

# A Polynomial Algorithm for **3-sat**

M Angela Weiss

weiss@ime.usp.br, Universidade de São Paulo, São Paulo - Brasil

January 4, 2012

## Abstract

We present a new method for solving **3-sat** formulas. In this method, the formulas are grouped in digraphs representing unsatisfiable **2-sat** formulas in which we decide satisfiability of the whole formula. We show that using this method we polynomially decide if a given formula **3-sat** is satisfiable or not, solving, in this way, the classic question whether  $\mathbf{P} = \mathbf{NP}$ .

© M Angela Weiss 2012

## 1 Introduction

In this paper, we present a polynomial (in time and space) algorithm to decide **3-sat**. The class of **sat** problems was shown to be **NP-complete**. See Cook, in [1] and Cook and Reckhow in [2] and seminal papers of L. Levin, in [5] and [4]. The goals are to establish lower bounds in complexity, see Meyer and Sotckmeyer in [7] and Meyer in [6]. The literature in this area is rich of very nice surveys, like [3] and [8].

A preliminary version of this paper was published as a technical report ([10]). Another version, based on tableaux methods was posted in

[www.ime.usp.br/~weiss](http://www.ime.usp.br/~weiss)  
(version 2011.05.30)

This new version, simpler and shorter, was written following some advises of Professor Lew Gordeew, to whom I am deeply thankful.

Given a **3-sat** formula the idea is to group clauses with at most the disjunction of two literals then we decide if the groups do represent all possible

choices of grouping the formulas (undecidable) or no (decidable). All the procedures we describe are polynomially bounded.

## 2 Basic Definitions

**Definition 2.1** A **3-sat** formula  $\Psi$  is a conjunction of a number, say  $\mathbf{n}$ , of a disjunction of at most three literals. We write

$$\Psi \equiv (l_1^1 \vee l_1^2 \vee l_1^3) \wedge \cdots \wedge (l_{\mathbf{n}}^1 \vee l_{\mathbf{n}}^2 \vee l_{\mathbf{n}}^3) \equiv C_1 \wedge \cdots \wedge C_{\mathbf{n}}$$

A subformula  $C_k \equiv l_k^1 \vee l_k^2 \vee l_k^3$ ,  $1 \leq k \leq \mathbf{n}$  of  $\Psi$  is called a clause.

The set of literals of a formula  $\Psi$  is denoted by  $\text{Letter}(\Psi)$ .

A pair of a literal together with its negation is called a conjugated pair.

**Note:** We require that no clause  $C_k$  of  $\Psi$ ,  $1 \leq k \leq \mathbf{n}$  have a pair of conjugated literals among its literals. Indeed, if  $C_k \equiv l_k^1 \vee p \vee \neg p$ , then  $C_k \equiv l_k^1$ .

**Notation 2.2** Consider the **3-sat** formula

$$\psi \equiv (p \vee q_{11} \vee q_{12}) \wedge (p \vee q_{21} \vee q_{22}) \wedge \cdots \wedge (p \vee q_{t1} \vee q_{t2})$$

The factorization of  $\psi$  by  $p$ , denoted  $p \vee S_p$ , is the rewritten of  $\Psi$  as (the equivalent) formula

$$p \vee ((q_{11} \vee q_{12}) \wedge (q_{21} \vee q_{22}) \wedge \cdots \wedge (q_{t1} \vee q_{t2})) \quad (1)$$

The literal  $p$  is called the factor of  $S_p$ .

**Definition 2.3** Given a **3-sat** formula  $\Psi$ , a factorization of  $\Psi$  by a set of labels  $\mathbb{L}abel \subseteq \{p_{01}, \neg p_{01}, \dots, p_{0k_0}, \neg p_{0k_0}, p_{11}, \neg p_{11}, \dots, p_{1k_1}, \neg p_{1k_1}\}$  (a set of literals in  $\Psi$ ) is a rewritten of  $\Psi$  by a formula  $\Psi' \equiv (\bigwedge_{l \in \mathbb{L}abel} (l \vee S_l)) \wedge S_{\top}$  so that  $\Psi$  is satisfiable iff  $\Psi'$  is satisfiable as well. We require,

- $S_l$  is a factorization of some clauses of  $\Psi$  by a literal  $l \in \mathbb{L}abel$  or, for some  $1 \leq i \leq k$  and  $1 \leq j \leq k_h$ ,  $S_{l_{ij}}$  is the symbol  $\top$ ;
- For all  $1 \leq l \leq k_0$ ,  $p_{0l}$  and  $\neg p_{0l}$  do not belong to the set of literals of any  $S_{p_{ij}}$
- For all  $1 \leq l_1 < l_2 \leq k_1$ ,  $i = 1$ ,  $p_{1l_1}$  and  $\neg p_{1l_1}$  do not belong to the set of literals of  $S_{p_{1l_2}}$  or  $S_{\neg p_{1l_2}}$ ;

- $S_{\top}$  is at most a 2-sat formula.

**Proposition 2.4** Any 3-sat formula  $\Psi \equiv C_1 \wedge \dots \wedge C_n$  admits a partition.

**Proof:** If the set of literals of  $\Psi$ ,  $\text{Letter}(\Psi) = \{p_1, \dots, p_r, l_1, \neg l_1, \dots, l_s, \neg l_s\}$  is so that for all  $p_i \in \{p_1, \dots, p_r\}$  its conjugates does not belong to  $\text{Letter}(\Psi)$ , then replace  $\Psi$  by  $C_1 \wedge \dots \wedge C_m$ , where  $\{C_1, \dots, C_m\}$  are all the clauses of  $\Psi$  that contains no literals of the set  $\{p_1, \dots, p_r\}$  among their set of literals. In this way, we can assume that  $\text{Letter}(\Psi) = \{l_1, \neg l_1, \dots, l_s, \neg l_s\}$ .

