Discrete Applied Mathematics 283 (2020) 123–132

Contents lists available at ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

The complete set of minimal simple graphs that support unsatisfiable 2-CNFs

Vaibhav Karve, Anil N. Hirani *

Department of Mathematics, University of Illinois at Urbana-Champaign, 1409 W. Green Street, Urbana, IL 61801, United States of America

ARTICLE INFO

Article history:
Received 27 January 2019
Received in revised form 24 October 2019
Accepted 23 December 2019
Available online 20 January 2020

Keywords:
Boolean satisfiability
Conjunctive normal form
Propositional logic
Graph minors
Topological minors
Edge contraction
Subdivision

ABSTRACT

A propositional logic sentence in conjunctive normal form that has clauses of length at most two (a 2-CNF) can be associated with a multigraph in which the vertices correspond to the variables and edges to clauses. We show that every 2-CNF that has been reduced under the application of certain tautologies, is equisatisfiable to a 2-CNF whose associated multigraph is, in fact, a simple graph. Our main result is a complete characterization of graphs that can support unsatisfiable 2-CNF sentences. We show that a simple graph can support an unsatisfiable reduced 2-CNF sentence if and only if it contains any one of four specific small graphs as a topological minor. Equivalently, all reduced 2-CNF sentences supported on a given simple graph are satisfiable if and only if all subdivisions of those four graphs are forbidden as subgraphs of the original graph.

1. Introduction

Given a sentence in propositional logic, the *satisfiability decision problem* is to determine if there exists a truth assignment for the variables that makes the sentence true. Let V be a countably infinite set of Boolean variables, $\neg V$ be the set $\{\neg x\colon x \in V\}$ of negations, and let the symbols $\top$ and $\bot$ be *True* and *False* values of variables. We define the *set of literals obtained from* V to be the set $V \cup \neg V \cup \{\top, \bot\}$.

A *Conjunctive Normal Form (CNF) on* V is a conjunction of one or more clauses, where each *clause* is a disjunction of literals. A CNF (or clause) is *reduced* if it is unchanged under application of all the tautologies listed below

$$x \vee \top = \top, \quad x \wedge \top = x, \quad x \vee x = x, \quad x \vee \neg x = \top,$$
$$x \vee \bot = x, \quad x \wedge \bot = \bot, \quad x \wedge x = x.$$

It is a fact that every CNF (or clause) is logically equivalent to a reduced CNF (or clause). The *length* of a reduced clause is the number of literals in the clause. For $k \in \mathbb{N}$, a reduced CNF is a *reduced k-CNF* if all of its clauses have length at most k. In particular, the CNFs $\top$ and $\bot$ are k-CNFs for every $k \in \mathbb{N}$. We will refer to the satisfiability decision problem as SAT and the satisfiability problem for k-CNFs as k-SAT.

Instead of algorithmic issues our aim in this paper is to study the structure of unsatisfiable 2-CNF sentences in a sense that will be made precise later. For completeness, here we briefly summarize the relevant fundamental algorithmic results for satisfiability problems. The 2-SAT problem is in P. An $\mathcal{O}(n^4)$ algorithm was given by Krom [9] and a linear-time

* Corresponding author.
E-mail addresses: vkarve2@illinois.edu (V. Karve), hirani@illinois.edu (A.N. Hirani).

https://doi.org/10.1016/j.dam.2019.12.017
0166-218X/

algorithm by Even, Itai and Shamir [6] and Aspvall, Plass and Tarjan [1]. All solutions of a given 2-CNF sentence can be listed efficiently using an algorithm by Feder [7].

In contrast with the algorithmic tractability of 2-SAT, the SAT problem in general is NP-complete as was shown by Cook and by Levin independently [5,10]. As part of the proof of the NP-completeness of SAT, they also proved that every logical sentence can be rewritten as a CNF while changing its length by no more than a constant factor. Schaefer's dichotomy theorem states necessary and sufficient conditions under which a finite set S of relations over the Boolean domain yields polynomial-time or NP-complete problems when the relations of S are used to constrain some of the propositional variables [12]. Thus [12] gives a necessary and sufficient condition for SAT-type problems to be in P vs. NP. This result can also be reformulated in terms of Boolean Constraint Satisfaction Problems (Boolean CSPs) [4] leading to a more algebraic version of Schaefer's dichotomy theorem as well as recent generalizations of the theorem to larger classes of relations [3].

Our paper relates properties of certain graphs to satisfiability. An early connection between satisfiability and graphs was in the proof of NP-completeness of various graph problems, such as the clique decision problem and the vertex cover problem, by Karp [8]. One of the linear-time algorithms for 2-SAT [1] mentioned above also related graphs and satisfiability in its use of strongly-connected graph components as a tool for deciding satisfiability, while [3] provides an analog of Schaefer's dichotomy result for the propositional logic of graphs instead of Boolean logic

We explore the structures of unsatisfiable 2-CNFs by relating reduced 2-CNFs to graphs and examining which graphs can support unsatisfiable sentences. Given a 2-CNF, an associated multigraph can be created by treating the variables as vertices and clauses as edges. Since multiple clauses may involve the same two variables, it is not immediate that it is sufficient to consider graphs rather than multigraphs. That it indeed is so is the content of Section 4. In Section 5 we prove results about the relation between graphs that support unsatisfiable sentences. Theorem 3 in that section shows that the family of simple graphs that can support an unsatisfiable sentence is closed under graph homeomorphism while Remark 4 shows that a graph can support an unsatisfiable sentence if one of its subgraphs can. Theorem 6 shows that if a graph can support an unsatisfiable sentence, then the graphs obtained by edge-contractions at edges not contained in triangles can. Section 7 is about connectivity properties of graphs that we need to prove the main result. The main result of this paper is Theorem 18 in which we give a complete characterization of graphs that can support unsatisfiable sentences. This is given in the form of a finite set of obstructions to supporting only satisfiable sentences. In Section 9 we discuss our conclusions as they relate to the computational complexity of searching for obstructions (forbidden topological minors) and the possibilities for generalizing our result to 3-CNFs and their associated multi-hypergraphs.

