Proceedings of the International MultiConference of Engineers and Computer Scientists 2010 Vol I,
IMECS 2010, March 17 - 19, 2010, Hong Kong

Minimizing Variable-Weighted X3SAT

Stefan Porschen, Galyna Plagge *

Abstract—**In this paper, we present an upper bound of $O(2^{0.1625n})$ for the minimum-weight exact 3-satisfiability problem (MINW-X3SAT) getting as input 3-CNF formulas over n real-valued weighted propositional variables. This problem is NP-hard and the best previous result is an exact algorithm solving MINW-XSAT with no restrictions on clause length in time $O(2^{0.2441n})$ [9].**

Keywords: weighted exact 3-satisfiability, branch-and-reduce algorithm, minimum perfect matching, NP-completeness

1 Introduction

The propositional satisfiability problem (SAT) for conjunctive normal form (CNF) formulas is a prominent problem, namely one of the first problems that have been proven to be NP-complete [3]. SAT has a wide range of applications because many problems can be encoded as a SAT problem. Weighted satisfiability problems, by which we mean that the propositional variables are the weighted objects, provide natural generalizations of SAT and also have important applications, e.g. in the area of code generation [1, 7].

The present paper is devoted to study a variant of weighted SAT, namely the weighted exact 3-satisfiability problem. The corresponding decision problem X3SAT is known to be NP-complete [11]. We investigate here the computational time complexity of optimizing solutions of X3SAT for weighted CNF formulas containing no clauses of length greater than three (3-CNF) as input. Note that the weighted variant of a search problem can increase its computational complexity considerably as is demonstrated by the transition from 2-SAT to variable weighted 2-SAT generalizing the vertex-weighted vertex cover problem. Thus we are motivated to consider the weighted optimization variant of X3SAT.

Recently, XSAT and X3SAT attracted much attention [2, 5, 10, 6]. However the first breakthrough-result [8] dates back to the year 1980 and provides an algorithm deciding XSAT in $O(2^{0.2441n})$ time, for input formulas over n variables. Until 2003 this bound has been the best known, then, based on the same techniques, it has been improved to $O(2^{0.2325n})$ [2]. The best known result for unweighted X3SAT provides a solution in $O(2^{0.1379n})$ time [2]. The weighted optimization problem MINW-XSAT was treated for the first time in [9]. This algorithm essentially uses the branching strategy provided in [8], and benefits from simplification steps preserving the minimum weight XSAT status of each intermediate weighted formula. It solves MINW-XSAT in worst case time $O(2^{0.2441n})$.

We want to use the restriction on the clause length to reach some improvement for 3-CNF input instances. Our algorithms are based on the same methods as the one from [9], however, using certain branching steps we achieve time $O(2^{0.1625n})$.

*Institut für Informatik, Universität zu Köln, D-50969 Köln, Germany, Email: {porschen,plagge}@informatik.uni-koeln.de

2 Basic definitions and notation

A *literal* is a Boolean variable $a \in \{0,1\}$ or its negation $\overline{a}$. We denote the *complement* of a literal l as $\overline{l}$. A *clause* C is the disjunction of different literals and is represented as a literal set. A *k-clause* is a clause that contains exactly k literals. A CNF formula F is a conjunction of different clauses and is represented as a clause set. A k-CNF formula is a formula that contains only clauses of a maximum length k. For a given formula F (resp. clause C), we denote the set of contained variables by $V(F)$ (resp. $V(C)$). Similarly, $V(l)$ denotes the underlying variable of a literal l. $\mathrm{pol}(l)$ denotes the polarity of a literal in a fixed clause of a formula. By $V_+(F)$ (resp. $V_-(F)$) we denote the set of all variables occurring positive (resp. negated). We call a variable (resp. a literal corresponding to it) *unique* if it occurs in the formula only once. We distinguish between the *length* of a formula $\|F\|$ and the number of its clauses $|F|$. Let CNF denote the set of all formulas and let CNF_+ denote the set of *positive monotone formulas*, i.e., each clause contains only unnegated variables. $F \in \mathrm{CNF}_+$ is called a *matching formula*, if each $a \in V(F)$ occurs at most twice in F. A CNF formula F is called *linear* if each pair of distinct clauses F has at most one variable in common. Given a formula F and a variable a, the formula $F[a \leftarrow 1]$ (resp. $F[a \leftarrow 0]$) is obtained from F by setting the value of a to 1 (resp. 0). The restriction of a mapping $f : A \to B$ to a subset $A_0 \subseteq A$ is denoted as $f|_{A_0}$.

