# Model counting for ♯2SAT problem in outerplanar graphs.

Marco A. López[1], J. Raymundo Marcial-Romero[1], José A. Hernández[1], and Guillermo De Ita[2]

[1] Facultad de Ingeniería, UAEM
[2] Facultad de Ciencias de la Computación, BUAP
mlopezm158@alumno.uaemex.mx, jrmarcialr@uaemex.mx,
xoseahernandez@uaemex.mx, deita@cs.buap.mx

**Abstract.** The satisfiability problem for formulas in two conjunctive Normal Form (2SAT) is solved in polynomial time, and ♯2SAT which is the count version of 2SAT is ♯P-complete. It has been shown that for certain types of formulas, ♯2SAT can be computed in polynomial time. In this paper we define a new method, based on embedded cycles, to compute ♯2SAT on the so-called outerplanar formulas. Our algorithm's time complexity is given by $O(n+m)$ where $n$ is the number of variables and $m$ the number of clauses of the formula. Although the time complexity is similar to other methods, experimental results show that the new method is faster.

## 1 Introduction

♯SAT (the problem of model counting for a Boolean formula) concerns especially to artificial intelligence (AI), and has a direct relationship with the automated theorem proving, as well as to approximate reasoning [1–3]. ♯SAT is a ♯P complete problem, even for formulas in two conjunctive normal form, so for complete methods, only exponential time algorithms are known. The exact algorithm with the best bound until now was presented by Wahlström [4], who provides an $O(1.2377^n)$-time algorithm, where $n$ is the number of variables of the formula. Exists some formula classes where, ♯2SAT can be solved in linear time [1]. Relevant classes of this formulas are monotone formulas and cactus formulas [5].

In practice there are some tools, called SAT solvers to solve ♯SAT in efficient time. sharpSAT is a SAT solver reported in literature as the fastest exact and randomized tool, it uses a component cache system that ensures a minimum evaluation on subformulas, generated in the formula decomposition, which are similar.

Besides sharpSAT, in [5] they present an algorithm that uses formula decomposition until achieve a certain type of formulas called cactus. The evidence presented [5] shows that using formula decomposition and a cactus formula as a base case, computing models is faster than the strategy used by sharpSAT .

In [6] a method for counting models in the so calles outerplannar formulas is presented. The method is based on a treewidth decomposition.

In this paper we present a new method to count models in the so called outerplanar formulas. This method to compute $\sharp$2SAT on outerplanar formulas is based on a transformation in the input formula on its constraint signed graph. Evidences show that, using outerplanar formulas as a base case is efficient compared to [5] and sharpSAT.

Another special class of graphs contained into outerplanar graphs is the class of polygonal array graphs that has been widely used in mathematical chemistry, since they are molecular graphs used to represent the structural formula of chemical compounds. In particular, hexagonal arrays are the graph representations of an important subclass of benzenoid molecules, unbranched catacondensed benzenoid molecules, which play a distinguished role in the theoretical chemistry of benzenoid hydrocarbons [7, 8].

In our case, we are more interested in the application of counting models on conjunctive normal form formulas as a medium to develop methods for approximate reasoning. For example, for computing the degree of belief on propositional formulas, or for building Bayesian models. It is relevant to know how many models are maintained while input conjunctive normal form formulas are being updating [2, 9, 10].

This method works using the embedded form of outerplanar graphs, those embedded graphs are then used to compute $\sharp$2SAT in linear time.

The paper is organized as follows, in Section 2 the preliminaries are established. In Section 3 an embedded graph representation of outerplanar formulas is presented. In Section 4, our main procedure is presented, in section 5 the results of the paper are shown and finally, the Conclusion.