2. Preliminaries

For a reduced CNF S and a variable v, we denote by $S[v \coloneqq \top]$ the reduced CNF obtained by

(a) setting all occurrences of v in S to $\top$,
(b) setting all occurrences of $\neg v$ in S to $\bot$, and
(c) reducing the CNF thus obtained, that is, applying all the tautologies listed in Section 1 to the CNF in order to obtain a reduced CNF.

Similarly, the reduced CNF obtained by setting v to $\bot$, setting $\neg v$ to $\top$, and then performing the tautological reductions is denoted by $S[v \coloneqq \bot]$. If we set multiple variables to true or false, we use the notation $S[\{a, b\} \coloneqq \top;\ \{c, d\} \coloneqq \bot]$ as short-hand for $S[a \coloneqq \top][b \coloneqq \top][c \coloneqq \bot][d \coloneqq \bot]$ and so on for variables a, b, c and d. Note that the order in which we set variables to true or false is not important, as it yields logically equivalent results.

We recall some special types of CNFs. A reduced CNF S is *true* if $S = \top$, *false* if $S = \bot$, and *nontrivial* if S is neither true nor false. A reduced CNF S is *satisfiable* if there is a subset V_1 of V such that $S[V_1 \coloneqq \top; V - V_1 \coloneqq \bot]$ is true. A reduced CNF is *unsatisfiable* if it is not satisfiable. Two reduced CNFs, S and S' are *equisatisfiable*, denoted by $S \sim S'$, if either both are satisfiable or both are unsatisfiable. We note that logical equivalence is a more stringent condition than equisatisfaction.

3. Graph associated with a reduced CNF

We define an operation $|\cdot| : V \cup \neg V \to V$ as $|x| = |\neg x| = x$, for every $x \in V$. Let $\alpha = \bigvee_{i=1}^{k} a_i$, where $k \in \mathbb{N}$ and $a_i \in V \cup \neg V$ for $i \in \{1, \dots, k\}$. We extend the domain of the operation $|\cdot|$ to include clauses like α by defining $|\alpha| = \{|a_i| : i \in \{1, \dots, k\}\}$.

Let S be a nontrivial reduced 2-CNF. We can write $S = \bigwedge_{j=1}^{n} \alpha_j$, where $n \in \mathbb{N}$ and each α_j is a clause of length 1 or 2. To S we associate a multigraph $\mathcal{MG}(S)$ having vertex set $\bigcup_{j=1}^{n} |\alpha_j|$ and having an edge $|\alpha_j|$ for every $j \in \{1, \dots, n\}$. Note that a variable x and its negation $\neg x$ are both represented by a single vertex (also labeled x) in $\mathcal{MG}(S)$. The 2-CNF S being nontrivial guarantees that $\mathcal{MG}(S)$ is a nonempty multigraph. S being reduced ensures that there are no self-loops in $\mathcal{MG}(S)$, since self-loops can only occur due to clauses $(x \vee \neg x)$ and $(x \vee x)$.

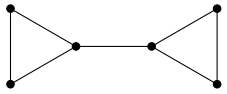
If a multigraph G is the associated multigraph of a 2-CNF S, then we say S is *supported on* G. A nontrivial 2-CNF S is *simple* if its associated graph is simple. In such a case, we drop the prefix "multi-" and denote the associated graph $\mathcal{MG}(S)$ simply by $\mathcal{G}(S)$. We denote by $\mathcal{U}$ the following family of graphs

$$\mathcal{U} = \{\, \mathcal{G}(S) \colon S \text{ is an unsatisfiable simple 2-CNF} \,\}.$$

Our aim in this paper is to characterize the elements of $\mathcal{U}$. First, we note that $\mathcal{U}$ is nonempty since

$$S = (a \vee b) \wedge (\neg a \vee c) \wedge (\neg b \vee c) \wedge (\neg c \vee d) \wedge (\neg d \vee e) \wedge (\neg d \vee f) \wedge (\neg e \vee \neg f),$$

is an unsatisfiable simple 2-CNF supported on the following graph.



4. Simple CNFs suffice

In this section, we show that every nontrivial reduced 2-CNF is equisatisfiable to a simple 2-CNF. Thus when studying satisfiability of 2-CNFs we only need to consider those that are simple.

First, we make the observation that for any $a \in V$, there are at most two length 1 clauses, namely a and $\neg a$. Such clauses involving a single variable a (or its negation) will be referred to as (a)-*clauses*. For every $a, b \in V$, there are at most four length 2 clauses, namely $(a \vee b)$, $(a \vee \neg b)$, $(\neg a \vee b)$ and $(\neg a \vee \neg b)$. Consequently, for any reduced 2-CNF S, edges in $\mathcal{MG}(S)$ have a maximum multiplicity of 4. Such clauses involving *both* a and b (or their negations) will be referred to as (a, b)-*clauses*.

