Implicates and reduction techniques for temporal logics *

I.P. de Guzmán, M. Ojeda-Aciego, A. Valverde

Dept. Matemática Aplicada, Universidad de Málaga,
P.O. Box 4114, 29080. Málaga, SPAIN
E-mail: {guzman,aciego,a_valverde}@ctima.uma.es

Reduction strategies are introduced for the future fragment of a temporal propositional logic on linear discrete time, named FNext. These reductions are based in the information collected from the syntactic structure of the formula, which allow the development of efficient strategies to decrease the size of temporal propositional formulas, viz. new criteria to detect the validity or unsatisfiability of subformulas, and a strong generalisation of the pure literal rule. These results, used as an inner processing step, allow to improve the performance of any automated theorem prover.

Keywords: Automated deduction, temporal logics, Δ-reductions, implicates

AMS Subject classification: 03B45,68T15

1. Introduction

The temporal dimension of information, the change of information over time and knowledge about how it changes has to be considered by many Artificial Intelligence systems. There is obvious interest in designing computationally efficient temporal formalisms, specially when intelligent tasks are considered, such as planning relational actions in a changing environment, building common sense reasoning into a moving robot, in supervision of industrial processes, ...

Temporal logics are widely accepted and frequently used for specifying concurrent and reactive agents (which can be either physical devices or software processes), and in the verification of temporal properties of programs. To verify a program, one specifies the desired properties of the program by a formula in temporal logic. The program is correct if all its computations satisfy the formula.

* Partially supported by Spanish CICYT project TIC97-0579-C02-02 and EC action COST-15.

However, in its generality, an algorithmic solution to the verification problem is hopeless. For *propositional* temporal logic, checking the satisfiability of a formula can be done algorithmically, and theoretical work on the complexity of program verification is being done [3]. The complexity of satisfiability and determination of truth in a particular finite structure are considered for different propositional linear temporal logics in [7].

Linear-time temporal logics have proven [5] to be a successful formalism for the specification and verification of concurrent systems; but have a much wider range of applications, for instance, in [2] a generalisation of the temporal propositional logic of linear time is presented, which is useful for stating and proving properties of the generic execution sequence of a parallel program. On the other hand, relatively complete deductive systems for proving *branching* time temporal properties of reactive systems [4] have been recently developed.

In recent years, several fully automatic methods for verifying temporal specifications have been introduced, in [6] a tableaux calculus is treated at length; a first introduction to the tableaux method for temporal logic can be seen in [8]. However, the scope of these methods is still very limited. Theorem proving procedures for temporal logics have been traditionally based on syntactic manipulations of the formula A to be proved but, in general, do not incorporate the substitution of subformulas in A like in a rewrite system in which the rewrite relation preserves satisfiability. One source of interest of these strategies is that can be easily included into any prover, specifically into those which are non-clausal.

In this work we focus on the development of a set of reduction strategies which, through the efficient determination and manipulation of lists of unitary implicant and implicates, *investigates exhaustively* the possibility of decreasing the size of the formula being analysed. The interest of such a set of reduction techniques is that the performance of a given prover for linear-time temporal logic can be improved because the size of a formula can be decreased, at a polynomial cost, as much as possible *before* branching.

Lists of unitary models, so-called Δ-lists, are associated to each node in the syntactic tree of the formula and used to study whether the structure of the syntactic tree has or has not direct information about the validity of the formula. This way, either the method ends giving this information or, otherwise, it decreases the size of the problem before applying the next transformation. So, it is possible to decrease the number of branchings or, even, to avoid them all.

The ideas in this paper generalise the results in [1], in a self-contained way, by explicitly extending the reduction strategy to linear-time temporal logic and, what is more important, by complementing the information in the Δ-lists by means of the so-called $\widehat{\Delta}$-sets. The former allow derivation of an equivalent and smaller formula; the latter also allow derivation of a smaller formula, not equivalent to the previous one, but equisatisfiable.

The paper is organised as follows:

- Firstly, preliminary concepts, notation and basic definitions are introduced: specifically, it is worth to note the definition of literal and the way some of them will be denoted.
- Secondly, Δ-lists, our basic tool, are introduced; its definition integrates some reductions into the calculation of the Δ-lists. The required theorems to show how to use the information collected in those lists are stated.
- Later, the $\widehat{\Delta}$-sets are defined and results that use the information in these sets are stated. One of these is a generalisation of the pure literal rule.

2. Preliminary Concepts and Definitions

In this paper, our object language is the future fragment of the Temporal Propositional Logic FNext with linear and discrete flow of time, and connectives $\neg$ (negation), $\wedge$ (conjunction), $\vee$ (disjunction), F (sometime in the future), G (always in the future), and $\oplus$ (tomorrow); $\mathcal{V}$ denotes the set of propositional variables p, q, r, ... (possibly subscripted) which is assumed to be completely ordered with the lexicographical order, e.g. $p_n \leq q_m$ for all n, m, and $p_n \leq p_m$ if and only if $n \leq m$. Given $p \in \mathcal{V}$, the formulas p and $\neg p$ are the classical literals on p.

Definition 1. Given a classical propositional literal ℓ, the *(temporal) literals*[1]on ℓ, denoted $\texttt{Lit}(\ell)$, are those wff (well-formed formulas) of the form $\oplus^n \ell$, $F\oplus^n \ell$, $G\oplus^n \ell$, $FG\ell$, $GF\ell$ for all $n \in \mathbb{N}$.

If ℓ is a classical propositional literal, we denote $\vartheta\ell$ to mean a literal on ℓ, where ϑ is said to be its *temporal prefix*; if $\vartheta\ell$ is a literal, then $|\vartheta|$ denotes the number of temporal connectives in ϑ, and $\overline{\vartheta\ell}$ denotes its opposite literal, where $\overline{F} = G$, $\overline{G} = F$, $\overline{FG} = GF$, $\overline{GF} = FG$ and $\overline{\oplus} = \oplus$.

[1] We will use the notation $\oplus^n$ to denote the n-folded application of the connective $\oplus$.

$\neg\oplus A$	$\equiv$	$\oplus\neg A$	$\oplus FA$	$\equiv$	$F\oplus A$	$\oplus GA$	$\equiv$	$G\oplus A$
FFA	$\equiv$	$F\oplus A$	GGA	$\equiv$	$G\oplus A$	$FGFA$	$\equiv$	GFA
$GFGA$	$\equiv$	FGA	$FG\oplus A$	$\equiv$	FGA	$GF\oplus A$	$\equiv$	GFA
$\oplus \bigvee A_i$	$\equiv$	$\bigvee \oplus A_i$	$\oplus \bigwedge A_i$	$\equiv$	$\bigwedge \oplus A_i$	$\neg FA$	$\equiv$	$G\neg A$
$\neg GA$	$\equiv$	$F\neg A$	$F(\bigvee_{i\in J} A_i)$	$\equiv$	$\bigvee_{i\in J} FA_i$	$G(\bigwedge_{i\in J} A_i)$	$\equiv$	$\bigwedge_{i\in J} GA_i$

Figure 1.

