A Polynomial Algorithm for 3-sat

M Angela Weiss
Draft Version
weiss@ime.usp.br, Universidade de São Paulo, São Paulo - Brasil

Abstract

We present a new method for solving 3-sat formulas where 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 \( P = NP \).

MSC 03D15 Keywords: Computational Complexity, SAT, Boolean Logic

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 in [10]. Previous versions of this paper, the algorithms based on it as well as examples are posted in
www.ime.usp.br/~weiss

The version, presented here 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 (SAT) A formula $\chi$ in Boolean Logic is said satisfiable if there is a valuation $v$ onto the set of atoms of $\chi$ so that $v(\chi)$ is true. If no such valuation exists, that is $v(\chi)$ is false for any valuation, then we say that $\chi$ is unsatisfiable.

Definition 2.2 A 3-sat formula $\Psi$ is a conjunctions of a number, say $n$, of a disjunction of at most three literals. We write $\Psi \equiv (l_1^1 \lor l_2^1 \lor l_3^1) \land \cdots \land (l_1^n \lor l_2^n \lor l_3^n) \equiv C_1 \land \cdots \land C_n$ A subformula $C_k \equiv l_1^k \lor l_2^k \lor l_3^k, 1 \leq k \leq n$ of $\Psi$ is called a clause.

The set of literals of a formula $\Psi$ is denoted by 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 $\Psi, 1 \leq k \leq n$ have a pair of conjugated literals among its literals. Indeed, if $C_k \equiv l_1^k \lor p \lor \neg p$, then $C_k \equiv l_1^k$.

Lemma 2.3 Let $\Psi \equiv C_1 \land \cdots \land C_m \land C_{m+1} \cdots \land C_n$ be so that Letter$(\Psi) = \{p_1, \ldots, p_r, q_1, \neg q_1, \ldots, q_s, \neg q_s\}$ and $\Psi' \equiv C_1 \land \cdots \land C_m$, where $\{C_1, \ldots, C_m\}$ are the clauses of $\Psi$ that do not contain literals from the set $\{p_1, \ldots, p_r\}$ among their literals. Then, $\Psi$ is unsatisfiable if and only if $\Psi'$ is unsatisfiable.

Proof: Clearly if $\Psi'$ is unsatisfiable, then $\Psi \equiv \Psi' \land (C_{m+1} \land \cdots \land C_n)$ is unsatisfiable. On the other side, if $\Psi$ is unsatisfiable, then any valuation will make it false. In particular, all the valuations $v$ where for all $s \in \{p_1, \ldots, p_r\}$, $v(s)$ is true. Thus, the conjunction $\chi \equiv C_{m+1} \land \cdots \land C_n$ is so that $v(\chi)$ is true and all the valuations $v$, restricted to the literals $\{q_1, \neg q_1, \ldots, q_s, \neg q_s\}$ make $\Psi$ false. Hence, $\Psi'$ is unsatisfiable.

Notation 2.4 Consider the 3-sat formula

$$\psi \equiv (p \lor q_{11} \lor q_{12}) \land (p \lor q_{21} \lor q_{22}) \land \cdots \land (p \lor q_{t1} \lor q_{t2})$$
The factorization of $\psi$ by $p$, denoted $p \lor S_p$, is the rewritten of $\Psi$ as (the equivalent) formula

$$p \lor ((q_{11} \lor q_{12}) \land (q_{21} \lor q_{22}) \land \cdots \land (q_{t1} \lor q_{t2}))$$

(1)

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

Case $\psi \equiv p$, then $S_p \equiv \top$.

**Definition 2.5** Given a 3-sat formula $\Psi$, a partition of $\Psi$ by a set of labels $\text{Label} \subseteq \{q_1, \neg q_1, \ldots, q_k, \neg q_k\}$ (a set of literals in $\Psi$) is a formula $\Psi' \equiv (\bigwedge_{l \in \text{Label}} (l \lor S_l)) \land S_{\top}$ 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 $l \in \text{Label}$, the symbol $\bot$ or the symbol $\top$;
- For all $1 \leq l_1 < l_2 \leq k$, $q_{l_1}$ and $\neg q_{l_1}$ do not belong to the set of literals of $S_{q_{l_2}}$ or $S_{\neg q_{l_2}}$;
- $S_{\top}$ is at most a 2-sat formula.