The exact 3-satisfiability problem (X3SAT) asks in its decision version, whether there exists a truth assignment $t : V(F) \leftarrow \{0,1\}$, setting exact one literal to 1 in each clause of F. We call such an assignment t an *x-model*, and we denote with X3SAT the set of all exact satisfiable 3-CNF formulas. In the search version of X3SAT one has to decide whether $F \in$ X3SAT, and in the positive case to

ISBN: 978-988-17012-8-2
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

find an x-model of F. We obtain an optimization variant of X3SAT, when weights are assigned to the variables. A *weighted formula* is a pair (F, w) consisting of $F \in$ 3-CNF and a weight function $w : V(F) \to \mathbb{R}$. The problem MINW-X3SAT for a weighted formula (F, w) asks whether $F \in$ X3SAT, and in the positive case one has to find a minimum x-model for F, i.e., an x-model t with the least weight among all x-models of F. The weight of a model is defined by $w(t) = \sum_{a \in V(F)} w(a)t(a)$. MINW-X3SAT is an NP-hard optimization problem.
We observe that the empty set is also a formula ($\emptyset \in$ CNF), which is exactly satisfiable. However, a formula F containing the empty clause ($\square \in F$) cannot be exactly satisfiable.

3 Structure of the Algorithm

The main method of our algorithm solving MINW-X3SAT is *branching*. Branching is a technique, which, for a given formula F, treats two new formulas $F[a \leftarrow 0]$ and $F[a \leftarrow 1]$, where $a \in V(F)$ is a variable of F. Both new formulas obviously contain at least one variable less than F. Branching splits the original problem in two problems with a smaller number of variables and treats these recursively. We note that F is exact satisfiable if and only if one of the smaller formulas $F[a \leftarrow 0]$ or $F[a \leftarrow 1]$ is exact satisfiable. Via recursive calls of branching steps we get a binary tree structure. This *branching tree* is generated in a depth first search manner. Its root corresponds to the weighted input formula (F, w) and all other nodes correspond to those weighted formulas that are calculated by branching at a variable of its parent node formula and simplifying afterwards. The leaf nodes of the tree correspond to weighted matching formulas. For matching formulas as input, we can solve MINW-X3SAT in polynomial time (see Section 5).
Each leaf defines an unique path to the root in the branching tree. Suppose that each simplification step performed at every weighted node formula preserves its *minimum weight* X3SAT *status*, i.e., a minimum weight X3SAT solution for the non-simplified formula can be obtained in polynomial time from the minimum weight X3SAT solution for the simplified one. Hence we want to traverse each path bottom up to the root setting variables according to the current path and performing the reverse simplification steps. This way we obtain a MINW-X3SAT solution with respect to the current path. A global MINW-X3SAT solution can be obtained by simply comparing the weights of path solutions.
By storing the branching and simplification information on a stack we are able to obtain the correct inverse transformations for each step in constant time when traversing bottom up along that path to the root node.
The algorithm uses two parameters *currsol* and *sol* for storing the current local, resp. the current global solution. Both parameters are initialized with **nil**. The weights of corresponding solutions, i.e., *currsol_weight* and *sol_weight*, are initialized with ∞. Further, a stack S is used for storing the history of the path from the root to the current node in the branching tree. For a current node, this history consists of the simplifications made in each predecessor node during a Procedure Simplify and also by assignments of the branching variables.
If the current formula is a matching formula, then a leaf of the tree is reached. The Procedure MinPerfMatch searches for a minimum weight X3SAT solution for the current weighted matching formula. Afterwards, in the Procedure ReturnToRoot, the operations stored on the stack are performed on the found local solution inversely and in reversed order yielding a minimum solution with respect to the current branching tree path.

Algorithm MINW-X3SAT
Input: $F \in$ 3-CNF; $w : V(F) \to \mathbb{R}$
Output: Solution for (F, w) if $F \in$ X3SAT, else **nil**.

```
(01) if contradiction in variable assignment then return nil
(02) Simplify(F, w, currsol, S);
(03) if □ ∈ F then return nil
(04) if F is a matching formula then
(05)      MinPerfMatch(F, w, currsol, S);
(06)      if currsol ≠ nil then ReturnToRoot(F, w, currsol, S);
(07)      if sol_weight > currsol_weight then sol ← currsol;
(08) if ∃a ∈ V(F) occurring ≥ 4 times then
(09)      MINW-X3SAT(F[a ← 1], w, currsol, S);
(10)      MINW-X3SAT(F[a ← 0], w, currsol, S);
(11) if {ā, x1, y1}, {a, x2, y2}, {a, x3, y3} ∈ F then
(12)      MINW-X3SAT(F[a ← 1], w, currsol, S);
(13)      MINW-X3SAT(F[a ← 0], w, currsol, S);
(14) if {a, x1, y1}, {a, x2, y2}, {a, x3, y3}, {x3, l1, l2}, {x1, x2, z} ∈ F
        s.t. V(pi) = V(xi), i = 1, 2, V(z) ∈ {V(x3), V(y3)} then
(15)      MINW-X3SAT(F[x3 ← 1], w, currsol, S);
(16)      MINW-X3SAT(F[x3 ← 0], w, currsol, S);
(17) if {a, x1, y1}, {a, x2, y2}, {a, x3, y3} ∈ F then
(18)      MINW-X3SAT(F[a ← 1], w, currsol, S);
(19)      MINW-X3SAT(F[a ← 0], w, currsol, S);
```