1. Choose the pair  $\{l_1, \neg l_1\}$ ;
2. Partition  $\{C_1, \dots, C_m\}$  in three sets,  $L_1, L_2$  and  $L_3$ , where  $L_1$  is the set of clauses that contains  $l_1$  among their literals,  $L_2$  is the set of clauses that contains  $\neg l_1$  among their literals and no clause in  $L_3$  contains  $l_1$  or  $\neg l_1$  among its literals;
3. Factorize  $\bigwedge_{C \in L_1} C$  by  $l_1$  and factorize  $\bigwedge_{C \in L_2} C$  by  $\neg l_1$ .

After obtaining a formula  $\Psi \equiv (l_1 \vee S_{l_1}) \wedge (\neg l_1 \vee S_{\neg l_1}) \wedge (\bigwedge_{C \in L_3} C)$ , recursively repeat the operation of factorization until obtain a set of conjugated pairs  $\mathbb{Label}_0 = \{p_{01}, \neg p_{01}, \dots, p_{0k}, \neg p_{0k}\}$  and the (equivalent to  $\Psi$ ) formula

$$(p_{01} \vee S_{p_{01}}) \wedge (\neg p_{01} \vee S_{\neg p_{01}}) \wedge \dots \wedge (p_{0k} \vee S_{p_{0k}}) \wedge (\neg p_{0k} \vee S_{\neg p_{0k}}) \wedge (\bigwedge_{C \in L_4} C)$$

where each  $S_l$  is a partition of  $l \in \mathbb{Label}_0$ . Besides,

1. For all  $1 \leq i, j \leq k_0$ , the literals  $p_{0i}$  and  $\neg p_{0i}$  do not belong to the set of literals of  $S_{p_{0j}}, S_{\neg p_{0j}}$  nor to any  $C \in L_4$ .
2. For all  $C \in L_4$ , for all literals  $q \in C$ , there is a  $l \in \mathbb{Label}_0$  so that either  $q$  or  $\neg q$  is a literal of  $S_l$ .

In 2, we ensure that we cannot add to  $\mathbb{Label}_0$  a new pair of conjugated literals and 1 remains true.

Let  $(\bigwedge_{C \in L_4} C)$  and  $p_{11}$  and  $\neg p_{11}$  be so that the clauses of  $\Psi$  contain at least  $p_{11}$  and  $\neg p_{11}$  among their literals. Then

- If the set of clauses of  $\Psi$  that contain  $p_{11}$  among their set of literals is empty, define  $S_{p_{11}} = \top$ . Else, factorize by  $p_{11}$  all the clauses of  $(\bigwedge_{C \in L_4} C)$  that contain  $p_{11}$  among their set of literals and add  $p_{11}$  to the set of labels. If  $p_{11}$  appear as literal of some  $S_l, l \in \mathbb{Label}$ , add  $p_{11}$  to the set of *necessarily false literals*, *Necfs*;

- Perform the same procedure for  $\neg p_{11}$ ;
- Let  $L_5$  be the set of clauses of  $(\bigwedge_{C \in L_4} C)$  that do not contain  $p_{11}$  nor  $\neg p_{11}$  among their literals.

If  $(\bigwedge_{C \in L_5} C)$  is at most a **2-sat** formula, stop the procedure, else, repeat the above procedure for a set of literals  $p_{12}, \neg p_{12}, \dots, p_{1k_1}, \neg p_{1k_1}$  until obtain a **2-sat** formula  $S_{\top}$ .  $\dashv$

From now on, we work with a partitioned formula  $\Psi$  whose set of labels  $\mathbb{L}abel = \{l_1, \dots, l_k\}$  is fixed.

**Proposition 2.5**  *$\Psi$  is satisfiable iff a partitioned version of it is satisfiable.*

**Proof:** Suppose that  $\Psi \equiv C_1 \wedge \dots \wedge C_m \wedge C_{m+1} \dots \wedge C_n$ ,  $\mathbb{L}etter(\Psi) = \{p_1, \dots, p_r, l_1, \neg l_1, \dots, l_s, \neg l_s\}$  and  $\Psi' \equiv C_1 \wedge \dots \wedge C_m$ , where  $\{C_1, \dots, C_m\}$  are the clauses of  $\Psi$  that does not contain literals from the set  $\{p_1, \dots, p_r\}$  among their literals.

Clearly if  $\Psi'$  is unsatisfiable, then  $\Psi \equiv \Psi' \wedge (C_{m+1} \wedge \dots \wedge C_n)$  is unsatisfiable. On the other side, if  $\Psi$  is unsatisfiable, then any valuation would make it false. In particular, the valuations where for all  $s \in \{p_1, \dots, p_r\}$ ,  $v(s)$  is true. Equivalently  $\Psi'$  is false for all valuation. That is,  $\Psi'$  is unsatisfiable.