## 2 Preliminaries

Let $X = \{x_1, \ldots, x_n\}$ be a set of $n$ Boolean variables. A literal is either a variable $x_i$ or a negated variable $\overline{x}_i$. As usual, for each $x_i \in X$, we write $x_i^0 = \overline{x}_i$ and $x_i^1 = x_i$. A clause is a disjunction of different literals (sometimes, we also consider a clause as a set of literals). For $k \in N$, a $k$-clause is a clause consisting of exactly $k$ literals and, a $(\leq k)$-clause is a clause with at most $k$ literals. A variable $x \in X$ appears in a clause $c$ if either the literal $x^1$ or $x^0$ is an element of c.

A Conjunctive Normal Form (CNF) $F$ is a conjunction of clauses (we also call $F$ a Conjunctive Form). A $k$-CNF is a CNF containing clauses with at most $k$ literals.

We use $\nu(Y)$ to express the set of variables involved in the object $Y$, where $Y$ could be a literal, a clause or a Boolean formula. $Lit(F)$ is the set of literals which appear in a CNF $F$, i.e. if $X = \nu(F)$, then $Lit(F) = X \cup \overline{X} = \{x_1^1, x_1^0, \ldots, x_n^1, x_n^0\}$. We also denote $\{1, 2, \ldots, n\}$ by $[[n]]$.

An assignment $s$ for $F$ is a Boolean function $s : \nu(F) \rightarrow \{0, 1\}$. An assignment can be also considered as a set which does not contain complementary literals. If $x^\epsilon \in s$, being $s$ an assignment, then $s$ turns $x^\epsilon$ true and $x^{1-\epsilon}$ false, $\epsilon \in \{0, 1\}$.

Considering a clause $c$ and assignment $s$ as a set of literals, $c$ is satisfied by $s$ if and only if $c \cap s \neq \emptyset$, and if for all $x^{\epsilon} \in c$, $x^{1-\epsilon} \in s$ then $s$ falsifies $c$.

If $F_1 \subset F$ is a formula consisting of some clauses of $F$, then $\nu(F_1) \subset \nu(F)$, and an assignment over $\nu(F_1)$ is a partial assignment over $\nu(F)$.

Let $F$ be a Boolean formula in CNF, $F$ is satisfied by an assignment $s$ if each clause in $F$ is satisfied by $s$. $F$ is contradicted by $s$ if any clause in $F$ is contradicted by $s$. A model of $F$ is an assignment for $\nu(F)$ that satisfies $F$. We will denote as $SAT(F)$ the set of models for the formula $F$.

Given a CNF $F$, the SAT problem consists on determining if $F$ has a model. The $\sharp$SAT problem consists of counting the number of models of $F$ defined over $\nu(F)$. $\sharp$2-SAT denotes $\sharp$SAT for formulas in 2-CNF.

### 2.1 The signed primal graph of a 2-CF

There are some graphical representations of a CNF (see e.g. [11]), we use here the signed primal graph of a two conjunctive normal form.

Let $F$ be a 2-CNF, its signed primal graph (constraint graph) is denoted by $G_F = (V(F), E(F))$, with $V(F) = \nu(F)$ and $E(F) = \{\{\nu(x), \nu(y)\} : \{x, y\} \in F\}$, that is, the vertices of $G_F$ are the variables of $F$, and for each clause $\{x, y\}$ in $F$ there is an edge $\{\nu(x), \nu(y)\} \in E(F)$. For $x \in V(F)$, $\delta(x)$ denotes its degree, i.e. the number of incident edges to $x$. Each edge $c = \{\nu(x), \nu(y)\} \in E$ is associated with an ordered pair $(s_1, s_2)$ of signs, assigned as labels of the edge connecting the literals appearing in the clause. The signs $s_1$ and $s_2$ are related to the literals $x^{\epsilon}$ and $y^{\delta}$, respectively. For example, the clause $\{x^0, y^1\}$ determines the labelled edge: "$x \overset{-+}{\text{—}} y$" which is equivalent to the edge "$y \overset{+-}{\text{—}} x$".