4 Procedure Simplify

This section is devoted to state the transformation rules used in our algorithm. Each rule performs a transformation on a weighted formula, which is a pair (F, w)
Before beginning, we state an easy but useful assertion proved in [9]. Let $T(F)$ denote the set of all x-models of F. Similarly, for a given weight function $w : V(F) \to \mathbb{R}$, let $T_{\min}(F, w) \subset T(F)$ denote the set of all minimum x-models of (F, w).

Lemma 1 *[9] For $F, F' \in$ CNF and two weight functions $w : V(F) \to \mathbb{R}$, $w' : V(F') \to \mathbb{R}$, assume that there exists a bijection $B : T(F) \to T(F')$, $t \mapsto t'$ such that*

$$(*) : w(t) = w'(t') + \alpha,$$

where $\alpha \in \mathbb{R}$ is a constant independent of t, t'.

ISBN: 978-988-17012-8-2
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Then $B_{\min} := B|_{T_{\min}(F,w)}$ *is a bijection between the two sets* $T_{\min}(F,w)$ *and* $T_{\min}(F',w')$*; and so we have* $|T_{\min}(F,w)| = |T_{\min}(F',w')|$.

Next we state the rules used in our Simplify procedure. Each rule provides a transformation from formula F to formula F'. The corresponding transformation modifying the weight function w to w' accordingly is presented in a Lemma, in each case.
The transformations performed in the Procedure Simplify are invertible and preserve the minimum weight X3SAT status of weighted formulas. It is understood, without explicitly mentioning, that all transformations are managed by the stack.
Each rule followed by a Lemma stating This lemma proposes that there exists a bijection between minimum x-models of (F,w) and (F',w'). Hence, a minimum solution for (F,w) can be obtained from a minimum solution for (F',w').
Rule (1): If a clause C occurs in F more than once, all except for one occurrences of C should be removed.
Rule (2): If a clause $C \in F$ contains the same literal l twice or more, then l must be set to 0 in F'.
Rule (3): If $C \in F$ contains only one literal l, then l must be set to 1 in F'.
Rule (4): If $a \in V(F)$ occurs only negated, then substitute $\overline{a} \leftarrow a$.

Lemma 2 *Given* $F \in$ 3-CNF, $w : V(F) \to \mathbb{R}$. *Let* $a \in V(F)$ *be a variable occuring in the formula* F *only negated. Let* F' *be a formula obtained from* F *by the substitution* $\overline{a} \leftarrow a$ *and* $w' : V(F') = V(F) \to \mathbb{R}$ *the same weight function as* w*, except* $w'(a) := -w(a)$.
Then we have $|T_{\min}(F,w)| = |T_{\min}(F',w')|$.

Proof. $F \in$ X3SAT iff $F' \in$ X3SAT, because an x-model $t' : V(F') \to \{0,1\}$ explicitly defines the x-model $t : V(F) \to \{0,1\}$ by

$$t(v) := \begin{cases} 1 - t'(a), & v = a \\ t'(v), & \text{otherwise} \end{cases}$$

Relation $(*)$ from Lemma 1 can be established as follows:

$$\begin{aligned} w(t) &= w(a)t(a) + \sum_{v \in V(F)-\{a\}} w(v)t(v) \\ &= -w'(a)(1-t'(a)) + \sum_{v \in V(F')-\{a\}} w'(v)t'(v) \\ &= \sum_{v \in V(F')} w'(v)t'(v) - w'(a) \\ &= w'(t') - w'(a) \quad \square \end{aligned}$$

Rule (5): If a clause $C \in F$ contains more than one unique variable, then we set these variables to 0 in F' except one with minimum weight.
Rule (6): For a 2-clause $C = \{x,y\} \in F$ with $V(x) \neq V(y)$, we can set $x \leftarrow \overline{y}$ in F'. The following lemma demonstrates that these transformations preserve the minimum weight X3SAT status.

