Model Counting for 2SAT Based on Graphs by Matrix Operators

C. Guillén, A. López*, G. De Ita†

Abstract—**Counting the models of Boolean formulae is known to be intractable but pops up often in diverse areas. We focus in a restricted version of the problem. In particular, our results are based on matrix operators and Hadamard product for counting models of Boolean formulae consisting of chains and embedded cycles. We obtain an efficient algorithm such that the input is a Boolean formula Σ in 2-CNF and the output can be either a charged Boolean formula Σ' simpler than Σ or the number of models of Σ (the charge of a Boolean formula Σ is introduced as a vector in $\mathbb{N}^2$, which contains information about the number of models of Σ). In the latter case, Σ belongs to a tractable class of Boolean formulae in 2-CNF for $\#SAT$ that contains the classes 2μ-2SAT and *Acyclic*-2HORN.**

Keywords: ***$\#2SAT$, Matrix Operators , Hadamart Product***

1 Introduction

The problem $\#SAT$, that consists of counting the number of models for a Boolean formula (BF) in conjunctive normal form (CNF), arises often in diverse areas such as logic, graph theory, and artificial intelligence [9, 10, 8, 5]. This problem is known to be intractable, even for cases with strict restrictions such as 3μ-2MON and 3μ-2HORN formulae [6]. A formula in 3μ-2MON(HORN) is composed of monotone(Horn) CNF formulae with clauses of length 2 with no variable occurring more than 3 times. There has been a growing interest to identify restricted cases of BFs and develop efficient algorithms for them [5, 6, 7]. The investigation in this direction is important not only for revealing the tractability frontiers, but also because they provide a collection of techniques useful for our understanding of these problems and reach new results [5]. In this direction, Vadhan [7] proved that counting the number of satisfying assignments of formulae in 2μ-2MON can be done in polynomial time using recurrence formulae. Roth [5] extended this result to 2μ-2SAT and Acyclic-2HORN (formula 2HORN, where every connected component of its corresponding graph is a directed tree). Russ [6] proposed other way to prove the tractability of the class 2μ-2SAT, regarding the problem as a problem on sink-free graph orientations.

In this paper, we show results for counting models of Boolean formulae (BFs) in Conjunctive Normal Form (CNF) containing "simple chains" and "nested cycles". For obtaining the number of models of a formula, we introduce the concept of "charge" of a variable x in a BF Σ, defined as the ordered pair (m, n), where m and n are the number of models of Σ with value 1 and 0 in the variable x, respectively. Hadamard product and matrix operators are also used by establishing relations between the number of models of a formula Σ and the charges of some variables in subformulae of Σ. These results lead to identify a tractable class of BFs in 2CNF, denoted by $\mathcal{C}_{mg}$, that contains the classes 2μ-2SAT and *Acyclic*-2HORN. The class $\mathcal{C}_{mg}$, is determined by the structure (multigraph) of its members, that is, $\Sigma \in \mathcal{C}_{mg}$ iff Σ is a BF in 2CNF s. t. it can be disarticulate in connected components that are either simple chain or nested cycles.

We also provide an efficient algorithm, called #SAT_2CNF s. t. for a given BF Σ in 2-CNF as input, returns two possible outputs: a charged BF free of chains and nested cycles, or the number of models of Σ. The algorithm #SAT_2CNF can be used for several aims: (1) identify the class $\mathcal{C}_{mg}$ ($\Sigma \in \mathcal{C}_{mg}$ iff #SAT_2CNF return the number of models of Σ), (2) obtain the number of models of any Σ in $\mathcal{C}_{mg}$, and (3) if Σ is a BF in 2CNF s. t. $\Sigma \notin \mathcal{C}_{mg}$, then #SAT_2CNF returns Σ' a charged BF (see definition 2). In this case, Σ' have the same number of models than Σ (definitions 3 and 4), and it decrease in the number of its variables when Σ contains subformulae in the class $\mathcal{C}_{mg}$.

The paper is organized as follows: In Section 2 are defined some preliminaries and technical tools. Section 3, defines the multigraph induced by a BF. Sections 4 to 5 are devoted to counting models of chains and nested cycles of BFs. In Section 6, are defined the concepts of Boolean Formula charged and #Sat-equivalent. In section 7, we present the algorithm #SAT_2CNF and analyze its complexity in time.

*Instituto Nacional de Astrofísica Óptica y Electrónica, Tonantzintla Puebla, México, C.P. 72840. & Tel/Fax: 2663152/ Email:cguillen@inaoep.mx, allopez@inaoep.mx

†Faculty of Computer Sciences, Universidad Autónoma de Puebla, deita@cs.buap.mx

(Advance online publication: 17 November 2007)

2 Preliminaries

The set of *natural numbers* (or nonnegative integers) is denoted by $\mathbb{N}$. For every positive integer n, the set $\{1, ..., n\}$ is denoted as [n] and the set $\{0, 1\}$ is denoted by $\mathbb{B}$. The cardinality of a set A is denoted by $\#A$. A *Boolean formula* (BF) Σ in CNF of n Boolean variables $X = \{x_1, x_2, ..., x_n\}$ consists of a set of clauses $c_1, c_2, ..., c_m$, where each *clause* is a set of literals over the variables X, a *literal* ℓ is either a variable x, or its negation $\neg x$. A BF Σ in 2CNF is a formula in CNF s.t. each clause of Σ has at most two literals.

The set X of variables of Σ is denoted by $Var(\Sigma)$, $Lit(\Sigma)$ refers to the set of literals of Σ, and $Var(\ell)$ denotes the *underlying variable* of $\ell \in Lit(\Sigma)$. Let $sgn : Lit(\Sigma) \to \{+, -\}$ be the sign function, defined by $sgn(\ell) = +$ if $\ell = Var(\ell)$, $sgn(\ell) = -$ if $\ell = \neg Var(\ell)$. Given $x \in Var(\Sigma)$, we define the *degree* of x as the number of occurrences of x in Σ without care of its sign, this is denoted by $deg_\Sigma(x)$.

