June 12, 2012

The Algorithm

Assume that given a set of literals F, the procedures,

1. $PST(F)$ takes atoms from F;

2. $NGT(F)$ takes the negation of atoms from F;

3. $\sim (F)$ give us the set $\{\neg l | l \in F\}$;

4. $Fst(F)$ takes the firs element of F.

Given a **3−SAT** formula PSI, there are procedures

1. $LGT(PSI)$ give us the maximum number of literals in the clauses of PSI;

2. $PSI \setminus C$ writes the disjunction of all the clauses of PSI except the clauses C.

Given a clause $C \equiv (a \vee b \vee c)$,

1. $C \setminus a$ gives us the clause $C \equiv (b \vee c)$

2. $Fst(C)$ gives the set $S = \{a\}$ and $Rest(C)$ gives the clause $c = (b \vee c)$. If $C \equiv a$, $Fst(C)$ gives the set $\{a\}$ and the emptyset.

Algorithm 1 Factorize PSI by p, `Fac`(Ψ, p)

Input: $\Psi \equiv (p \vee q_{12} \vee q_{13}) \wedge (p \vee q_{22} \vee q_{23}) \wedge \cdots \wedge (p \vee q_{t2} \vee q_{t3})$

  $S_p \leftarrow \emptyset$
  for $1 \leq i \leq t$ **do**
    Read C_i
    $S_p \leftarrow S_p \wedge (C_i \setminus p)$
  end for

Output: S_p

Algorithm 2 $Letter(C)$

% Given a clause C, write the set of literals in C

  $Letter(C) \leftarrow \emptyset$
  while $C \neq \emptyset$ **do**
    $Letter(C) \leftarrow Letter(C) \cup Fst(C)$
    $C \leftarrow Rest(C)$
  end while

Algorithm 3 $Letter(PSI)$

% Given a 3-SAT formula PSI, write the set of literals in PSI

$Letter(PSI) \leftarrow \emptyset$
for $1 \leq i \leq LGT(PSI)$ **do**
  $Letter(PSI) \leftarrow Letter(PSI) \cup Letter(C_i)$
end for

Algorithm 4 Filter Partition, $Fil(PSI)$

% Given a 3-SAT formula PSI, erase clauses that contain a literal l whose conjugated $\neg l$ does not belong to the set of literals of PSI

$var1 \leftarrow Letter(PSI)$
$var2 \leftarrow \{l \in var1 | \neg l \notin var1\}$
while $var2 \neq \emptyset$ **do**
  for all $C \in PSI$ **do**
    if $Letter(C) \cap var2 \neq \emptyset$ **then**
      $PSI \leftarrow PSI \setminus C$
    end if
  end for
  $var1 \leftarrow Letter(PSI)$
  $var2 \leftarrow \{l \in var1 | \neg l \notin var1\}$
end while
if $Letter(PSI) = \emptyset$, stop computation with output PSI **is satisfiable then**

end if

Algorithm 5 Partition Part1

Input: $\Psi \equiv (q_{11} \vee q_{12} \vee q_{13}) \wedge \cdots \wedge (q_{t1} \vee q_{t2} \vee q_{t3}) \equiv C_1 \wedge \cdots \wedge C_t$

```
  Label ← ∅
  S_G ← Ψ
  S_⊤ ← ∅
  PTTΨ ← ∅
  Str ← LetterS_G
  FS ← ∅
  LitAll ← Letter(PSI)
  while LGT(S_G)=3 AND PST(Letter(G))∩∼NGT(Letter(_G)) ≠ ∅ do
    Str ← Letter(S_G)
    p ← FstLetter(S_G)
    Label ← Label ∪ {p, ¬p}
    C_p ← ⊥
    C_¬p ← ⊥
    for all Clause C of Ψ do
      if p ∈ C then
        C_p ← C_p ∧ C
        S_G ← S_G \ C
      else
        if ¬p ∈ C then
          C_¬p ← C_¬p ∧ C
          S_G ← S_G \ C
        end if
      end if
    end for
  end while
  for all p ∈ Label do
    S_p ← Fac(C_p, p)
    PTTΨ ← PTTΨ ∧ (p ∨ S_P)
    litAll ← litAll \ ({p} ∪ LetterS_p))
  end for
```