The notion of *temporal negation normal formula*, denoted tnnf, is recursively defined as follows:

1. Any literal is a tnnf.
2. If A and B are tnnf, then $A\vee B$ and $A\wedge B$ are tnnf, which are called disjunctive and conjunctive tnnf, respectively.
3. If A is a disjunctive tnnf, then GA and FGA are tnnf.
4. If A is a conjunctive tnnf, then FA and GFA are tnnf.
5. A formula is a tnnf if and only if it can be constructed by the previous rules.

For formulas in tnnf, we will write $\overline{p}$ for the classical negated literal $\neg p$.

As usual, a classical clause is a disjunction of literals and a classical cube is a conjunction of literals. In addition, a G-clause is a formula GB where B is a classical clause, and a F-cube is a formula FB in which B is a classical cube.

The transformation of any wff into tnnf is linear by recursively applying the transformations induced by the double negation, the de Morgan laws and the equivalences in Fig. 1. By using the associative laws we will consider expressions like $A_1 \vee \cdots \vee A_n$ or $A_1 \wedge \cdots \wedge A_n$ as formulas.

We will use the standard notion of tree and address of a node in a tree. Given a tnnf A, the *syntactic tree* of A, denoted by T_A, is defined as usual. An address η in T_A will mean, when no confusion arises, the subformula of A corresponding to the node of address η in T_A; the address of the root node will be denoted ε.

If T_C is a subtree of T_A, then the *temporal order of T_C in T_A*, denoted $ord_A(C)$, is the number of temporal ancestors of T_C in T_A, that is the number of temporal connectives $(F, G, mannana)$ in which scope is the formula C.

We will also use lists with its standard notation; `nil` denotes the empty list. Elements in a list will be written in juxtaposition.

If α and β are lists of literals and $\vartheta\ell$ is a literal, $\vartheta\ell \in \alpha$ denotes that $\vartheta\ell$ is an element of α; and $\alpha \subseteq \beta$ means that all elements of α are elements of β. If $\alpha = \vartheta_1\ell_1\vartheta_2\ell_2\dots\vartheta_n\ell_n$, then $\overline{\alpha} = \overline{\vartheta_1\ell_1}\,\overline{\vartheta_2\ell_2}\dots\overline{\vartheta_n\ell_n}$.

Definition 2. A *temporal structure* is a tuple $S = (\mathbb{N}, <, h)$, where $\mathbb{N}$ is the set of natural numbers, $<$ is the standard strict ordering on $\mathbb{N}$, and h is a *temporal interpretation*, which is a function $h : \mathcal{L} \longrightarrow 2^{\mathbb{N}}$, where $\mathcal{L}$ is the language of the logic, satisfying:

1. $h(\neg A) = \mathbb{N} \smallsetminus h(A)$; $\quad h(A \vee B) = h(A) \cup h(B)$
2. $h(A \to B) = (\mathbb{N} \smallsetminus h(A)) \cup h(B)$; $\quad h(A \wedge B) = h(A) \cap h(B)$
3. $t \in h(FA)$ iff t' exists with $t < t'$ and $t' \in h(A)$
4. $t \in h(GA)$ iff for all t' with $t < t'$ we have $t' \in h(A)$
5. $t \in h(\oplus A)$ iff we have $t + 1 \in h(A)$

A formula A is said to be *satisfiable* if there exists a temporal structure $S = (\mathbb{N}, <, h)$ such that $h(A) \neq \varnothing$; if $t \in h(A)$, then h is said to be a *model of* A in t; if $h(A) = \mathbb{N}$, then A is said to be *true in the temporal structure* S; if A is true in every temporal structure, then A is said to be *valid*, and we denote it $\models A$.

Formulas A and B are said to be *equisatisfiable* if A is satisfiable iff B is satisfiable; $\equiv$ denotes the semantic equality, i.e. $A \equiv B$ if and only if for every temporal structure $S = (\mathbb{N}, <, h)$ we have that $h(A) = h(B)$; finally, the symbols $\top$ and $\perp$ mean truth and falsity, i.e. $h(\top) = \mathbb{N}$ and $h(\perp) = \varnothing$ for every temporal structure $S = (\mathbb{N}, <, h)$.

If Γ_1 and Γ_2 are sets of subformulas in A and X and Y are subformulas, then the expression $A[\Gamma_1/X, \Gamma_2/Y]$ denotes the formula obtained after substituting in A every occurrence of elements in Γ_1 by X and every occurrence of elements in Γ_2 by Y.

If η is an address in T_A and X, then the expression $A[\eta/X]$ is the formula obtained after substituting in A the subtree rooted in η by X.

3. Adding Information to the Tree: Δ-lists

The idea underlying the reduction strategy we are going to introduce is the use of information given by partial assignments. We associate to each tnnf A two lists of literals denoted $\Delta_0(A)$ and $\Delta_1(A)$ (the associated Δ-lists of A) and two sets of lists, denoted $\widehat{\Delta_0}(A)$ and $\widehat{\Delta_1}(A)$, whose elements are obtained out of the associated Δ-lists of the subformulas of A.

The Δ-lists and the $\widehat{\Delta}$-sets are the key tools of our method to reduce the size of the formula being analysed. These reductions allow to study its satisfiability with as few branching as possible.

In a nutshell, $\Delta_0(A)$ and $\Delta_1(A)$ are, respectively, lists of temporal implicates and temporal implicants of A. The purpose of these lists is two-fold:

1. To transform the formula A into an equivalent and smaller-sized one (see Sect. 3.3).
2. To be used in the definition the $\widehat{\Delta_0}$ and $\widehat{\Delta_1}$ sets (see Sect. 4), which will be used to transform the formula A into an equisatisfiable and smaller-sized one. Furthermore, information to build a countermodel (if it exists) is provided.

The sense in which we mean temporal implicant/implicate is the following:

Definition 3.

- A literal $\vartheta\ell$ is a *temporal implicant* of A if $\models \vartheta\ell \to A$.
- A literal $\vartheta\ell$ is a *temporal implicate* of A if $\models A \to \vartheta\ell$.

3.1. The Lattices of Literals

Definition 4. For each classical propositional literal ℓ we define an ordering in $\texttt{Lit}(\ell) \cup \{\bot, \top\}$ as follows:

1. $\vartheta\ell \leq \varrho\ell$ if and only if $\models \vartheta\ell \to \varrho\ell$
2. $\vartheta\ell \leq \top$ for all (possibly empty) ϑ.
3. $\vartheta\ell \geq \bot$ for all (possibly empty) ϑ.

Each set $\texttt{Lit}(\ell) \cup \{\bot, \top\}$ provided with this ordering is a lattice, depicted in Figure 2. For each literal $\vartheta\ell$ we will consider its upward and downward closures defined, respectively as:

$$\vartheta\ell{\uparrow} = \{\varrho\ell \mid \vartheta\ell \leq \varrho\ell\} \qquad\qquad \vartheta\ell{\downarrow} = \{\varrho\ell \mid \varrho\ell \leq \vartheta\ell\}$$

If Γ is a set of literals, then we define $\Gamma{\uparrow} = \bigcup_{l\in\Gamma} l{\uparrow}$ and $\Gamma{\downarrow} = \bigcup_{l\in\Gamma} l{\downarrow}$.