Lemma 3 *Given* $F \in$ 3-CNF, $w : V(F) \to \mathbb{R}$ *and* $C = \{x,y\} \in F$. *Then there exists a weighted formula* (F',w') *such that* C *is no more in* F' *and every* $t' \in T_{\min}(F',w')$ *defines exactly one* $t \in T_{\min}(F,w)$.

Proof. Let $V(x) = a$ and $V(y) = b$. We know that $V(C) \cap V(F-C) \neq \emptyset$, because otherwise C would contain two unique variables, which is not permitted by Rule (5). We define F' as the formula obtained from F by removing C and substituting either $y := \overline{x}$ or $\overline{y} := x$. Then we have $V(F') = V(F) - \{b\}$ and $(\dagger) F \in$ X3SAT iff $F' \in$ X3SAT because an assignment to a explicitly defines an assignment to b, so that the clause C is exact satisfiable. We distinguish two cases to define w':
Case (a): $C = \{a, \overline{b}\}$ (analog: $C = \{\overline{a}, b\}$). Then each x-model t of F necessarily fulfills $t(a) = t(b)$. We define w' as

$$w'(v) := \begin{cases} w(a) + w(b), & v = a \\ w(v), & v \in V(F') - \{a\} \end{cases}$$

Now we define a bijection $B : T(F) \to T(F')$ as follows: for each assignment $t \in T(F)$, let $B(t) := t' = t|_{V(F')} \in T(F')$ and, for each assignment $t' \in T(F')$, we set $B^{-1}(t') := t$ with

$$t(v) := \begin{cases} t'(v), & v \in V(F') \\ t'(a), & v = b \end{cases} \in T(F)$$

Notice that the map B is well defined because of $(\dagger)$ and is a bijection, because for every given t the value $B(t)$ is uniquely determined.
Now we prove that the relation $(*)$ in Lemma 1 holds true. Let $t \in T(F)$, then we have with $t(a) = t(b)$:

$$\begin{aligned} w(t) &= w(a)t(a) + w(b)t(b) + \sum_{v \in V(F)-\{a,b\}} w(v)t(v) \\ &= [w(a) + w(b)]t'(a) + \sum_{v \in V(F')-\{a\}} w(v)t'(v) \\ &= w'(t') \end{aligned}$$

The preconditions of Lemma 1 are fulfilled, so $B_{\min} := B|_{T_{\min}(F,w)}$ is the bijection between $T_{\min}(F,w)$ and $T_{\min}(F',w')$.
Case (b): $C = \{a,b\}$ (analogous: $C = \{\overline{a}, \overline{b}\}$). Then each x-model t of F necessarily fulfills $t(a) = 1 - t(b)$. We define w' as

$$w'(v) := \begin{cases} w(a) - w(b), & v = a \\ w(v), & v \in V(F') - \{a\} \end{cases}$$

Similar to (a), we define a map $B : T(F) \to T(F')$ by $B(t) := t' = t|_{V(F')} \in T(F')$ and $B^{-1}(t') := t$ with

$$t(v) := \begin{cases} t'(v), & v \in V(F') \\ 1 - t'(a), & v = b \end{cases} \in T(F)$$

ISBN: 978-988-17012-8-2
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

This map is also well defined and a bijection. We check now the relation $(*)$ from Lemma 1:

$$\begin{aligned} w(t) &= w(a)t(a) + w(b)t(b) + \sum_{v \in V(F)-\{a,b\}} w(v)t(v) \\ &= w(a)t'(a) + w(b)(1-t'(a)) + \sum_{v \in V(F')-\{a\}} w(v)t'(v) \\ &= [w(a)-w(b)]t'(a) + \sum_{v \in V(F')-\{a\}} w(v)t'(v) + w(b) \\ &= w'(t') + w(b) \end{aligned}$$

finishing the proof.□

Rule (7): If a clause $C \in F$ with $|C| = 3$ contains a complemented pair of variables $x, \overline{x}$, set the third literal y to 0 in F' and C does not occur in F'.