Remark 1. Given a reduced CNF S and $a, b \in V$, we make the following observations:

(a) $S \wedge a \wedge \neg a$ is unsatisfiable.
(b) $S \wedge a \sim S[a := \top]$.
(c) $S \wedge (a \vee b) \wedge (a \vee \neg b) \wedge (\neg a \vee b) \wedge (\neg a \vee \neg b)$ is unsatisfiable.
(d) $S \wedge (a \vee b) \wedge (a \vee \neg b) \wedge (\neg a \vee b) \sim S[\{a, b\} := \top]$.
(e) $S \wedge (a \vee b) \wedge (a \vee \neg b) \sim S[a := \top]$.
(f) $S \wedge (a \vee b) \wedge (\neg a \vee \neg b) \sim S[b := \neg a]$

The proofs of these statements are elementary. For example, to prove (f), we note that any truth assignment satisfying the CNF must assign a and b to opposite truth values in order to satisfy the two (a, b)-clauses. Thus, such a truth assignment must also satisfy $S[b := \neg a]$. Conversely, any truth assignment satisfying $S[b := \neg a]$, can be turned into a truth assignment that satisfies the CNF by additionally setting b to the negation of the assignment for a.

Lemma 2. *Let S be a reduced 2-CNF. There exists a reduced 2-CNF S', such that S' is equisatisfiable to S, and is either trivial or simple.*

Proof. We can construct S' from S by repeatedly applying a series of steps. For any variable a, if S has an (a)-clause, then by Remark 1, the CNF S is either trivially false or is equisatisfiable to a reduced 2-CNF without any (a)-clauses. By this reasoning, we can eliminate all (a)-clauses from S.

For every (a, b)-clause that appears more than once in S, by exchanging a with $\neg a$ and/or b with $\neg b$ and/or a with b, we can ensure that S is in one of the forms that appear on the left side of Remark 1 (c)–(f)). Using Remark 1, we can therefore reduce S to an equisatisfiable reduced 2-CNF not containing any (a, b)-clauses. Repeating this process for every variable pair whose clauses appear more than once in S, and after reapplying the previous steps and any further tautological reductions, we can obtain S' such that S' is either trivial or simple. □

Throughout the remainder of this paper, when we refer to graphs, we will always mean simple graphs unless stated otherwise.

5. $\mathcal{U}$ Is closed under graph homeomorphism

In this section we look at three different graph operations – subgraphing, subdivision and edge-contraction at edges not contained in triangles – and study whether $\mathcal{U}$ is closed under these operations. The definitions for these graph operations are standard and can be found in any graph text on graph theory, for example [2].

A graph G is a *subgraph* of a graph H if both the vertex set and edge set of G are subsets of the vertex and edge sets of H. We will show that if a subgraph of a graph is in $\mathcal{U}$, then the graph itself must also be in $\mathcal{U}$. *Edge-contraction at an edge* (u, v) of a graph results in a graph in which the vertices u and v are merged into a single new vertex w and all edges incident on u or v are now incident on w. In order to avoid the creation of multi-edges we restrict the edge-contraction operation to edges not contained in triangles. This is a necessary restriction and without it one would lose the correspondence between simple 2-CNFs and their associated simple graphs. We will show that if G can be obtained from a graph in $\mathcal{U}$ via a series of edge-contractions at edges not contained in triangles, then G must be in $\mathcal{U}$.

A *subdivision of an edge* (u, v) in a graph yields a graph containing one new vertex w, and with an edge set replacing (u, v) by two new edges (u, w) and (w, v). A *subdivision of a graph* G is a graph resulting from the subdivision

of edges in G. Two graphs are *homeomorphic* if they are subdivisions of the same graph. We will show that if two graphs are homeomorphic, then either both are in $\mathcal{U}$ or neither is.

A graph G is a *topological minor* of a graph H if a subdivision of G is a subgraph of H. We note that the relation of being a subgraph, a subdivision, or a topological minor, are each reflexive as well as anti-symmetric.

We will produce a complete list of graphs whose appearance as a topological minor is an obstruction to satisfiability of 2-CNFs. It is possible to embed simple graphs into $\mathbb{R}^3$ (by mapping distinct vertices to distinct points) and to allow them to inherit the subspace topology of $\mathbb{R}^3$. Once embedded, homeomorphic graphs are also homeomorphic in the topological sense and topological graph minors are simply topological subspaces [2]. In that sense, the ability to support unsatisfiable sentences is a topological property of graphs.

Theorem 3. *If simple graphs G and H are homeomorphic, then $G \in \mathcal{U}$ if and only if $H \in \mathcal{U}$.*

Proof. Since G and H are homeomorphic graphs, there exists a graph K such that both G and H are subdivisions of K. It is enough to prove that $K \in \mathcal{U}$ if and only if $G \in \mathcal{U}$. In fact, it is enough to prove that $K \in \mathcal{U}$ if and only if $G' \in \mathcal{U}$, where G' is a graph obtained via a single subdivision at an arbitrary edge (u, v) in K. We denote by w the new vertex in G' created by the subdivision.

