Theoretical Computer Science 99 (1992) 315-326
Elsevier

Note

Tseitin's formulas revisited

Jean-Denis Fouks

Université de Haute Alsace, Labo Math-info, 4 Rue des Frères Lumière, 68093 Mulhouse Cedex France

Communicated by M. Nivat
Received February 1991

Abstract

Fouks, J.D., Tseitin's formulas revisited, Theoretical Computer Science 99 (1992) 315-326.

G being a graph, we define its cyclomatic cohesion $\gamma(G)$. Then, using Tseitin's method (1970), we construct a contradictory formula $C(G)$ and prove our main theorem: *Every resolution of $C(G)$ contains, at least, $2^{\gamma(G)}$ distinct clauses*. A similar result was obtained by Urquhart (1987) with a different method valid only for a specific family of graphs.

0. Introduction

Hard examples for regular resolution (i.e. resolution where an eliminated boolean may not appear again) were proved first by Tseitin [10], who has introduced a method associating a formula with a graph.

The intractibility of general resolution was obtained first by Haken [7]. With a similar method, Urquhart has shown a specific family of Tseitin's formulas to be hard. A probabilistic result is also given by Chvatàl and Szemerédi [2].

But for a general graph, the complexity of resolution on the corresponding Tseitin's formula was still unknown. We link it with a cyclomatic property of the graph: the cyclomatic cohesion.

0.1. Definitions and notations

We consider a finite set $\underline{B}$ of "booleans". $\forall \underline{b} \in \underline{B}$, $(\underline{b}, +)$ $((\underline{b}, -))$ is called the *positive* (*negative*) *literal of support* $\underline{b}$ and is denoted by b (b'). A literal, positive or negative, is denoted by l, and the opposite one by l'.

A finite set of literals whose supports are distinct from one another is called a *clause*. We shall identify a clause with a partial map from $\underline{B}$ to $\{+,-\}$. For instance, the clause $\{1, 2', 3\}$ represents the map $\underline{1} \mapsto +, \underline{2} \mapsto -, \underline{3} \mapsto +$. The empty clause is denoted by $\wedge$.

Let c be a clause; the product of the signs of its literals is called the *signature* of c and is denoted by $\mathrm{sgn}(c)$. By convention, $\mathrm{sgn}(\wedge) = +$. The definition set of c is called the *support* of c and is denoted by $\underline{c}$. The set of literals $\{l, l' \in c\}$ is a clause denoted by c'. For instance, if $c = \{1, 2', 3\}$, then $\mathrm{sgn}(c) = -$, $\underline{c} = \{\underline{1}, \underline{2}, \underline{3}\}$ and $c' = \{1', 2, 3'\}$.

A finite set of clauses is called a *formula*.

A map from $\underline{B}$ to $\{+,-\}$ is called a *valuation* ($+$ may be seen as "true" and $-$ as "false"). A valuation is identified with a clause of support $\underline{B}$.

A clause c is said to be *satisfied* by a valuation v if $c' \not\subset v$. A valuation v is said to be a *solution* of a formula C if, $\forall c \in C$, c is satisfied by v. The set of solutions of C is denoted by $\mathrm{sol}(C)$; C is said to be *contradictory* if $\mathrm{sol}(C) = \emptyset$ and satisfiable otherwise.

If $|c_1 \cap c'_2| = 1$, then the clause $(c_1 - c'_2) \cup (c_2 - c'_1)$ is called the *resolvent* of c_1 and c_2. It is said to be obtained by *annihilating the support of* $c_1 \cap c'_2$ and is denoted $c_1 \bigtriangledown c_2$.

For instance, $\{1, 2', 3\} \bigtriangledown \{2, 3, 4\} = \{1, 3, 4\}$ and $\{1\} \bigtriangledown \{1'\} = \wedge$, but $\{1, 2', 3\}$ and $\{2, 3', 4\}$ do not define a resolvent.

Let C be a formula and (R, E) be a binary tree with root ra. If φ is a map from R to the set of clauses on $\underline{B}$ such that

- if r is a leaf then $\varphi(r)$ belongs to C,
- if r has two sons r_1 and r_2 then $\varphi(r)$ is the resolvent of $\varphi(r_1)$ and $\varphi(r_2)$,

then (R, E, φ) is said to be a *resolution* of C proving $\varphi(\mathrm{ra})$.

An example is shown in Fig. 1.

By Robinson's theorem [9], C is contradictory if and only if there exists a resolution of C proving $\wedge$.