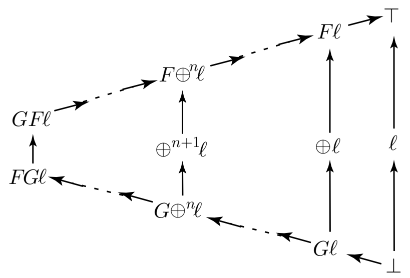


Figure 2. The lattice $\texttt{Lit}(\ell) \cup \{\perp, \top\}$

3.2. Definition of the $\boldsymbol{\Delta}$*-lists*

Definition 5. Given a tnnf A, we define $\Delta_0(A)$ and $\Delta_1(A)$ to be the lists of literals recursively defined below

$$
\begin{aligned}
&\Delta_0(\vartheta\ell) = \Delta_1(\vartheta\ell) = \vartheta\ell\\
&\Delta_0\left(\textstyle\bigwedge_{i=1}^n A_i\right) = \texttt{Union}_\wedge(\Delta_0(A_1), \dots, \Delta_0(A_n))\\
&\Delta_0\left(\textstyle\bigvee_{i=1}^n A_i\right) = \texttt{Intersection}(\Delta_0(A_1), \dots, \Delta_0(A_n))\\
&\Delta_1\left(\textstyle\bigwedge_{i=1}^n A_i\right) = \texttt{Intersection}(\Delta_1(A_1), \dots, \Delta_1(A_n))\\
&\Delta_1\left(\textstyle\bigvee_{i=1}^n A_i\right) = \texttt{Union}_\vee(\Delta_1(A_1), \dots, \Delta_1(A_n))\\
&\Delta_b\,(FA) = \texttt{Add}_\texttt{F}(\Delta_b(A)) \quad \text{for } b \in \{0, 1\}\\
&\Delta_b\,(GA) = \texttt{Add}_\texttt{G}(\Delta_b(A)) \quad \text{for } b \in \{0, 1\}
\end{aligned}
$$

The description of the operators involved in the definition above is the following:

1. The operators `Add` add a temporal connective to each element of a list of literals and simplify the results to a tnnf according to the rules in Fig. 1.
2. The two versions of `Union` arise because of the intended interpretation of the Δ-sets:
   (a) Elements in Δ_0 should contain *minimal* implicates in the lattices of literals $\texttt{Lit}(\ell)$. Therefore, we apply a union operator which preserves this minimality: the operator $\texttt{Union}_\wedge$.
   This operator gives us the union of two lists of literals on which the following reductions have been applied:
      i. $\vartheta\ell \wedge \vartheta\ell\!\uparrow = \vartheta\ell$, that is $\vartheta\ell \wedge \varrho\ell = \vartheta\ell$ for all $\varrho\ell \in \vartheta\ell\!\uparrow$,

ii. $\vartheta\ell \wedge \overline{\vartheta\ell}{\downarrow} = \bot$, that is $\vartheta\ell \wedge \varrho\overline{\ell} = \bot$ for all $\varrho\overline{\ell} \in \overline{\vartheta\ell}{\downarrow}$ and

iii. The pair $G\oplus^{n+1}\ell$ and $\oplus^{n+1}\ell$ is substituted by $G\oplus^{n}\ell$, for all n.

These reductions can be seen as follows: a pair of literals is substituted by its conjunction whenever it is either a literal or a logical constant; and this can only happen in the cases above.

(b) Elements in Δ_1 should contain *maximal* implicates in the lattices of literals $\mathtt{Lit}(\ell)$. Therefore, we apply a union operator which preserves this maximality: the operator $\mathtt{Union}_\vee$.
Δ_1-sets are considered to be disjunctively connected, so we use $\mathtt{Union}_\vee$. The disjunctive connection in Δ_1 means the application of the following rules $\vartheta\ell \vee \vartheta\ell{\downarrow} = \vartheta\ell$, $\vartheta\ell \vee \overline{\vartheta\ell}{\uparrow} = \top$, and the pair of literals $F\oplus^{n+1}\ell$ and $\oplus^{n+1}\ell$ is simplified to $F\oplus^{n}\ell$ in Δ_1, for all n.

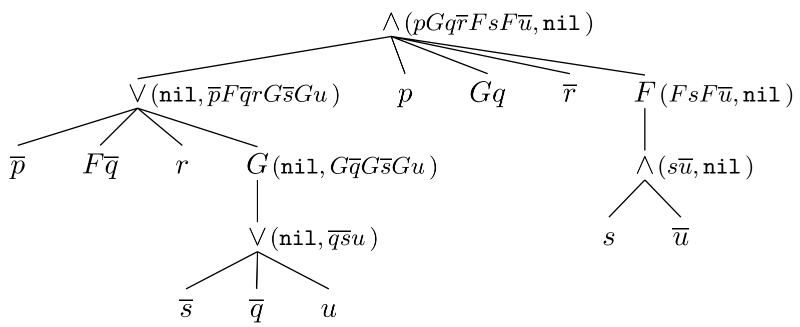
It is easy to see that, for all ℓ, we have that $\Delta_b(A) \cap \mathtt{Lit}(\ell)$ contains at most one literal in the set $\{F\oplus^k\ell, G\oplus^k\ell, FG\ell, GF\ell\}$ and, possibly, several of type $\oplus^k\ell$.

Definition 6. If a A is a tnnf, then to Δ-*label* A means to label each node η in A with the ordered pair $(\Delta_0(\eta), \Delta_1(\eta))$.

Example 7. Consider the formula

$$A = (\neg p \vee \neg Gq \vee r \vee G(\neg s \vee \neg q \vee u)) \wedge \neg(\neg p \vee \neg Gq \vee r \vee G(\neg s \vee u))$$

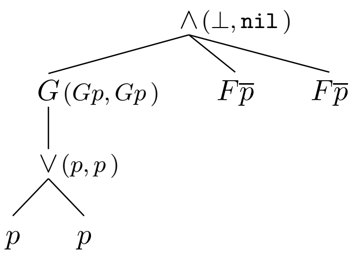
the Δ-labelled tree of A is[2]



Note that in node 1, literals $F\overline{q}$ and $G\overline{q}$ are collapsed into $F\overline{q}$, because of the disjunctive connection in Δ_1.

[2] For the sake of clarity, the Δ-labels of the leaves are not written.

Example 8. Let us study the validity of $A = G(\neg p \to p) \to (\neg Gp \to Gp)$. The Δ-labelled tree equivalent to $\neg A$ is



In this case, $\Delta_0(\varepsilon) = \bot$, because of the simplification of Gp and $F\overline{p}$ due to the conjunctive nature of the Δ_0-sets. We will see later that $\Delta_0(\varepsilon) = \bot$ implies that the input formula, that is $\neg A$, is unsatisfiable, therefore A is valid.

3.3. Information in the $\boldsymbol{\Delta}$*-lists*

As indicated above, the purpose of defining Δ_0 and Δ_1 is to collect implicants and implicates of A, as shown in the following theorem.

Theorem 9. Let A be a tnnf,

1. If $\vartheta\ell \in \Delta_0(A)$, then $\models A \to \vartheta\ell$.
2. If $\vartheta\ell \in \Delta_1(A)$, then $\models \vartheta\ell \to A$.

Proof.