A BF $\mathcal{C} = \{c_1, ..., c_{k-1}\}$ in 2CNF is a *simple chain* iff there is $\sigma : [k-1] \to [k-1]$ permutation s. t. $\#(Var(c_{\sigma(i)}) \cap Var(c_{\sigma(i+1)})) = 1$ for each $i \in [k-2]$ and $\mathcal{C}$ is a *simple cycle* iff $\mathcal{C}$ is a simple chain and $\#(Var(c_{\sigma(1)}) \cap Var(c_{\sigma(k-1)})) = 1$. If σ is the identity, then we say that $\mathcal{C}$ is a *ordered simple-cycle (simple-chain).*

An *assignment* s for Σ is a function $s : Var(\Sigma) \to \mathbb{B}$. Given any $A \subseteq Var(\Sigma)$, $s\mid_A$ denotes the restriction of s to A ($s\mid_A : A \to \mathbb{B}$ s.t. $s\mid_A(x) = s(x)$, for all $x \in A$). A literal ℓ is *satisfied* by the assignment s iff either $s(Var(\ell)) = 1$ and $\ell = Var(\ell)$ or $s(Var(\ell)) = 0$ and $Var(\ell) \neq \ell$. The assignment s *satisfies* the clause c iff s satisfies some literal in c, s *satisfies* the BF Σ iff s satisfies all clauses of Σ. $Sat(\Sigma)$ denotes the set of assignments on $Var(\Sigma)$ that satisfy the BF Σ.

Let Σ_1 and Σ_2 be BFs s.t. $Var(\Sigma_1) \cap Var(\Sigma_2) = \emptyset$, then it is clear that

$$\#Sat(\Sigma_1 \cup \Sigma_2) = \#Sat(\Sigma_1)\#Sat(\Sigma_2) \qquad (1)$$

Given any variable $x \in Var(\Sigma)$, $\#\mathbf{Sat}(\Sigma, x)$ denotes the ordered pair (m, n), where $m = \#Sat_{x=1}(\Sigma)$ and $n = \#Sat_{x=0}(\Sigma)$. We refer to $\#\mathbf{Sat}(\Sigma, x)$ as the ***charge*** of x in Σ. If Σ' is a subformula of Σ and $x \in Var(\Sigma')$, then the charge $\#\mathbf{Sat}(\Sigma', x)$ is called the ***partial charge*** of x in Σ' (or the ***induced charge*** by x in Σ').

Note that if (m, n) is the charge of any variable x in Σ, then $\#Sat(\Sigma) = m + n$. The pair (m, n) is also denoted by the 2×1 matrix $\begin{pmatrix} m \\ n \end{pmatrix}$.
We consider the *Hadamard product* "$\diamond$" on $\mathbb{N}^2$: $(m_1, n_1) \diamond (m_2, n_2) = (m_1 m_2, n_1 n_2)$.

Observe that "$\diamond$" is associative, distributive and commutative. If $\mathbf{q}_i \in \mathbb{N}^2$ for $i = 1, ..., k$, the product $\mathbf{q}_1 \diamond \mathbf{q}_2 \diamond \cdots \diamond \mathbf{q}_k$ is denoted by $\Diamond_{i=1}^{k} \mathbf{q}_i$. Also, we define the product "$\circledast$" of a 2×2 matrix by a charge, as follows:

$$\begin{pmatrix} m \\ n \end{pmatrix} \circledast \begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} ma & mb \\ nc & nd \end{pmatrix}$$

Note that if $\mathbf{p}$ and $\mathbf{q}$ are charges and T is a 2×2 matrix, then

$$(\mathbf{p} \circledast T)(\mathbf{q}) = \mathbf{p} \diamond (T\mathbf{q}) \qquad (2)$$

3 Multigraph Induced by a Formula in 2-CNF

Given a BF Σ in 2-CNF, one can choose an order in each clause of Σ. So if $c = \{\ell, r\} \in \Sigma$ we denote $c = (\ell, r)$ or $c = (r, \ell)$ depending on the chosen order. If $c = \{\ell\} \in \Sigma$, we denote $c = (\ell, \ell)$. Let Σ' be a BF with ordered clauses obtained from Σ choosing an order in each clause of Σ, note that $\#Sat(\Sigma') = \#Sat(\Sigma)$. Then, a BF Σ in 2-CNF with ordered clauses can be considered as a finite set of ordered pairs. Given a BF Σ in 2-CNF with ordered clauses, we define G_Σ the *multigraph induced* by Σ, as the edge-labelled directed multigraph built as follows: the set of nodes of G_Σ is $Var(\Sigma)$ and the set of edges of G_Σ is obtained from the set of clauses of Σ, where each clause (ℓ, r) defines the directed edge from the node $Var(\ell)$ (source) to the node $Var(r)$ (target) labelled with the concatenation of $sgn(\ell)$ and $sgn(r)$. We also denote for the clause $c = (\ell, r)$, $Sgn(c) = Sgn(\ell, r) = sgn(\ell)sgn(r)$. Note that a BF Σ in 2-CNF induces as many multigraphs as the number of ways to select an order in each of its clauses.

Example 1 *Consider the BF with ordered clauses* $\Sigma = \{(\neg x_1, x_2), (x_2, x_3), (\neg x_2, \neg x_4), (x_1), (x_1, x_3), (x_2, \neg x_3)\}$. *The multigraph* G_Σ *is given in la figura 1. Note that the unitary clause* (x_1) *increases in 2 the degree of* x_1*, i.e.* $deg_\Sigma(x_1) = 4$.