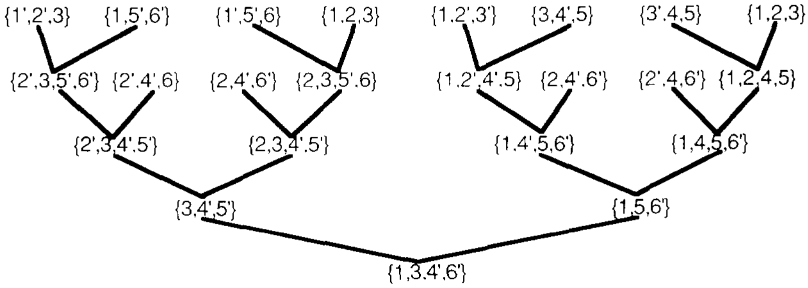

Fig. 1.

1. Resolution guide

1.1. Definitions

Let c be a clause on $\underline{B}$ and (S, A) a binary tree of height $|\underline{B}\setminus\underline{c}|$ with $2^{|\underline{B}\setminus\underline{c}|}$ leaves; c being identified with the root of (S, A), let α be a map from A to the set of literals on $\underline{B}\setminus\underline{c}$ such that

- $\forall a_1, a_2 \in A$, if a_1 and a_2 are in a same path of (S, A), then $\underline{\alpha(a_1)} \neq \underline{\alpha(a_2)}$;
- $\forall a_1, a_2 \in A$, if a_1 and a_2 have the same starting point, then $\alpha(a_1) = \alpha(a_2)'$.

Under these conditions, (S, A, α) is called a *resolution guide with goal* c on $\underline{B}$. By convention, if $\alpha(a)$ is a positive (negative) literal, a is said to be a left (right) edge. This convention will be applied to all figures.

For all $s \in S$, there exists a unique path $(s_1, \ldots, s_k)$ in (S, A) from $s_1 = \mathrm{sa}$ to $s_k = s$. The clause $c \cup \alpha\{(s_1, s_2), \ldots, (s_{k-1}, s_k)\}$ is called the *clause given by* s and is denoted by $c(s)$. In particular, $c(\mathrm{sa}) = c$.

Let d be a clause and s a vertex of (S, A). We say that d *eliminates* (*preserves*) s if d' is included (not included) in $c(s)$.

Lemma 1.1 (Resolution guide lemma). *Let* (R, E, φ) *be a resolution of a formula* C *proving a clause* c. *We can construct a part* $\Sigma = (S, A, \alpha)$ *of a resolution guide with goal* c' *canonically associated with* (R, E, φ). *It is called the* skeleton *of* (R, E, φ).

Proof. Initially, $S = \{\mathrm{sa}\}$, $c(\mathrm{sa}) = c'$ and an injective map ψ from S to R is defined by $\psi(\mathrm{sa}) = \mathrm{ra}$.

By hypothesis, during the construction, s, $c(s)$ and $r = \psi(s)$ are defined and are such that $\varphi(r) \subset c(s)'$.

Only three cases are possible:

Case 1: The clause $\varphi(\psi(s))$ is generated by annihilating $\underline{b}$ and $\underline{b} \notin \underline{c}(s)$. Then, we can construct two sons rss and lss of s and let $\alpha((s, \mathrm{rss})) = b'$ and $\alpha((s, \mathrm{lss})) = b$, so that $c(\mathrm{rss}) = c(s) \cup \{b'\}$ and $c(\mathrm{lss}) = c(s) \cup \{b\}$.

By the convention on the sons rsr and lsr of r, $b \in \varphi(\mathrm{rsr})$ and $b' \in \varphi(\mathrm{lsr})$. So $\varphi(\mathrm{rsr}) \subset \varphi(r) \cup \{b\} \subset c(\mathrm{rss})'$ ($\varphi(\mathrm{lsr}) \subset c(\mathrm{lss})'$) and we can let $\psi(\mathrm{rss}) = \mathrm{rsr}$ and $\psi(\mathrm{lss}) = \mathrm{lsr}$.

Case 2: The clause $\varphi(\psi(s))$ is generated by annihilating $\underline{b}$ and $\underline{b} \in \underline{c}(s)$. Let us suppose that $b \in c(s)$. We obtain $\varphi(\mathrm{lsr}) \subset \varphi(r) \cup \{b'\} \subset c(s)' \cup \{b'\} = c(s)'$; in that case, we modify ψ and let $\psi(s) = \mathrm{lsr}$. If $b' \in c(s)$, we let $\psi(s) = \mathrm{rsr}$.

Case 3: The clause $\varphi(\psi(s))$ belongs to C.