Let S_K be a reduced 2-CNF supported on K. Without loss of generality, we can write $S_K = S \wedge (u \vee v)$, where S is a simple 2-CNF not containing (u, v)-clauses. We then define a simple 2-CNF $S_{G'} = S \wedge (u \vee w) \wedge (\neg w \vee v)$, supported on G' and observe that any truth assignment satisfying $S_{G'}$ must also satisfy S_K because,

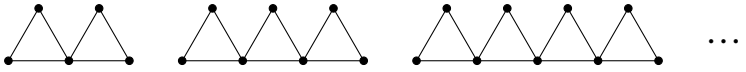
$$S_{G'} \sim (S_{G'}[w := \top]) \vee (S_{G'}[w := \bot]) = (S \wedge v) \vee (S \wedge u) = S_K.$$

Hence, if $K \in \mathcal{U}$, then $G' \in \mathcal{U}$. For the converse, we can write without loss of generality that any 2-CNF supported on G' has the form

$$S_{G'} = S \wedge (u \vee w) \wedge (w \vee v) \qquad \text{or} \qquad S_{G'} = S \wedge (u \vee w) \wedge (\neg w \vee v).$$

In both cases, we have $S_{G'} \sim S \wedge (u \vee v)$ and hence we can conclude that $G' \in \mathcal{U}$ implies $K \in \mathcal{U}$. □

In view of the theorem we just proved, it is natural to attempt a classification of all members of $\mathcal{U}$ up to homeomorphism. However, we observe that there are infinitely many graphs in $\mathcal{U}$ even up to homeomorphism. Given below is a (non-exhaustive) infinite list of graphs in $\mathcal{U}$, none of which are homeomorphic to each other.



Remark 4. If G and H are simple graphs such that G is a subgraph of H, then we note that $G \in \mathcal{U}$ implies $H \in \mathcal{U}$ because we can simply add clauses to an unsatisfiable CNF while preserving its unsatisfiability.

We note that the converse of this theorem is not true, that is, there exist graphs $G \notin \mathcal{U}$ and $H \in \mathcal{U}$ such that G is a subgraph of H. For example, the triangle graph is not in $\mathcal{U}$ (as shown in Remark 7) but  $\in \mathcal{U}$ (as shown in the proof of Lemma 16).

Corollary 5. *Let G be a topological minor of a simple graph H. If $G \in \mathcal{U}$, then $H \in \mathcal{U}$.*

Proof. By definition of topological minors, some subdivision G' of G is a subgraph of H. Since $G \in \mathcal{U}$, we conclude that $G' \in \mathcal{U}$ from Theorem 3. By Remark 4, we conclude $H \in \mathcal{U}$. □

We note here that the converse of Corollary 5 is not true for the same reason that the converse of Remark 4 is not. Corollary 5 motivates the following definition — a graph G is a *minimal unsatisfiability graph* if both of the following conditions hold

(a) $G \in \mathcal{U}$,
(b) for every proper topological minor G' of G, we have that $G' \notin \mathcal{U}$.

Furthermore, a set M of minimal unsatisfiability graphs is *complete* if every graph in $\mathcal{U}$ has a topological minor in M.

Once we know that minimal unsatisfiability graphs exist, we can form a set M by constructing the union of all sets of minimal unsatisfiability graphs. This set is complete because each graph in $\mathcal{U}$ will have some element of M as a topological minor — if not we simply add to M a minimal unsatisfiability graph that is a topological minor of this graph. Further, such a complete set must be unique because if not, then there is some minimal unsatisfiability graph G in complete set M' that is not in M. This would result in G having a proper topological minor in $\mathcal{U}$, violating the minimality of G. After proving the following result about the relation between edge-contraction and graphs in $\mathcal{U}$, the remainder of this paper is dedicated to finding this unique complete set of minimal unsatisfiability graphs.

Theorem 6. *Let G and H be simple graphs such that G can be obtained via a series of edge-contractions at edges of H not contained in triangles. If $H \in \mathcal{U}$, then $G \in \mathcal{U}$.*

Proof. If is enough to prove the theorem for the case when G can be obtained from H via a single edge-contraction, say at the edge (u, v) not contained in any triangles in H. We label the new vertex in G formed by the merger of u and v by w.

Suppose that $H \in \mathcal{U}$. Then, there exists an unsatisfiable simple 2-CNF S_H, supported on H. Without loss of generality, we can write

$$S_H = S_1 \wedge (u \vee v)$$

where S_1 is a simple 2-CNF not containing any (u, v)-clauses. If S_H is not in this form, we can exchange u with $\neg u$ and/or v with $\neg v$ to make the (u, v)-clause in S_H positive in both variables.

One can further factor S_1 such that

$$S_H = S_2 \wedge \left(\bigwedge_{a\in A} a \vee u\right) \wedge \left(\bigwedge_{b\in B} b \vee \neg u\right) \wedge (u \vee v),$$

where S_2 is a simple 2-CNF not containing any clauses incident on the vertex u, and the sets A and B are disjoint subsets of $(V \cup \neg V) - \{u, \neg u\}$. The sets A and B are disjoint because S_H is a simple 2-CNF.

Next, we factor S_2 such that

$$S_H = S_3 \wedge \left(\bigwedge_{a\in A} a \vee u\right) \wedge \left(\bigwedge_{b\in B} b \vee \neg u\right) \wedge \left(\bigwedge_{c\in C} c \vee v\right) \wedge \left(\bigwedge_{d\in D} d \vee \neg v\right) \wedge (u \vee v),$$

where S_3 is a simple 2-CNF not containing any clauses incident on vertices u or v, and the sets A, B, C, D are pairwise-disjoint subsets of $(V \cup \neg V) - \{u, \neg u, v, \neg v\}$. The sets C and D are disjoint because S_H is a simple 2-CNF. The sets A and C are disjoint since the edge (u, v) is not contained in a triangle. Disjointness of other pairs of sets follows similarly.