**Proposition 2.6** Any 3-sat formula $\Psi \equiv C_1 \land \cdots \land C_n$ admits a partition.

**Proof:** Due to Lemma 2.3, we can assume that $\text{Letter}(\Psi)$ is so that if $l$ is a literal of $\Psi$, so is its negation $\neg l$. Suppose that $\text{Letter}(\Psi) = \{l_1, \neg l_1, \ldots, l_s, \neg l_s\}$. Choose the pair $\text{Label}_0 = \{q_{01}, \neg q_{01}, \ldots, q_{0k_0}, \neg q_{0k_0}\} \subseteq \text{Letter}(\Psi)$ so that we partition the set of clauses $\{C_1, \ldots, C_m\}$ into

$$\{D_{q_{01}}, D_{\neg q_{01}}, \ldots, D_{q_{0k_0}}, D_{\neg q_{0k_0}}, D_1\}$$

where $D_l$ is the set of clauses that contains $l \in \text{Label}_0$ among their literals, $D_1$ is the set of clauses that do not contain literals of $\text{Label}_0$ among their literals and, moreover, for all $C \in D_1$, for all literals $p \in C$, there is a $l \in \text{Label}_0$ so that either $p$ or $\neg p$ is a literal of some clause of $D_l$.

For all $l \in \text{Label}_0$, $S_l$ is the factorization of $\bigwedge_{C \in D_l} C$ by $l$. Observe that if a set $D_l$ is unitary, $D_l = \{C\}$ and $C \equiv l$, then $C \equiv l \lor \bot$, so $S_l = \bot$. Obtain a set of conjugated pairs $\text{Label}_0 = \{q_{01}, \neg q_{01}, \ldots, q_{0k_0}, \neg q_{0k_0}\}$ and the (equivalent to $\Psi$) formula

$$(q_{01} \lor S_{q_{01}}) \land (-q_{01} \lor S_{-q_{01}}) \land \cdots \land S_{q_{0k_0}} \lor S_{q_{0k_0}}) \land (-q_{0k} \lor S_{-q_{0k}}) \land (\bigwedge_{C \in D_1} C)$$
where each $S_l$ is the factorization of the clauses of $\Psi$ that contain $l \in Label_0$.

Consider $Label_1 = \{q_{11}, q_{11}, \ldots, q_{k1}, \neg q_{k1}\}$ a set of conjugated literals and the formula

$$(q_{11} \lor S_{q_{11}}) \land (\neg q_{11} \lor S_{\neg q_{11}}) \land \cdots \land (q_{k1} \lor S_{q_{k1}}) \land (\neg q_{k1} \lor S_{\neg q_{k1}}) \land (\land_{C \in D_2} C)$$

where each $S_l$ is either the factorization of the clauses of $D_1$ that contain $l \in Label_1 = \{q_{11}, q_{11}, \ldots, q_{k1}, \neg q_{k1}\}$ or the symbol $\top$, case no clause of $D_2$ contain $l$ among its literals. Besides,

1. For all $1 \leq i, j \leq k_1$, the literals $q_{ij}$ and $\neg q_{ij}$ do not belong to the sets of literals of $S_{q_{ij}}, S_{\neg q_{ij}}$ nor to any $C \in D_2$;

2. For all $C \in D_2$, for all literals $l \in C$, there is a $l \in Label_0 \cup Label_1$ so that either $q$ or $\neg q$ is a literal of $S_l$;

For all $l \in Label_1$, there is a $p \in Label_0$ so that $l$ or $\neg l$ is a literal of $S_p$. Case $S_l$ is not the symbol $\bot$, we say that $l \in Label_1$ belong to the set of labels.

If $(\land_{C \in D_2} C)$ is at most a 2-sat formula, stop the procedure, else, keep repeating the above procedure for the sets of literals

$$\{q_{21}, \neg q_{21}, \ldots, q_{2k_2}, \neg q_{2k_2}\} \cdots \{q_{s_1}, \neg q_{s_1}, \ldots, q_{sk_s}, \neg q_{sk_s}\}$$

until obtain a formula

$$(q_{01} \lor S_{q_{01}}) \land (\neg q_{01} \lor S_{\neg q_{01}}) \land \cdots \land (q_{0k_0} \lor S_{q_{0k_0}}) \land (\neg q_{0k_0} \lor S_{\neg q_{0k_0}}) \land$$

$$(q_{21} \lor S_{q_{21}}) \land (\neg q_{21} \lor S_{\neg q_{21}}) \land \cdots \land (q_{2k_2} \lor S_{q_{2k_2}}) \land (\neg q_{2k_2} \lor S_{\neg q_{2k_2}}) \land \cdots$$