Lemma 4 *For $F \in$ 3-CNF and $w : V(F) \to \mathbb{R}$, let $C = \{x, \overline{x}, y\} \in F$ be a clause containing a complemented pair of variables. Let F' be a formula obtained from F by the substitution $y \leftarrow 0$ and the weight function is defined as follows: $w' : V(F') \to \mathbb{R}$ with $w' = w|_{V(F')}$. Then we have $|T_{\min}(F, w)| = |T_{\min}(F', w')|$.*

Proof. Let $a = V(y)$. We have F': $V(F') = V(F) \setminus \{a\}$, because of the definition of F', and $F \in$ X3SAT iff $F' \in$ X3SAT, because an x-model $t' : V(F') \to \{0,1\}$ of F' explicitly defines an x-model $t : V(F) \to \{0,1\}$ of F, namely:

$$t(v) := \begin{cases} 0, & v = a = y \\ 1, & v = a = \overline{y} \\ t'(v), & \text{otherwise} \end{cases}$$

We check now the relation $(*)$ from Lemma 1. We distinguish two cases: (1) $y = a$, the third literal in C is unnegated, hence $t(a) = 0$. Then:

$$w(t) = w(a)t(a) + \sum_{v \in V(F)-\{a\}} w(v)t(v) = w'(t')$$

(2) $y = \overline{a}$, the third literal in C is negated, hence $t(a) = 1$. Then:

$$w(t) = w(a)t(a) + \sum_{v \in V(F)-\{a\}} w(v)t(v) = w'(t') + w(a)$$

For both cases $(*)$ holds, therefore the assertion of this lemma is true. □

Rule (8): If F contains the clauses $\{a, x_1, y_1\}$, $\{a, x_2, y_2\}$, $\{a, x_3, y_3\}$ and C with $C \in \{\{x_1, x_2, x_3\}, \{\overline{x_1}, \overline{x_2}, x_3\}, \{\overline{x_1}, \overline{x_2}, \overline{x_3}\}\}$, then set a to 0 in F'.
To guarantee that $F \in$ X3SAT, one of the literals x_1, x_2 or x_3 must be set to 1. Thus we must set $a \leftarrow 0$. The following lemma proves that the MINW-X3SAT status holds. The proof of this lemma is similar to the one of Lemma 4.

Lemma 5 *For a formula $F \in$ 3-CNF and a function $w : V(F) \to \mathbb{R}$, let $\{a, x_1, y_1\}$, $\{a, x_2, y_2\}$, $\{a, x_3, y_3\}$ and C be clauses in F with $C \in \{\{x_1, x_2, x_3\}, \{\overline{x_1}, \overline{x_2}, x_3\}, \{\overline{x_1}, \overline{x_2}, \overline{x_3}\}\}$. Let F' be a formula obtained from F by the substitution $a \leftarrow 0$ and the weight function $w' : V(F') \to \mathbb{R}$ is defined as follows: $w' = w|_{V(F')}$. Then we have $|T_{\min}(F, w)| = |T_{\min}(F', w')|$.*

Rule (9): If F contains clauses $\{x, y, l_1\}$ and $\{x, y, l_2\}$, then set $l_1 \leftarrow l_2$ in F'.

Lemma 6 *Let $F \in$ 3-CNF, $w : V(F) \to \mathbb{R}$, $\{x, y, l_1\}, \{x, y, l_2\} \in F$. Then there exists a weighted formula (F', w') such $w' : V(F') \to \mathbb{R}$, $V(l_1)$ does not occur in F' and, for each $t' \in T_{\min}(F', w')$, exists exactly one $t \in T_{\min}(F, w)$.*

Rule (10): If a formula F contains clauses $\{x, y, l_1\}$ and $\{\overline{x}, y, l_2\}$, so set $l_1 \leftarrow \overline{x}$, $l_2 \leftarrow x$ and $y \leftarrow 0$ in F'.

Lemma 7 *Let $F \in$ 3-CNF, $w : V(F) \to \mathbb{R}$ and $\{x, y, l_1\}, \{\overline{x}, y, l_2\} \in F$. Then there exists a weighted formula (F', w') such that the variables $V(\{l_1, l_2, y\})$ do not occur in F' and, for each $t' \in T_{\min}(F', w')$, exists exactly one $t \in T_{\min}(F, w)$ and vice versa, for each $t \in T_{\min}(F, w)$, exists exactly one $t' \in T_{\min}(F', w')$.*

Rule (11): If a formula F contains clauses $\{x, y, l_1\}$ and $\{\overline{x}, \overline{y}, l_2\}$, set $l_1 \leftarrow 0$, $l_2 \leftarrow 0$ and $x \leftarrow \overline{y}$ in F'.
The following lemma can be proved similarly to Lemma 3.