Our claim follows on noticing that at each step of factorization,  $\Psi'$  is replaced by an equivalent formula.  $\dashv$

**Definition 2.6** *Given a partitioned 3-sat formula  $\Psi$ , define the cylindrical digraph  $\mathbb{C}lndr$ -graph generated by  $\Psi$ ,  $(L, \Rightarrow)$  as*

- $L \subseteq \mathbb{L}etter(\Psi) \cup \{\perp, \top\}$ ;
- If  $a \vee b$  is a clause of  $S_{l_1}, \dots, S_{l_k}$ ,  $l \in \mathbb{L}abel$  or  $l = \top$ , then  $\{a, \neg a, b, \neg b\}$  is contained in the set of vertexes,  $\neg a \Rightarrow b$  and  $\neg b \Rightarrow a$  are edges and both have as label the string  $l_1 \dots l_k$ ;
- If  $a \equiv a \vee \perp$  is a clause of  $S_{l_1}, \dots, S_{l_k}$ , then  $\{a, \neg a, \top, \perp\}$  is contained in the set of vertexes,  $\top \Rightarrow a$  and  $\neg a \Rightarrow \perp$  are edges in  $\mathbb{C}lndr$ -graph both with labels  $l_1 \dots l_k$ ;
- If  $a$  is necessarily false, then  $\{a, \neg a, \top, \perp\}$  is contained in the set of vertexes,  $a \Rightarrow \perp$  and  $\top \Rightarrow \neg a$  are edges of  $\mathbb{C}lndr$ -graph with label  $a$ ;

- If  $\top = S_l$ , form no edge.

**Definition 2.7** Let  $a$  and  $b$  be two vertexes of the cylindrical digraph.

1. A path from  $a$  to  $b$  is a subdigraph of the form

$$a \Rightarrow c^1 \Rightarrow c^2 \Rightarrow \dots \Rightarrow c^k \Rightarrow b$$

2. An interval between  $a$  and  $b$  is the (non empty) subdigraph that contains all paths from  $a$  to  $b$ . Denote the interval  $[a, b]$ ;
3. Given an interval  $[a, b]$  a path in the interval  $[a, b]$  is the set of all paths starting at  $a$  and ending at  $b$ .

If there is no path between  $a$  and  $b$ , we say that there is no interval between them.

**Definition 2.8** Recall that literal  $r \in \mathbb{L}abel \cap (\cup Letter(\Psi))$  is called necessarily false literal. The set of necessarily false literals is denoted by  $NecFls$ .

If an interval  $[\neg a, a]$  is non-empty digraph, then  $a$  is called a necessarily true literal. The set of necessarily true literals is denoted by  $Nec$ .

If  $\neg a \in NecFls$  or  $a$  is an isolated clause, the sequences  $\neg a \Rightarrow \perp \Rightarrow \top \Rightarrow a$  forms an interval.

**Definition 2.9** Given a cylindrical digraph, its set of **closed digraphs**,  $CSD$  is defined as the set of subdigraphs

- 1  $[\neg a, a] \rightarrow [a, \neg b] \rightarrow [\neg b, b]$
- 2  $[\neg a, a] \rightarrow [a, c] \rightarrow [c, \perp]$
- 3  $[\top, \neg d] \rightarrow [\neg d, c] \rightarrow [c, \perp]$

where 1 to 3 denotes the smallest subdigraph of the cylindrical digraph that contains, respectively the intervals,

- 1  $[\neg a, a]$ ,  $[a, \neg b]$  and  $[\neg b, b]$ , for all  $a, b \in Nec$ ;
- 2  $[\neg a, a]$ ,  $[a, c]$  and  $[c, \perp]$ , for all  $a \in Nec$ ,  $c \in NecFls$ ;
- 3  $[\top, \neg d]$ ,  $[\neg d, c]$  and  $[c, \perp]$ , for all  $d, c \in NecFls$ .

**Definition 2.10** Given an interval  $I = [a, b] \rightarrow [b, c] \rightarrow [c, d]$ , a (maximal) path in  $I$  is a path  $P$  that starts at  $a$  and ends at  $d$  and passes through (contains as an element)  $b$  and  $c$ ,

$$seq = a^0 \xrightarrow{l_1} a^1 \xrightarrow{l_2} a^2 \Rightarrow \dots \Rightarrow a^{r-1} \xrightarrow{l_r} a^r \quad (2)$$

where  $b$  and  $c$  belongs to  $\{a^1, a^2, a^{r-1}\}$ ,  $a_0 = a$  and  $a^r = d$ .

**Definition 2.11** Let  $C = \{p_1, \dots, p_m\} \subseteq \mathbb{Label}$  be a consistent set and  $\mathcal{F}$  the set of all mappings assigning to each  $1 \leq i \leq m$  a number between 0 and  $h_i$ , the number of clauses of  $S_{p_i}$ . Denote by  $(q_l \vee r_l)_p$  the  $l^{\text{th}}$  clause of  $S_p$ . For each  $f \in \mathcal{F}$ , let  $\Phi_f$  be the formula

$$\bigwedge_{i=1}^m (q_{f(p_i)} \vee r_{f(p_i)})_{p_i}$$

We are particularly interested in the maximal path in Formula 2. In that case, the set  $C$  is a choice of one element of the set of labels  $l_i$ ,  $1 \leq i \leq r$ . Denote by  $\Psi_{seq}$  be the conjunction of disjunctions,

$$(\neg a^0 \vee a^1) \wedge (\neg a^1 \vee a^2) \wedge \dots \wedge (\neg a^{r-1} \vee a^r) \quad (3)$$

**Theorem 2.12** For all maximal path  $seq$ ,  $\Psi_{seq}$  is unsatisfiable.

**Theorem 2.13**  $\Psi_f$  is unsatisfiable iff there is a  $\Psi_{seq}$  subformula of  $\Psi_f$ .

Use the following (well known) Proposition to prove the above Theorems.