In cases 1 and 2, the construction goes on recursively. In the third case, it comes to an end. As (R, E, φ) is finite, the algorithm will stop. $\square$

The two conditions on α, given above, are respected. So, the skeleton of (R, E, φ) is part of at least one resolution guide with goal c'. Note that every leaf l of Σ is eliminated by the clause $\varphi(\psi(l))$.

For instance, the skeleton of resolution in Fig. 1 is shown in Fig. 2.

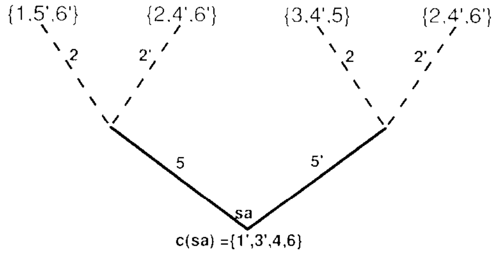

Fig. 2. The clause indicated above a leaf l is $\varphi(\psi(l))$.

Lemma 1.2 (Autumn lemma). *Let c be a clause on $\underline{B}$, C a formula and let Γ be a resolution guide with goal c'. If there exists a resolution (R, E, φ) of C proving c, then every leaf of Γ is eliminated by at least one clause in C.*

Proof. (i) Let Γ contain the skeleton Σ. Every leaf l of Γ is a descendant of some leaf l_0 of Σ and, so $c(l)$ contains $c(l_0)$. By the preceding proof, $\exists d\in C, d' \subset c(l_0)$, so that $d' \subset c(l)$ and l is eliminated by d.

(ii) Let Γ be any resolution guide with goal c' and let L be the set of leaves in Γ. $\{c(l), l\in L\}$ is the set of valuations containing c' and so does not depend on Γ. Point (i) shows us that, $\forall l\in L$, $\exists d\in C$, $d' \subset c(l)$ and l is eliminated.

2. Formula defined by a graph

2.1. *Alternated graphs*

The graphs we shall consider have a finite set X of "vertices" and a family $U = \{u_i\}_{i\in I}$ of "edges" which are unordered pairs (x, y) of vertices.

$\forall x\in X$, the set of edges (loops) containing x is called the *boundary* (*interior boundary*) of x and is denoted by $f_G(x)$ ($\mathrm{fi}_G(x)$); $f_G(x) \setminus \mathrm{fi}_G(x)$ is called the *exterior boundary* of x and is denoted by $\mathrm{fe}_G(x)$. If G is obvious, we write $f(x)$, $\mathrm{fi}(x)$ and $\mathrm{fe}(x)$.

The set of connected components of G is denoted by $\mathrm{co}(G)$, and the one containing a given vertex x is denoted by $\mathrm{co}(x, G)$.

$G = (X, U, \sigma)$ is called an *alternated graph* on $\underline{B}$ if

- (X, U) is a graph where $\underline{B}$ is the set of indices of U,
- σ is a map from X to $\{+, -\}$, called the sign.

In graphic representations, vertices of sign $+$ $(-)$ will be marked by ○ (●). An edge u_b will be frequently identified with its index $\underline{b}$.

Let $H = (Y, V)$ be a subgraph of G. The product $\prod_{y\in Y} \sigma(y)$ is called the *sign* of H and is denoted by $\sigma(H)$; H is said to be *even* if $\sigma(H)+$, and *odd* otherwise.

Let $G = (X, U, \sigma)$ be an alternated graph and x a vertex of G. The set of clauses $\{c = d_1 \cup d_2,\ d_1 = fe(x)$ and $\mathrm{sgn}(d_1) = -\sigma(x),\ \underline{d_2} = \underline{fi}(x)\}$ is said to be *associated with* x and is denoted by $G(x)$.

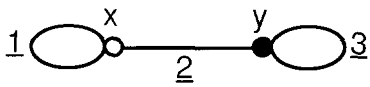

Fig. 3.

Example. Let G be the graph shown in Fig. 3.

The set $G(x)$ is $\{\{1, 2'\}, \{1', 2'\}\}$ and the set $G(y)$ is $\{\{2, 3\}, \{2, 3'\}\}$.

Let G be an alternated graph on $\underline{B}$. The set $\bigcup_{x\in X} G(x)$ is called the *formula defined* by G and is denoted by $C(G)$.