Lemma 8 *Let $F \in$ 3-CNF, $w : V(F) \to \mathbb{R}$ and $\{x, y, l_1\}$, $\{\overline{x}, \overline{y}, l_2\} \in F$. Let F' be a formula obtained from F by the substitution $l_1 \leftarrow 0, l_2 \leftarrow 0$, $x \leftarrow \overline{y}$ and the weight function is defined as follows:*

$$w'(v) := \begin{cases} w(b) + w(a), & v = b; \text{pol}(x) \neq \text{pol}(y) \\ w(b) - w(a), & v = b; \text{pol}(x) = \text{pol}(y) \\ w(v), & v \in V(F') \setminus \{b\} \end{cases}$$

with $V(x) = a$ and $V(y) = b$. Then we have $|T_{\min}(F, w)| = |T_{\min}(F', w')|$.

Rule (12): If F' is an independent subformula of F, i.e., $V(F') \cap V(F - F') = \emptyset$ with $|V(F')| \leq 10$, then we can solve MINW-X3SAT for F' in constant time, for example by brute force. It is clear that a MINW-X3SAT solution for the formula F is the union of MINW-X3SAT solutions for F' and $F - F'$. Thus replace F with $F - F'$, if F' is exact satisfiable.

5 Treating Matching Formulas

This section describes how to find a MINW-X3SAT solution for a matching formula $F \in$ 3-CNF in polynomial time. First, the Procedure MinPerfMatch converts a

ISBN: 978-988-17012-8-2
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

matching formula into a positive linear matching formula, so that the MINW-X3SAT status is preserved. Then we construct a certain graph G_F corresponding to the formula F, called *formula graph*, in a way that the minimum perfect matching of G_F corresponds to the MINW-X3SAT solution of F. It is known that the minimum perfect matching problem is solvable in polynomial time [4]. Hence, we can also solve the MINW-X3SAT problem for matching formulas in polynomial time. Primarily, we want to eliminate all occurrences of negated literals in F. The resulting formula F' is not necessarily in 3-CNF. It is only important that a minimum x-model of F' explicitly defines a minimum x-model of F.
The following lemma describes a transformation called *weighted simple resolution*. The weighted simple resolution for XSAT is due to [9].

Lemma 9 *Let $F = \{C_1, C_2, \ldots, C_r\} \in$ CNF be a matching formula and $w : V(F) \to \mathbb{R}$ be a weight function. Further, let $C_i = C_i' \cup \{a\}$ and $C_j = C_j' \cup \{\overline{a}\}$ be clauses in F. We define a new clause $C' := C_i' \oplus C_j'$, where $\oplus$ denotes the symmetrical difference of two clauses, i.e., C' contains the literals occurring only in one clause C_i' or C_j'. We define a new formula $F' := F \setminus \{C_i, C_j\} \cup \{C'\}$. Furthermore, let $w' : V(F') \to \mathbb{R}$ be a weight function. We distinguish two cases to construct w': (1) $C' = C_i' \oplus C_j'$ contains a complementary pair $b, \overline{b}$. Then we define w' as follows:*

$$w'(v) := \begin{cases} w(v), & v \in V(F') \setminus \{b\} \\ w(v) + w(a), & v = b \text{ and } \overline{b} \in C_i',\ b \in C_j' \\ w(v) - w(a), & v = b \text{ and } \overline{b} \in C_j',\ b \in C_i' \end{cases}$$

(2) $C' = C_i' \oplus C_j'$ does not contain a complementary pair. Then we define w' as follows:

$$w'(v) := \begin{cases} w(v), & v \in V(F' \setminus C_i') \\ w(v) + w(a), & v \in V_-(C_i' \setminus C_j') \\ w(v) - w(a), & v \in V_+(C_i' \setminus C_j') \end{cases}$$

Then we have $|T_{\min}(F, w)| = |T_{\min}(F', w')|$.

Now we want to transform a positive matching formula F into a linear positive matching formula F'. This transformation has to ensure that for each minimum x-model of F' a minimum x-model of F can be constructed.

Lemma 10 *Let $F = \{C_1, C_2, \ldots, C_r\} \in$ CNF be a positive matching formula with a weight function $w : V(F) \to \mathbb{R}$. Assume that $C_i, C_j \in F$ contain the same variables $v_1, v_2, \ldots, v_k$ with $k \geq 2$. Without loss of generality, let v_1 be a variable with the least weight, i.e., $w(v_1) \leq w(v_i)$ for all $i \in \{2, 3, \ldots, k\}$. We define new clauses $C_i' := C_i \setminus \{v_2, v_3 \ldots, v_k\}$, $C_j' := C_j \setminus \{v_2, v_3 \ldots, v_k\}$ and a new formula $F' := F \setminus \{C_i, C_j\} \cup \{C_i'\} \cup \{C_j'\}$ with a weight function $w' : V(F') \to \mathbb{R}$, $w' := w|_{V(F')}$.*
Then, for each $t' \in T_{\min}(F', w')$, we can find at least one $t \in T_{\min}(F, w)$.

