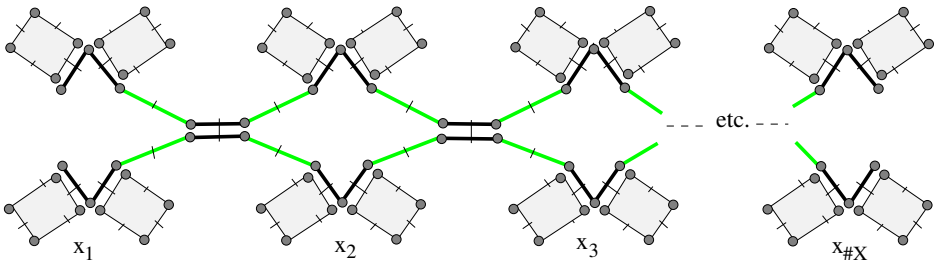


Hence, gadgets associated with clauses are all different and the gadget associated with a clause c_j cannot be matched with the gadget associated with

another clause c_k.

These clause gadgets are positionned in the plane like in the separable embedding μ of $G_{(X,F)}$. The first $2 \cdot \#c_j$ darts of the gadget associated with c_j are 2-sewn with the 8-dart faces corresponding to the literals of c_j, as illustrated in Fig. 11. Note that in Separable Planar 3-SAT instances every variable occurs in 2 or 3 clauses, and at least once positively and once negatively, so that no variable occurs more than twice positively or more than twice negatively. This ensures us that variable gadgets have enough 8-dart faces to connect clause gadgets with variable gadgets.

Definition of the 2G-map M'. The second nG-map M' is obtained from the first nG-map M by 2-unsewing choice-point darts of the two connecting chains as displayed below:



Also, for every clause c_j such that c_j contains $\#c_j$ literals, M' contains $\#c_j$ occurrences of the clause gadget associated with c_j. The first (resp. second and third, if any) two darts of the first (resp. second and third, if any) occurrence of this clause gadget are 2-sewn with the 8-dart face associated with the corresponding literal, as illustrated in Fig. 12.

Definition of the bound on the size s. We can now define the bound on the size of the common nG-map as $s = \#F \cdot (D + 1)$ where $D = 52 * \#X - 20$.

Proof of $(\exists M'', M'' \sqsubseteq^i M, M'' \sqsubseteq^i M', size_{(1,0)}(M'') \geq s) \Rightarrow (F$ is satisfiable). Let us first assume that M'' exists, and let us show that there exists a truth assignment A of X which satisfies F.

Let us first note that if we remove the choice-point darts (displayed in green), the nG-map M' is composed of $2 \cdot \#X$ different connected components (one for every truth value of every variable) and each connected component cannot contain more than two clause gadgets. Therefore, the only way to connect more than two clause gadgets is to keep in the submap M'' some

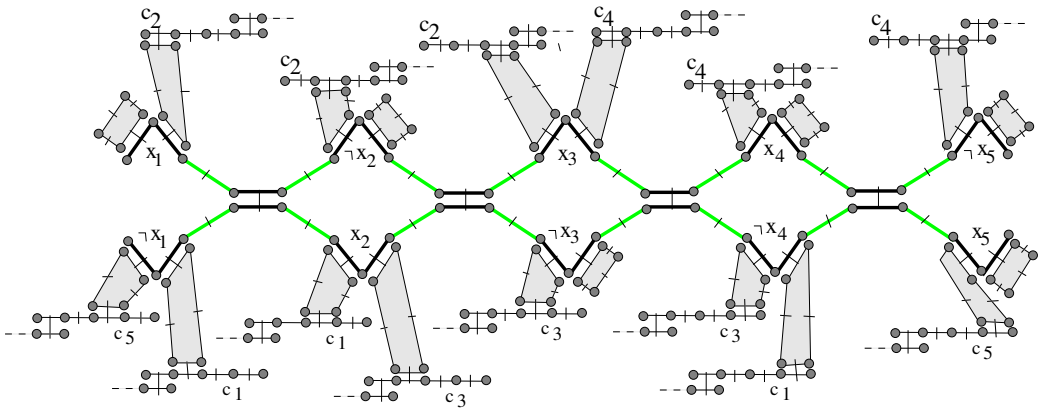
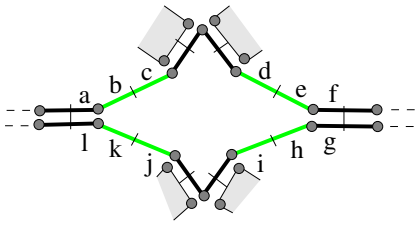