We then choose

$$S_G = S \wedge \left(\bigwedge_{x\in A\cup D} x \vee w\right) \wedge \left(\bigwedge_{y\in B\cup C} y \vee \neg w\right),$$

and note that S_G is a simple 2-CNF supported on G. The pairs of sets (A, D) and (B, C) being disjoint imply that no clauses are lost in the process of edge-contraction. The pairs (A, B), (A, C), (B, D) and (C, D) being disjoint imply that S_G is simple.

We prove that S_G is unsatisfiable by showing that from any truth assignment that satisfies S_G, we can obtain a truth assignment satisfying S_H, leading to a contradiction.

Given a truth assignment for S_G, we can extend it to a truth assignment for S_H by setting $[u := w]$ and $[v := \neg w]$. We have

$$S_H[u := w][v := \neg w] \;=\; S \wedge \left(\bigwedge_{x\in A\cup D} x \vee w\right) \wedge \left(\bigwedge_{y\in B\cup C} y \vee \neg w\right) \;=\; S_G,$$

is satisfiable. This contradicts the unsatisfiability of S_H. We conclude that S_G is unsatisfiable, and hence that $G \in \mathcal{U}$. □

The converse of this theorem is not true, that is, there does exist a graph $G \in \mathcal{U}$ that can be obtained via edge-contractions of a graph $H \notin \mathcal{U}$ at edges not contained in triangles of H. For example, consider the graphs $G =$  and $H =$  . The graph G is in $\mathcal{U}$ (as shown in Lemma 16) and can be obtained from an edge-contraction from H. However, the graph H is not in $\mathcal{U}$ (implied by Theorem 6 and Lemma 8 when combined with the fact that H can be reduced to $K_4 - e$ via other edge-contractions).

6. Graph families that are not in $\mathcal{U}$

Remark 7. We observe that the cycle graph C_n is not in $\mathcal{U}$ for any $n \geq 3$. As proof, we note that $C_3 \notin \mathcal{U}$ since we can simply enumerate all CNFs supported on C_3 and check that each CNF is satisfiable. The graph C_n is homeomorphic to C_3 for every $n \geq 3$. and Theorem 3 thus implies that $C_n \notin \mathcal{U}$.

Lemma 8. *Let K_4 denote the complete graph on four vertices. Let $K_4 - e$ denote the graph obtained by deleting a single edge e from K_4. The graph $K_4 - e$ is not in $\mathcal{U}$.*

Proof. We enumerate the vertex set of K_4 as $\{a, b, c, d\}$. Let e denote the edge $(c, d) \in K_4$. Every 2-CNF S supported on $K_4 - e$ can be written either in the form

$$S = (a \vee b) \wedge (a \vee c) \wedge (a \vee d) \wedge (b, c)\text{-clause} \wedge (b, d)\text{-clause}, \quad \text{or}$$

$$S = (a \vee b) \wedge (a \vee c) \wedge (\neg a \vee d) \wedge (b, c)\text{-clause} \wedge (b, d)\text{-clause}.$$

If S is not already in the desired form, then we can interchange each variable with its negation till it is.

In the first case, by setting $[a := \top]$ we obtain

$$S[a := \top] = (b, c)\text{-clause} \wedge (b, d)\text{-clause}.$$

This can be satisfied by making appropriate assignments for c and d so that they satisfy each of the clauses. In the second case, we can set $[a := \top]$ to obtain

$$S[a := \top] = d \wedge (b, c)\text{-clause} \wedge (b, d)\text{-clause}.$$

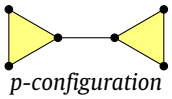
This resulting CNF can be satisfied by setting $[d := \top]$, by choosing an assignment for b that would satisfy the (b, d)-clause and by then choosing an assignment for c that would satisfy the (b, c)-clause. We conclude that S is always satisfiable, and therefore, the graph $K_4 - e$ is not in $\mathcal{U}$. □

Remark 9. We observe that tree graphs are not in $\mathcal{U}$ because every tree graph can be reduced via edge-contractions to a single-vertex graph, which is clearly not in $\mathcal{U}$.

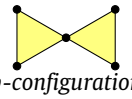
7. Structure of graphs with two or three cycles

In this section we prove four lemmas about the structure of graphs that have either two or three cycles. These structural results are needed for proving the results in Section 8, including the main result of this paper (Theorem 18).

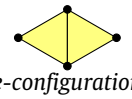
Lemma 10. *Every connected simple graph having two copies of C_3 as subgraphs has one of the following three graphs as a topological minor.*



p-configuration



v-configuration



e-configuration ($K_4 - e$)

Remark 11. For convenience, we label the three graphs as the p-configuration (p stands for path), the v-configuration (v stands for vertex) and e-configuration (e stands for edge) respectively. The two copies of C_3 have been filled in for easy visual identification.