2.2. *Transformations of graphs*

Let l be a literal on $\underline{B}$, the map $\sigma\backslash l$ from X to $\{+, -\}$ is defined as follows:

- If l is positive or if $\underline{l}$ is a loop, then $\sigma\backslash l=\sigma$.
- If l is negative and if $\underline{l}=(x, y)(x\neq y)$, then

$$(\sigma\backslash l)(x)=-\sigma(x), \qquad (\sigma\backslash l)(y)=-\sigma(y) \quad \text{and} \quad \forall z\notin(x, y), (\sigma\backslash l)(z)=\sigma(z).$$

The alternated graph $(X, U\backslash\{\underline{l}\}, \sigma\backslash l)$ is said to be obtained by *deletion* of l in G and is denoted by $G\backslash l$.

As, $\forall l_1, l_2, (\sigma\backslash l_1)\backslash l_2=(\sigma\backslash l_2)\backslash l_1$, we can naturally define $\sigma\backslash c$ for each clause c on $\underline{B}$. The alternated graph $(X, U\backslash\underline{c}, \sigma\backslash c)$ is said to be obtained by *deletion* of c in G and is denoted by $G\backslash c$. The graph $(X, U\backslash\underline{c})$ is denoted by $G\backslash\underline{c}$.

Let c be a clause on $\underline{B}$ and V be the subset of U indexed by $\underline{c}$. For each edge $v_b=(x, y)_b\in V$, let $\bar{v}_b=(\text{co}(x, G\backslash c), \text{co}(y, G\backslash c))_j$ and $\bar{V}$ be the family $\{\bar{v}_b\}_{b\in c}$. We define a sign σ/c on $\text{co}(G\backslash c)$ as follows:

$$\forall x\in X, \quad (\sigma/c)(\text{co}(x, G\backslash c))=\prod_{y\in\text{co}(x, G\backslash c)}(\sigma\backslash c)(y).$$

The alternated graph $(\text{co}(G\backslash c), \bar{V}, \sigma/c)$ is called the *quotient* of G by $G\backslash c$ and is denoted by G/c. The graph $(\text{co}(G\backslash c), V)$ is denoted by $G/\underline{c}$.

Example. Let G be given by Fig. 4 and $c=\{4, 11, 12, 14', 19, 22, 31\}$.

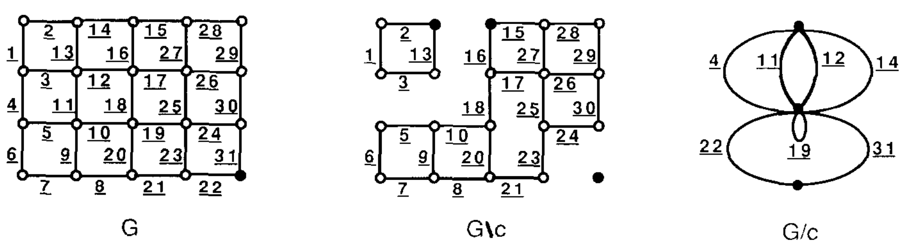

Fig. 4.

2.3. Cyclomatic notions

An edge u is called a cycle edge if $|\mathrm{co}(G\setminus\{u\})|=|\mathrm{co}(G)|$, and an isthmus otherwise.

The number $\nu(G)=|U|-|X|+|\mathrm{co}(G)|$ is called the *cyclomatic number* of G. For every graph G, $\nu(G)$ is a nonnegative integer (see [1]).

For $x\in X$, $\nu(G/f(x))$ is called the *cyclomatic degree* of x in G and is denoted by $\delta_G(x)$ or $\delta(x)$, if G is obvious.

The number $\delta(G)=\max\{\delta(x), x\in X\}$ is called the *cyclomatic degree* of G.

Let ω be a permutation of $\underline{B}$. The sequence $G_0=\mathrm{co}(G)$, $G_1=G/\{\omega(\underline{1})\},\ldots,$ $G_i=G/\omega[\underline{1},\ldots,\underline{i}],\ldots, G_{|\underline{U}|}=G$, is called the *sequence according to ω of the quotients of G*.

The number $\gamma(\omega)=\max\{\delta(G_{\underline{i}}), \underline{i}\in\underline{B}\}$ is called the *cyclomatic cohesion* of ω. As $G_{|\underline{U}|}=G$, we have $\delta(G)\leqslant\gamma(\omega)$ for every permutation ω of $\underline{B}$.

The number $\gamma(G)=\min\{\gamma(\omega), \omega$ describing the set of permutations of $\underline{B}\}$ is called the *cyclomatic cohesion* of G. For every graph, $\delta(G)\leqslant\gamma(G)$.