Algorithm 6 Partition Part 2

$FS \leftarrow \cup\{Letter(S_p)|p \in LABEL\}$
$Necfalse \leftarrow \emptyset$
$AUXLabel \leftarrow \emptyset$
$litAll \leftarrow \emptyset$
$Str \leftarrow LetterS_G$
while $LGH(S_G)=3$ **do**
  while $Str \neq \emptyset$ **do**
    if there is a conjugated pair $p, \neg p$ where at least one lies in $Str \setminus (litAll)$ **then**
      $C_p \leftarrow \emptyset$
      $C_{\neg p} \leftarrow \emptyset$
      if $p \in FS$ **then**
        $Necfalse \leftarrow Necfalse \cup \{p\}$
      end if
      if $\neg p \in FS$ **then**
        $Necfalse \leftarrow Necfalse \cup \{\neg p\}$
      end if
      for all Clause C of Ψ **do**
        if $p \in C$ **then**
          $C_p \leftarrow C_p \wedge C$
          $S_G \leftarrow S_G \setminus C$
        else
          if $\neg p \in C$ **then**
            $C_{\neg p} \leftarrow C_{\neg p} \wedge C$
            $S_G \leftarrow S_G \setminus C$
          end if
        end if
      end for
      if $C_p \neq \emptyset$ **then**
        $S_p \leftarrow Fac(C_p, p)$
        $Label \leftarrow Label \cup \{p\}$
        $AuxLabel \leftarrow AuxLabel \cup \{p\}$
      end if
      if $C_{\neg p} \neq \emptyset$ **then**
        $S_{\neg p} \leftarrow Fac(C_{\neg p}, \neg p)$
        $Label \leftarrow Label \cup \{\neg p\}$
        $AuxLabel \leftarrow AuxLabel \cup \{\neg p\}$
      end if
      for all $l \in AUXLabel$ **do**
        $PTT\Psi \leftarrow PTT\Psi \wedge (l \vee S_l)$
        $litAll \leftarrow litAll \setminus (\{l\} \cup Stract(S_l)$
      end for
      $FS \leftarrow FS \cup (\mathbf{Stract}(S_P) \cup \mathbf{Stract}(S_{\neg p}))$
    end if
  end while
  $S_G \leftarrow S_G \setminus PTT\Psi$
end while
$S_\top \leftarrow S_G$

Algorithm 7 Cylindrical Digraph

Input: $Ptt(\Psi)$, $Labels = \{p_1, \neg p_1, \ldots, p_s, \neg p_s, p_{s+1} = \top\}$, $NecFs$

```
Nectrue ← ∅
V ← ∅
E ← ∅
for all a, b ∈ LiteralΨ do
  label(a, b) ← ∅
end for
for all l ∈ Labels do
  for all Clauses C of S_l do
    if a ∨ b is a disjunction of C then
      if (a ≠ ⊥) AND (b ≠ ⊥) then
        V ← V ∪ {a, b, ¬a, ¬b}
        E ← E_0 ∪ {¬a ⇒ b, ¬b ⇒ a}
        label(¬a, b) ← label(¬a, b) ∪ {l}
        label(¬b, a) ← label(¬b.a) ∪ {l}
      else {Suppose b = ⊥}
        V ← V ∪ {a, ¬a, ⊥, ⊤}
        E ← E ∪ {¬a ⇒ ⊥, ⊤ ⇒ a}
        label(¬a, ⊥) ← label(¬a, ⊥) ∪ {l}
        label(⊤, a) ← label(⊤, a) ∪ {l}
        Nectrue ← Nectrue ∪ {a}
      end if
    end if
  end for
end for
for all a ∈ Necfs do
  V ← V ∪ {a, ¬a, ⊥, ⊤}
  E ← E ∪ {¬a ⇒ ⊥, ⊤ ⇒ a}
  label(¬a, ⊥) ← label(¬a, ⊥) ∪ a
end for
for all S_l do
  if S_l = ⊥ AND l ∉ Necfs then
    V ← V ∪ {⊤, ⊥}
    E ← E ∪ {⊥ ⇒ ⊤}
    label(⊥ ⇒ ⊤) ← {a}
  end if
end for
Output: V, E, label(a, b), Nectrue
```

Algorithm 8 Write Non Empty Intervals $[\neg a, q]$

Input $Cyl = (V, E, label(a, b))$, $Nectrue$, $NecFs$

We write all the sequences starting at $\neg a$, for all $\neg a \in E$.