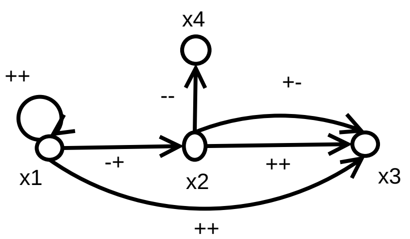

Fig. 1 Multigraph G_Σ

4 Matrix Operators and Chains

To illustrate the underlying ideas, we start counting the number of models of the BFs $\Sigma_1 = \{(\ell)\}$ and $\Sigma_2 =$

$\{(\ell, r)\}$. For Σ_1, it is clear that $\#Sat(\Sigma_1) = 1$ and the charge of $Var(\ell)$ in Σ_1 is $(1,0)$ if $sgn(\ell) = +$, and is $(0,1)$ if $sgn(\ell) = -$. So, we obtain

$$\#\mathbf{Sat}(\Sigma_1, Var(\ell)) = \pi_{sgn(\ell)} \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

where π_+ and π_- are the *projection operators* given by

$$\pi_+ = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, \ \pi_- = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \tag{3}$$

Now, analyzing all the cases for $Sgn(\ell, r)$ of Σ_2, we have $\#Sat(\Sigma_2) = 3$ and the charges of $Var(\ell)$ and $Var(r)$ are given by $\#\mathbf{Sat}(\Sigma_2, Var(\ell)) = (2,1)$ for $Sgn(\ell, r) \in \{++, +-\}$ and $\#\mathbf{Sat}(\Sigma_2, Var(\ell)) = (1,2)$ for $Sgn(\ell, r) \in \{-+, --\}$ symmetrically, $\#\mathbf{Sat}(\Sigma_2, Var(r)) = (2,1)$ for $Sgn(r, \ell) \in \{++, +-\}$ and $\#\mathbf{Sat}(\Sigma_2, Var(\ell)) = (1,2)$ for $Sgn(r, \ell) \in \{-+, --\}$. Summarizing, we have $\#\mathbf{Sat}(\Sigma_2, Var(r))$=$T_{Sgn(\ell,r)}(1,1)$ and $\#\mathbf{Sat}(\Sigma_2, Var(\ell))$=$T_{Sgn(r,\ell)}(1,1)$, where the operators $T_{Sgn(\ell,r)}$(chain operators) are defined as:

$$T_{++} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}, T_{+-} = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \tag{4}$$

$$T_{-+} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, T_{--} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} \tag{5}$$

Given a BF Σ, following the previous idea the question is whether it is possible to find relations between $\#Sat(\Sigma)$ and the matrix operators $\pi_{sgn(\ell)}$ and $T_{Sgn(\ell,r)}$ in (5) and (6). We find an answer for a class of BFs (see Theorem 1). Let consider the BFs: $\Sigma_3 = \{(x,y),(y,z)\}$, $\Sigma_4 = \{(x,y),(\neg y,z)\}$, $\Sigma_5 = \{(x,y),(\neg y,\neg z)\}$ and $\Sigma_6 = \{(x,y),(y,\neg z)\}$. We can easily obtain the relations: $\#\mathbf{Sat}(\Sigma_3, z) = T_{++}T_{++}(1,1) = (3,2)$, $\#\mathbf{Sat}(\Sigma_4, z) = T_{-+}T_{++}(1,1) = (3,1)$, $\#\mathbf{Sat}(\Sigma_5, z) = T_{--}T_{++}(1,1) = (1,3)$ and $\#\mathbf{Sat}(\Sigma_6, z) = T_{+-}T_{++}(1,1) = (2,3)$. If $\Sigma = \{(x_1,x_2),(x_2,x_3),...,(x_n,x_{n+1})\}$, it is not hard to check that $\#\mathbf{Sat}(\Sigma, x_1) = T_{++}^n(1,1)$. Observe also that T_{++} is the Fibonacci Q-Matrix, and that for every positive integer n, the powers of T_{++} are given by

$$T_{++}^n = \begin{pmatrix} F_{n+1} & F_n \\ F_n & F_{n-1} \end{pmatrix} \tag{6}$$

where $F_0 = 0, F_1 = 1, F_{n+1} = F_n + F_{n-1}$. Then $\#\mathbf{Sat}(\Sigma, x_1) = (F_{n+2}, F_{n+1})$. However, the powers of T_{--}, T_{+-} and T_{-+} are given by

$$T_{--}^n = \begin{pmatrix} F_{n-1} & F_n \\ F_n & F_{n+1} \end{pmatrix}, \ T_{+-}^n = (T_{-+}^n)^t = \begin{pmatrix} 1 & 0 \\ n & 1 \end{pmatrix}.$$

We can establishes a relation between the partial charges and the total charge of one common variable of two BFs.

Remark 1 *Let Σ_1 and Σ_2 be BFs s.t. $Var(\Sigma_1) \cap Var(\Sigma_2) = \{x\}$, then $\#\boldsymbol{Sat}(\Sigma_1 \cup \Sigma_2, x) = \#\boldsymbol{Sat}(\Sigma_1, x) \diamond \#\boldsymbol{Sat}(\Sigma_2, x)$.*

Remark 2 *Let Σ_1 and Σ_2 be BFs, s.t. $Var(\Sigma_1) \cap Var(\Sigma_2) = \emptyset$, $c = (\ell, r)$ a clause with $Var(\ell) = x \in Var(\Sigma_1)$, $Var(r) = y \in Var(\Sigma_2)$ and $\boldsymbol{p}$, $\boldsymbol{q}$, the charges of $Var(\ell)$ and $Var(r)$ in Σ_1 and Σ_2 respectively, then for $i \in \mathbb{B}$:*

$$\#\boldsymbol{Sat}_{x=i}(\Sigma, y) = \ \boldsymbol{q} \diamond (T_{Sgn(c)}\pi_i(\boldsymbol{p})) \tag{7}$$

$$\#\boldsymbol{Sat}(\Sigma, y) = \ \boldsymbol{q} \diamond (T_{Sgn(c)}(\boldsymbol{p})) \tag{8}$$

where $\Sigma = \Sigma_1 \cup \Sigma_2 \cup \{c\}$, $\pi_1 = \pi_+$, $\pi_0 = \pi_-$.

Using the chain operators, from remark 2 and equation (3), we get $\#Sat(\Sigma)$.