1. By structural induction on A.
   (a) If $A = \vartheta\ell$, then the result is obvious.
   (b) If either $A = FB$ or $A = GB$, then the result follows from the definition of the operators `Add` and the fact that if $\models B \to C$ then $\models FB \to FC$ and $\models GB \to GC$.
   (c) If $A = B \wedge C$, then by the definition of the Δ-lists we have two possibilities:
      i. Either $\vartheta\ell \in \Delta_0(B)$ or $\vartheta\ell \in \Delta_0(C)$. Now, by the induction hypothesis, either $\models B \to \vartheta\ell$ or $\models C \to \vartheta\ell$, and $\models A \to \vartheta\ell$ in either case.
      ii. $\vartheta\ell \equiv \vartheta'\ell \wedge \vartheta''\ell$ where $\vartheta'\ell \in \Delta_0(B)$ and $\vartheta''\ell \in \Delta_0(C)$, that is $\vartheta\ell$ has been introduced by a simplification. By induction hypothesis, we

have $\models B \to \vartheta'\ell$ and $\models C \to \vartheta''\ell$, therefore $\models A \to (\vartheta'\ell \wedge \vartheta''\ell)$ and we have that $\models A \to \vartheta\ell$.

(d) If $A = B \vee C$, then $\vartheta\ell \in \Delta_0(B) \cap \Delta_0(C)$. Now, by the induction hypothesis, $\models B \to \vartheta\ell$ and $\models C \to \vartheta\ell$, therefore $\models A \to \vartheta\ell$.

2. By duality.

□

The theorem above will be used in the equivalent form stated below:

Corollary 10. Let A be a tnnf,

1. If $\vartheta\ell \in \Delta_0(A)$, then $A \equiv A \wedge \vartheta\ell$.
2. If $\vartheta\ell \in \Delta_1(A)$, then $A \equiv A \vee \vartheta\ell$.

As a literal is satisfiable, by Corollary 10 item 2, we have the following result:

Corollary 11. If $\Delta_1(A) \neq \texttt{nil}$, then A is satisfiable. In addition, if $\vartheta\ell \in \Delta_1(A)$, then any model of $\vartheta\ell$ in t is a model of A in t.

3.4. Strong Meaning-Preserving Reductions

A lot of information can be extracted from the Δ-lists as consequences of Corollary 10. The first result is a structural one, for it says that either one of the Δ-lists is empty, or both are equal and singletons.

Corollary 12. If $\Delta_1(A) \neq \texttt{nil} \neq \Delta_0(A)$, then there exists $\vartheta\ell$ such that $\Delta_1(A) = \Delta_0(A) = \vartheta\ell$. In this case, if A is not a literal, then $A \equiv \vartheta\ell$.

Proof. Follows easily by structural induction on A. □

The corollary below states conditions on the Δ-lists which allow to determine the validity or unsatisfiability of the formula we are studying.

Theorem 13. Let A be a tnnf, then

1. (a) If $\Delta_0(A) = \perp$, then $A \equiv \perp$.

   (b) If $A = \bigwedge_{i=1}^{n} A_i$ in which a conjunct A_{i_0} is a clause such that $\overline{\Delta_1(A_{i_0})} \subseteq \Delta_0(A)\!\uparrow$, then $A \equiv \perp$.

(c) If $A = \bigwedge_{i=1}^{n} A_i$ in which a conjunct A_{i_0} is a G-clause GB such that $\overline{\mathtt{Add}_{\oplus}(\Delta_1(B))} \subseteq \Delta_0(A)\uparrow$, then $A \equiv \perp$.

2. (a) If $\Delta_1(A) = \top$, then $A \equiv \top$.

   (b) If $A = \bigvee_{i=1}^{n} A_i$ in which a disjunct A_{i_0} is a cube such that $\overline{\Delta_0(A_{i_0})} \subseteq \Delta_1(A)\downarrow$, then $A \equiv \top$.

   (c) If $A = \bigvee_{i=1}^{n} A_i$ in which a disjunct A_{i_0} is an F-cube FB such that $\overline{\mathtt{Add}_{\oplus}(\Delta_0(B))} \subseteq \Delta_1(A)\downarrow$, then $A \equiv \top$.

Proof.

1.a This first item is trivial.

1.b Recall that, by Corollary 10, we have

$$A \equiv \bigwedge_{k=1}^{r} \varrho_k \ell_k \wedge A$$

where the $\varrho_k \ell_k$ are the elements in $\Delta_0(A)$.
If $\Delta_1(A_{i_0}) = \vartheta_1 \ell_1 \cdots \vartheta_m \ell_m$, we have that $A_{i_0} \equiv \bigvee_{j=1}^{m} \vartheta_j \ell_j$; thus,

$$A \equiv A_{i_0} \wedge A \equiv \bigvee_{j=1}^{m} (\vartheta_j \ell_j \wedge A) \equiv \bigvee_{j=1}^{m} \left(\vartheta_j \ell_j \wedge \bigwedge_{k=1}^{r} \varrho_k \ell_k \wedge A \right)$$

let us show that each disjunct is unsatisfiable.
For a fixed $\vartheta_j \ell_j$, as $\overline{\Delta_1(A_{i_0})} \subseteq \Delta_0(A)\uparrow$, there exists $\varrho_{k_j} \ell_j \in \Delta_0(A)$ such that $\varrho_{k_j} \ell_j \leq \overline{\vartheta_j \ell_j}$, but this implies that $\vartheta_j \ell_j \wedge \varrho_{k_j} \ell_j \equiv \perp$ and, therefore the disjunct $(\vartheta_j \ell_j \wedge \bigwedge_{k=1}^{r} \varrho_k \ell_k \wedge A)$ is unsatisfiable.

1.c If $\Delta_1(B) = \vartheta_1 \ell_1 \cdots \vartheta_m \ell_m$, we have that $B \equiv \bigvee_{j=1}^{m} \vartheta_j \ell_j$. By using the fact that $GB \equiv \oplus B \wedge GB$ we have

$$A \equiv A \wedge GB \equiv A \wedge \oplus B$$

So, to deduce the unsatisfiability of A it suffices to prove that $A \wedge \oplus B$ is unsatisfiable; consider the following equivalences

$$A \wedge \oplus B \equiv A \wedge \oplus \bigvee_{j=1}^{m} \vartheta_j \ell_j \equiv A \wedge \bigvee_{j=1}^{m} \mathtt{Add}_{\oplus}(\vartheta_j \ell_j)$$

now, by using the hypothesis $\overline{\mathtt{Add}_{\oplus}(\Delta_1(B))} \subseteq \Delta_0(A)\uparrow$, the argument applied in item 1.b works to prove that $A \wedge \oplus B \equiv \perp$.

2. Follows by duality.

□

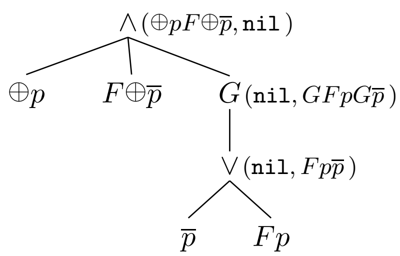
The following definition gives a name to those formulas which have been simplified by using the information in the Δ-lists.