Theorem 2.1 (Satisfiability theorem). *If all the connected components of G are even then $|\mathrm{sol}(C(G))|=2^{\nu(G)}$. Otherwise, $C(G)$ is contradictory. (See [5].)*

2.4. Towers and keeps

A vertex $\mathrm{co}(x, G\setminus\underline{c})$ in $G/\underline{c}$, such that $\delta(\mathrm{co}(x, G\setminus\underline{c}))\geqslant\gamma(G)$, is called a *tower* of G with respect to $\underline{c}$.

Let ω be a permutation of $\underline{B}$. If there is one $\underline{i}\in\underline{B}$ such that $\underline{c}=\omega[\underline{1},\ldots,\underline{i}]$, then ω is said to be an *extension* of $\underline{c}$, and a tower with respect to $\underline{c}$ is called a tower of ω with index $\underline{i}$.

If, for any extension ω of $\underline{c}$, $\mathrm{co}(x, G\setminus\underline{c})$ contains a tower of ω, then $\mathrm{co}(x, G\setminus\underline{c})$ is called a *rampart* of G.

If $\mathrm{co}(x, G\setminus\underline{c})$ is a tower and if, $\forall\underline{d}\subset\underline{c}$, $\mathrm{co}(x, G\setminus\underline{d})$ is a rampart, then $\mathrm{co}(x, G\setminus\underline{c})$ is called a *keep* of G.

Lemma 2.2 (The rampart lemma). *Let $\mathrm{co}(x, G\setminus\underline{c})$ be a rampart, ω an extension of $\underline{c}$ and k the least index for a tower of ω contained in $\mathrm{co}(x, G\setminus\underline{c})$.*

Then, $\forall i\in[|\underline{c}|,\ldots,k]$, at least one among the components of $\mathrm{co}(G/\omega[\underline{1},\ldots,\underline{i}])$ contained in $\mathrm{co}(x, G\setminus\underline{c})$ is a rampart.

Proof. We note first that, $\forall y\in X, \forall i, i', i'>i$, $\mathrm{co}(y, G\setminus\omega[\underline{1},\ldots,\underline{i'}])\subset\mathrm{co}(y, G\setminus\omega[\underline{1},\ldots,\underline{i}])$. Let us now suppose that, for some $i>|\underline{c}|$, none of the components Y of $G/\omega[\underline{1},\ldots,\underline{i}]$, contained in $Z=\mathrm{co}(x, G\setminus\omega[\underline{1},\ldots,\underline{i}])$, is a rampart.

For any Y contained in Z, there exists, by hypothesis, a permutation ω_Y of its edges such that any extension of $\omega[\underline{1},\ldots,\underline{i}]$ using ω_Y would have no tower in Y.

As Z is a disjoint union of components Y, there exists an extension ω' of $\omega[\underline{1},\ldots,\underline{i}]$ using $\{\omega_Y, Y\subseteq Z\}$ which has no tower contained in Z.

This leads us to a contradiction. □

Theorem 2.3 (The keep theorem). *Let $G=(X, U)$ be a connected graph and ω a permutation of $\underline{B}$. Then there exists at least one tower of ω which is a keep.*

Proof. (i) Let us first see that there is a sequence of ramparts, the last one being a tower.

For all $x \in X$, $\mathrm{co}(x, G\backslash\emptyset)=G$ is clearly a rampart.

Let us suppose that $\mathrm{co}(x, G\backslash\omega[\underline{1},\ldots,\underline{i}])$ is a rampart but not a tower.

Three cases are possible:

Case 1: $\omega(\underline{i+1})\notin\mathrm{co}(x, G\backslash\omega[\underline{1},\ldots,\underline{i}])$; then we have that $\mathrm{co}(x, G\backslash\omega[\underline{1},\ldots,\underline{i}])=\mathrm{co}(x, G\backslash\omega[\underline{1},\ldots,\underline{i+1}])$.

Case 2: $\omega(\underline{i+1})$ is an isthmus of $\mathrm{co}(x, G\backslash\omega[\underline{1},\ldots,\underline{i}])$. In that case, $\mathrm{co}(x, G\backslash\omega[\underline{1},\ldots,\underline{i}])$ is divided into two components, at least one being a rampart by the rampart lemma.

Case 3: $\omega(\underline{i+1})$ is a cycle of $\mathrm{co}(x, G\backslash\omega[\underline{1},\ldots,\underline{i}])$. In that case, a new loop on $\mathrm{co}(x, G\backslash\omega[\underline{1},\ldots,\underline{i}])$ is created and $\delta(\mathrm{co}(x, G\backslash\omega[\underline{1},\ldots,\underline{i+1}]))=\delta(\mathrm{co}(x, G\backslash\omega[\underline{1},\ldots,\underline{i}]))+1$; $\mathrm{co}(x, G\backslash\omega[\underline{1},\ldots,\underline{i+1}])$ is a rampart and may be a tower.