Figure 12: Map M' associated with the Separable Planar 3-SAT instance displayed in Fig. 4. Clause gadgets contain $D+1$ darts but we have only displayed the first 10 darts.

choice-point darts. However, given a variable x_i, M'' cannot contain both upper choice-point darts and lower choice-point darts. More precisely, let us consider the pattern associated with a variable x_i in M',



and show that if M'' contains one or more darts of $\{b, c, d, e\}$ then it cannot contain any dart of $\{k, j, i, h\}$ (and *vice versa*). Indeed, if M'' contains b, then it cannot contain k as these darts are 2-sewn in M and 2-free in M', and it cannot contain h, i or j as M'' must be connected and all the paths of sewn darts which connect b to a dart of $\{h, i, j\}$ either go through k (via a and l), which is not possible as M'' cannot contain k, or go through e and h (via f and g), which is not possible as e and h are 2-sewn in M and 2-free in M'. Therefore, for each variable x_i, M'' cannot contain both the upper and the lower part of the connecting chain, *i.e.*, we cannot recover both label x_i and label $\neg x_i$ in the truth assignment.

Now, let us show how to build the truth assignment A from M''. Each variable $x_i \in X$ is assigned in A to the truth value associated with the part of the connecting chains which belongs to M''. It may happen that M'' does

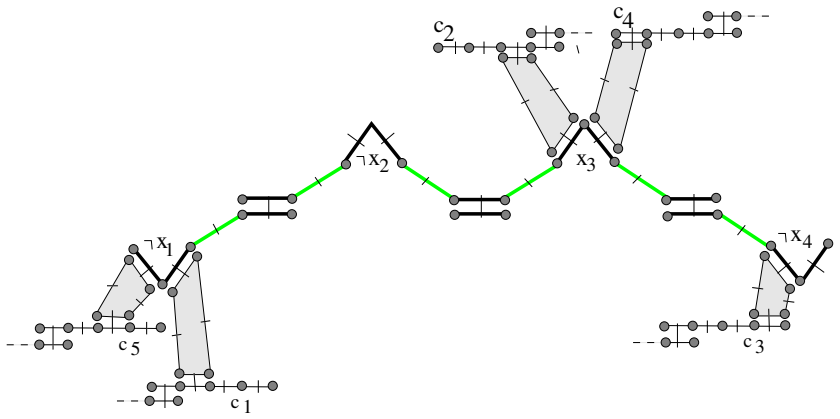


Figure 13: Common connected submap M'' of the nG-maps of figures 11 and 12. M'' has a size greater than $s = 5 \cdot (D+1)$ as it contains one gadget for each clause $c_1, \ldots, c_5$. The truth assignment built from M'' is $A = \{\neg x_1, \neg x_2, x_3, \neg x_4, x_5\}$. Note that M'' does not contain any part of the connecting chains associated with x_5 as all clauses are satisfied whatever the truth value of x_5 is. Therefore, x_5 is arbitrarily set to true in A.