Formally, let $S = \{+, -\}$ be a set of signs. A graph with labelled edges on a set $S$ is a pair $(G, \psi)$, where $G = (V, E)$ is a graph, and $\psi$ is a function with domain $E$ and range $S$. $\psi(e)$ is called the label of the edge $e \in E$. Let $G = (V, E, \psi)$ be a signed primal graph with labelled edges on $S$x$S$. Let $x$ and $y$ be vertices in $V$, if $e = \{x, y\}$ is an edge and $\psi(e) = (s, s')$, then $s(resp.s')$ is called the adjacent sign to $x(resp.y)$. We say that a 2-CNF $F$ is a path, cycle, a tree, or an outerplanar graph, if its signed constraint graph $G_F$ represents a path, cycle, a tree, an outerplanar graph, respectively. We will omit the signs on the graph if all of them are $+$.

Notice that a signed primal graph of a 2-CNF can be a multigraph since two fixed variables can be involved in more than one clause of the formula forming so parallel edges. Furthermore, a unitary clause is represented by a loop (an edge to join a vertex to itself). A polynomial time algorithm to process parallel edges and loops to solve $\sharp$SAT has been shown in [1].

Let $\rho$ : 2-CNF $\to G_F$ be the function whose domain is the space of Boolean formulas in 2-CNF and codomain the set of multi-graphs, $\rho$ is a bijection. So any 2-CNF formula has a unique signed constraint graph associated via $\rho$ and viceversa, any signed constraint graph $G_F$ has a unique formula associated.

### 2.2 Cumulative operations

We define a set of cumulative operations as *macro*, in this paper a macro must be constructed using the method shown in [5], using the fact that a *macro* is a linear equation of the form $M = \alpha x + \beta$ where $\alpha$ and $\beta$ represent the models already counted and $x$ the models to be computed. Since the models of an outerplannar formula always belongs to a simple cycle, $M = \{\alpha_\alpha + \beta_\alpha, \alpha_\beta + \beta_\beta\}$, where we omit the variable $x$. A *macro* contains four elements, so we need to perform operations using two more elements than [5].

A new set of equations must be constructed, associated to every pair of signs $(\epsilon, \delta)$ of an edge $\{x^\epsilon, y^\delta\}$, in a graph.

$$(\alpha_\alpha+\beta_\alpha, \alpha_\beta+\beta_\beta) = \begin{cases} (\alpha_{\beta_{-1}} + \beta_{\beta_{-1}}, (\alpha_{\alpha_{-1}} + \alpha_{\beta_{-1}}) + (\beta_{\alpha_{-1}} + \beta_{\beta_{-1}})), & \text{if } (\epsilon_i, \delta_i) = (-,-) \\ ((\alpha_{\alpha_{-1}} + \alpha_{\beta_{-1}}) + (\beta_{\alpha_{-1}} + \beta_{\beta_{-1}}), \alpha_{\beta_{-1}} + \beta_{\beta_{-1}}), & \text{if } (\epsilon_i, \delta_i) = (-,+) \\ (\alpha_{\alpha_{-1}} + \beta_{\alpha_{-1}}, (\alpha_{\alpha_{-1}} + \alpha_{\beta_{-1}}) + (\beta_{\alpha_{-1}} + \beta_{\beta_{-1}})), & \text{if } (\epsilon_i, \delta_i) = (+,-) \\ ((\alpha_{\alpha_{-1}} + \alpha_{\beta_{-1}}) + (\beta_{\alpha_{-1}} + \beta_{\beta_{-1}}), \alpha_{\alpha_{-1}} + \beta_{\alpha_{-1}}), & \text{if } (\epsilon_i, \delta_i) = (+,+) \end{cases}$$

Counting on acyclic graphs, like tree or paths, using a different equation than [5] gives a new panorama, counting in graphs like cactus graphs with always non intersecting cycles, we can use this set of operations and define a macro as the cumulative operations in a simple and non intersecting cycle.