Proof. We construct this list of topological minors from the bottom up. Two copies of C_3 can be put together to create a connected simple graph in three ways

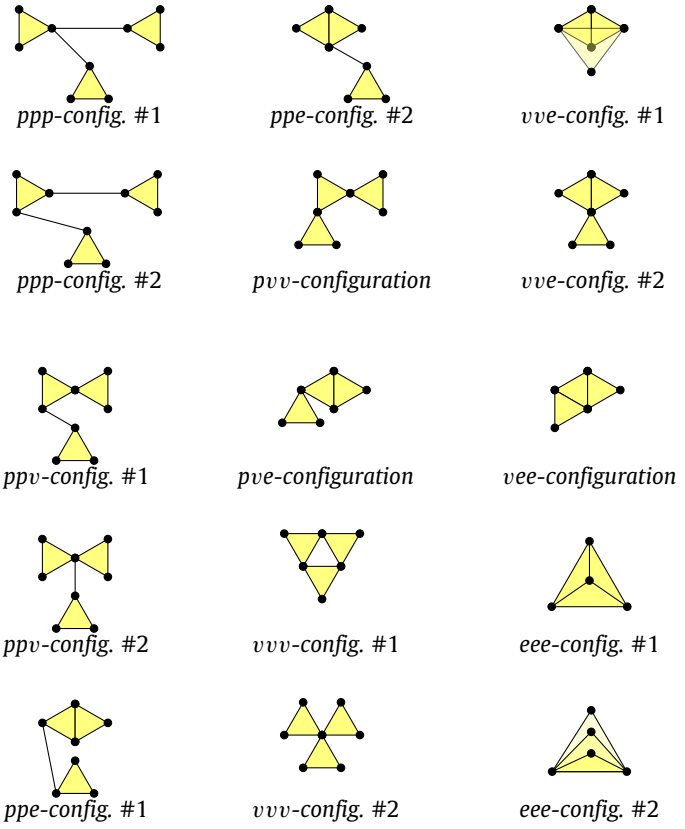
(a) the two copies of C_3 are disjoint,
(b) they share one vertex,
(c) they share two vertices (that is, they share an edge).

In the first case, since the graph is connected, the two copies of C_3 must be connected by one or more paths. Hence, the graph has the p-configuration as a topological minor. In the second case, the graph has the v-configuration as a subgraph and therefore, also as a topological minor. Similarly, in the third case, the graph has the e-configuration as a subgraph and therefore, also as topological minor. □

Lemma 12. *Every connected simple graph having at least two cycles has one of the three graphs in Lemma 10 as a topological minor.*

Proof. Let G be a connected simple graph with two or more cycles. If any two cycles share one or more edges, then $K_4 - e$ is a topological minor of G. If no pair of cycles shares any edges, but at least one pair shares a vertex, then the v-configuration is a topological minor of G. If no pair of cycles shares an edge or a vertex, then using the connectedness of G, we infer that there must be a path connecting vertices in every pair of cycles. Thus, in this case, the p-configuration is a topological minor of G. □

Lemma 13. *Every connected simple graph having three or more copies of C_3 as subgraphs has one of the following* 15 *graphs as a topological minor.*



Remark 14. Labels for each of the 15 graphs (like *ppp*-config. #1) are explained as part of the proof. The three copies of C_3 that we use for this labeling have been filled in for the sake of easy visual identification.

Proof. We arrive at this list of topological minors by constructing them from the bottom up. From the proof of Lemma 10, we know that every pair of C_3 can be joined in one of three ways. Graphs with three or more copies of C_3 will have at least $\binom{3}{2} = 3$ distinct pairs of C_3.

To enumerate all possible joinings of these three pairs of C_3, we form all three-letter words formed using the letters $\{p, v, e\}$, with repetition allowed, but order being irrelevant. For example, the word *ppe* would correspond to the family of graphs where the two pairs of C_3 are joined via the p-configuration, while the third pair of C_3 is joined via the e-configuration. We also note that some words correspond to multiple non-isomorphic configurations. We analyze each case separately below

(a) *ppp*-configuration. After joining the first pair of C_3 in a p-configuration, the third copy of C_3 can be added in two different non-isomorphic ways (such that the first and third, as well as the second and third are joined via the p-configuration). Thus, we obtain only two configurations in this case — *ppp*-config. #1 and *ppp*-config. #2.

(b) ppv-configuration. After joining the first pair of C_3 in a v-configuration, the third copy of C_3 can be added in two non-isomorphic ways. Thus, we obtain only two configurations in this case, namely ppv-config. #1 and ppv-config. #2.

(c) *ppe*-configuration. After joining the first pair of C_3 in an e-configuration, the third copy of C_3 can be added in two non-isomorphic ways. Thus, we obtain only two configurations in this case, namely *ppe*-config. #1 and *ppe*-config. #2.

(d) pvv-configuration. After joining the first pair of C_3 in a v-configuration, the third copy of C_3 can be added in only one way. Thus, we obtain only one configurations in this case, namely the pvv-configuration.

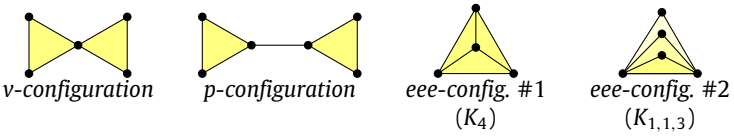
(e) pve-configuration. After joining the first pair of C_3 in an e-configuration, the third copy of C_3 can added in only one way. Thus, we obtain only one configurations in this case, namely the pve-configuration.

(f) *pee*-configuration. After joining the first pair of C_3 in an e-configuration, the third copy of C_3 cannot be added in a way that it is shares an edge with the first and is connected by a path to the second. Thus, we do not obtain configurations in this case.

(g) vvv-configuration. After joining the first pair of C_3 in an v-configuration, the third copy of C_3 can be added in two non-isomorphic ways. Thus, we obtain only two configurations in this case, namely vvv-config. #1 and vvv-config. #2.