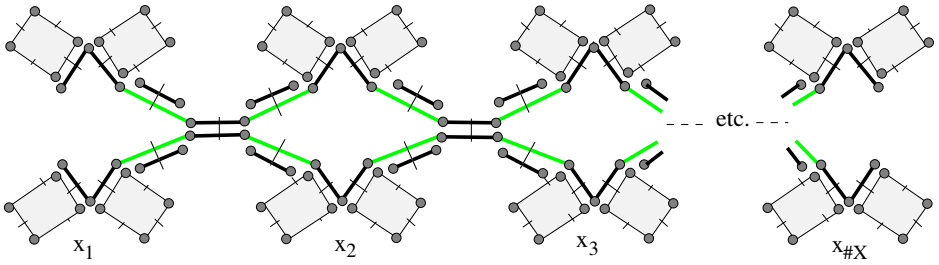
not contain any part of the connecting chains associated with x_i. This occurs when all the clauses of formula F can be satisfied whatever the truth value of the first variable x_1 or the last variable $x_{\#X}$, as illustrated in Fig. 13. In this case, A arbitrarily assigns the corresponding variable to true.

Let us show that A satisfies all clauses of F. This is a direct consequence of the fact that $size_{(1,0)}(M'') \geq s$ with $s = \#F \cdot (D+1)$. Indeed, $size_{(1,0)}(M'') \geq s$ iff M'' contains $\#F$ clause gadgets as each clause gadget contains $D+1$ darts whereas the whole variable gadget has D darts. As M contains $\#F$ clause gadgets which are all different, M'' must contain exactly one occurrence of each clause gadget, *i.e.*, one for each different clause of F. Each clause c_j is satisfied by at least one literal, *i.e.*, the literal which connects the gadget associated with c_j to the connecting chains in M''. For example, in Fig. 13, c_1 is satisfied by $\neg x_1$, c_2 by x_3, c_3 by $\neg x_4$, c_4 by x_3, and c_5 by $\neg x_1$.

Proof of $(F$ *is satisfiable*$) \Rightarrow (\exists M'', M'' \sqsubseteq^i M, M'' \sqsubseteq^i M', size_{(1,0)}(M'') \geq s)$. Let us assume that there exists a truth assignment A of X which satisfies F and let us show how to derive from A a nG-map M'' which is isomorphic to induced submaps of M and M' and whose size is greater than s. We build M'' from M' by removing some darts from it. For each variable x_i, we remove from M' the part of the connecting chain which corresponds to the negation of the truth value of x_i in A (together with its two adjacent 8-edge faces).

For each clause c_j, we choose one literal of c_j which is satisfied by A and we remove from M the clause gadgets corresponding to the other literals of c_j. We obtain a connected nG-map which contains $\#F$ clause patterns so that its size is greater than $\#F \cdot (D+1)$. One can easily check that this nG-map is also an induced submap of M.

Proof for the partial case. Let us now consider the partial case. To extend the previous proof to the partial case, we have to modify the connecting chains in the nG-map M'. The goal is to forbid one to keep in a common submap the two choice-point darts associated with the two different truth values of a same variable. To this aim, we 2-sew each choice-point dart with new darts which are 1-free as displayed below:



Also, we modify the weights used to define the size of an nG-map. Indeed, in the partial case, if ω_2 is set to 0, a maximum common submap may be obtained by removing all seams of the smallest nG-map while keeping all its darts (as the size of an nG-map only depends on its number of darts). Therefore, we set both ω_1 and ω_2 to 1 so that the size of an nG-map is equal to its number of darts and seams. Finally, we modify clause gadgets so that the size of each clause gadget is greater than the size of the whole variable gadget (all 8-dart faces and the two connecting chains), and define the bound on the size s as $\#F$ times the size of a clause gadget.

7. Conclusion

In this paper, we have proved that the search of a non connected pattern submap in a target nG-map is an $\mathcal{NP}$-complete problem, by reduction of Separable Planar 3-SAT. Nevertheless, we have shown that it is Fixed-Parameter Tractable, by describing an FPT-algorithm whose time complexity is exponential only in the number of connected components in the pattern nG-map. Our algorithm may be extended to plane graphs in a rather straightforward

way. In this case, the dimension n of the nG-maps is equal to 2 and the time complexity of the algorithm becomes $\mathcal{O}(v'^k + vv'^2)$ where v and v' are the number of vertices of the pattern and the target graphs. This complexity may be compared to the complexity of the FPT-algorithm proposed by Dorn [23] for planar subgraph isomorphism, where the fixed parameter is the number of vertices v of the pattern graph. In this case, Dorn gives an FPT-algorithm which is linear in v' and exponential in v (instead of the number of connected components k in our case). Of course, we do not solve the same problem: We consider plane graphs whereas Dorn considers planar graphs.