Example 2 *Let $\Sigma = \{c_1, ..., c_5\}$ were $c_1 = (\neg x_1, x_2)$, $c_2 = (\neg x_5, x_2)$, $c_3 = (x_2, x_3)$, $c_4 = (x_6, x_3)$, $c_5 = (\neg x_3, \neg x_4)$. G_Σ is depicted in figure 2.*

Let $q_i^0 = (1,1)$ for $i = 1, ..., 6$ and $\Sigma_i = \Sigma \setminus \{c_1, ..., c_i\}$ for $i = 1, ..., 5$. Then

$$\begin{aligned} q_2 &= \#\mathbf{Sat}(\Sigma_1, x_2) = q_2^0 \diamond T_{-+}q_1^0 = (2,1) \\ q_2' &= \#\mathbf{Sat}(\Sigma_2, x_2) = q_2 \diamond T_{-+}q_5^0 = (4,1) \\ q_3 &= \#\mathbf{Sat}(\Sigma_3, x_3) = q_3^0 \diamond T_{++}q_2' = (5,4) \\ q_3' &= \#\mathbf{Sat}(\Sigma_4, x_3) = q_3 \diamond T_{++}q_6^0 = (10,4) \\ q_4 &= \#\mathbf{Sat}(\Sigma_5, x_4) = q_4^0 \diamond T_{--}q_3' = (4,14) \end{aligned}$$

therefore $\#Sat(\Sigma) = 18$.

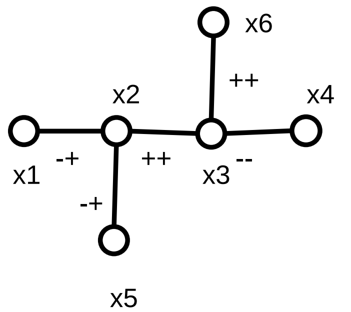

Figure 2. Multigraph G_Σ, example 2.

5 Matrix Operators and Cycles

By a similar analysis as that of section 4, we obtain the set of operators for counting the number of models of simple cycles. The following definition summarizes this analysis.

Definition 1 *(Cycle Operators). Let $\Psi_{s_{i,j}} : \mathbb{N}^4 \to \mathbb{N}^4$ be the operators defined for each combination of signs $s_{i,j} \in \{++, +-, -+, --\}$ as follows:*

$$\Psi_{++}\begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} a & b \\ c & 0 \end{pmatrix}, \ \Psi_{-+}\begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} a & 0 \\ c & d \end{pmatrix}$$

$$\Psi_{--}\begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} 0 & b \\ c & d \end{pmatrix}, \ \Psi_{+-}\begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} a & b \\ 0 & d \end{pmatrix}$$

Example 3 *Let* $\Sigma = \{(w_1, w_2), (\neg w_2, w_3), (w_3, \neg w_1)\}$, $T_{1,2} = T_{++}$ *and* $T_{2,3} = T_{-+}$. *Observe that*

$$\#\boldsymbol{Sat}(\Sigma, w_3) = \Psi_{+-} T_{-+} T_{++} \begin{pmatrix} 1 \\ 1 \end{pmatrix} =$$

$$\Psi_{+-} \begin{pmatrix} 2 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 2 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 3 \\ 0 \end{pmatrix}$$

Remark 3 *Let* $\Sigma = \{c_1, ..., c_k\}$ *a ordered simple-cycle (as in section 2) with* $k \geq 2$. *Assuming that* $c_i = (\ell_i, r_i)$, $Var(\ell_i) = x_i$ *for* $i \in [k]$, *then*

$$\#\boldsymbol{Sat}(\Sigma, x_1) = (\Psi_{Sgn(c_k)}(\hat{T}_k))(\boldsymbol{q}_1) \tag{9}$$

where $\hat{T}_k = \prod_{i=1}^{k-1} T_{Sgn(c_{k-i})}$ *and* $\boldsymbol{q}_1 = (1, 1)$.

For each $k \in \mathbb{N}$ *and chain operators* T_{s_j}, $s_j \in \{++, +-, -+, --\}$, $j \in [k]$, *we define the operator* $Cycle(T_{s_1}, ..., T_{s_k}) = \Psi_{s_k}(T_{s_{k-1}} \cdots T_{s_1})$. *Note that* $\#\boldsymbol{Sat}(\Sigma, x_1) = Cycle(T_{Sgn(c_1)}, ..., T_{Sgn(c_k)})(\boldsymbol{q}_1)$.

In particular when every literal of Σ is positive, we have $\hat{T}_k = T_{++}^{k-1}$, and using (8), we obtain

$$\#\mathbf{Sat}(\mathcal{C}, x_1) = (\Psi_{++}\hat{T}_k)q_1 = (\Psi_{++}T_{++}^{k-1})q_1 =$$

$$\begin{pmatrix} F_k & F_{k-1} \\ F_{k-1} & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} F_{k+1} \\ F_{k-1} \end{pmatrix}$$

where F_{k+1} and F_{k-1} are Fibonacci numbers. Note also that any connected component of a BF Σ in 2μ-2SAT is either a simple chain or simple cycle. Using Theorems 1 and 2, with equation (3), we can get $\#Sat(\Sigma)$.

6 Processing Embedded Cycles

A set of simple cycles $C = \{\mathcal{C}_1, ..., \mathcal{C}_j\}$ is a set of *embedded cycles* iff there is a permutation $\sigma : [j] \to [j]$ and $e_i \in \mathcal{C}_{\sigma(i)}$ for $i = 1, ..., j$ s. t.

$$\mathcal{C}_{\sigma(1)} \setminus \{e_1\} \subset \mathcal{C}_{\sigma(2)} \setminus \{e_2\} \cdots \subset \mathcal{C}_{\sigma(j)} \setminus \{e_j\} \tag{10}$$