In case 1, the induction hypothesis is still true at stage $i+1$.

In case 2, the induction hypothesis is verified at stage $i+1$ if we replace x by one of the rampart's vertices.

In case 3, if a tower is not obtained, the induction hypothesis is still true.

So, a tower of ω will be obtained.

(ii) *This tower is a keep.* Let $\underline{i}$ be its index and $\underline{d}\subset\omega[\underline{1},\ldots,\underline{i}]$. For any extension ω' of $\underline{d}$, let k be the least integer such that $\omega'[\underline{1},\ldots,\underline{k}]\supset\omega[\underline{1},\ldots,\underline{i-1}]$. Only two cases are possible:

Case 1: The component $\mathrm{co}(x, G\backslash\underline{d})$ contains a tower of ω' with index $<\underline{k}$.

Case 2: The component $\mathrm{co}(x, G\backslash\omega[\underline{1},\ldots,\underline{i-1}])$ being a rampart, one of its components in $\mathrm{co}(G\backslash\omega'[\underline{1},\ldots,\underline{k}])$ is also a rampart by the rampart lemma.

In both cases, $\mathrm{co}(x, G\backslash\underline{d})$ contains a tower of ω'. So, $\mathrm{co}(x, G\backslash\underline{d})$ is a rampart and $\mathrm{co}(x, G\backslash\underline{c})$ is a keep. □

2.5. *Simple clauses*

In that part, $G(X, U, \sigma)$ is an odd connected alternated graph on a set $\underline{B}$ of booleans.

Let c be a clause. An odd connected component of $G\backslash c$ is called a *territory* of c.

Remark. If $c\in G(x)$, then $(\{x\}, \emptyset)$ is a territory of c but this territory may not be the only one.

If G is given by Fig. 5, then the clause $\{1, 2', 3, 4'\}$ belongs to $G(x)$ and admits three territories.

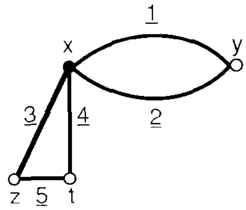

Fig. 5.

A clause c which admits only one territory is said to be *simple*. Its only territory is denoted by $\mathrm{ter}(c)$.

Remark. Every subclause of a simple clause is simple.

Theorem 2.4 (Inclusion theorem). *Let x be a vertex and $\tau_x(G)$ be the graph obtained from G by reversing $\sigma(x)$.*

(a) *A valuation v belongs to* $\mathrm{sol}(C(\tau_x(G)))$ *if and only if v' is a simple clause with territory* $(\{x\}, \emptyset)$.

(b) *If c is a simple clause and if $x \in \mathrm{ter}(c)$, then c' is included in $2^{\nu(G-c)}$ solutions of* $C(\tau_x(G))$.

(c) *The subset of simple clauses of $C(G)$ is the only minimal contradictory subformula.*

(d) $\forall x \in X$, *$G(x)$ contains exactly $2^{\delta(x)}$ simple clauses.*

The proofs may be found in [5].

3. Resolution of $C(G)$

3.1. *Climbing procedure*

Let G be an odd connected alternated graph on $\underline{B}$, let c be a simple clause whose territory $\mathrm{ter}(c)$ is a rampart, and (R, E, φ) be a resolution of $C(G)$ proving c specified as follows:

$R = \{1, \ldots, n\}$ and $\mathrm{ra} = 1$ is the root.

$\forall r \in R$, $\mathrm{lsr}[r]$ is the left son of r if it exists and 0 otherwise.

$\mathrm{rsr}[r]$ is the right son of r if it exists and 0 otherwise.

$\varphi[r]$ is the clause contained in r.

$\mathrm{li}[r]$ is the only literal of $\varphi[\mathrm{rsr}[r]] \setminus \varphi[r]$ if r is not a leaf, and 0 otherwise.

By convention on rsr, $\mathrm{li}[r]$ is a positive literal.

Results. A set V of booleans (edges) and a set SC of simple clauses.