Also, we have proved that the computation of the maximum common connected submap of two nG-maps is $\mathcal{NP}$-hard, by reduction of Separable Planar 3-SAT. A first consequence of this result is that the computation of the map edit distance of [10] is also $\mathcal{NP}$-hard as we have shown in [10] that there exist edit costs for which the map edit distance may be derived in polynomial time from the maximum common submap. Another consequence is that the computation of a maximum connected common plane subgraph is $\mathcal{NP}$-hard (even though plane subgraph isomorphism is polynomial when the pattern graph is connected). Indeed, the reduction described in Section 6 may be extended to plane graphs in a rather straighforward way.

Determining whether the maximum common submap is approximable or not is an interesting perspective. Indeed, on the one hand, the problem of computing the maximum common connected subgraph of two graphs is deeply related to the computation of a maximum clique in the product of the graphs [33], and Max Clique is known to be hard to approximate [34]. But on the other hand, the maximum common connected subgraph was recently shown to be solvable in polynomial time for outerplanar graphs of bounded degree [25], which leaves room for further investigations in the framework of nG-maps.

Acknowledgement. This work has been partially supported by the French National Agency (ANR), project Solstice ANR-13-BS02-01.

References

[1] A. Rosenfeld, Adjacency in digital pictures, Information and Control 26 (1) (1974) 24–33.

[2] A. Trémeau, P. Colantoni, Regions adjacency graph applied to color

image segmentation, IEEE Transactions on Image Processing 9 (2000) 735–744.

[3] W. Kropatsch, Building irregular pyramids by dual graph contraction, in: IEEE Conference on Vision, Image and Signal Processing, Vol. 142, 1995, pp. 366–374.

[4] A. Pinz, H. Bischof, W. Kropatsch, G. Schweighofer, Y. Haxhimusa, A. Opelt, A. Ion, Representations for cognitive vision: A review of appearance-based, spatio-temporal, and graph-based approaches, Electronic letters on computer vision and image analysis 7 (2) (2008) 35–61.

[5] P. Lienhardt, N-dimensional generalized combinatorial maps and cellular quasi-manifolds, Computational Geometry and Applications 4 (3) (1994) 275–324.

[6] D. Fradin, D. Meneveaux, P. Lienhardt, A hierarchical topology-based model for handling complex indoor scenes., Computer Graphics Forum 25 (2) (2006) 149–162.

[7] J. P. Braquelaire, L. Brun, Image segmentation with topological maps and inter-pixel representation, Visual Communication and Image representation 9 (1) (1998) 62–79.

[8] G. Damiand, C. Solnon, C. de la Higuera, J.-C. Janodet, E. Samuel, Polynomial algorithms for subisomorphism of nd open combinatorial maps, Computer Vision and Image Understanding (CVIU) 115 (7) (2011) 996–1010.

[9] C. Solnon, G. Damiand, C. D. L. Higuera, J.-C. Janodet, On the complexity of submap isomorphism, in: GbR, Vol. 7877 of LNCS, Springer, 2013, pp. 21–30.

[10] C. Combier, G. Damiand, C. Solnon, From maximum common submaps to edit distances of generalized maps, Pattern Recognition Letters 33 (15) (2012) 2020–2028.

[11] H. Bunke, K. Shearer, A graph distance metric based on the maximal common subgraph, Pattern Recognition Letters 19 (3-4) (1998) 255–259.