## 3 Outerplanar 2-CNF Formulas

An outerplanar 2-CNF formula is one whose signed primal graph is ourterplanar e.g the graph has a planar drawing for which all vertices belong to the outer face of the drawing. Outerplanar graphs may be characterized (analogously to Wagner's theorem for planar graphs) by the two forbidden minors $K_4$ and $K_{2,3}$, or by their Colin de Verdière graph invariants. They have Hamiltonian cycles if and only if they are biconnected, in which case the outer face forms the unique Hamiltonian cycle. Every outerplanar graph is 3-colorable, and has degeneracy and treewidth at most 2 [12]. The outerplanar graphs are a subset of the planar graphs, of the serial-parallel graphs, and of the circle graphs.

### 3.1 Counting on outerplanar graphs

Using the representation of an outerplanar graph as a graph with embedded cycles, there is a method to solve the most internal cycles as simple cycles, and replacing the set of vertices that form the cycle with a macro. In this way it is possible to count models on any outerplanar graph its representation as embedded graph is known.

### 3.2 Common edges identification

One characteristic of the outerplanar graphs is that they do not contain the subgraph $K_{2,3}$, this subgraph can be obtained with two cycles intersected by two edges, that is, there exists two common edges between a pair of cycles. With this we found that an outerplanar graph has at most one edge between any pair of cycles (Fig. 1).

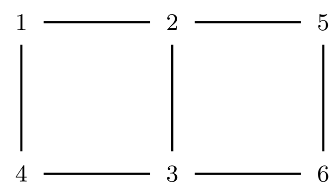


**Fig. 1.** Simple cycles intersected in one edge

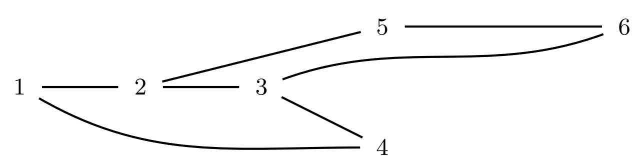


**Fig. 2.** Simple cycles intersection in a depth first search construction

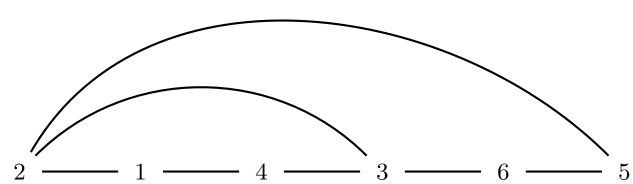


**Fig. 3.** Embedded graph of two intersected simple cycles

Identification of common edges is done by constructing an expansion tree of the graph, which we can find a set of back edges or cycles within the graph, this is necessary to identify intersection edges that belongs to the expansion tree or back edges (Fig. 2).

To substitute an edge of the expansion tree for one of the back edges, for a new construction of a embedded graph, an edge of the expansion tree can become in one of the back edges in the embedded graph (Fig. 3), the following considerations allow this exchange to be made between the sets of tree and back edges:

- If a pair of the back edges intersect on a tree edge, the back edge that closes at a low level, on the tree, is replaced by the intersecting tree edge.
- If two back edges intersect on a tree edge, and they form two sub-partitions, that is, each one belongs to a different branch in the tree. The back edge that closes at a low level, on the tree, is replaced by the intersecting tree edge.

### 3.3 Embedded graph transformation

If we identify the common edges between cycles it is possible to find an expansion tree, which the common edges form the set of back edges, so that the back edges form an embedded graph.

By obtaining the common edges in a graph and eliminating them, it is possible to generate a set of possible paths, obtaining $C_n$, a simple cycle of $n$ vertices. This way it is possible to generate multiple expansion trees that can be seen as a set of embedded cycles (Fig. 4).

