Proceedings of the World Congress on Engineering 2015 Vol I
WCE 2015, July 1 - 3, 2015, London, U.K.

On Counting Planar Models

Stefan Porschen*, Tatjana Schmidt†

Abstract—**We investigate the computational complexity of #SAT for k-outerplanar formulas, a problem which in general is #P-complete. For 1-outerplanar formulas over n variables we solve #SAT in time $O(n^{5.13})$. More generally, we show that #SAT for k-outerplanar graphs, $k \geq 2$, can be solved in time $O(n^{1.7(2k+1)})$. Finally, we prove that #SAT for nested formulas runs in time $O(n^{8.5})$.**

Keywords: CNF-formula, satisfiability, k-outerplanar, counting-problem

1 Introduction

The propositional satisfiability problem (SAT) of conjunctive normal form (CNF) formulas is an essential combinatorial problem, namely one of the first problems that have been proven to be NP-complete [4]. More precisely, it is the natural NP-complete problem and thus lies at the heart of computational complexity theory. Moreover SAT plays a fundamental role in the theory of designing exact algorithms, and it has a wide range of applications because many problems can be encoded as a SAT problem via reduction [9, 8] due to the rich expressiveness of the CNF language. The applicational area is pushed by the fact that meanwhile several powerful solvers for SAT have been developed (cf. e.g. [12, 18] and references therein). Also from a theoretical point of view one is interested in classes for which SAT can be solved in polynomial time. There are known several subclasses of CNF restricted to which SAT behaves polynomial-time solvable, so for instance 2-CNF-SAT, where clauses have length at most two [1], and Horn-SAT [15], confer also [16]. Also CNF formulas for modelling industrial applications often admit a specific structure which sometimes can be described by graph-theoretic means. Clearly efficient algorithms for such instances are of high interest if achievable. Our paper specifically is devoted to study the problems SAT and #SAT for CNF formulas admitting a k-outerplanar graph structure. Recall that for unrestricted formulas #SAT means to count all solutions and that it is known to be a basic #P-complete problem [19]. Observe that #SAT also solves SAT for a given instance.

In this paper we exploit the separator theorem proved in [13]. It states that the vertex set of a planar graph can be partitioned into two sets V_1 and V_2 of at most $2n/3$ vertices each, plus a separator set S containing $O(n^{1/2})$ vertices, such that no vertex in V_1 is adjacent to a vertex in V_2. In [3] it is shown that the tree width of a k-outerplanar graph G is at most $3k-1$, in which case it is ensured that G admits a type-2 $\frac{1}{2}(n-(3k-1))$-separator of size at most $3k$, for positive integer k. Here, a *type-2 k-separator* is a set $U \subseteq V$, such that each connected component of the induced subgraph $G[V-U]$ contains at most k vertices. Given a simple graph $G=(V,E)$, recall that an induced subgraph over vertex set $U \subseteq V$ admits exactly those edges of G joining the vertices in U [7]. However, the separation approach due to [13] mentioned first seems to be more appropriate for our purposes. So, on that basis we first design an algorithm solving #SAT for 1-outerplanar formulas which then is generalized to the case of arbitrary fixed value k yielding a time complexity that is upper bounded by $O(n^{1.7(2k+1)})$, i.e., $O(n^{5.13})$ for $k=1$. Finally, nested formulas defined in [10] are treated. A nested formula turns out to have a 2-outerplanar graph structure, so counting its models over n variables never consumes more than $O(n^{8.5})$ time.

2 Notation and Preliminaries

Let CNF denote the set of duplicate-free conjunctive normal form formulas over propositional variables $x \in \{0,1\}$. A *positive (negative) literal* is a (negated) variable. The *complement* of a literal ℓ is its negation $\overline{\ell}$. Each formula $C \in$ CNF is considered as a clause set, and each clause $c \in C$ is represented as a literal set, so clauses are not permitted to contain a literal more than once. For a formula C, clause c, by $V(C), V(c)$ we denote the set of variables (neglecting negations), contained in C resp. in c. Given a clause c and $x \in V(c)$, let $\ell(x) \in c$ denote the literal over x in c.