**Proposition 2.14 (Papadimitreou, [9])** Let  $\Phi \equiv (a_0 \Rightarrow a_2) \wedge (a_3 \Rightarrow a_4) \wedge \dots \wedge (a_n \Rightarrow a_{n+1})$  be a **2-sat** formula (use only **implies** and **and** as Boolean connectives). Consider a digraph whose vertexes are the union of the literals of  $\Phi$  together with their negations and whose edges are  $a_i \Rightarrow a_{i+1}$  and  $\neg a_{i+1} \Rightarrow \neg a_i$ ,  $0 \leq 1 < n$ . Then  $\Phi$  is unsatisfiable iff there are paths from  $a_i$  to  $\neg a_i$  and from  $\neg a_i$  to  $a_i$  for some  $0 \leq i < n$ .

**Proof of 2.12 and 2.13:**

**2.12:** Applying Definition 2.9, we obtain three kind of formulas,

$$(a \vee p^1) \wedge (\neg p^1 \vee p^2) \wedge \dots \wedge (\neg p^r \vee a) \wedge (\neg a \vee p^{r+1}) \wedge (\neg p^{r+1} \vee p^{r+2}) \wedge \dots \wedge (\neg p^{s-1} \vee \neg b) \wedge (b \vee p^{s+1}) \wedge (\neg p^{s+1} \vee p^{s+2}) \wedge \dots \wedge (\neg p^{t-1} \vee b) \quad (4)$$

$$(a \vee p^1) \wedge (\neg p^1 \vee p^2) \wedge \dots \wedge (\neg p^r \vee a) \wedge (\neg a \vee p^{r+1}) \wedge (\neg p^{r+1} \vee p^{r+2}) \wedge \dots \wedge (\neg p^{s-1} \vee c) \wedge \neg c \quad (5)$$

$$\neg d \wedge (d \vee p^r) \wedge (\neg p^r \vee p^{r+1}) \wedge \dots \wedge (\neg p^{r-1} \vee c) \wedge \neg c \quad (6)$$

The above three formulas are clearly unsatisfiable, but the associated pair of paths are not so clear. Observe that

$$\begin{aligned} & \neg d \wedge (d \vee p^r) \wedge (\neg p^r \vee p^{r+1}) \wedge \dots \wedge (\neg p^{r-1} \vee c) \wedge \neg c \equiv \\ & (\perp \vee \neg d) \wedge (d \vee p^r) \wedge (\neg p^r \vee p^{r+1}) \wedge \dots \wedge (\neg p^{r-1} \vee c) \wedge (\perp \vee \neg c) \end{aligned}$$

and that  $\perp \Rightarrow \top$  belongs to any path.

Digraph associated, respectively, to formulas 4, 5 and 6,

$$\begin{aligned} & (\neg a \Rightarrow p^1), (p^1 \Rightarrow p^2), \dots, (p^r \Rightarrow a), (a \Rightarrow p^{r+1}), (p^{r+1} \Rightarrow p^{r+2}), \dots, \\ & (p^{s-1} \Rightarrow \neg b), (\neg b \Rightarrow p^{s+1}), (p^{s+1} \Rightarrow p^{s+2}), \dots, (p^{t-1} \Rightarrow b) \\ & (\neg p^1 \Rightarrow a), (\neg p^2 \Rightarrow \neg p^1), \dots, (\neg a \Rightarrow \neg p^r), (\neg p^{r+1} \Rightarrow \neg a), \\ & (\neg p^{r+2} \Rightarrow \neg p^{r+1}), \dots, (b \Rightarrow \neg p^{s-1}), (\neg p^{s+1} \Rightarrow b), (\neg p^{s+2} \Rightarrow \neg p^{s+1}), \dots, \\ & (\neg b \Rightarrow \neg p^{t-1}) \end{aligned}$$

$$\begin{aligned} & (\neg a \Rightarrow p^1), (p^1 \Rightarrow p^2), \dots, (p^r \Rightarrow a), (a \Rightarrow p^{r+1}), (p^{r+1} \Rightarrow p^{r+2}), \dots, \\ & (p^{s-1} \Rightarrow c), (c \Rightarrow \perp) \\ & (\neg p^1 \Rightarrow a), (\neg p^2 \Rightarrow \neg p^1), \dots, (\neg a \Rightarrow \neg p^r), (\neg p^{r+1} \Rightarrow \neg a), (\neg p^{r+2} \Rightarrow \neg p^{r+1}), \\ & \dots, (\neg c \Rightarrow \neg p^{s-1}), (\top \Rightarrow \neg c) \end{aligned}$$

$$\begin{aligned} & (\top \Rightarrow \neg d), (\neg d \Rightarrow p^r), (p^r \Rightarrow p^{r+1}), \dots, (p^{r-1} \Rightarrow c), (c \Rightarrow \perp) \\ & (d \Rightarrow \perp), (\neg p^r \Rightarrow d), (\neg p^{r+1} \Rightarrow \neg p^r), \dots, (\neg c \Rightarrow \neg p^{r-1}), (\top \Rightarrow \neg c) \end{aligned}$$