(h) vve-configuration. After joining the first pair of C_3 in an e-configuration, the third copy of C_3 can be added in two non-isomorphic ways. Thus, we obtain only two configurations in this case, namely vve-config. #1 and vve-config. #2.
(i) vee-configuration. After joining the first pair of C_3 in an e-configuration, the third copy of C_3 can added in only one way. Thus, we obtain only one configurations in this case, namely the vee-configuration.
(j) eee-configuration. After joining the first pair of C_3 in an e-configuration, the third copy of C_3 can be added in two non-isomorphic ways. Thus, we obtain only two configurations in this case, namely eee-config. #1 and eee-config. #2. □

Lemma 15. *Every connected simple graph having three or more cycles has one of the following four graphs as a topological minor.*



Proof. It is enough to show that every graph listed in Lemma 13 has one of the four graphs listed above as a topological minor. The p-config. is a subgraph of both the ppp-configs., both the ppv-configs., both ppe-configs., the pvv-config., and the pve-config. Of the remaining configurations, both the vvv-configs., both the vve-configs., and the vee-config. contain the v-config. as a subgraph.

The first eee-configuration is isomorphic to K_4 while the second is isomorphic to $K_{1,1,3}$. Hence, all 15 configurations have at least one of the four graphs listed above as a topological minor. □

8. The complete set of minimal unsatisfiability graphs

We note that a simple graph G is in $\mathcal{U}$ if and only if some connected component of G is in $\mathcal{U}$. We proceed to make some more observations about the graph-family $\mathcal{U}$.

Lemma 16. *The four graphs in Lemma 15 are minimal unsatisfiability graphs.*

Proof. The following unsatisfiable 2-CNFs

$$S_1 = (a \vee b) \wedge (\neg a \vee c) \wedge (\neg b \vee c) \wedge (\neg c \vee d) \wedge (\neg c \vee e) \wedge (\neg d \vee \neg e),$$
$$S_2 = (a \vee b) \wedge (\neg a \vee c) \wedge (\neg b \vee c) \wedge (\neg c \vee d) \wedge (\neg d \vee e) \wedge (\neg d \vee f) \wedge (\neg e \vee \neg f),$$
$$S_3 = (a \vee b) \wedge (a \vee c) \wedge (\neg a \vee d) \wedge (\neg b \vee \neg c) \wedge (b \vee \neg d) \wedge (c \vee \neg d) \text{ and}$$
$$S_4 = (a \vee b) \wedge (\neg a \vee d) \wedge (b \vee c) \wedge (\neg b \vee d) \wedge (\neg b \vee e) \wedge (\neg c \vee \neg d) \wedge (\neg d \vee \neg e),$$

have as their associated graphs the four graphs listed in Lemma 15. Hence these graphs are in $\mathcal{U}$. Since the four listed graphs are not subdivisions of other graphs, to prove minimality, it suffices to show that every proper subgraph of these graphs is not in $\mathcal{U}$.

Every proper subgraph of the v-configuration is either a subgraph of  or  . Both subgraphs can be reduced via edge-contractions to C_3. By Remark 7 and Theorem 6, we conclude that neither subgraph is in $\mathcal{U}$.

Every proper subgraph of the p-configuration is either a subgraph of  ,  , or  . The first two subgraphs can be reduced via edge-contractions to proper subgraphs of the v-configuration and thus by Theorem 6, we conclude they are not in $\mathcal{U}$. The third subgraph  is not in $\mathcal{U}$ because of Remark 7.

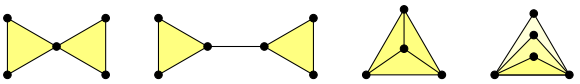
Since every proper subgraph of K_4 is also a subgraph of $K_4 - e$, using Remark 4 and Lemma 8, we conclude that none of the proper subgraphs of K_4 are in $\mathcal{U}$.

Every proper subgraph of $K_{1,1,3}$ is either a subgraph of  or $K_{2,3}$ (), both of which can be reduced via edge-contractions to $K_4 - e$. We conclude that none of the proper subgraphs of $K_{1,1,3}$ are in $\mathcal{U}$. □

Corollary 17. *Every connected simple graph having three or more cycles is in $\mathcal{U}$.*

Proof. From Lemma 15 we know that every connected simple graph having three or more cycles has one of the four graphs listed in that lemma as a topological minor. Since we proved in Lemma 16 that all four of these graphs are in $\mathcal{U}$, the result follows from Corollary 5. □

Theorem 18. *A simple graph G can support an unsatisfiable 2-CNF if and only if G contains at least one of the following four graphs as a topological minor:*



Proof. The four graphs belong to $\mathcal{U}$, as shown in Lemma 16. The "if" part of the theorem then follows from Corollary 5.

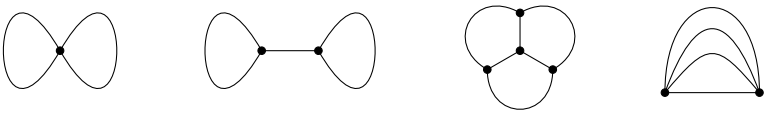
To prove the converse, we assume that $G \in \mathcal{U}$. We can also assume that G is connected (else simply restrict to the connected component of G that is in $\mathcal{U}$) and has minimum degree greater than 1 (else simply perform edge-contractions to get rid of pendant edges).