The satisfiability problem (SAT) asks whether a formula $C \in$ CNF has a *model*, which is a truth assignment $t: V(C) \to \{0,1\}$ assigning at least one literal in each clause of C to 1. The counting version #SAT of SAT is to determine the number $N(C)$ of models of a formula $C \in$ CNF. Given C and $x \in V(C)$ by $C(x=\epsilon)$ we denote the formula resulting from C by evaluation of the assignment $x=\epsilon$. Hence $C(x=\epsilon)$ contains only those clauses that are still unsatisfied by $x=\epsilon$, for $\epsilon \in \{0,1\}$, and from which the literal $\ell(x)$ is removed. More generally, given a truth assignment t, and $U=\{x_{i_1},\ldots,x_{i_r}\} \subseteq V(C)$, by $C_{t(x_{i_1},\ldots,x_{i_r})}$ we denote the formula obtained from C by

*Mathematics Group, Department 4, HTW Berlin, D-10313 Berlin, Germany, Email: porschen@htw-berlin.de.
†Waidmarkt 18, D-50676 Köln, Germany.

ISBN: 978-988-19253-4-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

its evaluation according to $t(x_{i_1}), \ldots, t(x_{i_r})$.

Moreover, an embedding of a planar graph is called *1-outerplanar* (or simply outerplanar) if all vertices lie on the exterior face. For $k \geq 2$, a planar graph G is *k-outerplanar*, if G admits an embedding such that the deletion of all vertices on the outer face yields a $(k-1)$-outerplanar graph. Given a CNF formula $C = \{c_1, \ldots, c_m\}$ with variable set $V(C) = \{x_1, \ldots, x_n\}$ we define its *(formula) graph* G_C as follows: Its vertex set is $C \cup V(C)$, and its edge set is $\{\{c_i, x_j\} : x_j \in V(c_i), 1 \leq i \leq m, 1 \leq j \leq n\}$. A vertex in in G_C from C is called a *clause-vertex* abbreviated by c-vertex, and a vertex in G_C from $V(C)$ is called a *variable-vertex* throughout abbreviated by v-vertex. We call a formula *k-outerplanar* if its graph is k-outerplanar, for fixed positive integer $k \geq 1$. Clearly the correspondence between formulas and formula graphs is uniquely determined by the definition above. Therefore having a graph G we often write $N(G)$ for the number of models of the underlying formula. Similarly, given C we denote by $G_C(x = \epsilon)$ the formula graph of $C(x = \epsilon)$ as defined above, where $\epsilon \in \{0, 1\}$. A *model* of a formula graph always means the model of the underlying formula.

3 #SAT for Outerplanar Formulas

First we aim at a polynomial-time algorithm for counting all models of an outerplanar formula. The basis is a devide-and-conquer approach resting on the separator theorem for planar graphs due to [13]. This result states that the vertex set of a planar graph of n vertices can be partitioned into three parts V_1, V_2, S such that no edge joins a vertex in V_1 with a vertex in V_2; neither V_1 nor V_2 contains more than $2n/3$ vertices, and the *separator set* S contains no more than $2\sqrt{2}\sqrt{n}$ *separator vertices*. Throughout this section we use a nice variant of the separator theorem above [14] for the special case of outerplanar graphs stating that the separator set then has at most two vertices. It is well known that a separator set for outerplanar graphs can be computed in linear time [3].

Let us emphasize some notions concerning cycle-free graph patterns which can be treated as special cases in our algorithm which is described below. A connected outerplanar formula graph G without cycles is called a *tree*. A tree specifically is called a *v-tree* if each pair of intersecting paths is allowed to intersect in a v-vertex only. If an arbitrary path of a v-tree is fixed it will be refered to as its *main path*. Observe that a tree which is no v-tree has a c-vertex that is adjacent to at least three v-vertices. Let P be an arbitrary path and let α be a fixed model over the variables $x_1, \ldots, x_n$ of the formula underlying P. We write $M(\alpha(x_i))$ for the number of all different models of P, in which the variables $x_i, \ldots, x_n$ are set according to α and the variables $x_1, \ldots, x_{i-1}$ can be set arbitrarily as long as P is satisfied. Similarly, we write $M(\overline{\alpha(x_i)})$ for the number of all different models of P, where the variables $x_{i+1}, \ldots, x_n$ are set according to α, $x_i = \overline{\alpha(x_i)}$ and the variables $x_1, \ldots, x_{i-1}$ can be assigned arbitrarily as long as P is satisfied. Next we state our algorithm for counting all models of an outerplanar formula. This algorithm works recursively using subprocedures when the graph of the input formula is a tree or a path. These subprocedures are described below.