We say that a BF Σ has a *structure of embedded cycles* iff there is a set of embedded cycles C s. t. $\Sigma = \cup_{\mathcal{C} \in C} \mathcal{C}$. Observe that if Σ_1 and Σ_2 are two simple cycles s. t. $\Sigma_1 \cap \Sigma_2 = \{c\}$, then $(\Sigma_1 \cup \Sigma_2) - \{c\}$ is a simple cycle that contains to $\Sigma_1 - \{c\}$. Indeed, we have that $\Sigma_1 = \{c_1, ..., c_{n-1}, c\}$ and $\Sigma_2 = \{e_1, ..., e_{m-1}, c\}$, then $(\Sigma_1 \cup \Sigma_2) - \{c\} = \{c_1, ..., c_{n-1}, e_1, ..., e_{m-1}\}$, $Var(c) = \{x, y\}$. Also, if Σ_1 and Σ_2 are simple cycles s. t. $\#(\Sigma_1 \setminus \Sigma_2) = 1$, yields $\Sigma_1 \setminus \{c\} \subset \Sigma_2 \setminus \{e\}$ for some $e \in \Sigma_2 \setminus \Sigma_1$. Therefore, $\Sigma_1 \cup \Sigma_2$ has a structure of embedded cycles if some of the following conditions is fulfilled

$$\#(\Sigma_1 \cap \Sigma_2) = 1, \; \#(\Sigma_1 \setminus \Sigma_2) = 1, \; \#(\Sigma_2 \setminus \Sigma_1) = 1 \tag{11}$$

Conversely, it is not hard to check that if $\{\Sigma_1, \Sigma_2\}$ is a set of embedded cycles, then Σ_1, Σ_2 satisfy some of the conditions given in (11). Therefore, $\Sigma_1 \cup \Sigma_2$ has a structure of embedded cycles iff Σ_1, Σ_2 satisfies some of the conditions given in (11). For checking (11), we consider all clause as a nonordered pair.

Given a set C of simple cycles to verify the set of pairs in C s. t. satisfies some constrains from (11), can be done in polynomial time, since both " $\cap$ " and " $\setminus$ " are operations that consume linear time.

Example 4 *Let* $\Sigma_1 = \{a_1, a_2, a_3, a_4, a_5\}$ *where* $a_1 = \{\neg x_3, x_4\}$, $a_2 = \{\neg x_1, \neg x_2\}$, $a_3 = \{\neg x_4, x_5\}$, $a_4 = \{x_1, \neg x_5\}$, $a_5 = \{x_2, \neg x_3\}$ *and* $\Sigma_2 = \{b_1, b_2, b_3\}$ *where* $b_1 = \{\neg x_3, x_4\}$, $b_2 = \{x_2, \neg x_3\}$, $b_3 = \{\neg x_4, \neg x_2\}$. *Choosing* $\sigma_1 = (3, 4, 2, 5, 1\}$ *and* $\sigma_2 = (2, 1, 3)$ *permutations for* Σ_1 *and* Σ_2 *respectively, we have* Σ_1 *and* Σ_2 *are simple cycles. Observe that* $a_1 = b_1$, $a_5 = b_2$ *and as* $\Sigma_1 \setminus \Sigma_2 = \{b_3\}$, *then* $\Sigma_1 \cup \Sigma_2$ *has a structure of embedded cycles (see figure 3).*

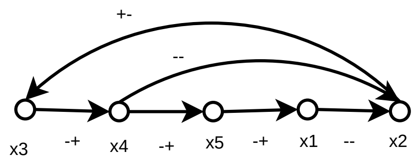

Fig. 3 Graph induced by $\Sigma_1 \cup \Sigma_2$.

Remark 4 *Given* Σ_1 *and* Σ_2 *simple cycles s. t.* $\Sigma_1 \cup \Sigma_2$ *has a structure of embedded cycles and* $\Sigma_1 \cap \Sigma_2 = \{c\}$, *we can obtain* $\#Sat(\Sigma_1 \cup \Sigma_2)$ *as follows*

$$\#\boldsymbol{Sat}(\Sigma_1 \cup \Sigma_2, x) = Cycle(U, T_{Sgn(e_1)}, ..., T_{Sgn(e_m)})(\boldsymbol{q}_1)$$

$$U = Cycle(T_{Sgn(c_1)}, ..., T_{Sgn(c_k)}, T_{Sgn(c)})$$

where $c = (l, r)$, $x = Var(l)$, $\Sigma_1 = \{c_1, ..., c_k, c\}$ *and* $\Sigma_2 = \{c, e_1, ..., e_m\}$ *are ordered.*

Example 5 *Let* $\Sigma_1 = \{(z, x), (x, y), (y, \neg z)\}$ *and* $\Sigma_2 = \{(y, u), (u, w), (\neg w, z), (y, \neg z)\}$. *We have that* $\Sigma_1 \cap \Sigma_2 = \{c\}$ *where* $c = (y, \neg z)$. *We obtain that*

$$U = Cycle(T_{Sgn(z,x)}, T_{Sgn(x,y)}, T_{Sgn(y,\neg z)}) =$$

$$= \Psi_{+-}(T_{++}T_{++}) = \begin{pmatrix} 2 & 1 \\ 0 & 1 \end{pmatrix} \textit{ and}$$

$$Cycle(U, T_{Sgn(y,u)}, T_{Sgn(u,w)}, T_{Sgn(\neg w,z)}) =$$

$$= \Psi_{-+}(T_{++}T_{++}U) = \Psi_{-+} \begin{pmatrix} 4 & 3 \\ 2 & 2 \end{pmatrix} = \begin{pmatrix} 4 & 0 \\ 2 & 2 \end{pmatrix}$$

therefore $\#Sat(\Sigma_1 \cup \Sigma_2) = 8$.

Remark 5 *Given* Σ_1 *and* Σ_2 *simple cycles s. t.* $\Sigma_1 \cup \Sigma_2$ *has a structure of embedded cycles and* $\Sigma_1 \setminus \Sigma_2 = \{c\}$, *we* *can obtain* $\#Sat(\Sigma_1 \cup \Sigma_2)$ *as follows*

$$\#\boldsymbol{Sat}(\Sigma_1 \cup \Sigma_2, x) = Cycle(U, T_{Sgn(e_1)}, ..., T_{Sgn(e_m)})(\boldsymbol{q}_1)$$

$$U = Cycle(T_{Sgn(c_1)}, ..., T_{Sgn(c_k)}, T_{Sgn(c)})$$

where $c = (l, r)$, $x = Var(l)$, $\Sigma_1 = \{c_1, ..., c_k, c\}$ *and* $\Sigma_2 = \{c_1, ..., c_k, e_1, ..., e_m\}$ *are ordered.*