[12] G. Damiand, C. De La Higuera, J.-C. Janodet, E. Samuel, C. Solnon, Polynomial algorithm for submap isomorphism: Application to searching patterns in images, in: GbR, Vol. 5534 of LNCS, Springer, 2009, pp. 102–112.

[13] J. E. Hopcroft, J. K. Wong, Linear time algorithm for isomorphism of planar graphs, in: STOC, ACM, 1974, pp. 172–184.

[14] E. M. Luks, Isomorphism of graphs of bounded valence can be tested in polynomial time, J. Comput. Syst. Sci. 25 (1) (1982) 42–65.

[15] X. Jiang, H. Bunke, Optimal quadratic-time isomorphism of ordered graphs, Pattern Recognition 32 (7) (1999) 1273–1283.

[16] S. Sorlin, C. Solnon, A parametric filtering algorithm for the graph isomorphism problem, Constraints 13 (4) (2008) 518–537.

[17] K. Riesen, S. Fankhauser, H. Bunke, P. J. Dickinson, Efficient suboptimal graph isomorphism, in: GbRPR, Vol. 5534 of Lecture Notes in Computer Science, 2009, pp. 124–133.

[18] M. Garey, D. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman, 1979.

[19] D. W. Matula, Subtree isomorphism in o(n5/2), in: P. H. B. Alspach, D. Miller (Eds.), Algorithmic Aspects of Combinatorics, Vol. 2 of Annals of Discrete Mathematics, Elsevier, 1978, pp. 91 – 106.

[20] M. M. Syslo, The subgraph isomorphism problem for outerplanar graphs, Theoretical Computer Science 17 (1) (1982) 91 – 97.

[21] T. Horváth, J. Ramon, S. Wrobel, Frequent subgraph mining in outerplanar graphs, Data Min. Knowl. Discov. 21 (3) (2010) 472–508.

[22] D. Eppstein, Subgraph isomorphism in planar graphs and related problems, J. Graph Algorithms Appl. 3 (3).

[23] F. Dorn, Planar subgraph isomorphism revisited, in: STACS, Vol. 5 of LIPIcs, 2010, pp. 263–274.

[24] T. Akutsu, A polynomial time algorithm for finding a largest common subgraph of almost trees of bounded degree, IEICE Trans. Fundam. (1993) 1488–1493.

[25] T. Akutsu, T. Tamura, A polynomial-time algorithm for computing the maximum common subgraph of outerplanar graphs of bounded degree, in: Mathematical Foundations of Computer Science, Vol. 7464 of Lecture Notes in Computer Science, Springer, 2012, pp. 76–87.

[26] L. Schietgat, J. Ramon, M. Bruynooghe, A polynomial-time maximum common subgraph algorithm for outerplanar graphs and its application to chemoinformatics, Annals of Mathematics and Artificial Intelligence (2013) 1–34.

[27] S. A. Cook, The complexity of theorem-proving procedures, in: ACM Symposium on Theory of Computing, 1971, pp. 151–158.

[28] D. Lichtenstein, Planar formulae and their uses, SIAM J. Comput. 11 (2) (1982) 329–343.

[29] D. E. Knuth, A. Raghunathan, The problem of compatible representatives, SIAM J. Discrete Math. 5 (3) (1992) 422–427.

[30] G. Damiand, Y. Bertrand, C. Fiorio, Topological model for two-dimensional image representation: definition and optimal extraction algorithm., Computer Vision and Image Understanding 93 (2) (2004) 111–154.

[31] C. Solnon, Alldifferent-based filtering for subgraph isomorphism, Artif. Intell. 174 (12-13) (2010) 850–864.

[32] C. Combier, G. Damiand, C. Solnon, Measuring the distance of generalized maps, in: GbR, LNCS, Springer, 2011, pp. 82–91.

[33] I. Koch, Enumerating all connected maximal common subgraphs in two graphs, Theor. Comput. Sci. 250 (1-2) (2001) 1–30.

[34] D. Zuckerman, Linear degree extractors and the inapproximability of max clique and chromatic number, Theory of Computing 3 (1) (2007) 103–128.