Algorithm 1
Input: $C \in$ CNF outerplanar.
Output: $N(C)$.

1. Compute G_C.
2. As long as there is a c-vertex c_j in G_C, which is adjacent to a single v-vertex x_i only, then fix the value of x_i such that c_j is satisfied. Then remove all c-vertices which are satisfied by x_i, remove vertex x_i and all its incident edges from G_C.
3. If there is a c-vertex that is adjacent to no v-vertex (corresponding to an empty clause), then C is unsatisfiable and the procedure stops with output $N(C) = 0$.
4. If G_C is a path, then C is satisfiable and $N(C) = N(G_C)$ is determined by Procedure_Path.
5. If G_C is a tree, then $N(G_C)$ is determined by Procedure_Tree.
6. (a) Fix two vertices x_i and c_j such that G_C is partitioned into two subgraphs G_C^1 and G_C^2, which both have x_i und c_j in common and neither G_C^1 nor G_C^2 contains more than $\frac{2n}{3}$ v-vertices. In addition, there is no edge joining a vertex from $G_C^1 \setminus \{x_i, c_j\}$ and a vertex from $G_C^2 \setminus \{x_i, c_j\}$. For simplicity let the formula underlying G_C^a be denoted as C^a, for $a = 1, 2$.
   (b) Derive $G_C^a(x_i = 1)$ $(a = 1, 2)$ by eliminating all those c-vertices which are satisfied by $x_i = 1$, the v-vertex x_i and all its incident edges. Analogously derive $G_C^a(x_i = 0)$ by removing from G_C^a $(a = 1, 2)$ all c-vertices which are satisfied by $x_i = 0$, moreover remove x_i and all its incident edges. Further obtain $G_C^a(x_i = \epsilon) \setminus \{c_j\}$ by removing c_j and all incident edges from $G_C^a(x_i = \epsilon)$, for $a = 1, 2$ and $\epsilon \in \{0, 1\}$.
   (c) Compute $N(G_C^a(x_i = \epsilon))$, and $N(G_C^a(x_i = \epsilon) \setminus \{c_j\})$, for $a = 1, 2$, and $\epsilon \in \{0, 1\}$, recursively by applying Algorithm 1 consecutively to the following subgraphs.
   $G_C^1(x_i = 1)$,
   $G_C^1(x_i = 0)$,
   $G_C^2(x_i = 1)$,
   $G_C^2(x_i = 0)$,

ISBN: 978-988-19253-4-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Proceedings of the World Congress on Engineering 2015 Vol I
WCE 2015, July 1 - 3, 2015, London, U.K.

$G_C^1(x_i = 1) \setminus \{c_j\}$,
$G_C^1(x_i = 0) \setminus \{c_j\}$,
$G_C^2(x_i = 1) \setminus \{c_j\}$,
$G_C^2(x_i = 0) \setminus \{c_j\}$.

(d) Compute

$$\begin{aligned} N(C) = \max\{&N(C^1(x_i=1)\setminus\{c_j\})\cdot N(C^2(x_i=1)),\\ &N(C^2(x_i=1)\setminus\{c_j\})\cdot N(C^1(x_i=1))\}\\ &+\max\{N(C^1(x_i=0)\setminus\{c_j\})\cdot N(C^2(x_i=0)),\\ &N(C^2(x_i=0)\setminus\{c_j\})\cdot N(C^1(x_i=0))\}\end{aligned}$$

It remains to formulate both the subprocedures refered to above for managing the cases where the formula graph is a tree or more specifically is a single path. In the latter case it only remains to treat paths with v-vertices at both ends.