Example 6 *Let* $\Sigma_1 = \{(x, y), (y, z), (z, x)\}$ *and* $\Sigma_2 = \{(x, y), (y, z), (z, w), (w, x)\}$. *We have that* $\Sigma_1 \setminus \Sigma_2 = \{c\}$ *where* $c = (x, z)$. *We obtain that*

$$U = Cycle(T_{Sgn(x,y)}, T_{Sgn(y,z)}, T_{Sgn(z,x)}) =$$

$$= \Psi_{++}(T_{++}T_{++}) = \begin{pmatrix} 2 & 1 \\ 1 & 0 \end{pmatrix} \text{ and}$$

$$Cycle(U, T_{Sgn(z,w)}, T_{Sgn(w,x)}) =$$

$$= \Psi_{++}(T_{++}U) = \Psi_{++}\begin{pmatrix} 3 & 1 \\ 2 & 1 \end{pmatrix} = \begin{pmatrix} 3 & 1 \\ 2 & 0 \end{pmatrix}$$

therefore $\#Sat(\Sigma_1 \cup \Sigma_2) = 6$.

In general, if Σ has a structure of embedded cycles, we use the remarks 4 and 5 recursively for obtaining $\#Sat(\Sigma)$.
For example let $\Sigma = \{(x, y), (y, z), (z, x), (z, w), (w, x), (w, u), (u, x)\}$, If $\Sigma_1 = \{(x, y), (y, z), (z, x)\}$, $\Sigma_2 = \{(x, y), (y, z), (z, w), (w, x)\}$ and $\Sigma_3 = \{(x, y), (y, z), (z, w), (w, u), (u, x)\}$, then Σ_1, Σ_2 and Σ_3 are cycles s.t. $\Sigma_1 \setminus \Sigma_2 = \{(z, x)\}$, $\Sigma_2 \setminus \Sigma_3 = \{(w, x)\}$ and $\Sigma = \Sigma_1 \cup \Sigma_2 \cup \Sigma_3$, then

$$U = Cycle(T_{Sgn(x,y)}, T_{Sgn(y,z)}, T_{Sgn(z,x)}) =$$

$$= \Psi_{++}(T_{++}T_{++}) = \begin{pmatrix} 2 & 1 \\ 1 & 0 \end{pmatrix} \text{ and}$$

$$U' = Cycle(U, T_{Sgn(z,w)}, T_{Sgn(w,x)}) =$$

$$= \Psi_{++}(T_{++}U) = \Psi_{++}\begin{pmatrix} 3 & 1 \\ 2 & 1 \end{pmatrix} = \begin{pmatrix} 3 & 1 \\ 2 & 0 \end{pmatrix}$$

$$Cycle(U', T_{Sgn(w,u)}, T_{Sgn(u,x)}) =$$

$$= \Psi_{++}(T_{++}U) = \Psi_{++}\begin{pmatrix} 5 & 1 \\ 3 & 1 \end{pmatrix} = \begin{pmatrix} 5 & 1 \\ 3 & 0 \end{pmatrix}$$

therefore $\#Sat(\Sigma_1 \cup \Sigma_2) = 9$.

7 Algorithm $\#ChainsCycles$

We can enlarge the idea showed in section 6 for computing the number of models of BFs consisting of chains and embedded cycles. The concept of charge allows to store the partial information of counting, in one or more variables of a BF. We see some these cases, for this, first we consider the following algorithm.

The remarks 1, 2 and 3 allow to design an efficient algorithm that, given a BF in 2-CNF, it is reduced to a charged-formula in 2-CNF (a formula with charged nodes and with the same number of models that the original formula). When a BF in 2-CNF is reduced to a charged variable, then the algorithm computes its number of models. We refer to this algorithm as the algorithm $\#ChainsCycles$, that uses the procedures *chain_contraction* and *cycle_contraction*.

Procedure chain_contraction. Given a BF, the procedure chain_contraction trims the chains of its associated graph, until the graph is reduced to a node or to a graph without nodes of degree 1. The effect of this procedure on the BF is the reduction to a BF without variables of degree 1, or the computation of its number of models. The minimum of the degrees of the variables of Σ is denoted by $mindeg(\Sigma)$. It is not hard to see that the complexity in time of this algorithm is $O(m^2)$, where m is the number of clauses of Σ, since finding $mindeg(\Sigma)$ can be done in time of $O(m)$, and the internal loop takes time $O(m)$.

Procedure ***chain_contraction*(Σ)**

Input: Boolean Formula Σ, q_i for $i \in [\#Var(\Sigma)]$
Output: Either Σ' where $mindeg(\Sigma') \geq 2$ or $\#Sat(\Sigma)$
B1) **While** $\Sigma \neq \emptyset$ or $mindeg(\Sigma) = 1$ **do**
B2) **Let** $(\ell, r) \in \Sigma$ s.t. $deg_\Sigma(\ell) = 1$ **then**
B3) $q_r = (T_{\ell,r} q_\ell) \diamond q_r$
B4) $deg_\Sigma(r) = deg_\Sigma(r) - 1$
B5) $\Sigma = \Sigma \setminus \{(\ell, r)\}$
B6) **endWhile**
B7) **Return** $\Sigma,\ q_r$

Procedure cycle_ contraction. A cycle s.t. all variable in it, with one possible exception, are of degree 2 is called a *cycle-leaf*. The set of cycle-leaves of a BF Σ is denoted by $Cl(\Sigma)$. Given a BF Σ and its associated multigraph G_Σ, the procedure cycle_contraction trims every cycle-leaf of G_Σ, until reducing G_Σ to a node or to a multigraph without cycle-leaves. When G_Σ is reduced to a node, we get the number of models of Σ. The line C2 of this procedure takes time of $O(m)$, where m is the number of clauses of Σ. Since we can find the set of cycle-leaves in linear time from the set of fundamental-cycles of G_Σ, then the complexity of the internal loop (lines C2-C5) is $O(m^3)$. Therefore, cycle_contraction has a time complexity of $O(m^4)$.