Proof. According to the definition of F' we have $V(F') = V(F) \setminus \{v_2, v_3 \ldots, v_k\}$. $F \in$ XSAT iff $F' \in$ XSAT, because an x-model $t' : V(F') \to \{0, 1\}$ explicitly defines an x-model $t : V(F) \to \{0, 1\}$ by

$$t(v) := \begin{cases} t'(v), & v \in V(F') \\ 0, & v \in \{v_2, v_3 \ldots, v_k\} \end{cases}$$

And vice versa, we can obtain an x-model $t' : V(F') \to \{0, 1\}$ of F' from one of F:

$$t'(v) := \begin{cases} t(v), & v \in V(F') \setminus \{v_1\} \\ 1, & v = v_1, \exists i \in \{1, 2, \ldots, k\} t(v_i) = 1 \\ 0, & v = v_1, \forall i \in \{1, 2, \ldots, k\} t(v_i) = 0 \end{cases}$$

Let $t' \in T_{\min}(F', w')$. We want to prove that

$$t(v) = \begin{cases} t'(v), & v \in V(F') \\ 0, & v \in \{v_2, v_3 \ldots, v_k\} \end{cases} \in T_{\min}(F, w).$$

Obviously, t is an x-model of F and $w(t) = w'(t')$. We demostrate that t is minimal. Assumed, there exists $t_0 \in T(F)$ with $t_0 \neq t$ and $w(t_0) < w(t)$. We define t_0' as follows:

$$t_0'(v) = \begin{cases} t_0(v), & v \in V(F') \setminus \{v_1\} \\ 1, & v = v_1, \exists i \in \{1, 2, \ldots, k\} t_0(v_i) = 1 \\ 0, & v = v_1, \forall i \in \{1, 2, \ldots, k\} t_0(v_i) = 0 \end{cases}$$

t_0' is an x-model of F', because $t_0 \in T(F)$. It must be fulfilled $w'(t_0') \leq w(t_0)$, because $w'(v_1) = w(v_1) \leq w(v_i)$, $\forall i \in \{2, 3, \ldots, k\}$. Generally it follows that $w'(t_0') \leq w(t_0) < w(t) = w'(t') \Rightarrow w'(t_0') < w'(t')$, That means that we have an x-model of F' with less weight than t'. This is a contradiction to the assumption $t' \in T_{\min}(F', w')$. So our assumption is false and $t \in T_{\min}(F, w)$. □

At first, we apply Lemma 9 and, afterwards, the Rules (2) and (7) as often as possible, i.e., until no more negated variables occur in the current formula. After that, we apply Lemma 10 to the resulting formula, until linearity is achieved. Then we can construct a formula graph G_F, whose minimum perfect matching yields a minimum x-model for the final formula F.

6 Running Time

In this section we analyse the time complexity of the Algorithm MINW-X3SAT. The running time of the algorithm is generally determined by the running time of recursive calls of itself. Hence, except a polynomial factor, the total running time $T(n)$ is defined by the number of leaves of the branching tree. The maximum number of leaves is determined by the maximum of the solutions of the recurrence inequalities, which correspond to the branching steps.
For each branching step we want to determine the number of deleted variables. Then we can construct and solve

ISBN: 978-988-17012-8-2
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

the corresponding recurrences. The dominating recurrence yields the running time of the algorithm which is $T(n) \leq T(n-9)+T(n-4)$ according to the next result.

Lemma 11 *regarding the branching steps in Algorithm* MINW-X3SAT *one obtains the following recurrences: (1) lines 09-10:* $T(n) \leq T(n-9)+T(n-5)$,
(2) lines 12-13: $T(n) \leq T(n-7)+T(n-6)$,
(3) lines 15-16: $T(n) \leq T(n-9)+T(n-5)$,
(4) lines 18-19: $T(n) \leq T(n-9)+T(n-4)$.

Theorem 1 *Algorithm* MINW-X3SAT *is correct and has worst case time complexity* $O(2^{0.16254n})$.