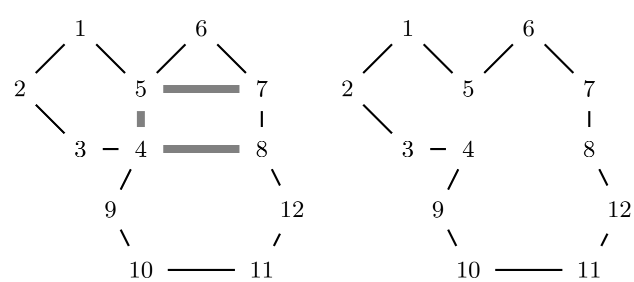


**Fig. 4.** Outerplanar graph and common edges identification

With these graphs it can be shown that no matter which vertex the construction of the new expansion tree begins on, adding the common edges will always result in an embedded graph (Fig. 5).

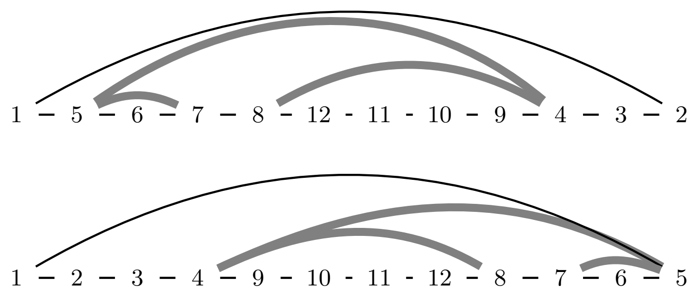


**Fig. 5.** Embedded graph transformations from Fig. 4 from the vertex labeled as 1.

By constructing an embedded graph it is possible to assign an order in which each embedded cycle must be solved, inner cycles first. This way all cycles on outerplanar formulas can be handled independently.

## 4 Computing ♯2SAT on outerplanar 2-CNF formulas

If $F$ consists of disconnected sub-formulas then $\sharp 2SAT(F) = \prod_{i=1}^{k} \sharp 2SAT(F_i)$ where $F_i, i = 1, \ldots, k$, are the disconnected sub-formulas of $F$ [3]. The time complexity for computing $\sharp 2SAT(F)$, denoted as $T(\sharp 2SAT(F))$, is given by the rule $T(\sharp 2SAT(F)) = max\{T(\sharp 2SAT(F_i)) : F_i$ is a disconnected subformula of $F\}$. Thus, a first decomposition of the formula is done via its connected components, and from here on, we consider only outerplanar connected formulas.

Due that we can assign an order to solve inner cycles first, in embedded graphs, and replace that cycle with a macro. We can compute ♯2SAT in a outerplanar graph in linear time.

We built a linear equation, macro, that represents the values $m_i = (\alpha_\alpha + \beta_\alpha, \alpha_\beta + \beta_\beta)$ on each cycle in the embedded graph. And a linear equation for each node $(\alpha_\alpha + \beta_\alpha, \alpha_\beta + \beta_\beta)$.

Giving the first transformation in Fig. 5, we can handle two cycles at same time and perform two operation sets, $M_1$ and $M_2$, where the vertices $\{5, 6, 7\} \in M_1$ and $\{8, 12, 11, 10, 9, 4\} \in M_2$.

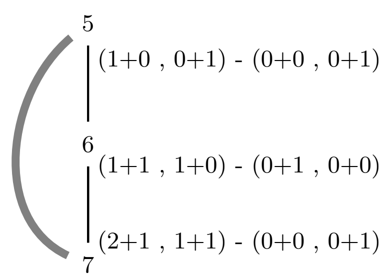


**Fig. 6.** Embedded cycle counting, $M_1$.

The final operation gives a new macro $M_1 = (2 + 1, 1 + 0)$, a second macro analogously can be obtained $M_2 = (8+5, 5+0)$. A third macro can be computed, replacing the embedded cycles, we have that the vertices $\{5, 7, 8, 4\} \in M_3$ and $\{M_1, M_2\} \in M_3$ (Fig. 7).