Recursive procedure Treat(s, d)

```
Treat(s, d);
  b:=li[ψ[s]];
  if cond=false
    then
      {V:=V∪[b];
       if a vertex x̄ of G/V, contained in ter(c), is a keep
         then cond:=true;
      };
    else if b∉V then b:=0;
  while (b∈d) do
    {if b∈d then ψ[s]:=lsr[ψ[s]] else ψ[s]:=rsr[ψ[s]];
     b:=li[ψ[s]];
    };
  if b≠0
    then
      {e:=e+1; /* create the left son of s*/
       β[e]:=b; lss[s]:=e; pred[e]:=s; ψ[e]:=lsr[ψ[s]]; Treat(e, d∪{b});
       e:=e+1; /* create the right son of s*/
       β[e]:=b'; rss[s]:=e; pred[e]:=s; ψ[e]:=rsr[ψ[s]]; Treat(e, d∪{b'});
      };
    else
  if (cond=true and φ(ψ(s)) is simple and x∈ter(φ[ψ[s]])) then
    SC:=SC∪[φ[ψ[s]]];
```

Climb is defined as

$$\{\psi[1]:=1;\ e:=1;\ \beta[1]:=0;\ V:=\underline{c};\ \mathrm{cond}:=\mathrm{false};\ \mathrm{SC}:=\emptyset;\ \mathrm{Treat}(1, c')\}.$$

Lemma 3.1 (Iteration lemma). *The set* SC *given by Climb is a nonempty set of simple clauses whose territories are ramparts.*

Proof. Let $\Sigma=(S, A, \alpha)$ be the skeleton of (R, E, φ). By the inclusion theorem and the autumn lemma, we prove that $\varphi(R)$ must contain at least one clause in $\{G(x), x\in\mathrm{ter}(c)\}$ and, hence, that $\alpha(A)$ contains every literal whose support is an edge in $\mathrm{ter}(c)$.

As $\mathrm{ter}(c)$ is a rampart and V increases edge by edge, by the keep theorem, at a stage of the construction, V is such that there exists a vertex $\bar{x}$ of G/V, contained in $\mathrm{ter}(c)$, which is a keep.

Let Γ be a resolution guide with goal c' on V and Φ be the set of leaves in the binary tree constructed by Climb. By the autumn lemma, every leaf of Γ is eliminated by a clause in $\varphi(\psi(\Phi))$.

Let DL be the set of leaves l in Γ, for which $c(l)'$ is simple and $\mathrm{ter}(c(l)')=\bar{x}$. As a subclause of a simple clause is simple, $\forall l\in\mathrm{DL}$, l is eliminated by a simple clause whose territory, containing $\bar{x}$, is a rampart.

3.2. The fall-of-the-keep algorithm

Data An odd connected and alternated graph on $\underline{B}$ and a resolution (R, E, φ) of $C(G)$ proving $\wedge$ (which is a simple clause whose territory G is a rampart).

Algorithm

{FC:=∅; Applied Climb to (R, E, φ); FC:=SC;
While ($\exists c \in$ FC and ter(c) is not a keep) **do**
{applied Climb to the subtree of (R, E, φ) proving c; FC:=(FC $\setminus$ [c]) $\cup$ SC;
};
};

Lemma 3.2 (Counting lemma). *The fall-of-the-keep algorithm stops and* $\sum_{c\in \mathrm{FC}} 2^{\nu(G\setminus c)-\nu(G)} \geqslant 1$.

Proof. Every clause $c \in$ FC is simple by construction and ter(c) containing a keep $\bar{x}$ is a rampart. If ter(c) $\neq \bar{x}$, then it contains more than one vertex of G and so $c \notin C(G)$. So c is the root of a subresolution of (R, E, φ) and we can apply Climb to this data; (R, E, φ) being finite, the process will stop because it works from the root to the leaves.

We construct a resolution guide Γ associated with the fall-of-the-keep algorithm in the following way:

At the beginning, Γ is reduced to a single distinguished leaf of weight 1.

At the current stage, let us suppose that a distinguished leaf l is eliminated by a clause c in SC. If Climb is applied to this clause c, then a subtree of root l is added to Γ and the weight of l is divided into the new distinguished leaves.

Let c be a clause. We denote by $\nu(c)$ the number $\nu(G)-\nu(G\setminus c)$ and prove that the total weight eliminated by c in Γ is less than $2^{-\nu(c)}$.

The weight eliminated by $\wedge$ is 1.

Let b be a boolean not belonging to $\underline{c}$ and associated with a cycle edge of $G\setminus c$. Extending Γ, if necessary, we can suppose that, for every distinguished leaf l, $\overline{c(l)} \supset \underline{c} \cup \underline{b}$. Then, by the inclusion theorem, the total weight eliminated by $c \cup \{b\}$ (or $c \cup \overline{b'}\}$) is exactly half of the total weight eliminated by c. This weight is reduced if, in fact, $\underline{c} \cup \underline{b} \not\subset \overline{c(l)}$ for some distinguished leaves l of Γ.