and the respective pair of paths,

$$\begin{aligned} & (\neg a \Rightarrow p^1), (p^1 \Rightarrow p^2), \dots, (p^r \Rightarrow a) \\ & (a \Rightarrow p^{r+1}), (p^{r+1} \Rightarrow p^{r+2}), \dots, (p^{s-1} \Rightarrow \neg b), (\neg b \Rightarrow p^{s+1}), (p^{s+1} \Rightarrow p^{s+2}), \\ & \dots, (p^{t-1} \Rightarrow b), (b \Rightarrow \neg p^{s-1}), \dots, (\neg p^{r+2} \Rightarrow \neg p^{r+1}), (\neg p^{r+1} \Rightarrow \neg a) \end{aligned}$$

$$\begin{aligned} & \perp \Rightarrow \top \\ & (\top \Rightarrow \neg c), (\neg c \Rightarrow \neg p^{s-1}), \dots, (\neg p^{r+2} \Rightarrow \neg p^{r+1}), (\neg p^{r+1} \Rightarrow \neg a), \\ & (\neg a \Rightarrow p^1), (p^1 \Rightarrow p^2), \dots, (p^r \Rightarrow a), (a \Rightarrow p^{r+1}), (p^{r+1} \Rightarrow p^{r+2}), \dots, \\ & (p^{s-1} \Rightarrow c), (c \Rightarrow \perp) \end{aligned}$$

$$\begin{aligned} & \perp \Rightarrow \top \\ & (\top \Rightarrow \neg d), (\neg d \Rightarrow p^r), (p^r \Rightarrow p^{r+1}), \dots, (p^{r-1} \Rightarrow c), (c \Rightarrow \perp) \end{aligned}$$

**2.13:** Given a **2-sat** formula  $\psi$ , consider a pair of paths associated to  $\psi$ ,

$$\begin{aligned} & (a \Rightarrow b^1), (b^1 \Rightarrow b^2), \dots, (b^n \Rightarrow \neg a) \\ & (\neg a \Rightarrow c^1), (c^1 \Rightarrow c^2), \dots, (c^{n^2} \Rightarrow a) \end{aligned}$$

write

$$(a \Rightarrow b^1), (b^1 \Rightarrow b^2), \dots, (b^n \Rightarrow \neg a), (\neg a \Rightarrow c^1), (c^1 \Rightarrow c^2), \dots, (c^{n^2} \Rightarrow a)$$

Choose literals  $\eta$  and  $\zeta$  so that there are paths  $\eta$  to  $\neg\eta$  and  $\zeta$  to  $\neg\zeta$  so that for no conjugated pair  $\mu$  and  $\neg\mu$  there is a path between  $\mu$  and  $\neg\mu$  contained

in between the paths between  $\eta$  and  $\neg\eta$  or in between  $\zeta$  to  $\neg\zeta$ . Observe that at least  $a$  and  $\neg a$ ,  $\neg a$  and  $a$  can form sequences, so, the smallest sequence can be written. Truncate the above sequence erasing all elements before  $\eta$  and erasing all elements after  $\neg\zeta$ . Obtain intervals

$$[\eta, \neg\eta] \rightarrow [\neg\eta, \neg\zeta] \rightarrow [\neg\zeta, \zeta]$$

where the first and third intervals are associated to necessarily false or necessarily true, according to Definition 2.9.  $\dashv$

**Definition 2.15** Consider  $\mathcal{P}$ , the set of all maximal paths in the intervals in  $\mathcal{CSD}$ . To each  $seq \in \mathcal{P}$ , let  $P_{seq}$  be the set of all consistent sets so that  $\mathbf{v} \in P_{seq}$  iff for all edge  $c \Rightarrow d$  in  $seq$ , there is a literal  $l \in \mathbf{v}$  that belongs to the set of labels of  $c \Rightarrow d$ .

Let  $\mathcal{V}$  be the set of all valuations so that  $v \in \mathcal{V}$  iff there is a maximal path  $seq$  and a consistent set  $\mathbf{v} \in P_{seq}$  in which  $v(p) = \perp$  for all  $p \in \mathbf{v}$ .

Conclude (Theorems 2.12 and 2.13) that all the valuation in  $\mathcal{V}$  are the valuation that falsifies  $\Psi$ . Thus,  $\Psi$  is unsatisfiable iff  $\mathcal{V}$  entails all possible consistent combination over the set of literals. That is  $\mathcal{V} = 2^{\mathbb{L}abel}$ , the set of all mappings from  $\mathbb{L}abel$  onto  $\{0, 1\}$ , with the restriction that for all literal  $p$ ,  $f(p) = 0$  iff  $f(\neg p) = 1$ . Of course, if, for example  $\mathbb{L}abel = \{a, \neg a, b, \neg b\}$ , then if  $\mathbf{v} = \{a\}$ , then we extend  $\mathbf{v}$  to  $\mathbf{v}_1 = \{a, b\}$  and  $\mathbf{v}_2 = \{a, \neg b\}$ . Both valuations  $\mathbf{v}_1$  and  $\mathbf{v}_2$  would make  $\Psi$  false. We resume our decision problem on the verification whether  $\mathcal{V}$  entails all possible consistent combination over the set of labels or no.