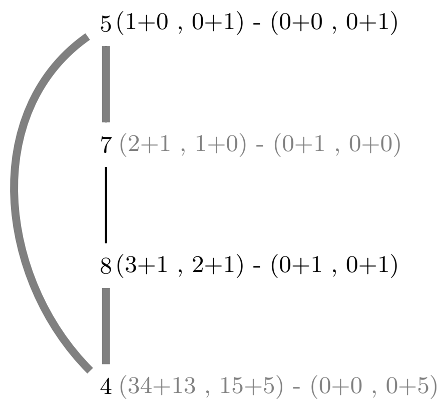


**Fig. 7.** Embedded cycle counting, $M_3$.

As a result we have $M_3 = (34 + 13, 15 + 0)$, to obtain a new set of values when we replace an inner cycle by a macro we need to apply another set of equations. If we have that $M_i = (\alpha_{\alpha_i} + \beta_{\alpha_i}, \alpha_{\beta_i} + \beta_{\beta_i})$ and a node with values $(\alpha_{\alpha_{-1}} + \beta_{\alpha_{-1}}, \alpha_{\beta_{-1}} + \beta_{\beta_{-1}})$, to obtain $(\alpha_\alpha + \beta_\alpha, \alpha_\beta + \beta_\beta)$:

- $\alpha_\alpha = \alpha_{\alpha_i} * \alpha_{\alpha_{-1}} + \beta_{\alpha_i} * \alpha_{\beta_{-1}}$
- $\beta_\alpha = \alpha_{\alpha_i} * \alpha_{\beta_{-1}} + \beta_{\alpha_i} * \beta_{\beta_{-1}}$
- $\alpha_\beta = \alpha_{\beta_i} * \alpha_{\alpha_{-1}} + \beta_{\beta_i} * \alpha_{\beta_{-1}}$
- $\beta_\beta = \alpha_{\beta_i} * \alpha_{\beta_{-1}} + \beta_{\beta_i} * \beta_{\beta_{-1}}$

This set of new equations only needs to use when a macro is applied, then we can compute the next macro $M_4$ where $\{1, 5, 4, 3, 2\} \in M_4$ and $M_3 \in M_4$, then $M_4 = (109 + 83, 62 + 0)$. Finally with the initial values of $(1 + 0, 0 + 1)$ we can obtain the total models in this example, $\sharp SAT(M_4) = (192, 62) = 254$

A relevant property of a macro, as defined in this paper, is the possibility to represent cumulative operations via symbolic variables, making macros indistinguishable from individual operators. If subsequences of operators are repeated, a hierarchy of macros can represent a more compactly plan than a simple operator sequence, replacing each occurrence of a repeating subsequence with a macro [13].

The correctness of our method is based in the following Theorem from [6].

**Theorem 1.** *Let $F_1$ and $F_2$ be two formulas in 2-CNF. If $F_1 \cap F_2 = \{x_1^\epsilon, x_2^\delta\}$, e.g. a single clause then*

$$\begin{aligned}\sharp 2SAT(F_1 \cup F_2) = & \sharp 2SAT(F_1 \mid_{\{x_1^\epsilon, x_2^\delta\} \subseteq s}) \times \sharp 2SAT(F_2 \mid_{\{x_1^\epsilon, x_2^\delta\} \subseteq s}) + \\ & \sharp 2SAT(F_1 \mid_{\{x_1^\epsilon, x_2^{\delta-1}\} \subseteq s}) \times \sharp 2SAT(F_2 \mid_{\{x_1^\epsilon, x_2^{\delta-1}\} \subseteq s}) + \\ & \sharp 2SAT(F_1 \mid_{\{x_1^{\epsilon-1}, x_2^\delta\} \subseteq s}) \times \sharp 2SAT(F_2 \mid_{\{x_1^{\epsilon-1}, x_2^\delta\} \subseteq s})\end{aligned}$$