Proof. *Correctness.* The correctness of the algorithm follows from the correctness of the simplifying transformations and the branching. The correctness of the branching remains to be shown. I.e., after a branching step, we want to obtain the MINW-X3SAT solution t for (F,w) from the solutions t'_0 for $(F[a \leftarrow 0], w(V(F[a \leftarrow 0])))$ and t'_1 for $(F[a \leftarrow 1], w(V(F[a \leftarrow 1])))$. We set t, so that $w(t) = \min\{w(t_0), w(t_1)\}$ with

$$t_i(x) = \begin{cases} t'_i(x), & x \in V(F) \setminus \{a\} \\ i, & x = a \end{cases}, \ i = 0, 1.$$

If there are no solutions for both formulas $(F[a \leftarrow 0], w(V(F[a \leftarrow 0]))$ and $(F[a \leftarrow 1], w(V(F[a \leftarrow 1]))$, then there is no solution for the formula (F,w). We assume that there exists a model $t' \in T(F)$ with $w(t) < w(t')$. We distinguish two cases:
(1) $t'(a) = 0$. Then $w(t') \geq w(t_0)$, because of $t_0(a) = 0$ and $t_0|_{V(F[a \leftarrow 0])} = t'_0 \in T_{\min}(F[a \leftarrow 0], w|_{V(F[a \leftarrow 0])})$. Furthermore, by definition of t, we have $w(t) \leq w(t_0)$. Hence, $w(t') \geq w(t_0) \geq w(t)$ is a contradiction to the assumption.
(2) $t'(a) = 1$. Then, similarly as above, $w(t') \geq w(t_1)$, because of $t_1(a) = 1$ and $t_1|_{V(F[a \leftarrow 1])} = t'_1 \in T_{\min}(F[a \leftarrow 1], w|_{V(F[a \leftarrow 1])})$. We must also have t: $w(t) \leq w(t_1)$. Hence, $w(t') \geq w(t_1) \geq w(t)$ is a contradiction to the assumption.
So, our assumption is false and $t \in T_{\min}(F,w)$. The correctness of the algorithm follows by induction based on the invariant above.
Running time. The recurrence $T(n) \leq T(n-9) + T(n-4)$ dominates all other recurrences yielding $T(n) = 1.119252667^n = 2^{0.16254n}$. □

7 Concluding Remarks and Open Problems

In this paper we developed an algorithm solving minimum weight exact 3-satisfiability in time $O(2^{0.16254n})$, where n is the number of weighted variables. The presented algorithm significantly improves on the previously known time bound $O(2^{0.2441n})$ for unrestricted variable-weighted CNF's.

However the up to now best bound for deciding (unweighted) X3SAT is $O(2^{0.1379n})$ [2]. So the investigation whether this bound can also be achieved for the minimization problem MINW-X3SAT is left for future work.

References

[1] Aho, A.V., Ganapathi, M., Tjiang, S.W.: Code Generation Using Tree Matching and Dynamic Programming. ACM Trans. Programming Languages and Systems, 11 (1989) 491-516

[2] Byskov, J.M., Ammitzboll Madsen, B., Skjernaa, B.: New Algorithms for Exact Satisfiability. Theoretical Comp. Science 332 (2005) 515-541

[3] Cook, S.A.: The Complexity of Theorem Proving Procedures. In: Proceedings of the 3rd ACM Symposium on Theory of Computing, pp. 151-158, 1971

[4] Cook W., Rohe, A.: Computing Minimum-Weight Perfect Matchings. INFORMS Journal on Computing 11 (1999) 138-148

[5] Dahllöf, V., Jonsson, P., Beigel, R.: Algorithms for four variants of the exact satisfiability problem. Theoretical Comp. Sci. 320 (2004) 373-394

[6] Kulikov, A., An upper bound $O(2^{0.16254n})$ for Exact 3-Satisfiability. Journal of Mathematical Sciences, 126 (2005) 1995-1999

[7] Liao, S., Kreutzer, K., Tjiang, S.W., Devadas, S.: A New Viewpoint on Code Generation for Directed Acyclic Graphs. ACM Trans. Design Automation of Electronic Systems, 3 (1998) 51-75

[8] Monien, B., Speckenmeyer, E., Vornberger, O.: Upper Bounds for Covering Problems. Methods of Operations Research 43 (1981) 419-431

[9] Porschen, S.: On variable-weighted exact satisfiability problems. Annals of Mathematics and Artificial Intelligence 51 (2007) 27-54

[10] Porschen, S., Randerath, B., Speckenmeyer, E.: Exact 3-Satisfiability is Decidable in Time $O(2^{0.16254n})$. Annals of Mathematics and Artificial Intelligence 43 (2005) 173-193

[11] Schaefer, T.J.: The complexity of satisfiability problems. In: Proceedings of the 10th ACM Symposium on Theory of Computing, pp. 216-226, 1978

ISBN: 978-988-17012-8-2
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)