for all $\neg a \in V$ {Step for recursive $[\neg a, q]$, for all $\neg a \in V$} **do**
  $V_{\neg a} \leftarrow \{\neg a\}$
  $E_{\neg a} \leftarrow E \setminus \{(a, c)|(a, c) \in E\}$
  for all $(b, c) \in E$ **do**
    $label(b, c)_{[\neg a,q]} \leftarrow \emptyset$
  end for
  for all $b \in V_{\neg a}$ **do**
    if $b \in V_{[\neg a,q]}$ AND $(b, c) \in E$ **then**
      $c \in V_{[\neg a,q]}$
      $E_{[\neg a,q]} \leftarrow (b, c)$
      $label(b, c)_{[\neg a,q]} \leftarrow label(b, c)$
    end if
  end for
end for

Algorithm 9 Write Non Empty Intervals $[\neg a, b]$

for all $\neg a \in V$ AND $b \in V$ **do**
  if $b \in V_{[\neg a,q]}$ **then**
    $E_{[\neg a,b]} \leftarrow \emptyset$
    $V_{[\neg a,b]} \leftarrow \{\neg a, a\}$
    if $(c, d) \in E \setminus E_{[\neg a,b]}$ AND $d \in V_{[\neg a,a]}$ and $c \neq \neg a$ **then**
      $V_{[\neg a,b]} \leftarrow V_{[\neg a,b]} \cup \{c\}$
      $E_{[\neg a,b]} \leftarrow E_{[\neg a,b]} \cup \{(c, d)\}$
      $label(b, c)_{[\neg a,b]} \leftarrow label(b, c)_{[\neg a,q]}$
    end if
  else $\{b \notin V_{[\neg a,q]}\}$
    $[\neg a, b] = \emptyset$
  end if
  if $[\neg a, a] \neq \emptyset$ **then**
    $Nectrue \leftarrow Nectrue \cup \{a\}$
  end if
end for

Algorithm 10 Closed Intervals

$count \leftarrow 1$
for all $a \neq b$ Necessarily True **do**
  if $[a, \neg b] \neq \emptyset$ or a and b are conjugated **then**
    $V_{[\neg a,a]\to[a,\neg b]\to[\neg b,b]} \leftarrow (V_{[\neg a,a]} \cup V_{[a,\neg b]} \cup V_{[\neg b,b]}) \times \{count\}$
    if (c, j) and (d, j) belong to $V_{[\neg a,a]\to[a,\neg b]\to[\neg b,b]}$ and $(c, d) \in E_{[\neg a,a]} \cup E_{[a,\neg b]} \cup E_{[\neg b,b]}$ **then**
      $((c, j), (d, j)) \in E_{[\neg a,a]\to[a,\neg b]\to[\neg b,b]}$
      $label_{[\neg a,a]\to[a,\neg b]\to[\neg b,b]}(((c, j), (d, j))) \leftarrow label(c, d)$
    end if
  end if
  $count \leftarrow count + 1$
end for
for all a Necessarily True and b Necessarily False **do**
  if $[a, b] \neq \emptyset$ OR $a = b$ (in that case, $[a, \neg b] = \emptyset$) **then**
    $V_{[\neg a,a]\to[a,\neg b]\to[b,\perp]} \leftarrow (V_{[\neg a,a]} \cup V_{[a,b]} \cup V_{[b,\perp]}) \times \{count\}$
    if (c, j) and (d, j) belong to $V_{[\neg a,a]\to[a,\neg b]\to[\neg b,\perp]}$ and $(c, d) \in E_{[\neg a,a]} \cup E_{[a,\neg b]} \cup E_{[\neg b,\perp]}$ **then**
      $((c, j), (d, j)) \in E_{[\neg a,a]\to[a,\neg b]\to[\neg b,\perp]}$
      $label_{[\neg a,a]\to[a,\neg b]\to[\neg b,\perp]}(((c, j), (d, j))) \leftarrow label(c, d)$
    end if
  end if
  $count \leftarrow count + 1$
end for
for all $a \neq b$ Necessarily False **do**
  if $[\neg a, b] \neq \emptyset$ **then**
    $V_{[\top,\neg a]\to[\neg a,b]\to[b,\perp]} \leftarrow V_{[\top,\neg a]} \cup V_{[\neg a,b]} \cup V_{[b,\perp]} \times \{count\}$
    if (c, j) and (d, j) belong to $V_{[\top,\neg a]\to[\neg a,b]\to[b,\perp]}$ and $(c, d) \in E_{[\neg a,a]} \cup E_{[a,\neg b]} \cup E_{[\neg b,b]}$ **then**
      $((c, j), (d, j)) \in E_{[\top,\neg a]\to[\neg a,b]\to[b,\perp]}$
      $label_{[\top,\neg a]\to[\neg a,b]\to[b,\perp]}(((c, j), (d, j))) \leftarrow label(c, d)$
    end if
  end if
  $count \leftarrow count + 1$
end for