Procedure_Path
Input: Path P starting/ending with a v-vertex.
Output: $N(P)$.

1. Let $\{x_1, \ldots, x_n\}$ be the variable set of the formula underlying P and let $\alpha : \{x_2, \ldots, x_n\} \longrightarrow \{0, 1\}$ be the following model for P: For all $i = 2, \ldots, n$

$$\alpha(x_i) = \begin{cases} 1, & \text{if } x_i \in c_{i-1} \\ 0, & \text{if } \overline{x_i} \in c_{i-1} \end{cases}$$

By definition α satisfies all clauses of P. So the value of x_1 is irrelevant for satisfying P.

2. Initially: $M(\alpha(x_2)) = 2$ and $M(\overline{\alpha(x_2)}) = 1$.

3. For $i = 3$ to n do
$M(\alpha(x_i)) = M(\alpha(x_{i-1})) + M(\overline{\alpha(x_{i-1})})$ and

$$M(\overline{\alpha(x_i)}) = \begin{cases} M(\alpha(x_{i-1})), \text{if } \ell(x_{i-1}) \in c_{i-1} \cap c_{i-2} \\ M(\overline{\alpha(x_{i-1})}), \text{else.} \end{cases}$$

4. $N(P) = M(\alpha(x_n)) + M(\overline{\alpha(x_n)})$.

Procedure_Tree
Input: Tree B.
Output: $N(B)$.

1. If B is a v-tree such that there is a v-vertex x_p in B which is adjacent to at least three c-vertices. Then proceed as follows. Let HP be the main path of B and let x_p be a v-vertex of the main path, which is adjacent to $l \geq 1$ c-vertices $c_1, \ldots, c_l$ that do not lie on HP and let α be a model for HP. For every v-vertex x_p of the main path, let $B^1_{x_p}, \ldots, B^l_{x_p}$ be those subtrees of B intersecting HP in x_p. Recursively compute them by applying Algorithm 1 to $B^i_{\alpha(x_p)}$ and to $B^i_{\overline{\alpha(x_p)}}$, for $i = 1, \ldots, l$. Here $B^i_{\alpha(x_p)}$ denotes the tree corresponding to the formula underlying $B^i_{x_p}$ when it is evaluated according to $\alpha(x_p)$, similarly for $B^i_{\overline{\alpha(x_p)}}$. By Algorithm 1 which calls Procedure_Path the main path HP is treated. As soon as x_p is reached we have

$$\begin{aligned} M(\alpha(x_p)) = &\left[M(\alpha(x_{p-1})) + M\left(\overline{\alpha(x_{p-1})}\right)\right] \\ &\cdot N\left(B^1_{\alpha(x_p)}\right)\cdot \ldots \cdot N\left(B^l_{\alpha(x_p)}\right)\end{aligned}$$

and

$$M(\overline{\alpha(x_p)}) = \begin{cases} M(\alpha(x_{p-1}))\cdot \Pi_{i=1}^l N(B^i_{\overline{\alpha(x_p)}}), \text{if } \ell(x_{p-1}) \in c_{p-1} \cap c_{p-2} \\ M(\overline{\alpha(x_{p-1})})\cdot \Pi_{i=1}^l N(B^i_{\overline{\alpha(x_p)}}), \text{else.} \end{cases}$$

2. If B contains a c-vertex c_i, which is adjacent to $r \geq 3$ v-vertices $x_{i_1}, \ldots, x_{i_r}$ then partition B into k subtrees $B_{x_{i_1}} \ldots, B_{x_{i_r}}$ such that every subtree is connected with the other $r-1$ subtrees by c_i only. Let $B_{x_{i_j}}$ denote the subtree including x_{i_j}, for $1 \leq j \leq r$, without the c-vertex c_i. For every $B_{x_{i_j}}$, $1 \leq j \leq r$, compute $N(B_{x_{i_j}})$ as follows:

   (a) If $B_{x_{i_j}}$ is a path, then $N(B_{x_{i_j}})$ is determined by Procedure_Path.