So, by induction, the weight eliminated by c is less than $2^{-\nu(c)}$.

As all the leaves in Γ are eliminated, we conclude that $\sum_{c\in \mathrm{FC}} 2^{-\nu(c)} \geqslant 1$.

Theorem 3.3 (Fundamental theorem). *Let G be an odd connected alternated graph on $\underline{B}$. If (R, E, φ) is a resolution of $C(G)$ proving $\wedge$, then* $|\varphi(R)| \geqslant 2^{\gamma(G)}$.

Proof. The fall-of-the-keep algorithm gives us a set FC of simple clauses belonging to $\varphi(R)$. For all $c \in$ FC, ter(c) is a keep and, consequently, $\nu(G)-\nu(G\setminus c) \geqslant \gamma(G)$.

By the counting lemma, $\sum_{c\in \mathrm{FC}} 2^{\nu(G\setminus c)-\nu(G)} \geqslant 1$. So, $|\varphi(R)| \geqslant |\mathrm{FC}| \geqslant 2^{\gamma(G)}$.

Corollary 3.4. *Let G be an odd alternated graph. An optimal resolution of $C(G)$ is obtained by the Davis–Putnam procedure applied with an optimal order on the booleans.*

3.3. *The Margulis graphs*

Margulis [8] defines the following family of graphs:

$\forall n \in \mathbb{N}^*$, let X'_n and X''_n be disjoint sets in one-to-one mapping with $Z/nZ \times Z/nZ$ and let $X_n = X'_n \cup X''_n$. Let U_n be obtained by joining each element (x, y) in X'_n to the following elements in X''_n: (x, y), $(x+1, y)$, $(x, y+1)$, $(x, x+y)$ and $(-y, x)$.

(X_n, U_n) is called the *Margulis graph* with index n and is denoted by Mar_n. Galil [6] proves the following result (The Galil–Margulis theorem):

There exists a real $k > 0$ such that, $\forall n \in \mathbb{N}^*$, $\gamma(\mathrm{Mar}_n) \geqslant kn^2$.

We conclude: *For infinitely many $n \geqslant 1$, there are unsatisfiable formulas G_n, over $O(n)$ variables which contain $O(n)$ clauses, such that every resolution of G_n contains at least 2^n distinct clauses.*

4. Conclusion

Resolution, whose complexity is exponentially growing, is intractable on the formulas defined with the Margulis graphs.

Our result generalizes, by applying to any graph, the result obtained by Urquhart [11]. So, we obtain not only an exponentially growing lower bound but a counting method for the number of distinct clauses and we can conclude that the most efficient resolution of a Tseitin formula is given by the Davis–Putnam procedure [3] with an optimal order on the booleans.

Acknowledgment

I thank Professor J.C. Spehner for many helpful discussions and suggestions during this work.

References

[1] C. Berge, *Theorie des Graphes et ses Applications* (Dunod, Paris, 1958).

[2] V. Chvatal and E. Szemeredi, Many hard examples for resolution, *J. ACM* **35** (4) (1988) 759–768.

[3] M. Davis and H. Putnam, A computing procedure for quantification theory, *J. ACM* **1** (1960) 201–215.

[4] J.D. Fouks, Sur la résolution du problème de satisfiabilité, Doctoral Thesis n°175, UHA, 68093 Mulhouse Cedex, France, 1991.

[5] J.D. Fouks and J.C. Spehner, Meta-resoluton, an algorithmic formalisation, Publ. n°54, UHA, 68093 Mulhouse Cedex, France, 1991.
[6] Z. Galil, On the complexity of resolution and the Davis–Putnam procedure, *Theoret. Comput. Sci.* **4** (1977) 23–46.
[7] A. Haken, The intractibility of resolution, *Theoret. Comput. Sci.* **39** (1985) 297–308.
[8] G.A. Margulis, Explicit constructor of concentrators, *Problems of information transmission* **9** (1973) 325–332.
[9] J.A. Robinson, A machine-oriented logic based on the resolution principle, *J. ACM* **12** (1965) 23–41.
[10] G.S. Tseitin, On the complexity of derivations in the propositional calculus, in: A.O. Slisenko, ed., *Structures in Constructive Mathematics and Mathematical Logic, Part II* (Consultants Bureau, New York, 1970) 115–125.
[11] A. Urquhart, Hard examples for resolution, *J. ACM* **34** (1) (1987) 209–219.