For all  $seq \in \mathcal{P}$ , let  $label(seq)$  be the set of all labels of  $seq$ . Using (the classical) interpretation of literals as sets,  $\mathcal{V}$  is contained in

$$\tau \equiv \bigcup_{seq \in \mathcal{P}} \bigcap_{l \in label(seq)} \cup \{v \in l\}$$

Denoting here the complementary of a set by the symbol  $\sim$ ,

$$\sim \tau \equiv \bigcap_{seq \in \mathcal{P}} \bigcup_{l \in label(seq)} \cap \{\sim v \in l\}$$

Let us reduce the question whether  $\sim \tau$  is empty, equivalently, whether  $\Psi$  is satisfiable or not, to a polynomial search over  $\mathcal{CSD}$ .

**Definition 2.16** A set of set of labels  $\mathbf{C} = \{l_1, \dots, l_n\}$  is incompatible if there is a conjugated pair of literals  $p$  and  $\neg p$  in  $\cup \mathbf{C}$  so that for some  $i$  and  $j$  (not necessarily  $i \neq j$ )  $p$  belongs to  $l_i$  and  $\neg p$  belongs to  $l_j$ . If otherwise, we say that  $\mathbf{C}$  is compatible and denote  $comp(l_1, \dots, l_n)$ .

If some  $l_i$  contains  $S_\top$  then  $\mathbf{C}$  is incompatible.

In order to obtain  $\sim \tau$  empty, we require that all antichains (over the partial order generated by the closed digraphs) must be incompatible. So, we search for an antichain in the closed digraph whose label are a non compatible set. The task of deciding if  $\Psi$  is satisfiable resumes on finding an antichain in  $\mathcal{CSD}$  whose set of labels are compatible.

### 3 Algorithm II - Polynomial Decision

We will give the algorithm to decide if a given  $\mathcal{CSD}$  has antichains or no. Our algorithm is based on a partial order naturally given over the  $\mathcal{CSD}$ .

**Definition 3.1** Given a  $\mathcal{CSD} = (V, E, \Rightarrow)$  its sets of source, spillway and roots,  $\mathbb{F}$ ,  $\mathbb{S}$  and  $\mathbb{R}$  is given, respectively, by

$$\begin{aligned}\mathbb{F} &= \{a \mid (\exists c \in V \exists c' \in V (c \neq c') \wedge (a \Rightarrow c \in E) \wedge (a \Rightarrow c' \in E)) \\ &\quad \vee \nexists c (c \Rightarrow a \in E)\} \\ \mathbb{S} &= \{b \mid (\exists c \in V \exists c' \in V (c \neq c') \wedge (c \Rightarrow b \in E) \wedge (c' \Rightarrow b \in E)) \\ &\quad \vee \nexists c (b \Rightarrow c \in E)\} \\ \mathbb{R} &= \{a \in V \mid \nexists b (b \Rightarrow a \in E)\} \subseteq \mathbb{F}\end{aligned}$$

**Definition 3.2** Call an interval  $[a, b]$  contained in  $\mathcal{CSD}$  a tabular array,  $\text{MAXLIN}$ , if both  $a$  and  $b$  are source or spillway and no element in the digraph  $[a, b]$  is a source or a spillway. The interval has  $j$  paths, each of them called a linear path of the form,

$$S^i = a \Rightarrow c^{i1} \Rightarrow c^{i2} \Rightarrow \dots \Rightarrow c^{ik_i} \Rightarrow b$$

$1 \leq i \leq j$ , where  $c^{is} \notin \mathbb{F}$  or  $c^{is} \notin \mathbb{S}$ .

We give next the algorithm that search and give a partial order to the intervals  $\text{MAXLIN}$  in a given  $\mathcal{CSD}$ .

**Notation 3.3** Associate to each linear path (see Definition 3.2) a Capital letter called its path label.

**Definition 3.4** The ordered set of labels,  $\mathbb{U}$  is the set given by  $r \in \mathbb{U}$  if

1.  $r$  is a path label;

2.  $\{A_1, \dots, A_j\} \subseteq \mathbb{U}$ ,  $B$  is an path label,  $r = (A_1 \bullet \dots \bullet A_j)B$ .

The set  $\mathbb{U}$  is partially ordered with the order given by the transitive closure of the relation  $r < s$  if

$$(s = (A \bullet \dots \bullet A_j)B) \wedge (\exists 1 \leq i \leq j (r = A_i))$$

Each  $A_i$  is called a sequent of  $s$ . Sequents of each  $A_i$  are sequents of  $s$  as well.

**Definition 3.5** *The ordered set of labels associated to a closed digraph, associated to each linear path is the subset of  $\mathbb{U}$  so that,*

1. If  $\text{seq}$  is a linear path between  $a$  and  $b$ ,  $a$  a root, its ordered label is its path label;
2. If  $B$  is the path label associated to the linear path between  $a$  and  $b$  and if all the linear paths between  $c_1^j$  and  $a$  have their linear paths, respectively given by  $A_1, \dots, A_j$ , then the set of labels associated the path between  $a$  and  $b$  is  $(A_1 \bullet \dots \bullet A_j)B$ .

The paths in an interval in **MAXLIN** are ordered by the order inherited from the order over  $\mathbb{U}$ .