   (b) If $B_{x_{i_j}}$ is a v-tree then it has a v-vertex x_p which is adjacent to at least three c-vertices. In this case compute $N(B_{x_{i_j}})$ by applying Step 1.

   (c) If $B_{x_{i_j}}$ has a c-vertex which is adjacent to at least three v-vertices, then compute $N(B_{x_{i_j}})$ by applying Step 2.

   (d) Compute

$$N(B) = \Pi_{j=1}^r N(B_{x_{i_j}}) - \Pi_{j=1}^r N(B_{x_{i_j}\neg c_i})$$

   Here $N(B_{x_{i_j}\neg c_i})$, for $j = 1, \ldots, r$, denotes the number of all models of $B_{x_{i_j}}$ where the variable x_{i_j} is set such that it does not satisfy the clause c_i.

The main result of this section is as follows:

Theorem 1 *The counting problem* #SAT *for outerplanar formulas with n variables is solvable in time $O(n^{5,13})$. For outerplanar formulas whose graph is either free of cycles or consists of disjoint chordless cycles only* #SAT *can be solved in linear time.*

PROOF. We establish the theorem by proving the correctness and stated time complexity of Algorithm 1 starting with analysing its time complexity. Let $C \in$ CNF be defined over n variables and admitting a connected outerplanar graph G_C. If G_C is a tree, thus free of cycles, the

ISBN: 978-988-19253-4-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

number of models of C can be determined in linear time. As we visit each vertex of G_C as well in Procedure_Tree as also in Procedure_Path only once, each of both procedures takes linear running time. The same argument holds in case G_C consists of pairwise disjoint and chordless cycles only, because by setting an arbitrary variable of such a cycle yields a path.

If G_C has cycles, then we treat G_C recursively by the separator theorem: We determine two vertices x_i and c_j such that G_C is partitioned in two subgraphs G_C^1 and G_C^2 which have only the two vertices x_i and c_j in common and it holds that neither G_C^1 nor G_C^2 contains more than $\frac{2n}{3}$ v-vertices. Further there is no edge connecting a vertex from $G_C^1 \setminus \{x_i, c_j\}$ with a vertex from $G_C^2 \setminus \{x_i, c_j\}$. From G_C^1 we get $G_C^1(x_i = \epsilon)$ by setting $x_i = \epsilon$ in G_C^1 then eliminating all clauses which are satisfied by $x_i = \epsilon$ ($\epsilon \in \{0, 1\}$) and further eliminating x_i and all incident edges. Analogously one obtains from G_C^2 the subgraphs $G_C^2(x_i = \epsilon)$, for $\epsilon \in \{0, 1\}$. Then from $G_C^a(x_i = \epsilon)$ build $G_C^a(x_i = \epsilon) \setminus \{c_j\}$, by eliminating vertex c_j, for $a = 1, 2$ and $\epsilon \in \{0, 1\}$. Then apply Algorithm 1 to the following eight subgraphs:
$G_C^1(x_i = 1)$,
$G_C^1(x_i = 0)$,
$G_C^2(x_i = 1)$,
$G_C^2(x_i = 0)$,
$G_C^1(x_i = 1) \setminus \{c_j\}$,
$G_C^1(x_i = 0) \setminus \{c_j\}$,
$G_C^2(x_i = 1) \setminus \{c_j\}$ and
$G_C^2(x_i = 0) \setminus \{c_j\}$.

As soon as a subgraph is free of cycles, we can compute the number of all its models in linear time by Procedure_Tree or by Procedure_Path.

Let $T(n)$ be the running time to compute the number of all models of an outerplanar formula with n variables. Then we obtain the following recurrence for the running time

$$T(n) = 8 \cdot T\left(\frac{2}{3}n\right) + O(n) + O(1), n \geq 2.$$

where $T(1) = O(1)$. The separator set for an outerplanar graph can be computed in $O(n)$ time. In every step of the recursion eight new subgraphs are obtained, to each of which Algorithm 1 is applied. Since with every separation step the variable set has diminished to at most $2/3$ of the variable set of the previous graph the recursion tree has maximal depth l satisfying $(2/3)^l n = 1$. Therefore $l = \log_{3/2}(n) = \log_2(n)/\log_2(3/2)$. Combining the solutions of the different subgraphs obviously consumes constant time. Thus the solution of the recurrence is $T(n) = O(n^{\log_{3/2} 8}) = O(n^{5.13})$ proving the claimed time complexity.