$$(q_{s_1} \lor S_{q_{s_1}}) \land (\neg q_{s_1} \lor S_{\neg q_{s_1}}) \land \cdots \land (q_{sk_s} \lor S_{q_{sk_s}}) \land (\neg q_{sk_s} \lor S_{\neg q_{sk_s}}) \land$$

$$(\land_{C \in E} C)$$

where the $\land_{C \in E} C$, denoted $S_\top$, is a 2-sat formula,

1. For all $1 \leq i \leq j \leq s$, $1 \leq t \leq k_i$ and $1 \leq r \leq k_j$ the literals $q_{it}$ and $\neg q_{it}$ do not belong to the sets of literals of $S_{q_{ir}}, S_{\neg q_{ir}}$;

2. For all $1 \leq i \leq s$ and $1 \leq r \leq k_i$, there is a $0 \leq j < i$ and $1 \leq t \leq k_j$ so that either $q_{ir}$ or $\neg q_{ir}$ belongs to $S_{p_{jr}}$ or to $S_{\neg p_{jr}}$;

3. For all $p_{ir}$, there is a $p_{js}, j < s$ so that $p_{ir}$ or $\neg p_{ir}$ is a literal of $S_{p_{js}}$ or of $S_{\neg p_{js}}$. 

4
The label associated to \( S_\top \) is \( \top \).

We work with a partitioned formula \( \Psi \) whose set of labels, from now on denoted \( \text{Label} = \{q_1, \ldots, q_k\} \), is fixed.

**Definition 2.7** If there is a formula \( S_p \) and a literal \( l \in \text{Label} \) so that \( l \) is a literal of \( S_p \), then \( l \) is called necessarily false. The set of necessarily false literals is denoted by \( \text{NecFls} \).

**Observation 2.8** If \( C = \{C_1, \ldots, C_n\} \) is the set of clauses of \( \Psi \), we have shown in Proposition 2.6 that there is a partition of \( C \) by sets \( \mathcal{P} \mathcal{T} \subseteq \{D_{q_1}, D_{\neg q_1}, \ldots, D_{q_k}, D_{\neg q_k}, T\} \), \( T \) the clauses of \( \Psi \) with at most two literals, \( D_l, l \in \{q_i, \neg q_i\}, \) clauses in \( \Psi \) that contain \( l \).

The sets of clauses in \( \mathcal{P} \mathcal{T} \) are pairwise disjoint.

**Proposition 2.9** \( \Psi \) is unsatisfiable if and only if a partitioned version of it is unsatisfiable.

**Proof:** Our claim follows on noticing that at each step of factorization, \( \Psi' \) is replaced by an equivalent formula. More precisely, we wrote a partition of the clauses of \( \Psi' \), \( \mathcal{P} \mathcal{T} \subseteq \{D_{q_1}, D_{\neg q_1}, \ldots, D_{q_k}, D_{\neg q_k}, T\} \). Some of the sets \( D_l \) are the symbol \( l \lor \top \), that will not spoil our valuation for \( l \lor \top \) is always true. We have that

\[
\Psi' \equiv C_1 \land \cdots \land C_m \equiv \bigwedge_{l \in \text{Label}} (l \lor S_l) \equiv (l \lor (C \in \mathcal{P} \mathcal{T})) \equiv \bigwedge_{l \in \text{Label}} (l \lor S_l) \land (S_\top)
\]

**Definition 2.10** Given a partitioned 3-sat formula \( \Psi \), define the cylindrical digraph \( \mathcal{C} \text{Indr-graph} \) generated by \( \Psi \), \( (L, \Rightarrow) \) as

- \( L \subseteq \text{Letter}(\Psi) \cup \{\bot, \top\} \);
- If \( a \lor b \) is a clause of \( S_{l_1}, \ldots, S_{l_k} \), \( l \in \text{Label} \) 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 set \( \{l_1 \ldots l_k\} \);
- If \( a \equiv a \lor \bot \) is a clause of \( S_{l_1}, \ldots, S_{l_k} \), then \( \{a, \neg a, \top, \bot\} \) is contained in the set of vertexes, \( \top \Rightarrow a \) and \( \neg a \Rightarrow \bot \) are edges in \( \mathcal{C} \text{Indr-graph} \) both with label \( \{l_1 \ldots l_k\} \);
• If \( a \) is necessarily false, then \( \{a, \neg a, \top, \bot\} \) is contained in the set of vertexes, \( a \Rightarrow \bot \) and \( \top \Rightarrow \neg a \) are edges of Clndr-graph with label \( \{a\} \);
• If \( \top = S_I \), form no edge;
• For all \( p \) so that \( S_p \equiv \bot \), then \( \bot \Rightarrow \top \) is an edge whose label contains \( p \).