Procedure ***cycle_contraction*(Σ)**
Input: BF Σ, $q_i,\ i \in [\#Var(\Sigma)]$
Output: Either Σ' where $Cl(\Sigma') = \emptyset$ or $\#Sat(\Sigma)$,
C1) **While** $\Sigma \neq \emptyset$ and $Cl(\Sigma) \neq \emptyset$ **do**
C2) **Let** $(C, w) \in C\Sigma$
C3) $q_w = Cycle(C, w)$
C4) $deg_\Sigma(w) = deg_\Sigma(w) - 2$

C5) $\Sigma = \Sigma \setminus C$
C6)**endWhile**
C7)**Return** $\Sigma,\ q_w$

Algorithm #ChainsCycles. The algorithm *#ChainsCycles* alternates procedures *cycle_contraction* and *chain_contraction* for reducing the associated multigraph of a BF to a node or to a simplified graph without cycle-leaves and chains. So, if Σ is a BF in $\mathcal{C}_{mg}$, then Σ is reduced to a node.

Algorithm *#ChainsCycles*

Input: BF Σ in $2-FC$
Output: Either Σ'' where $Cl(\Sigma) = \emptyset$ and $\Sigma_{deg1} = \emptyset$ or $\#Sat(\Sigma)$
S1) $C\Sigma = \{C : C \text{ is cycle of } \Sigma\}$
S2) $deg_\Sigma(x) = degree\ of\ x, \forall x \in Var(\Sigma)$
S3) **While** $\Sigma \neq \emptyset$ and ($Cl(\Sigma) \neq \emptyset$ or $\Sigma_{deg1} \neq \emptyset$)
S4) $\Sigma = chain\_contraction(\Sigma)$
S5) $\Sigma = cycle\_prunning(\Sigma)$
S6) **endWhile**
S7) **Return** $\Sigma,\ q_w$

Given a BF Σ with m clauses and n variables, the time complexity of an algorithm for generating a set of fundamental-cycles of G_Σ is $O(n^3)$ [2, 4]. Therefore, the complexity of the algorithm *#Sat_2CNF* is polynomial. This algorithm is illustrated in the following examples.

Example 7 *Let* $\Sigma_1 = \{(x_1, x_2), (\neg x_2, x_3), (x_3, x_1), (x_3, x_4), (x_4, x_5), (x_5, x_3), (x_5, x_6), (x_6, \neg x_7), (x_6, x_8), (x_8, x_9), (x_9, x_{10}), (x_{10}, x_8), (x_{10}, x_{11}), (x_2), (\neg x_4)\}$. *The associated graph of* G_Σ *is depicted in Fig. 4.*

*1.- **Lines S1-S4.** We obtain that chain_contraction returns* $\Sigma_1 = \Sigma_1 \setminus \{(x_{10}, x_{11}), (x_6, x_7)\}$ *updating the charges of* x_6 *and* x_{10} *in* Σ_1*, as follows:* $q_6 = T_{-+}q_7 \diamond q_6 = (2, 1)$, $q_{10} = T_{++}q_{11} \diamond q_{10} = (2, 1)$.
*2.- **Lines S3-S5.** Here, the procedure cycle_contraction updates the formula and the charges as follows:* $q_2 = Cycle(C_2, x_2) = (1, 0)$, $q_4 = Cycle(C_4, x_4) = (0, 1)$, $q_8 = Cycle(C_5, x_8) = (6, 1)$, $\Sigma_1 = \Sigma_1 \setminus (C_2 \cup C_4 \cup C_5)$, $q_3 = Cycle(C_1, x_3) = (2, 0)$, $\Sigma = \Sigma_1 \setminus C_1$, $q_5 = Cycle(C_3, x_5) = (2, 0)$ *and* $\Sigma_1 = \Sigma_1 \setminus C_3$. *So we get* $cycle\_contraction(\Sigma_1) = \{(x_5, x_6), (x_6, x_8)\}$.
*3.- **Lines S3-S4.** Finally,* $chain\_contraction(\Sigma_1) = (28, 10)$, *since* $q_6 = T_{++}q_5 \diamond q_6 = (2, 2) \diamond (2, 1) = (4, 2)$, $q_6 = T_{++}q_8 \diamond q_6 = (7, 5) \diamond (4, 2) = (4, 2) = (28, 10)$. *Therefore,* $\#Sat(\Sigma_1) = 38$.

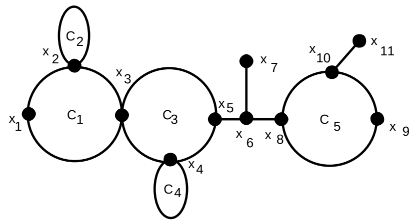

Fig. 4. Graph G_{Σ_1} example 7

Example 8 *Let* $\Sigma_2 = \Sigma_1 \cup \{(\neg x_3, x_5)\}$.
Following the lines from example 4 we get essentially the same, except that in step 2 from example 5, we obtain $cycle\_contraction(\Sigma_2) = C_3 \cup \{(\neg x_3, x_5)(x_5, x_6), (x_6, x_8)\}$. *Finally, from lines S3-S4, we get* $\Sigma_2'' = chain\_contraction(\Sigma_2) = C_3 \cup \{(\neg x_3, x_5)\}$, $q_3 = (2, 0)$, $q_4 = (0, 1)$ *and* $q_5 = (19, 14)$. *The reduced multigraph* $G_{\Sigma_2''}$ *is depicted in Fig. 5.*

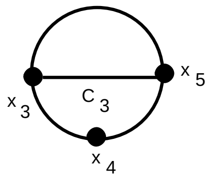

Fig. 5. Graph $G_{\Sigma_2''}$ example 8

Algorithm #EmbeddedCycles. The algorithm *#EmbeddedCyles* has as input a simplified BF Σ^* without cycle-leaves and chains, that is, first a BF Σ is preprocessing by *#ChainsCycles* for obtain Σ^*. When Σ^* has a structure of embedded cycles, the algorithm *#EmbeddedCyles* obtains $\#Sat(\Sigma)$. The collection of pairs of fundamental cycles that satisfy some condition from (11), is denoted by $EmCy(\Sigma)$, this set is determined in polynomial time, since the fundamental cycles and the conditions (11) are obtained in polynomial time. Given $\mathcal{C}$ and $\mathcal{D}$ cycles that satisfy some conditions from (11), we denote by $proCycle(\mathcal{C}, \mathcal{D})$, the procedures given in the remarks 4 and 5.