Regarding the correctness of Algorithm 1, first consider Procedure_Path which is applied to P. Here the following holds true. For every $i \in \{2, \dots, n\}$, $M(\alpha(x_i))$ is the number of all models assigning the variables $x_i, \dots, x_n$ according to α. Moreover $M(\overline{\alpha(x_i)})$ is the number of all models assigning the variables $x_{i-1}, \dots, x_n$ according to α, but $x_i = \overline{\alpha(x_i)}$. Hence

$$N(P) = M(\alpha(x_n)) + M(\overline{\alpha(x_n)})$$

provides the number of all models for the path P. Similarly, regarding Procedure_Tree one obtains the following. If B is a v-tree then Algorithm 1 using Procedure_Path is applied to the main path HP. For every variable x_p of the main path which is adjacent to $l > 2$ c-vertices $c_1, \dots, c_l$ the following invariant is valid:

$$M(\alpha(x_p)) = (M(\alpha(x_{p-1})) + M(\overline{\alpha(x_{p-1})}))\Pi_{i=1}^{l} N(B^i_{\alpha(x_p)})$$

and

$$M(\overline{\alpha(x_p)}) = \begin{cases} M(\alpha(x_{p-1})) \cdot \Pi_{i=1}^{l} N(B^i_{\overline{\alpha(x_p)}}), \text{if } \ell(x_{p-1}) \in c_{p-1} \cap c_{p-2} \\ M(\overline{\alpha(x_{p-1})}) \cdot \Pi_{i=1}^{l} N(B^i_{\overline{\alpha(x_p)}}), \text{else.} \end{cases}$$

That means B is partitioned into paths such that one only needs to treat recursively each single path by Algorithm 1 and finally one has to combine the number of the solutions at the common variables.

If B is a tree with a c-vertex c which is adjacent to at least three v-vertices then we partition B into r subtrees, where r is the number of the v-vertices which are adjacent to c, such that every two of the r subtrees have only vertex c in common. Then remove c from every subtree and compute for every subtree the number of all its models separately. Next multiply all these numbers and subtract the number of all truth assignments not satisfying c.

If G_C is neither a path nor a tree then G_C must have a cycle. Then we treat G_C by the divide and conquer strategy using the separator theorem. □

4 The Case of k-Outerplanar Formulas

Here we extend the concepts of the previous section in order to treat #SAT for CNF formulas with k-outerplanar graphs, for $k \geq 2$. A k-outerplanar graph can easily be partitioned into two subgraphs with at most $2k$ common separator vertices by the separator theorem [2]: Let G be any n-vertex k-outerplanar graph. The vertex set of G can be partitioned into three parts V_1, V_2, S such that no edge joins a vertex in V_1 with a vertex in V_2, with $|V_i| \leq 2n/3$, $i = 1, 2$, $|S| \leq 2k$, and separator set S can be computed in linear time.

Algorithm 2
Input: C k-outerplanar, $|V(C)| = n$.
Output: $N(C)$.

1. If G_C is outerplanar, then compute $N(C)$ by Algorithm 1.

ISBN: 978-988-19253-4-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

2. As long as there is a c-vertex c_j in G_C which is adjacent to one v-vertex x_i only, we set x_i such that it satisfies c_j and simplify the formula. That means we remove all c-vertices satisfied by x_i. Further we remove x_i and all its incident edges.

3. If there is an isolated c-vertex (empty clause) then the formula is unsatisfiable and the procedure terminates with output $N(C) = 0$.