Definition 14. Let A be an tnnf then it is said that A is:

1. *finalizable* if either $A = \top$, or $A = \bot$ or $\Delta_1(A) \neq \texttt{nil}$.
2. A tnnf verifying either (a) or (b) or (c) of item 1 in Theorem 13 is said to be Δ_0-conclusive.
3. A tnnf verifying either (a) or (b) or (c) of item 2 in Theorem 13 is said to be Δ_1-conclusive.
4. A tnnf A such that $\vartheta\ell \in \Delta_0(A) \cap \Delta_1(A)$ is said to be $\vartheta\ell$-simple.
5. A tnnf A is said to be Δ-*restricted* if it has no subtree which is either Δ_0-conclusive, or Δ_1-conclusive, or $\vartheta\ell$-simple.
6. To Δ-*restrict* a tnnf A means to substitute each Δ_1-conclusive formula by $\top$, each Δ_0-conclusive formula by $\bot$, and each $\vartheta\ell$-simple formula by $\vartheta\ell$; and then eliminate the constants $\top$ and $\bot$ by applying the 0-1 laws.
   Note that Δ-restricting is a meaning-preserving transformation.

Example 15. Given the transitivity axiom $A = FFp \to Fp$; the tnnf equivalent to $\neg A$ is $F\oplus p \wedge G\overline{p}$; since $\Delta_0(F\oplus p \wedge G\overline{p}) = \bot$, we have that $\neg A$ is Δ_0-conclusive, therefore $\neg A$ is unsatisfiable and A is valid.

Example 16. Given the formula $A = \oplus p \wedge \oplus F\overline{p} \wedge G(p \to Fp)$, its Δ-labelled tree is



This tree is Δ_0-conclusive, since $\overline{\texttt{Add}_\oplus(\Delta_1(31))} = \oplus pF\oplus\overline{p} \subseteq \Delta_0(\varepsilon)\uparrow$. In fact, what we have in this example is $\overline{\texttt{Add}_\oplus(\Delta_1(31))} = \Delta_0(\varepsilon)$

3.5. *Weak Meaning-Preserving Reductions*

The aim of this section is to give more general conditions allowing to use the information in the Δ-lists which has not been able to be used by the strong reductions. Specifically, a strong reduction uses the information in the Δ-lists in a *strong sense*, that is, to substitute a whole subformula by either $\top$, or $\bot$, or a literal. As in the propositional case, sometimes this is not possible and we can only use the information in a *weak sense*, that is, to decrease the size of the formula by eliminating literals depending on the elements of the Δ-lists.

The following notation is used in the statement of some results hereafter:

- If $\mathcal{S}$ is a set of literals in a tnnf A, then $\mathcal{S}^0$ denotes the set of all the occurrences of literals $\vartheta\ell \in \mathcal{S}$ of temporal order 0 in A
- $\texttt{Lit}(\ell, n) = \{\eta \mid \eta = \vartheta\ell \text{ and } |\vartheta| + ord_A(\eta) \geq n+1\}$

Theorem 17. Let A be a tnnf and $\vartheta\ell$ a literal in A:

1. If $\vartheta\ell \in \Delta_0(A)$, then $A \equiv \vartheta\ell \wedge A[(\vartheta\ell\uparrow)^0/\top, (\overline{\vartheta\ell}\downarrow)^0/\bot]$
2. If $\vartheta\ell \in \Delta_1(A)$, then $A \equiv \vartheta\ell \vee A[(\vartheta\ell\downarrow)^0/\bot, (\overline{\vartheta\ell}\uparrow)^0/\top]$

Proof.

1. By Corollary 10 we have that $A \equiv \vartheta\ell \wedge A$, so we only have to show that
$$\vartheta\ell \wedge A \equiv \vartheta\ell \wedge A[(\vartheta\ell\uparrow)^0/\top, (\overline{\vartheta\ell}\downarrow)^0/\bot]$$
By structural induction
   (a) If A is a literal, then the result is trivial.
   (b) If A is either FB or GB, there there are no literals of temporal order 0, and the result follows from the fact that $A[(\vartheta\ell\uparrow)^0/\top, (\overline{\vartheta\ell}\downarrow)^0/\bot] = A$.
   (c) If $A = A_1 * A_2$, with $* \in \{\wedge, \vee\}$, then by the induction hypothesis on both A_1 and A_2 we have that
$$\begin{aligned}\vartheta\ell \wedge A &\equiv (\vartheta\ell \wedge A_1) * (\vartheta\ell \wedge A_2)\\ &\equiv (\vartheta\ell \wedge A_1[(\vartheta\ell\uparrow)^0/\top, (\overline{\vartheta\ell}\downarrow)^0/\bot]) * (\vartheta\ell \wedge A_2[(\vartheta\ell\uparrow)^0/\top, (\overline{\vartheta\ell}\downarrow)^0/\bot])\\ &\equiv \vartheta\ell \wedge A[(\vartheta\ell\uparrow)^0/\top, (\overline{\vartheta\ell}\downarrow)^0/\bot]\end{aligned}$$
2. The proof is similar.

□

This theorem cannot be improved for an arbitrary literal $\vartheta\ell$; although, for some particular cases, it is possible to get more literals reduced, as shown in Theorem 19, which generalises the result in Theorem 17, by dropping the restriction of temporal order 0 for all the literals in the upward/downward closures. In the proof we will use the results stated in the lemma below:

Lemma 18. The following equivalences hold in FNext:

$$F(FGC \wedge B) \equiv FGC \wedge FB$$
$$F(GFC \wedge B) \equiv GFC \wedge FB$$

Theorem 19. Let A be a tnnf,

1. If $\vartheta\ell \in \Delta_0(A)\!\uparrow$ with $\vartheta\ell \in \{FG\ell, GF\ell\} \cup \{G\oplus^n\ell \mid n \in \mathbb{N}\}$, then
$$A \equiv \vartheta\ell \wedge A[\vartheta\ell\!\uparrow\!/\top, \overline{\vartheta\ell}\!\downarrow\!/\bot]$$
2. If $\vartheta\ell \in \Delta_1(A)\!\downarrow$ with $\vartheta\ell \in \{FG\ell, GF\ell\} \cup \{F\oplus^n\ell \mid n \in \mathbb{N}\}$, then
$$A \equiv \vartheta\ell \vee A[\vartheta\ell\!\downarrow\!/\bot, \overline{\vartheta\ell}\!\uparrow\!/\top]$$

Proof.