**Definition 3.6** *Define the order over the set of labels in a **CSD** by naming each label  $l$  associated to an edge  $e$  by  $(r, i)$ , where  $r \in \mathbb{U}$  is the ordered set of labels associated to the linear path where  $e$  lies and  $i$  is a numeration of the place the edge occupies in the linear path,  $1 \leq i \leq k_i$ . Let  $\prec$  be the lexicographic order over the Cartesian product of  $\mathbb{U}$  and the natural numbers.*

Consider an edge  $c \Rightarrow d$  in the closed digraph and its set of labels  $l_{(r,i)}$ . Associated to  $l_{(r,i)}$ , define a set of set of labels called  $\mathbf{Nest}(l_{(r,i)})$ . The dual of  $\mathbf{Nest}(l_{(r,i)})$  is  $\mathbf{setarray}(l_{(r,i)})$  in the sense that

$$\begin{aligned} \mathbf{Nest}(l_{(r,i)}) &= \{l_{(s,j)} \mid l_{(r,i)} \in \mathbf{setarray}(l_{(s,j)})\} \\ \mathbf{setarray}(l_{(r,i)}) &= \{l_{(s,j)} \mid l_{(r,i)} \in \mathbf{Nest}(l_{(s,j)})\} \end{aligned}$$

**Definition 3.7** *A set of labels  $\{l_{(r_1, i_1)}, \dots, l_{(r_k, i_k)}\}$  is complete if  $\{r_1, \dots, r_k\}$  contains an antichain.*

Let  $\sqsubseteq$  be an arbitrary linear order over the set of labels.

**Algorithm 3.8** Consider the set of labels in a CSD. Denote each label as  $l_{r,i}$ ,  $r \in \mathbb{U}$  and  $i$  the enumeration of the edge associated to the label.

1. For all  $l_{r,i}$  and  $l_{s,j}$  so that  $r$  and  $s$  are not comparable in the order  $<$ ,  $1 \leq i \leq k_r$  and  $1 \leq j \leq k_s$ , add  $l_{(r,i)}$  to **setarray**( $l_{(s,j)}$ ) if both are compatible. Recall that  $l_{(r,i)}$  is comparable with  $l_{(r,i)}$  and therefore,  $l_{(r,i)}$  belongs to its setarray and nest;
2. For all  $(r,i)$ , for all  $(s,j)$ , erase  $l_{(s,j)}$  from **setarray**( $l_{(r,i)}$ ) (and  $l_{(r,i)}$  from **Nest**( $l_{(s,j)}$ )) if the below intersection is not complete.

$$(\mathbf{setarray}(l_{(r,i)}) \cup \mathbf{Nest}(l_{(r,i)})) \cap (\mathbf{setarray}(l_{(s,j)}) \cup \mathbf{Nest}(l_{(s,j)}))$$

If all labels were erased, stop the computation with the output **There is no compatible antichain**. Else, the computation stops because the set of label was kept unchanged and the output is **There are compatible antichains**. Obtain the output **Solve**, a list with **setarray** and **Nest** from all labels.

**Definition 3.9** If the solution of a CSD (Algorithm 3.8) is non empty, define its solution set, **Sol**, is the set of all labels in **Solve**,  $\{l_{(r_1,i_1)}, \dots, l_{(r_s,i_s)}\}$ , so that each label  $l_{(r_m,i_m)} \in \mathbf{Nest}(l_{(r_n,i_n)})$ . Besides, each set in **Sol** is maximal regarding to subsets, that is, if to set  $I_1$  and  $I_2$  satisfy the condition of **Sol** and  $I_1 \subsetneq I_2$ , then  $I_1 \notin \mathbf{Sol}$ .

**Theorem 3.10** The solution set coincides with the set of all compatible antichains in CSD.

**Proof:** Suppose that  $S = \{l_{(i_1,r_1)}, l_{(i_2,r_2)}, \dots, l_{(i_n,r_n)}\}$  is a set of complete and compatible labels. Clearly  $S$  belongs to the solution set.

Reciprocally, we must show that a non empty output generates only antichains of compatible labels. Once we made a choice of compatible (non complete) labels, say  $T = \{l_{(i_1,r_1)}, \dots, l_{(i_n,r_n)}\}$ , if we cannot extend our choice by adding to  $T$  a compatible label  $l_{(i_{n+1},r_{n+1})}$ , then all labels in  $T$  would be erased.  $\dashv$

Henceforth, we can take for granted that an output  $\emptyset$  means that there is no compatible antichain and conversely. Naming all antichains is an (un-necessary) expsize task.

## 4 Bounds in Computation

The Algorithm we propose to decide **3-sat** has the following polynomially bounded steps

1. Factorize  $\Psi$  (Definition 2.3);
2. Write the cylindrical digraph (Definition 2.6);
3. Write the set of closed digraph (Definition 2.9),  $\mathcal{CSD}$  and estimate the size of  $\text{Nectrue}$  and  $\text{Necfalse}$
4. Search for source, spillway and roots (Definition 3.1);
5. Write the ordered set of labels associated to a closed digraph (Definition 3.5);
6. Perform Algorithm 3.8.

Observation: The size of  $\text{Cyl}$  is given by

- $V$ , the set of vertexes, is bounded by  $L$ , the number of literals of  $\Psi$ ;
- $E$ , the set of edges. For each  $a \in V$ , there are at most  $L$  edges of the form  $a \Rightarrow b$ ,  $b \in V$ , thus  $|E| \leq L^2$ .

The number of intervals given in Definition 2.9, denoted by  $\mathcal{I}$  is bounded by all combinations of

- 1  $[\neg a, a]$ ,  $[a, \neg b]$  and  $[\neg b, b]$ , for all  $a, b \in \text{Nec}$ ;
- 2  $[\neg a, a]$ ,  $[a, c]$  and  $[c, \perp]$ , for all  $a \in \text{Nec}$ ,  $c \in \text{NecFls}$ ;
- 3  $[\top, \neg d]$ ,  $[\neg d, c]$  and  $[c, \perp]$ , for all  $d, c \in \text{NecFls}$ .

and the number of intervals generated by the three combinations is bounded by the square of the number of vertexes  $V$ , that is  $L^2$ .