*Proof.* In order to satisfy $F_1 \cup F_2$ the clause $\{x_1^\epsilon, x_2^\delta\}$ has to be satisfied, so either $\{x_1^\epsilon, x_2^\delta\} \subseteq s$ or $\{x_1^\epsilon, x_2^{\delta-1}\} \subseteq s$ or $\{x_1^{\epsilon-1}, x_2^\delta\} \subseteq s$. The computation of the satisfying assignments of $F_1 \cup F_2$ is given by

$$\begin{aligned}\sharp 2SAT(F_1 \cup F_2) = &\sharp 2SAT(F_1 \cup F_2 \mid_{\{x_1^\epsilon, x_2^\delta\} \subseteq s}) + \\ &\sharp 2SAT(F_1 \cup F_2 \mid_{\{x_1^\epsilon, x_2^{\delta-1}\} \subseteq s}) + \\ &\sharp 2SAT(F_1 \cup F_2 \mid_{\{x_1^{\epsilon-1}, x_2^\delta\} \subseteq s})\end{aligned}$$

Assigning truth values to the variables $x_1$ and $x_2$ to satisfy $\{x_1^\epsilon, x_2^\delta\}$ in $F_1 \cup F_2$ gives two disconnected formula, by the hypothesis that $F_1 \cap F_2 = \{x_1^\epsilon, x_2^\delta\}$, so the conclusion holds. □

The previous theorem states that if we know the models of $F_1$ where the truth values of the variables $x_1$ and $x_2$ which joint $F_1$ to another formula $F_2$ via a clause $\{x_1^\epsilon, x_2^\delta\}$ are known, then we can substitute the models where $x_1^\epsilon$ and $x_2^\delta$ appears in $F_1$ into those of $F_2$ considering the truth values of $x_1$ and $x_2$ in $F_2$.

## 5 Results

We implement our proposal and compare its runtime against sharpSAT which to the best of our knowledge is the leading sequential implementation.Additionally, in Table 1, we compare our proposal based on embedded cycles against our previous version of markSAT based on bags [6]. Other outerplannar formulas, Tables [2, 3, 4], represent polygonal tree graphs where each polygon has three to eight sides. It is work to said that this implementation is sound and complete hence the exact number of models is computed in all of them.

Table 1 shows instances of polygonal chains, with three sides each polygon, which provides the maximum number of edges in outerplanar graphs.

| | | Time in Seconds | | |
|---|---|---|---|---|
| Variables | Clauses | sharpSAT | markSAT [6] | markSAT |
| 5002 | 10001 | 4.898 | 1.726 | 0.192 |
| 10002 | 20001 | 19.335 | 6.927 | 0.268 |
| 15002 | 30001 | 43.411 | 21.821 | 0.508 |
| 20002 | 40001 | 77.283 | 47.379 | 0.824 |
| 25002 | 50001 | 121.271 | 83.371 | 1.215 |
| 30002 | 60001 | 174.213 | 129.012 | 1.647 |
| 35002 | 70001 | 237.553 | 186.094 | 2.193 |
| 40002 | 80001 | 310.091 | 249.291 | 2.782 |

**Table 1.** Formulas whose signed constraint graph is outerplannar and polygonal chain, using three sides polygons.

Table 2 shows instances of polygonal trees, with six sides each polygon.

| | | Time in Seconds | |
|---|---|---|---|
| Variables | Clauses | sharpSAT | markSAT |
| 40002 | 50001 | 1.793 | 0.492 |
| 60002 | 75001 | 3.036 | 0.753 |
| 80002 | 100001 | 4.766 | 1.515 |
| 100002 | 125001 | 6.822 | 2.593 |

**Table 2.** Formulas whose signed constraint graph is outerplannar and polygonal tree, using six side polygons.

Table 3 shows results on polygonal trees using a side-randomizer, which generate graphs with polygons from three to eight sides each.

| | | Time in Seconds | |
|---|---|---|---|
| Variables | Clauses | sharpSAT | markSAT |
| 20173 | 25172 | 0.724 | 0.231 |
| 40331 | 50330 | 1.839 | 0.483 |
| 60980 | 75979 | 3.421 | 0.762 |
| 80797 | 100796 | 5.183 | 1.558 |
| 101146 | 126145 | 7.457 | 2.631 |