**Definition 2.11** Let \( a \) and \( b \) be two vertexes of the cylindrical digraph.

1. A path from \( a \) to \( b \) is a subdigraph of the form
   \[
   a \Rightarrow c^1 \Rightarrow c^2 \Rightarrow \cdots \Rightarrow c^k \Rightarrow b
   \]
2. An interval between \( a \) and \( b \) is the (non empty) subdigraph that contains all paths from \( a \) to \( b \). Denote the interval \([a, b]\). A path in the interval \([a, b]\) is the set of all paths starting at \( a \) and ending at \( b \).

A path between two edges \( e_1 \) and \( e_{k+1} \), denoted \( e_1, \ldots, e_{k+1} \) is a sequence like the above sequence with \( e_1 = a \Rightarrow c^1, e_2 = c^1 \Rightarrow c^2, \ldots, e_k = c^k \Rightarrow b \)

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

**Definition 2.12** Recall that literal \( r \in \text{Label} \cap (\cup \text{Letter}(\Psi)) \) is called necessarily false literal. The set of necessarily false literals is denoted by \( \text{NecFls} \).

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 \( \text{Nec} \).

If \( \neg a \in \text{NecFls} \) or \( a \) is an isolated clause, we consider the edges \( \neg a \Rightarrow \bot \) and \( \top \Rightarrow a \) in our computation. If \( a \) is an isolated clause, then \( a \) is necessarily true and its interval, \([\neg a, a]\) contains the (unique) path

\[
\neg a \Rightarrow \bot \Rightarrow \top \Rightarrow a
\]

If \( S_p \equiv \bot \), then \( \bot \Rightarrow \top \) forms a closed interval whose label is \( p \).

**Notation 2.13** Given a cylindrical digraph, denote

1. \([\neg a, a] \rightarrow [a, \neg b] \rightarrow [\neg b, b] \)
2. \([\neg a, a] \rightarrow [a, c] \rightarrow [c, \bot] \)
3. \([\top, \neg d] \rightarrow [\neg d, c] \rightarrow [c, \bot] \)
respectively, the smallest subdigraph of the cylindrical digraph that contains, respectively the intervals,
1 \([-a,a]\), \([a,-b]\) and \([-b,b]\), for all \(a,b \in \text{Nec}\);
2 \([-a,a]\), \([a,c]\) and \([c,\bot]\), for all \(a \in \text{Nec}, c \in \text{NecFls}\);
3 \([\top,-d]\), \([-d,c]\) and \([c,\bot]\), for all \(d,c \in \text{NecFls}\).
Digraphs as above are called generators.

**Definition 2.14** Given a cylindrical digraph, and the set of natural numbers, \(\text{N}\), its set of closed digraphs, \(\text{CSD}\) is defined as the set of digraphs \(\text{CSD} = (V,E,\text{label}(a,b))\), where, for all generator \(I\) as in notation 2.13 there is one and only one natural number \(j\), called index of \(I\) so that if \(x\) and \((x,y)\) are a vertex and an edge of \(I\), then, \((x,j)\) and \(((x,j),(y,j))\) are, respectively a vertex and an edge of \(\text{CSD}\).

**Definition 2.15** Given a closed digraph generated by \(I = [a,b] \rightarrow [b,c] \rightarrow [c,d]\), with index \(j\), a (maximal) path in \(I\) is a path \(P\) that starts at \((a,j)\) and ends at \((d,j)\) and passes through (contains as an element) \((b,j)\) and \((c,j)\),
\[
\text{seq} = (a^0,j) \xrightarrow{b} (a^1,j) \xrightarrow{c} (a^2,j) \xrightarrow{c} \cdots \xrightarrow{c} (a^{r-1},j) \xrightarrow{r} (a^r,j) \quad (2)
\]
where \(b\) and \(c\) belongs to \(\{a^1,a^2,a^{r-1}\}\), \(a^0 = a\) and \(a^r = d\).