Algorithm 11 Source, Root and Spillway

Input $Closed\ Digraph = (V, E, label(a, b), Nectrue, NecFs)$.

$Root \leftarrow \emptyset$
$Source \leftarrow \emptyset$
$Spillway \leftarrow \emptyset$
for all $a \in V$ **do**
  if $\nexists b(b \Rightarrow a \in E)$ **then**
    $Root \leftarrow Root \cup \{a\}$
  end if
  if $\exists c \in V \exists c' \in V(c \neq c') \wedge (a \Rightarrow c \in E)$ **then**
    $Source \leftarrow Source \cup \{a\}$
  end if
  if $(\exists c \in V \exists c' \in V(c \neq c') \wedge (a \Rightarrow c \in E) \wedge (a \Rightarrow c' \in E)$ **then**
    $Spillway \leftarrow Spillway \cup \{a\}$
  end if
end for

Algorithm 12 $up(e)$

if $(c, d) \in E$ is so that $c \in Root$ **then**
  $up(c, d) = \{e\}$
else
  $up(c, d) = \cup\{up(a, c)|a \in V\}$
end if

Algorithm 13 $seq(e1, e2)$?

Verify if there is a sequence between $e1$ and $e2$

if $up(e1) = up(e2)$ **then**
  if $\exists a \exists b \exists c((e1 = (a, b)\ AND\ e2 = (b, c))\ OR(e2 = (a, b)\ AND\ e1 = (b, c)))$ **then**
    $seq(e1, e2)$
    if $seq(e1, e3)\ AND\ seq(e3, e2)$ **then**
      $seq(e1, e2)$
    end if
  end if
else {there is no sequence}
  $notseq(e1, e2)$
end if

Algorithm 14 $comp(e1, e2)$?

if $up(e1) \subsetneq up(e2)\ OR\ up(e2) \subsetneq up(e1)$ **then**
  $comp(e1, e2)$
else
  if $(up(e1) = up(e2))\ AND\ seq(e1, e2)$ **then**
    $comp(e1, e2)$
  else
    $uncomp(e1, e2)$
  end if
end if

Algorithm 15 *antichain*$(V/\mathbb{U})$?

for all $e1 \in \mathbb{U}$ **do**
  if $\exists e2 \in V(comp(e1, e2)$ **then**
    $antichain(V)$
  else
    $noantichain(V)$
  end if
end for

Algorithm 16 *compatible*$(e1, e2)$?

if $(label(e1) \cup label(e2)) \cap \sim (label(e1) \cup label(e2)) \neq \emptyset$ {the union of the set of labels UNI intersected its set of the negation of the elements of UNI} **then**
  $\Xi(e1, e2)$
end if

Algorithm 17 $Nest(e1)$

for all $e1$ and $e2$ in E **do**
  if $uncomp((e1), (e2))\ AND\ \Xi(e1, e2)$ **then**
    $e1 \in Nest(e2)\ AND\ e2 \in Nest(e1)$
  end if
end for

Algorithm 18 SOLVE

$VERIFY \leftarrow E$
while $VERIFY \neq \emptyset$ **do**
  $AUX \leftarrow \emptyset$
  for all $uncomp((e1), (e2)) \in VERIFY$ **do**
    if $noantichain(Nest(e1) \cap Nest(e2))$ **then**
      $Nest(e1) \leftarrow Nest(e1) \setminus \{e2\}$
      $Nest(e2) \leftarrow Nest(e2) \setminus \{e1\}$
      $AUX \leftarrow AUX \cup Nest(e1) \cup Nest(e2)$
    end if
    $VERIFY \leftarrow AUX$
  end for
end while