**Table 3.** Formulas whose signed constraint graph is outerplannar and polygonal tree, using a side-randomizer.

Table 4 shows the running time of our proposal against sharpSAT using outerplanar graphs as base case in a general formula decomposition.

| | | Time in Seconds | |
|---|---|---|---|
| Variables | Clauses | sharpSAT | markSAT |
| 6000 | 14397600 | 230.728 | 33.824 |
| 6000 | 16197300 | 261.389 | 38.072 |
| 6500 | 10560875 | 179.876 | 24.982 |
| 6500 | 12673050 | 216.774 | 29.875 |
| 6500 | 14785225 | 254.181 | 34.756 |
| 6500 | 16897400 | 292.413 | 39.849 |

**Table 4.** Decomposition on General Formulas using outerplanar graphs as a base case.

## Conclusion

We present a new method for model counting in outerplanar graphs, with linear time complexity.

Our procedure requires the construction of the expansion tree of the outerplanar graphs, which in this case it is done in time $O(n)$, the number of vertices of the input formula. Once an expansion tree has been built a common edge identification on both the tree and their back edges is done in time complexity $O(m)$, where $m$ is the number of edges in the graph. A new expansion tree is built with non common edges, in time $O(n)$. Embedded edges can be added in time $O(\frac{n-1}{2})$, the maximum number of back edges in a outerplanar graph. Model counting is done in $O(n + m)$.

## References


1. Guillermo De Ita, Pedro Bello, and Meliza Contreras. New polynomial classes for #2SAT established via graph-topological structure. *Engineering Letters*, 15(2), 2007.
2. Darwiche A. On the tractability of counting theory models and its application to belief revision and truth maintenance. *Journal of Applied Non-classical Logics*, 11(1-2):11–34, 2001.
3. Dan Roth. On the hardness of approximate reasoning. *Artificial Intelligence*, 82(1-2):273–302, April 1996.
4. Magnus Wahlström. A Tighter Bound for Counting Max-Weight Solutions to 2SAT Instances. *Springer Berlin Heidelberg*, pages 202–213, 2008.
5. M. A. López, José Raymundo Marcial-Romero, Guillermo De Ita Luna, Héctor A. Montes Venegas, and Roberto Alejo. A linear time algorithm for solving #2SAT on cactus formulas. *CoRR, ams/1702.08581*, 2017.
6. Marco A. López, J. Raymundo Marcial-Romero, Guillermo De Ita, and Yolanda Moyao. A Linear Time Algorithm for Computing #2SAT for Outerplanar 2-CNF Formulas. *Lecture Notes in Computer Science*, 10880:72–81, 2018.
7. Shiu W. C. Extremal hosoya index and merrifield-simmons index of hexagonal spiders. *Discrete Applied Mathematics*, 156:2978–2985, 2008.
8. Stephan Wagner and Ivan Gutman. Maxima and minima of the hosoya index and the merrifield-simmons index. *Acta Applicandae Mathematicae*, 112(3):323–346, 2010.
9. Guillermo De Ita Luna. Polynomial Classes of Boolean Formulas for Computing the Degree of Belief. *Springer Berlin Heidelberg*, pages 430–440, 2004.
10. Evgeny Dantsin and Alexander Wolpert. An Improved Upper Bound for SAT. *Springer Berlin Heidelberg*, pages 400–407, 2005.
11. Stephan Szeider. On Fixed-Parameter Tractable Parametrizations of SAT. *Springer Berlin Heidelberg*, pages 188–202, 2004.
12. Bodlaender H.L. A linear time algorithm for finding tree-decompositions of small treewidth. *SIAM Journal of Computer*, 25(6):1305–1317, 1996.
13. C. Bäcström, A. Jonsson, and P. Jonsson. Automaton plans. *Journal of Artificial Intelligence Research*, 51:255–291, 2014.