If G has three or more cycles, then the result follows from Lemma 15. If G is connected and has two cycles, then by Lemma 12 it follows that either G has the first graph, the second graph, or $K_4 - e$ as a topological minor. This last case is in fact not possible because we note that a graph with two cycles having $K_4 - e$ as a topological minor can in fact be reduced via edge-contractions to $K_4 - e$. Using Lemma 8 and the contrapositive of Theorem 6, this would imply that $G \notin \mathcal{U}$, a contradiction.

If G has only one cycle, then G can be reduced to C_3 by edge-contractions. However, we can derive contradicting results from Remark 7 and Theorem 6, showing that in fact G cannot have only one cycle.

Finally, if G has no cycles, then G is a tree graph. We know from Remark 9 that tree graphs are not in $\mathcal{U}$, which is a contradiction. Thus G cannot have zero cycles. □

Remark 19. The four graphs in Theorem 18 can be embedded into $\mathbb{R}^2$ (as drawn) and are homeomorphic to the following 1-dimensional cell complexes respectively



One can immediately identify unsatisfiable sentences supported on these cell complexes. For the first one, this is the (unreduced) 2-CNF $(x \vee x) \wedge (\neg x \vee \neg x)$. For the second cell complex, the (unreduced) 2-CNF $(x \vee x) \wedge (\neg x \vee \neg y) \wedge (y \vee y)$ is unsatisfiable. For the third, the cell complex is the same as K_4 and any unsatisfiable 2-CNF supported on K_4 will suffice. For the fourth cell complex, the (unreduced) 2-CNF $(x \vee y) \wedge (\neg x \vee y) \wedge (x \vee \neg y) \wedge (\neg x \vee \neg y)$ is unsatisfiable.

9. Conclusion

In our main result, Theorem 18, we showed that given a simple graph, all reduced sentences supported on it will be satisfiable if and only if four specific graphs are forbidden as topological minors of the original graph.

The decision problem of determining if a fixed graph is present as a topological minor can actually be resolved in polynomial time as a consequence of the graph minor theorem [11]. Since our set of forbidden topological minors is finite we can therefore guarantee that the task of searching for them can also be accomplished in polynomial time.

We emphasize however that as we have stated earlier, our goal in this paper is the characterization of unsatisfiable reduced 2-CNFs as opposed to questions of algorithmic efficiency for solving the satisfiability problem for 2-CNFs for which efficient algorithms already exist. We hope that the techniques developed in this paper when generalized to hypergraphs might shed light on the structure of unsatisfiable 3-CNFs.

Generalizing to 3-CNFs poses several new challenges. For example, the associated graphs for 3-CNFs are in fact multi-hypergraphs. Simple hypergraphs (without edge multiplicities) do not suffice. This also increases the number of cases that need to be analyzed. Furthermore, there are various choices when generalizing graph operations like edge-subdivision and edge-contraction to hypergraphs.

Acknowledgments

The authors thank Yuliy Baryshnikov for suggesting the problem of studying satisfiability from the viewpoint of the underlying structure of the sentences and for early discussions on the subject. This work was funded in part by Campus Research Board Award RB19150 from the University of Illinois at Urbana-Champaign, United States of America.

References

[1] B. Aspvall, M.F. Plass, R.E. Tarjan, A linear-time algorithm for testing the truth of certain quantified boolean formulas, Inform. Process. Lett. 8 (3) (1979) 121–123, http://dx.doi.org/10.1016/0020-0190(79)90002-4.
[2] L.W. Beineke, R.J. Wilson, J.L. Gross, T.W. Tucker, Topics in Topological Graph Theory, Cambridge University Press Cambridge, 2009.
[3] M. Bodirsky, M. Pinsker, Schaefer's theorem for graphs, CoRR abs/1011.2894 (2010) arXiv:1011.2894.
[4] H. Chen, A rendezvous of logic, complexity, and algebra, ACM Comput. Surv. 42 (1) (2009) 2:1–2:32, http://dx.doi.org/10.1145/1592451.1592453.

[5] S.A. Cook, The complexity of theorem-proving procedures, in: Proceedings of the Third Annual ACM Symposium on Theory of Computing, STOC '71, ACM, New York, NY, USA, 1971, pp. 151–158, http://dx.doi.org/10.1145/800157.805047.
[6] S. Even, A. Itai, A. Shamir, On the complexity of timetable and multicommodity flow problems, SIAM J. Comput. 5 (4) (1976) 691–703, http://dx.doi.org/10.1137/0205048.
[7] T. Feder, Network flow and 2-satisfiability, Algorithmica 11 (3) (1994) 291–319, http://dx.doi.org/10.1007/BF01240738.
[8] R.M. Karp, Reducibility among combinatorial problems, in: Complexity of Computer Computations, Springer, 1972, pp. 85–103.
[9] M.R. Krom, The decision problem for a class of first-order formulas in which all disjunctions are binary, Math. Logic Quart. 13 (1–2) (1967) 15–20, http://dx.doi.org/10.1002/malq.19670130104.
[10] L.A. Levin, Universal enumeration problems, Problemy Peredači Informacii 9 (3) (1973) 115–116.
[11] N. Robertson, P. Seymour, Graph minors. XIII. the disjoint paths problem, J. Combin. Theory Ser. B 63 (1) (1995) 65–110, http://dx.doi.org/10.1006/jctb.1995.1006.
[12] T.J. Schaefer, The complexity of satisfiability problems, in: Proceedings of the Tenth Annual ACM Symposium on Theory of Computing, STOC '78, ACM, New York, NY, USA, 1978, pp. 216–226, http://dx.doi.org/10.1145/800133.804350.