**Definition 2.16** Let \(C = \{p_1,\ldots,p_m\} \subseteq \text{Label}\) be a consistent set and \(F\) 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 \((q_i \lor r_i)_{p_i}\) the \(l^{th}\) clause of \(S_{p_i}\).
For each \(f \in F\), let \(\Phi_f\) be the formula
\[
\bigwedge_{i=1}^{m} (q_{f(p_i)} \lor r_{f(p_i)})_{p_i}
\]
We are particularly interested in the maximal path in Formula 2.

**Definition 2.17** Consider \(P\), the set of all maximal paths in the intervals in \(\text{CSD}\). To each \(\text{seq} \in P\), let \(P_{\text{seq}}\) be the set of all consistent sets so that \(v \in P_{\text{seq}}\) iff for all edge \((c,j) \Rightarrow (d,j)\) in \(\text{seq}\), there is a literal \(l \in v\) that belongs to the set of labels of \((c,j) \Rightarrow (d,j)\).
Let \(V\) be the set of all valuations so that \(v \in V\) iff there is a maximal path \(\text{seq}\) and a consistent set \(v \in P_{\text{seq}}\) in which \(v(p) = \bot\) for all \(p \in v\).
In Definition 2.17, the set $C$ is a (consistent) set so that for each label $l_i$ in the path $2$ there is an element of $l_i$ in $C$ and, reciprocally each literal in $C$ belongs to a label in $2$ and, associated to seq in $2$, we have the $2$-sat formula $\Phi_{\text{seq}}$,

$$(-a^0 \lor a^1) \land (-a^1 \lor a^2) \land \cdots \land (-a^{r-1} \lor a^r)$$ (3)

We have that $P_{\text{seq}}$ is a subset of $\{\Phi_f | f \in \mathcal{F}\}$. We show that $P_{\text{seq}}$ is a kernel of unsatisfiable formulas of $\{\Phi_f | f \in \mathcal{F}\}$, as below stated.

**Theorem 2.18** For all maximal path seq, for all consistent choice $C$ in seq, $\Psi_{\text{seq}}$ is unsatisfiable.

**Theorem 2.19** If $\Phi_f$ is unsatisfiable, then there is a formula of $\chi \in \Phi_{\text{seq}}$ so that $\chi$ is a subformula of $\Psi_f$.

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

**Observation 2.20** Once we distinguish intervals by using its index, we omit writing its index to avoid overload our proofs.

**Proposition 2.21 (Papadimitreou, [9])** Let $\Phi \equiv (a_0 \Rightarrow a_2) \land (a_3 \Rightarrow a_4) \land \cdots \land (a_n \Rightarrow a_{n+1})$ 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 $-a_{i+1} \Rightarrow -a_i$, $0 \leq 1 < n$. Then $\Phi$ is unsatisfiable iff there are paths from $a_i$ to $-a_i$ and from $-a_i$ to $a_i$ for some $0 \leq i < n$.

**Proof of 2.18 and 2.19:**

**2.18:** Applying Definition 2.14, we obtain three kind of formulas, associated, respectively, to cases 1, 2, and 3,

$$
(a \lor p^1) \land (-p^1 \lor p^2) \land \cdots \land (-p^r \lor a) \land (a \lor p^{r+1}) \land (-p^{r+1} \lor p^{r+2}) \land \\
\cdots \land (-p^{s-1} \lor -b) \land (b \lor p^{s+1}) \land (-p^{s+1} \lor p^{s+2}) \land \cdots \land (-p^{t-1} \lor b)$$ (4)

$$
(a \lor p^1) \land (-p^1 \lor p^2) \land \cdots \land (-p^r \lor a) \land (a \lor p^{r+1}) \land (-p^{r+1} \lor p^{r+2}) \land \\
\cdots \land (-p^{s-1} \lor c) \land -c$$ (5)

$$
-d \land (d \lor p^r) \land (-p^r \lor p^{r+1}) \land \cdots \land (-p^{r-1} \lor c) \land -c$$ (6)
The above three formulas are clearly unsatisfiable, but the associated pair of paths are not so clear. Observe that

\[ \neg d \land (d \lor p') \land (\neg p' \lor p^{r+1}) \land \cdots \land (\neg p^{r-1} \lor c) \land \neg c \equiv (\bot \lor \neg d) \land (d \lor p') \land (\neg p' \lor p^{r+1}) \land \cdots \land (\neg p^{r-1} \lor c) \land (\bot \lor \neg c) \]

and that \( \bot \Rightarrow \top \) belongs to any path.