Algorithm *#EmbeddedCycles*

Input: BF Σ^* free of simple cycles and chains
Output: $\#Sat(\Sigma^*)$
E1) **While** $EmCy(\Sigma^*) \neq \emptyset$ **do**
E2) Let $(\mathcal{C}, \mathcal{D}) \in EmCy(\Sigma^*)$
E3) $proCycle(\mathcal{C}, \mathcal{D})$
E4) $EmCy(\Sigma^*) \setminus (\mathcal{C}, \mathcal{D})$
E5) **endWhile**
E6) **Return** $\#Sat(\Sigma^*)$

Example 9 *Let* $\Sigma = \{(x_1, x_2), (x_2, x_3), (x_3, x_4), (x_4, x_5), (x_5, x_6), (x_6, x_7), (x_7, x_8), (x_8, x_9), (x_9, x_{10}), (x_6, x_{11})\}$.
For simplicity, the integer i *denotes the variable* x_i *and* i' *the partial charge of the variable* x_i. *A cycle* $\mathcal{C} = \{(i_1, i_2), (i_2, i_3), ..., (i_k, i_{k+1})\}$ *is denoted by* $(i_1, ..., i_{k+1})$

$proCycle((1,2,3),(1,2,3,4,5))$
$:(1,2,3) \twoheadrightarrow (2',4,5)$

$$q_{2'} = \begin{pmatrix} 2 & 1 \\ 1 & 0 \end{pmatrix}$$

$proCycle((2',4,5),(4,5,6,7,8))$
$:(2',4,5) \twoheadrightarrow (4',6,7,8,9)$

$$q_{4'} = \begin{pmatrix} 4 & 3 \\ 3 & 0 \end{pmatrix}$$

$proCycle((8,9,10),(4',6,7,8,9))$
$:(8,9,10) \twoheadrightarrow (4',6,7,9')$

$$q_{9'} = \begin{pmatrix} 2 & 1 \\ 1 & 0 \end{pmatrix}$$

$proCycle((6,11,7),(4',6,7,11,9'))$
$6 = 7 = 11 = 6', (6,11,7) \twoheadrightarrow (4',6',9')$

$$q_{6'} = \begin{pmatrix} 2 & 1 \\ 1 & 0 \end{pmatrix}$$

$proCycle((4',6',9'))$
$(4',6',9') \twoheadrightarrow (6'')$

$$q_{6''} = \begin{pmatrix} 2 & 1 \\ 1 & 0 \end{pmatrix} \twoheadrightarrow \begin{pmatrix} 2 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 3 & 1 \\ 2 & 1 \end{pmatrix} = \begin{pmatrix} 8 & 3 \\ 3 & 1 \end{pmatrix}$$

$$\twoheadrightarrow \begin{pmatrix} 4 & 3 \\ 3 & 0 \end{pmatrix} \begin{pmatrix} 11 & 4 \\ 8 & 3 \end{pmatrix} \twoheadrightarrow \begin{pmatrix} 68 & 25 \\ 33 & 0 \end{pmatrix}$$

Therefore $\#Sat(\Sigma) = 126$.

8 Conclusions and Future Work

The approach that we have taken is to consider simple structures of BFs for concentrating the information of model counting (charge) in one of their nodes or variables. This information is compiled by means of matrix operators. Later, we establish a relation between counting models in a 2-CNF and the Hadamard product.

We have obtained a tractable superclass $\mathcal{C}_{mg}$ of 2μ-2SAT and Acyclic-2HORN, determined by the multigraph of its formulae. An extension of this class can be achieved if we can identify new procedures to compute in efficient way the charges of the subformulae appearing in a formula in 2-CNF. Therefore, it is possible to extend the algorithm $\#EmbeddedCycles$ to a new efficient algorithm that improves the reduction of the output formula or counts the number of models for the input formula.

Furthermore, it is possible to obtain improvements in the exponential bounds for known algorithms if we combine the $\#EmbeddedCycles$ procedure with the Davis-Putnam-Logemann-Loveland algorithm, modified for counting satisfying assignments.

References

[1] Dahllöf V., Jonsson P., Wahlström M., "Counting models for 2SAT and 3SAT formulae", *Theoretical Computer Science*, 332(1-3), pp. 265-291, 2005.

[2] Deo N., Prabhu G., Krishnamoorthy M., "Algorithms for Generating Fundamental Cycles in Graph", *ACM Transactions on Mathematical Software (TOMS)*, Vol. 8, No. 1, pp. 26-42, 1982.

[3] Fürer M., Kasiviswanathan S. P., "Algorithms for Counting 2-SAT Solutions and Colorings with Applications", *Electronic Colloquium on Computational Complexity* (033)(ECCC), 2005.

[4] Paton K.: "An Algorithm for Finding a Fundamental Set of Cycles of Graph". *Comm. ACM 12, 9, pp. 514-518, 1969.*

[5] Roth D.: "On the hardness of approximate reasoning", *Artificial Intelligence*, pp.273-302, 1996.

[6] Russ B.: *Randomized Algorithms: Approximation, Generation, and Counting*, Distinguished dissertations Springer, 2001.

[7] Vadhan. S.: *The Complexity of Counting*, B.S. thesis, Mathematics and Computer Science, Harvard College, Cambridge, Massachusetts, 1995.

[8] Vadhan Salil P., "The complexity of Counting in Sparse, Regular, and Planar Graphs", *SIAM Journal on Computing*, Vol. 31, No.2, pp. 398-427, 2001.

[9] Valiant L. "The complexity of enumeration and reliability problems". *SIAM Journal of Computing*, pp. 410-421, 1979.

[10] Valiant L. "The complexity of computing the permanent". *Theoretical Computing Science*, pp. 189-201, 1979.