1. For an easy and intuitive argument just consider that any literal occurring in A and satisfying $\varrho\ell \in \vartheta\ell\!\uparrow$, regardless of its temporal order, refers to instants of time covered by $\vartheta\ell$.
The formal proof follows by structural induction. The only difference w.r.t. that of Theorem 17 arises when A is either FB or GB; in the rest of the proof we will write C' to denote $C[\vartheta\ell\!\uparrow\!/\top, \overline{\vartheta\ell}\!\downarrow\!/\bot]$.
   (a) If $\vartheta\ell = G\oplus^n\ell$ then the only possibility for A is to be GB, therefore let us assume $G\oplus^n\ell \in \Delta_0(GB)\!\uparrow$:
$$\begin{aligned} G\oplus^n\ell \wedge GB &\equiv G\oplus^n\ell \wedge GG\oplus^n\ell \wedge GB \equiv G\oplus^n\ell \wedge G(G\oplus^n\ell \wedge B) \\ &\equiv G\oplus^n\ell \wedge G(G\oplus^n\ell \wedge B') \qquad \text{(Induction step)} \\ &\equiv G\oplus^n\ell \wedge GG\oplus^n\ell \wedge GB' \\ &\equiv G\oplus^n\ell \wedge GB' \end{aligned}$$

(b) If $GF\ell \in \Delta_0(GB)\upharpoonleft$, then

$$\begin{aligned} GF\ell \wedge GB &\equiv GF\ell \wedge GGF\ell \wedge GB \equiv GF\ell \wedge G(GF\ell \wedge B) \\ &\equiv GF\ell \wedge G(GF\ell \wedge B') && \text{(Induction step)} \\ &\equiv GF\ell \wedge GGF\ell \wedge GB' \\ &\equiv GF\ell \wedge GB' \end{aligned}$$

(c) If $GF\ell \in \Delta_0(FB)\upharpoonleft$, then $GF\ell \in \Delta_0(B)\upharpoonleft$ and

$$\begin{aligned} FB &\equiv F(GF\ell \wedge B') && \text{(Induction step)} \\ &\equiv GF\ell \wedge FB' && \text{(Lemma 18)} \end{aligned}$$

(d) If $FG\ell \in \Delta_0(FB)\upharpoonleft$, then $FG\ell \in \Delta_0(B)\upharpoonleft$ and

$$\begin{aligned} FB &\equiv F(FG\ell \wedge B') && \text{(Induction step)} \\ &\equiv FG\ell \wedge FB' && \text{(Lemma 18)} \end{aligned}$$

(e) If $FG\ell \in \Delta_0(GB)\upharpoonleft$, then $FG\ell \in \Delta_0(B)\upharpoonleft$ and

$$\begin{aligned} GB &\equiv G(FG\ell \wedge B') && \text{(Induction step)} \\ &\equiv GFG\ell \wedge GB' \\ &\equiv FG\ell \wedge GB' \end{aligned}$$

2. By duality.

□

Finally, in the particular cases when $\vartheta\ell$ equals either $G\oplus^n\ell$ or $F\oplus^n\ell$, then a number of additional reductions can be applied. These new reductions are stated in the theorem below.

Theorem 20. Let A be a tnnf and $\vartheta\ell$ a literal in A:

1. If $G\oplus^n\ell \in \Delta_0(A)$, then $A \equiv G\oplus^n\ell \wedge A[\texttt{Lit}(\ell,n)/\top, \texttt{Lit}(\overline{\ell},n)/\bot]$
2. If $F\oplus^n\ell \in \Delta_1(A)$, then $A \equiv F\oplus^n\ell \vee A[\texttt{Lit}(\ell,n)/\bot, \texttt{Lit}(\overline{\ell},n)/\top]$

Proof.

1. The intuition here is that any literal in A which is in $\texttt{Lit}(\ell,n)$ or $\texttt{Lit}(\overline{\ell},n)$ refers to instants of time covered by $G\oplus^n\ell$. The formal proof follows by induction on n.

(a) The base case $n = 0$ holds by Theorem 19, since $\mathtt{Lit}(\ell, 0) \subset G\ell\!\uparrow$ for any formula.

(b) Assume the theorem is true for $n = k-1$. The definition of the Δ-lists and the following equivalences can be used to prove the case $n = k$:

i. $G\oplus^k\ell \wedge (B_1 \vee B_2) \equiv (G\oplus^k\ell \wedge B_1) \vee (G\oplus^k\ell \wedge B_2)$

ii. $G\oplus^k\ell \wedge (B_1 \wedge B_2) \equiv (G\oplus^k\ell \wedge B_1) \wedge (G\oplus^k\ell \wedge B_2)$

iii.

$$\begin{aligned}
G\oplus^k\ell \wedge GB &\equiv G(G\oplus^{k-1}\ell \wedge B)\\
&\equiv G(G\oplus^{k-1}\ell \wedge B[\mathtt{Lit}(\ell, k-1)/\top, \mathtt{Lit}(\overline{\ell}, k-1)/\bot])\\
&\qquad\qquad\qquad\qquad\qquad\qquad\text{(Induction step)}\\
&\equiv G\oplus^k\ell \wedge G(B[\mathtt{Lit}(\ell, k-1)/\top, \mathtt{Lit}(\overline{\ell}, k-1)/\bot]))\\
&\equiv G\oplus^k\ell \wedge GB[\mathtt{Lit}(\ell, k)/\top, \mathtt{Lit}(\overline{\ell}, k)/\bot])
\end{aligned}$$

iv.

$$\begin{aligned}
G\oplus^k\ell \wedge FB &\equiv (G\oplus^k\ell \wedge \oplus B) \vee (G\oplus^k\ell \wedge \oplus FB)\\
&\equiv \oplus(G\oplus^{k-1}\ell \wedge B) \vee \oplus(G\oplus^{k-1}\ell \wedge FB)\\
&\equiv \oplus(G\oplus^{k-1}\ell \wedge B[\mathtt{Lit}(\ell, k-1)/\top, \mathtt{Lit}(\overline{\ell}, k-1)/\bot])\\
&\qquad \vee \oplus(G\oplus^{k-1}\ell \wedge (FB)[\mathtt{Lit}(\ell, k-1)/\top, \mathtt{Lit}(\overline{\ell}, k-1)/\bot])\\
&\qquad\qquad\qquad\qquad\qquad\qquad\text{(Induction step)}\\
&\equiv (G\oplus^k\ell \wedge \oplus B[\mathtt{Lit}(\ell, k-1)/\top, \mathtt{Lit}(\overline{\ell}, k-1)/\bot])\\
&\qquad \vee(G\oplus^k\ell \wedge \oplus F(B[\mathtt{Lit}(\ell, k-2)/\top, \mathtt{Lit}(\overline{\ell}, k-2)/\bot]))\\
&\equiv G\oplus^k\ell \wedge FB[\mathtt{Lit}(\ell, k)/\top, \mathtt{Lit}(\overline{\ell}, k)/\bot]
\end{aligned}$$

2. By duality

□

4. Adding Information to the Tree: $\widehat{\boldsymbol{\Delta}}$-sets

In the previous sections, the information in the Δ-lists has been used *locally*, that is, the information in $\Delta_b(\eta)$ has been used to reduce η. The purpose of defining a new structure, the $\widehat{\Delta}$-sets, is to allow the globalisation of the information, in that the information in $\Delta_b(\eta)$ can be *refined* by the information in its ancestors.

Given a Δ-restricted tnnf A, we define the sets $\widehat{\Delta_0}(A)$ and $\widehat{\Delta_1}(A)$, whose elements are pairs (α, η) where α is a *reduced* Δ-list (to be defined below) associated to a subformula B of A, and η is the address of B in A. These sets allow to transform the formula A into an equisatisfiable and smaller sized one, as seen in Section 4.2.

The application of the reductions in Theorems 17, 19 and 20 sometimes allows to substitute a whole subformula of A by either $\top$ or $\bot$, in the rest of the cases only literals are deleted; these literals will be called *reducible*. Theorem 21 below collects those cases in which we are allowed to substitute a whole subformula of A by either $\top$ or $\bot$.