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

\[
\begin{align*}
(\neg a \Rightarrow p^1), (p^1 \Rightarrow p^2), \ldots, (p^r \Rightarrow a), (a \Rightarrow p^{r+1}), (p^{r+1} \Rightarrow p^{r+2}), \ldots, \\
(p^{s-1} \Rightarrow \neg b), (\neg b \Rightarrow p^{s+1}), (p^{s+1} \Rightarrow p^{s+2}), \ldots, (p^{l-1} \Rightarrow b) \\
(\neg p^{l+2} \Rightarrow \neg p^{r+1}), \ldots, (b \Rightarrow \neg p^{s-1}), (\neg p^{s-1} \Rightarrow b)) \\
(\neg b \Rightarrow \neg p^{l-1})
\end{align*}
\]

\[
\begin{align*}
(\neg a \Rightarrow p^1), (p^1 \Rightarrow p^2), \ldots, (p^r \Rightarrow a), (a \Rightarrow p^{r+1}), (p^{r+1} \Rightarrow p^{r+2}), \ldots, \\
(p^{s-1} \Rightarrow c), (c \Rightarrow \bot) \\
(\neg p^1 \Rightarrow a), (\neg p^2 \Rightarrow \neg p^1), \ldots, (\neg a \Rightarrow \neg p^r), (\neg p^r \Rightarrow \neg a), \\
(\neg p^{r+2} \Rightarrow \neg p^{r+1}), \ldots, (\neg c \Rightarrow \neg p^{s-1}), (\top \Rightarrow \neg c) \\
(\bot \Rightarrow \neg d), (\neg d \Rightarrow p^r), (p^r \Rightarrow p^{r+1}), \ldots, (p^{l-1} \Rightarrow c), (c \Rightarrow \bot) \\
(d \Rightarrow \bot), (\neg p^r \Rightarrow d), (\neg p^{r+1} \Rightarrow \neg p^r), \ldots, (\neg c \Rightarrow \neg p^{r-1}), (\top \Rightarrow \neg c)
\end{align*}
\]

and the respective pair of paths,

\[
\begin{align*}
(\neg a \Rightarrow p^1), (p^1 \Rightarrow p^2), \ldots, (p^r \Rightarrow a) \\
(a \Rightarrow p^{r+1}), (p^{r+1} \Rightarrow p^{r+2}), \ldots, (p^{s-1} \Rightarrow \neg b), (\neg b \Rightarrow p^{s+1}), (p^{s+1} \Rightarrow p^{s+2}), \\
\ldots, (p^{l-1} \Rightarrow b), (b \Rightarrow \neg p^{s-1}), \ldots, (\neg p^{r+2} \Rightarrow \neg p^{r+1}), (\neg p^{r+1} \Rightarrow \neg a) \\
\bot \Rightarrow \top \\
(\bot \Rightarrow \neg c), (\neg c \Rightarrow \neg p^{s-1}), \ldots, (\neg p^{r+2} \Rightarrow \neg p^{r+1}), (\neg p^{r+1} \Rightarrow \neg a), \\
(\neg a \Rightarrow p^1), (p^1 \Rightarrow p^2), \ldots, (p^r \Rightarrow a), (a \Rightarrow p^{r+1}), (p^{r+1} \Rightarrow p^{r+2}), \ldots, \\
(p^{l-1} \Rightarrow c), (c \Rightarrow \bot)
\end{align*}
\]

\[
\begin{align*}
\bot \Rightarrow \top \\
(\bot \Rightarrow \neg d), (\neg d \Rightarrow p^r), (p^r \Rightarrow p^{r+1}), \ldots, (p^{l-1} \Rightarrow c), (c \Rightarrow \bot)
\end{align*}
\]

2.19: Given a unsatisfiable 2-sat formula \( \Phi_f, f \in \mathcal{F} \), consider a pair of paths associated to \( \Phi_f \),

\[
\begin{align*}
(a \Rightarrow b^1), (b^1 \Rightarrow b^2), \ldots, (b^a \Rightarrow \neg a) \\
(\neg a \Rightarrow c^1), (c^1 \Rightarrow c^2), \ldots, (c^{a_2} \Rightarrow a)
\end{align*}
\]
write

\((a \Rightarrow b^1), (b^1 \Rightarrow b^2), \ldots, (b^n \Rightarrow \neg a), (\neg a \Rightarrow c^1), (c^1 \Rightarrow c^2), \ldots, (c^{n_2} \Rightarrow a)\)

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 paths, so, the smallest path 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 2.14.