**Lemma 4.1** *Writing an interval is polynomial in space and time.*

**Proof:** We perform our search over the edges  $E$  which, recursively, is exhausted according we enlarge the set  $S$ , which leads, in the worse case, on a progression  $|E| + (|E| - 1) + \dots + 1$ . So, the search is bounded by the square of the elements of the set  $E$  and, thus, by the  $4^{\text{th}}$  power of the set of literals in  $\Psi$ .

Notice also that the number of interval is bounded by the number of necessarily true plus the number of necessarily false literal, that is, at most the square of the literals of  $\Psi$ . ◄

**Lemma 4.2** *Writing the set of roots, spillway, sources and the intervals MAXLIN is a polynomial task.*

**Proof:** Given a  $\mathcal{CSD}$ , let  $E$  be its set of edges and  $V$  be its set of vertexes.

1. The search for source and spillway requires a search over the edges, a linear search;
2. The search for the roots is a linear search over the set of sources;
3. Marking each path with *path label* depends on writing intervals, a poly-size task;
4. Writing the *ordered set of labels*,  $\mathbb{U}$ , depends on writing intervals, a polysize operation. The number of paths is bounded by the number of edges in a closed digraph.

–

**Lemma 4.3** *Writing the path labels and the ordered set of labels associated to a closed digraph is a polynomial task.*

**Proof:** The number of paths is bounded by the number of edges in the closed digraph. Writing the ordered labels depends on a search performed on, at most  $n + (n - 1) + \dots + 2 + 1$  over the number of edges. –

**Lemma 4.4** *There is a polynomial algorithm to decide whether there is an antichain in a set  $V$  in  $\mathbb{U}$ .*

**Proof:** Comparison between two elements of  $\mathbb{U}$  and  $V$  demands a  $|V| \times |\mathbb{U}| \leq |\mathbb{U}|^2$  operations, where  $|S|$  denotes the size of a set  $S$ . –

**Proposition 4.5** *The search for antichains in a set of labels  $V$  is polynomially bounded.*

**Proof:** The search depends on comparison between the sets  $\mathbb{U}$  and  $V$ , again bounded by the square of  $|\mathbb{U}|$ . –

**Proposition 4.6** *Algorithm 3.8 is polynomially bounded.*

**Proof:** Perform a polysize search over the Cartesian product of the set of labels (compare a label with other label).

In step 2, we compare sets of the size of the set of labels while performing the operations

$$(\mathbf{setarray}(l_{(r,i)}) \cup \mathbf{Nest}(l_{(r,i)})) \cap (\mathbf{setarray}(l_{(s,j)}) \cup \mathbf{Nest}(l_{(s,j)}))$$

In the worse case, we erase labels in macrobranches, in a **setarray** or in **Nest**, which can demand a search over space. If  $|l|$  is the number of labels, the space is  $|l|^2$ , the number of labels plus its set **setarray**  $\cup$  **Nest**. Thus, erasing label by label, we spend a number square of the space, thus  $|l|^4$ .

As the number of labels is bounded by the number edges in each interval in the set of closed digraphs  $\mathcal{CSD}$ . The number of closed digraphs, given in Definition 2.9 is, at most, the order square of the literals in  $\Psi$  and the number of edges in each interval is bounded by the number of edges in a cylindrical digraph, that is, the square of the number of literals. So, at most  $|\mathbf{Letter}(\Psi)|^2$  edges in each interval, whose number bounded by  $|\mathbf{Letter}(\Psi)|^2$   
 $\dashv$

## References

- [1] Cook S.A. The complexity of theorem proving procedures. *Proc 3rd STOC*, pages 151–158, 1971.
- [2] Cook S.A. Reckhow A.R. The relative efficiency of propositional proof systems. *J. of Symbolic Logic*, 44(1):36–50, 1979.
- [3] Stockmeyer L. J. Classifying the computational complexity problems. *The Journal of Symbolic Logic*, 52 No 1:1–43, 1987.
- [4] Levin, Leonid. Universal sorting problems. problems of information transmission. *Problemy Peredachi Informatsii*, 9:265–266, 1973.
- [5] Levin, Leonid. "universal search problems". *Problemy Peredachi Informatsii*, 6(4):265–266, 1984.
- [6] Meyer A.R. weak monadic second-order theory of successor is not elementary recursive. In R Parikh, editor, *Lecture Notes in Mathematics*, volume vol 453, pages 132–154. Springer-Verlag, New York, 1972-73.

- [7] Meyer A.R. Stockmeyer L. J. The equivalence problem for regular expressions with squaring requires exponential space. *Proceedings of the 13th IEEE Symposium on Switching and Automata Theory*, pages 125–129, 1972.
- [8] Pudlak P. Satisfiability - algorithms and logic. In *Mathematical Foundations of Computer Science*, pages 129–141. Springer, 1998.
- [9] Papadimitriou, C. H. *Computational Complexity*. Addison-Wesley, 1994.
- [10] M. A. Weiss. A polynomial algorithm for deciding 3-sat. Technical report, IME-USP. Mathematics Department, Universidade de Sao Paulo, 2007. weiss@ime.usp.br.