Theorem 21. Let A be a tnnf, B a subformula of A, and η the address in the tree of A of a subformula of B:

1. (a) If $\vartheta\ell$ is any literal satisfying $\vartheta\ell \in \Delta_0(\eta)\!\uparrow \cap\, (\Delta_1(B) \cup \overline{\Delta_0(B)})$ and $ord_B(\eta) = 0$, then $A \equiv A[\eta/\bot]$.

   (b) If $\vartheta\ell \in \{FG\ell, GF\ell\} \cup \{G\oplus^n\ell \mid n \in \mathbb{N}\}$ and satisfies $\vartheta\ell \in \Delta_0(\eta)\!\uparrow \cap\, (\Delta_1(B) \cup \overline{\Delta_0(B)})$, then $A \equiv A[\eta/\bot]$.

   (c) If $\vartheta\ell \in \Delta_0(\eta)\!\uparrow$, and $F\oplus^n\ell \in \Delta_1(B) \cup \overline{\Delta_0(B)}$, and $|\vartheta| + ord_B(\eta) \geq n+1$, then $A \equiv A[\eta/\bot]$.

2. (a) If $\vartheta\ell$ is any literal satisfying $\vartheta\ell \in \Delta_1(\eta)\!\downarrow \cap\, (\Delta_0(B) \cup \overline{\Delta_1(B)})$ and $ord_B(\eta) = 0$, then $A \equiv A[\eta/\top]$.

   (b) If $\vartheta\ell \in \{FG\ell, GF\ell\} \cup \{G\oplus^n\ell \mid n \in \mathbb{N}\}$ and satisfies $\vartheta\ell \in \Delta_1(\eta)\!\downarrow \cap\, (\Delta_0(B) \cup \overline{\Delta_1(B)})$, then $A \equiv A[\eta/\top]$.

   (c) If $\vartheta\ell \in \Delta_1(\eta)\!\downarrow$, and $G\oplus^n\ell \in \Delta_0(B) \cup \overline{\Delta_1(B)}$, and $|\vartheta| + ord_B(\eta) \geq n+1$, then $A \equiv A[\eta/\top]$.

Proof.

1.(a) Assume $\vartheta\ell \in \Delta_1(B) \cap \Delta_0(\eta)\!\uparrow$ (the case $\vartheta\ell \in \overline{\Delta_0(B)} \cap \Delta_0(\eta)\!\uparrow$ is similar) then Theorem 17 item 2 applied to subformula B, provides a number of substitutions of literals by logical constants. Specifically, as $\vartheta\ell \in \Delta_0(\eta)\!\uparrow$ there exists an element $\varrho\ell \in \Delta_0(\eta)$ such that $\varrho\ell \leq \vartheta\ell$ which can be substituted by $\bot$, recalling that $\eta \equiv \eta \wedge \varrho\ell$ we have that formulas A and $A[\eta/\bot]$ turn out to be equivalent.

1.(b-c), 2. These items are proved similarly.

□

This theorem can be seen as a generalisation of Theorem 13, in which a subformula B can be substituted by a constant even when that subformula *is not equivalent to* that constant.

The subformula at address η in A is said to be 0-*conclusive in A (resp.* 1-*conclusive in A)* if it verifies some of the conditions in item 1 (resp. item 2) above.

Definition 22. Given a tnnf A and an address η, the *reduced* Δ-lists for A, $\Delta_b^A(\eta)$ for $b \in \{0, 1\}$, are defined below,

1. If η is 0-conclusive in A, then $\Delta_0^A(\eta) = \bot$.
2. If η is 1-conclusive in A, then $\Delta_1^A(\eta) = \top$.
3. Otherwise, $\Delta_b^A(\eta)$ is the list $\Delta_b(\eta)$ in which the reducible literals have been deleted.

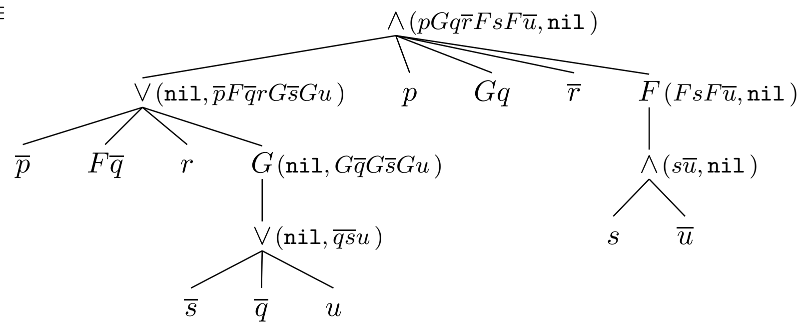
We define the sets $\widehat{\Delta_b}(A)$ as follows

$$\widehat{\Delta_b}(A) = \{(\Delta_b^A(\eta), \eta) \mid \ \eta \text{ is a non-leaf address in } T_A \text{ with } \Delta_b(\eta) \neq \texttt{nil}\}$$

If A is a tnnf, to *label* A means Δ-label A and to associate to the root of A the ordered pair $(\widehat{\Delta_0}(A), \widehat{\Delta_1}(A))$.

Example 23. From Example 7 we had the following tree

$A \equiv$



Note that literals $\overline{p}, F\overline{q}$ and r in Δ_1 of node 1 are reducible in A because of the occurrence of its duals in Δ_0 of the root. Similarly $G\overline{q}$ is also reducible in node 14, and $\overline{q}$ is reducible in 141. Therefore, the calculation of the $\widehat{\Delta}$-sets leads to

$$\widehat{\Delta}_0(A) = \{(pGq\overline{r}FsF\overline{u}, \varepsilon), (FsF\overline{u}, 5), (s\overline{u}, 51)\}$$

$$\widehat{\Delta}_1(A) = \{(G\overline{s}Gu, 1), (G\overline{s}Gu, 14), (\overline{s}u, 141)\}$$

4.1. Some words about the complexity of labelling

All the tests involved in the labelling process work on the Δ_b-lists of the given formula. The complexity of the Δ-labelling of a tnnf is linear both in space and in time, the proof is sketched below.

Let us estimate the ratio between the size of the set of Δ-lists and the size of the formula. The worst case is that of a balanced tree W of a tnnf in which binary and monary connectives alternate and all the literals in the branch get included in the Δ-lists of all their ancestors, not only their predecessor. Actually, this is the worst case for the calculation of Δ-lists but a trivial one for the satisfiability analysis with reductions.

As the number of nodes decreases geometrically each two levels of depth (in ascending order), then the number of elements in all the Δ-lists is bounded by four times the number of leaves of W. Therefore, we have at most a linear increase in size.

On the other hand, the labelling of a tnnf consists of the determination of the $\widehat{\Delta}$-sets of the root node, which is a subset of the set just bounded above. Thus, in the worst case (in which no b-conclusive nodes are present) the cardinal of this set together with the addresses of the nodes increases at most linearly w.r.t. the input tree.

Finally, the calculations involved in both determining each Δ-list and the tests of the hypotheses in Theorem 21 are based on intersections and unions on lists and comparisons between Δ-lists of consecutive nodes in the tree. As we use ordered lists, all these operations can be performed in linear time; note that the simplifications based on the lattice of literals can be done also in linear time and, in addition, can decrease the complexity of the union operation.

4.2. Satisfiability-Preserving Results

The use of the information in the $\widehat{\Delta}$-sets is analyzed in this section.