Conclude (Theorems 2.18 and 2.19) that valuations in \(\mathcal{V}\) are the valuations that falsify \(\Psi\). Thus, \(\Psi\) is unsatisfiable iff \(\mathcal{V}\) entails all possible consistent combinations over the set of literals. We resume our decision problem on the verification whether \(\mathcal{V} \subseteq 2^{\text{Label}}\), the set of all mappings from \text{Label} onto \(\{0, 1\}\), contains all possible consistent combination over the set of labels.

The set \(\mathcal{V}\) is contained in the set of consistent sets of literals,

\[\mathcal{T} \equiv \{cttcsd|\exists \text{seq} \in \mathcal{P}\forall \text{label} \in \text{seq}\exists p \in \text{label}(p \in cttcsd)\}\]

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

\[\sim \mathcal{T} \equiv \{cttcsd|\forall \text{seq} \in \mathcal{P}\exists \text{label} \in \text{seq}\forall p \in \text{label}(\neg p \in cttcsd)\}\]

Observe that \(\mathcal{T} \subseteq 2^{\text{Label}}\) and its complementary set is \(\sim \mathcal{T}\). So, we reduce our search to the verification whether all the sets in \(\sim \mathcal{T}\) are inconsistent (and \(\Psi\) is not valid) or no (\(\Psi\) is valid).

**Definition 2.22** A set of set of edges \(A = \{e_1, \ldots, e_n\}\) is incompatible if there is a conjugated pair of literals \(p\) and \(\neg p\) in the union of its set of labels, \(\cup \{l_i|1 \leq i \leq n\}\). If otherwise, we say that \(C\) is compatible and denote \(\Xi(e_1, \ldots, e_n)\).

If some \(l_i\) contains \(\top\) then \(C\) is incompatible.

All the sets in \(\sim \mathcal{T}\) are inconsistent iff all antichains (over a natural partial order generated by the closed digraphs) are incompatible. So, we search for antichains in the paths in the closed digraph. The task of deciding if \(\Psi\) is satisfiable resumes on finding an compatible antichain in \(\mathcal{CSD}\).
3 Polynomial Decision Algorithm

Notice that from now on, we are working with the set of closed digraphs, \( CSD \), and that a vertex or an edge in the cylindrical digraph, \( a \) or \((a, b)\) can be replied and renamed \((a, 1), \ldots, (a, r)\) and \(((a, 1), (b, 1)), \ldots, ((a, r), (b, r))\). Each vertex and edges are pairwise distinct accordingly to that combination in Notation 2.13. We now search for sets of antichains whose set of labels is incompatible over all the possible sequences in \( CSD \). Our search can be expsize long if we choose to name all possible sequences.

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

\[
\begin{align*}
F &= \{a | (\exists c \in V \exists c' \in V (c \neq c') \land (a \Rightarrow c \in E) \land (a \Rightarrow c' \in E)) \\
&\lor \neg \exists c (a \Rightarrow c \in E)\} \\
S &= \{b | (\exists c \in V \exists c' \in V (c \neq c') \land (c \Rightarrow b \in E) \land (c' \Rightarrow b \in E)) \\
&\lor \neg \exists c (b \Rightarrow c \in E)\} \\
R &= \{a \in V | \not\exists b (b \Rightarrow a \in E)\} \subseteq F
\end{align*}
\]

**Definition 3.2** Given an edge \( e \), \( up(e) \) is the set of all edges \( s \) so that \( s = x \Rightarrow y, x \in F \) and there is a path, \( s, b_1, \ldots, b_n, e \) between \( s \) and \( e \).

**Definition 3.3** Given two edges \( a \) and \( b \), we say that \( a \) and \( b \) are comparable, \( comp(a, b) \) if either

- \( up(a) \subsetneq up(b) \) or \( up(a) = up(b) \) and there is a path between \( a \) and \( b \) or;
- \( up(b) \subsetneq up(a) \) or \( up(a) = up(b) \) and there is a path between \( b \) and \( a \).

If otherwise, \( a \) and \( b \) are said incomparable, \( incomp(a, b) \).

A set of vertexes \( V \) has an antichain if for any vertex \( s \) in the set of closes intervals, there is a vertex \( a \in V \) so that \( comp(s, v) \).

**Proposition 3.4** There is a polynomial algorithm to verify if a set \( L \) of edges has an antichain in the set of closed digraphs

**Proof:** For all \( e_1 \in CSD \), search if there is a comparable \( e_2 \in L \). If the procedure ends with an empty set, there is an antichain, else, there is an edge \( e_1 \in CSD \) not comparable with any edge of \( L \), so \( L \) has no antichain. \( \square \)
**Definition 3.5** A set of edges \( \{e_1, \ldots, e_k\} \) is said **complete** if it contains an antichain.