4.
   - Determine the separator set $S = \{x_1, \ldots, x_l\}$ of G_C, $l \leq 2k$, and V_1, V_2 such that there is no edge joining a vertex of V_1 with a vertex of V_2 and $|V_1|, |V_2| \leq \frac{2n}{3}$.
   - Next consider the induced subgraphs $G_1 := G[V_1 \cup \{x_1, \ldots, x_l\})]$ and $G_2 := G[V_2 \cup \{x_1, \ldots, x_l\}]$ of G_C and let C^i denote the subformula of C underlying G_i, for $i = 1, 2$.
   - Let $t_1, \ldots, t_{2^l}$ be the distinct truth assignments over the variables $x_1, \ldots, x_l$. For each fixed t_j, $1 \leq j \leq 2^l$, the value $N(C^i_{t_j(x_1,\ldots,x_l)})$, for $i = 1, 2$, is computed by Algorithm 2. Recall that $C^i_{t_j(x_1,\ldots,x_l)}$ denotes the evaluation of C^i according to t_j: All satisfied clauses in C^i have to be removed from C_i, and the literals $\ell(x_1), \ldots, \ell(x_l)$ have to be removed from all remaining clauses accordingly, $i = 1, 2$.
   - Compute

$$N(C) = \sum_{j=1}^{2^l} \left(N(C^1_{t_j(x_1,\ldots,x_l)}) \cdot N(C^2_{t_j(x_1,\ldots,x_l)}) \right)$$

Theorem 2 *Algorithm 2 needs $O(n^{1.7(2k+1)})$ time to compute the number of all models for a k-outerplanar formula C with n variables, where $k \geq 1$ is a fixed integer.*

Proof. Let $T(n)$ denote be the number of iterations for determining $N(C)$ with C an arbitrary k-outerplanar formula of n variables. So, $T_k(1) = O(1)$ and for $n \geq 2$, $T_k(n) = 2^{2k+1} T_k\left(\frac{2}{3}n\right) + O(n) + O(2^{2k+1})$. For fixed values of k, the last equation simplifies to $T_k(n) = 2^{2k+1} T_k(\frac{2}{3}n) + O(n)$ whose solution is $T_k(n) = O(n^{1.7(2k+1)})$ finishing the proof. □

Next, nested formulas are discussed for which SAT can be decided in linear time [10]. We aim at showing that the number of models of a nested formula can be determined in $O(n^{8.5})$ polynomial-time. For our purposes the following characterization turns out to be useful [11]: A formula $C = \{c_1, \ldots, c_m\}$ is nested, if there is an ordering $V(C) = \{x_1, \ldots, x_n\}$ such that the graph $G^V_C := (V(C) \cup C, E)$ with $E = E(G_C) \cup \{\{x_i, x_{i+1}\} : 1 \leq i \leq n\}$ admits a planar embedding where the boundary of the outer face coincides with the cycle $x_1 \cdots x_m$.

Obviously the graph G_C of a nested formula C which is obtained from G^V_C by eliminating the edges between all v-vertices is at most 2-outerplanar. This can easily be seen by removing all vertices from the outer face of G_C: Removing all vertices of the outer face particularly means removing all the v-vertices and hence there are left only c-vertices. Since there is no edge between two c-vertices the remaining graph consists of isolated vertices only and thus is outerplanar. Therefore a nested formula is 2-outerplanar and according to Theorem 2 we obtain the following result.

Theorem 3 #SAT *for nested formulas can be solved in time $O(n^{8.5})$.*

5 Conclusions and Problems

We have discussed the class of k-outerplanar CNF formulas, for arbitrary fixed integer $k \geq 1$ and as the main result we proved that #SAT for formulas over n variables can be solved in $O(n^{1.7(2k+1)})$ time. Results for more specific so-called level-planar formulas can be found in [17]. It is an open problem whether #SAT for the class of nested formulas can be solved faster than in time $O(n^{8.5})$. Then it would be interesting to investigate whether #SAT for the class of 2-outerplanar formulas can be solved faster than in $O(n^{8.5})$ running time. As mentioned earlier a k-outerplanar graph has a type-2 $\frac{1}{2}(n - (3k - 1))$-separator of size at most $3k$ [3]. So it would be interesting to find out, whether this approach can be used to obtain a better running time for #SAT on k-outerplanar formulas. Clearly, there also is an indirect approach for classifying the complexity in principle based on monadic second order logic [5, 17], however in this paper we preferred an explicit approach. Finally, one could ask for fixed-parameter complexity [6] results in this context regarding the parameter k.