**Definition 3.6** For all edge \( e \) in \( CSD \), define \( \text{Nest}(e) \) as the set of all edges incomparable and compatible with \( e \). In other words
\[
\text{Nest}(e) = \{ c \in E | \text{imcomp}(e, c) \land \Xi(e, c) \}
\]

**Algorithm 3.7** For all incomparable pair of edges \( (e_1, e_2) \) while \( \text{Nest}(e_1) \cap \text{Nest}(e_2) \) is not complete, then erase both \( e_1 \) from \( \text{Nest}(e_2) \).

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 \( \text{Nest} \) from all labels.

**Definition 3.8** If the solution of a \( CSD \) (Algorithm 4.2) is **non empty**, its solution set, \( \text{Sol} \), is the set of all set of edges contained in **Solve**, \( \{e_1, \ldots, e_s\} \), so that for all \( 1 \leq i, j \leq s \) each edge \( e_i \in \text{Nest}(e_j) \).

**Theorem 3.9** The solution set coincides with the set of all compatible antichains in \( CSD \).

**Proof:** Suppose that \( S = \{e_1, \ldots, e_s\} \) is a set of incomparable edges with 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 a incomparable set of edges whose labels are compatible, say \( T = \{e_1, \ldots, e_s\} \), if we cannot extend our choice to an antichain by adding to \( T \) a compatible edge \( e_{s+1} \), then all labels in \( T \) would be erased.

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

## 4 Bounds in Computation

Summarily, we followed the polynomially bounded steps

1. Factorize \( \Psi \) (Definition 2.5);
2. Write the cylindrical digraph (Definition 2.10);

3. Write the set of closed digraph (Definition 2.14), $\mathcal{CSD}$ and estimate the size of $\text{Nectrue}$ and $\text{Necfalse}$

4. Search for source, spillway and roots (Definition 3.3);

5. Perform Algorithm 4.2.

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 $|E| \leq L^2$.

**Lemma 4.1** Writing a partition is polynomially bounded.

**Proof:** Making the choices of literals $\{q_{i1}, \neg q_{i1}, \ldots, q_{ik}, \neg q_{ik}\}$ causes, in the worse case, the elimination of the clauses of $\Psi$ one by one, thus, again, the order square in the number of clauses of $\Psi$ and, therefore, the order square in the number of literals of $\Psi$. $\dashv$

**Lemma 4.2** The number of intervals given in Definition 2.14, denoted by $I$ is bounded by the square of the number of vertexes of the cylindrical digraph.

**Proof:** The number of intervals given in Definition 2.14 is bounded by all combinations of

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

and the number of intervals generated by the three combinations in bounded by the square of the number of vertexes $V$, that is $V^2$ and, thus, the number of edges of that cylindrical digraph is bounded by the square of the edges of $\Psi$, that is bounded by the square of vertexes of $\Psi$, thus, $V^4$. $\dashv$

**Lemma 4.3** Writing the set of roots, spillway, sources and the intervals $\text{MAXLIN}$ 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 polysize task;

4. Writing the ordered set of labels, $\mathbb{U}$, depends on writing intervals, a poly size operation. The number of paths is bounded by the number of edges in a closed digraph.

---

**Lemma 4.4** There is a polynomial algorithm to decide whether there is an antichain in a set of vertexes $V$.

**Proof:** Comparison between two elements of the set of vertexes in the closed digraph, $\mathbb{U}$ and $V$ demands a $|V| \times |\mathbb{U}| \leq |\mathbb{U}|^2$ operations. 

---

**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 4.2 is polynomially bounded.

**Proof:** Perform a poly size search over the Cartesian product of the set edges. In step 2, we compare sets of the size of the set of labels while performing the operations $\text{Nest}(e_1) \cap \text{Nest}(e_2)$.

In the worse case, we erase edge by edge in the set of all $\text{Nest}$, which can demand a search over space. If $|E|$ is the number of edges, the space is $|E|^2$ in the set of all $\text{Nest}$. Thus, erasing edge by edge, we spend a number square of the space, that is, $|E|^4$. The number of closed digraphs, given in Definition 2.14 is, at most, the order square of the literals in $\Psi$ and the number of edges in each interval is bounded by the square of the number of edges in a cylindrical digraph.
References