References

[1] Aspvall, B., Plass, M.R., Tarjan, R.E., "A linear-time algorithm for testing the truth of certain quantified Boolean formulas," *Inform. Process. Lett.* pp. 121-123, 8/1979.

[2] Baker, B.S., "Approximation algorithms for NP-complete problems on planar graphs," *J. Assoc. Comput. Mach.*, pp. 153-180, 41/1994.

[3] Bodlaender, H.L., "A partial k-arboretum of graphs with bounded treewidth," *Theoret. Comp. Sci.*, pp. 46-52, 209/1998.

[4] Cook, S.A., "The Complexity of Theorem Proving Procedures," *3rd ACM Symposium on Theory of Computing*, pp. 151-158, 1971.

ISBN: 978-988-19253-4-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

[5] Courcelle, B., Makowsky, J.A., Rotics, U., "On the Fixed Parameter Complexity of Graph Enumeration Problems Definable in Monadic Second Order Logic," *Discr. Appl. Math.*, pp. 23-52, 108/2001.

[6] Downey, R.G., Fellows, M.R., *Parameterized Complexity*, Springer-Verlag, New York, 1999.

[7] Golumbic, M.C., *Algorithmic Graph Theory and Perfect Graphs*, Academic Press, New York, 1980.

[8] Gu, J., Purdom, P.W., Franco, J., Wah, B.W., "Algorithms for the Satisfiability (SAT) Problem: A Survey," in: D. Du, J. Gu, P. M. Pardalos (Eds.), *Satisfiability Problem: Theory and Applications*, DIMACS Workshop, March 11-13, 1996, DIMACS Series, V35, pp. 19-151, American Mathematical Society, Providence, Rhode Island, 1997.

[9] Karp, R.M., "Reducibility Among Combinatorial Problems," in: *Proc. Sympos. IBM Thomas J. Watson Res. Center, Yorktown Heights, N.Y.*, New York: Plenum, pp. 85-103, 1972.

[10] Knuth, D.E., "Nested satisfiability," *Acta Informatica*, pp. 1-6, 28/1990.

[11] Kratochvil, J., Krivanek, M., "Satisfiability of co-nested formulas," *Acta Informatica*, pp. 397-403, 30/1993.

[12] Le Berre, D., Simon, L., "The Essentials of the SAT 2003 Competition," *Lecture Notes Comp. Ssi.*, pp. 172-187, 2919/2004.

[13] Lipton, R.J., Tarjan, R.E., "A separator theorem for planar graphs," *SIAM J. Appl. Math.*, pp. 177-189, 36/1979.

[14] Maheshwari, A., Zeh, N., "External Algorithms for Outerplanar Graphs," *Lecture Notes Comp. Ssi.*, pp. 307-316, 1741/1999.

[15] Minoux, M., "LTUR: A Simplified Linear-Time Unit Resolution Algorithm for Horn Formulae and Computer Implementation," *Inform. Process. Lett.*, pp. 1-12, 29/1988.

[16] Schaefer, T.J., "The complexity of satisfiability problems," *Conference Record of the Tenth Annual ACM Symposium on Theory of Computing*, San Diego, California, pp. 216-226, 1978.

[17] Schmidt, T., *Computational complexity of SAT, XSAT and NAE-SAT for linear and mixed Horn CNF formulas*, dissertation, Univ. Köln, 2010.

[18] Speckenmeyer, E., Min Li, C., Manquinho V., Tacchella, A., (Eds.), "Special Issue on the 2007 Competitions," *J. Satisf. Boolean Modeling, Comp.*, 4/2008.

[19] Valiant, L., "The complexity of enumeration and reliability problems," *SIAM J. Comput.*, pp. 410-421, 9/1979.

ISBN: 978-988-19253-4-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)