Definition 24. A tnnf A is said to be *restricted* if it is Δ-restricted and satisfies the following:

- There are not elements $(\bot, \eta)$ in $\widehat{\Delta_0}(A)$.
- There are not elements $(\top, \eta)$ in $\widehat{\Delta_1}(A)$.

Remark 25. A restricted and equivalent tnnf can be obtained by using the 0-1 laws in conjunction with the elimination of conclusive subformulas in A, according to Theorem 21.

The following results will allow, by using the information in the $\widehat{\Delta}$-sets, to substitute a tnnf A by an equisatisfiable and smaller sized A'.

4.2.1. Complete Reduction

This section is named after Theorem 26, because after its application on a literal $G\oplus^n\ell$, gives an equisatisfiable formula whose only literals in ℓ are of the form $\oplus^n\ell$.

Theorem 26. Let A be a tnnf such that $G\oplus^n\ell \in \alpha$ for some $(\alpha,\varepsilon) \in \widehat{\Delta_0}(A)$, and consider the formulas

$$B = A[\mathtt{Lit}(\ell,n) \cup G\oplus^n\ell\!\uparrow/\top, \mathtt{Lit}(\overline{\ell},n) \cup F\oplus^n\overline{\ell}\!\downarrow/\bot]$$
$$C = B[G\oplus^k\ell/\oplus^{k+1}\ell \wedge \ldots \wedge \oplus^n\ell, F\oplus^k\overline{\ell}/\oplus^{k+1}\overline{\ell} \vee \ldots \vee \oplus^n\overline{\ell}]$$

if every literal on ℓ or $\overline{\ell}$ occurring in the formula C have temporal order 0, then A is satisfiable if and only if C is satisfiable.

Furthermore, if h is a model of C in t, then the interpretation h' such that $h'(q) = h(q)$ if $q \neq \ell$ and $h'(\ell) = h(\ell) \cup [t+n+1,\infty)$ is a model of A in t.

Proof. For all $m < n$ we have:

$$G\oplus^m\ell \equiv \oplus^{m+1}\ell \wedge \oplus^{m+2}\ell \wedge \cdots \wedge \oplus^n\ell \wedge \oplus^n G\ell$$
$$F\oplus^m\ell \equiv \oplus^{m+1}\ell \vee \oplus^{m+2}\ell \vee \cdots \vee \oplus^n\ell \vee \oplus^n F\ell$$

The equivalences above, the hypothesis $G\oplus^n\ell \in \Delta_0(A)$ and Theorems 19 and 20 lead to the equivalence

$$A \equiv G\oplus^n\ell \wedge B[G\oplus^k\ell/\oplus^{k+1}\ell \wedge \ldots \wedge \oplus^n\ell, F\oplus^k\overline{\ell}/\oplus^{k+1}\overline{\ell} \vee \ldots \vee \oplus^n\overline{\ell}] \equiv G\oplus^n\ell \wedge C$$

We only have to show that the deletion of $G\oplus^n\ell$ is satisfiability-preserving.

Note that the only occurrences of literals on ℓ (or $\overline{\ell}$) are of the form $\oplus^m\ell$. Since every literal on ℓ or $\overline{\ell}$ occurring in the formula C have temporal order 0, then the only temporal requirements on ℓ are in the next n instants of time.

Therefore, any model h of C in t can be easily extended to a model h' of A in t just by making $h'(\ell) = h(\ell) \cup [t + n + 1, \infty)$. $\square$

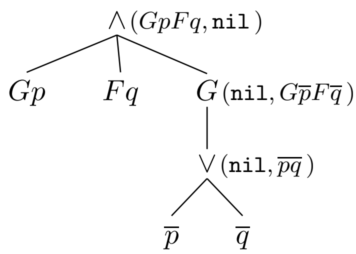
Definition 27. A tnnf A satisfying the hypothesis of the theorem above is said to be *completely reducible.*

Example 28. Given the density axiom $A = Fp \to FFp$; the formula $\neg A$ is equivalent to the tnnf $Fp \wedge G{\oplus}\overline{p}$.

We have that $\Delta_0(Fp \wedge G{\oplus}\overline{p}) = FpG{\oplus}\overline{p}$. Note that, as the conjunction of Fp and $G{\oplus}\overline{p}$ is not a literal, no simplification can be applied. In addition, its $\widehat{\Delta_0}$-set is $\{(FpG{\oplus}\overline{p}, \varepsilon)\}$, thus $\neg A$ is completely reducible.

Now applying Theorem 26, we get that $\neg A$ is satisfiable if and only if $\oplus p$ is satisfiable. Therefore $\neg A$ is satisfiable, a model being $h(\overline{p}) = [2, \infty)$, $h(p) = \{1\}$.

Example 29. Given the formula $A = (Gp \wedge Fq) \to F(p \wedge q)$, we have $\neg A \equiv Gp \wedge Fq \wedge G(\overline{p} \vee \overline{q})$; its Δ-restricted form is



and its $\widehat{\Delta}$-sets are:

$$\widehat{\Delta}_0(A) = \{(GpFq, \varepsilon)\} \qquad \widehat{\Delta}_1(A) = \{(G\overline{p}F\overline{q}, 3), (\overline{pq}, 31)\}$$

This formula is completely reducible, by an application of Theorem 26, the leaf in node 1 is deleted, and node 3 is substituted by $G\overline{q}$.

The resulting formula is $Fq \wedge G\overline{q}$, which is 0-conclusive and, therefore, unsatisfiable.

4.2.2. The Pure Literal Rule

The result introduced here is an extension of the well known *pure literal rule* for Classical Propositional Logic. Existing results in the bibliography allow a straightforward extension of the concept of pure literal. Our definition makes use of the $\widehat{\Delta}$-sets, which allow to focus only on those literals which are essential parts of the formula; this is because reducible literals *are not included* in the $\widehat{\Delta}$-sets.

Definition 30. Let A be a tnnf.

1. A classical literal ℓ is said to be $\widehat{\Delta}$-pure in A if a literal $\vartheta\ell$ occurs in $\widehat{\Delta_0}(A) \cup \widehat{\Delta_1}(A)$ and no literal on $\vartheta'\overline{\ell}$ occurs in $\widehat{\Delta_0}(A) \cup \widehat{\Delta_1}(A)$.
2. A classical literal ℓ is said to be $\widehat{\Delta}$-k-pure in A if $\oplus^k\ell$ occurs in an $(\alpha,\eta) \in \widehat{\Delta_0}(A)\cup\widehat{\Delta_1}(A)$ with $ord_A(\eta) = 0$, $\oplus^k\overline{\ell}$ does not occur in any $(\alpha,\eta) \in \widehat{\Delta_0}(A)\cup\widehat{\Delta_1}(A)$ with $ord_A(\eta) = 0$, and for any other literal $\vartheta\ell$ or $\vartheta'\ell$, occurring in some element $(\alpha,\eta) \in \widehat{\Delta_0}(A) \cup \widehat{\Delta_1}(A)$, we have $|\vartheta| + ord_A(\eta) > k$.

Theorem 31. Let A be a tnnf, ℓ a $\widehat{\Delta}$-pure literal in A, and B the formula obtained from A by the following substitutions

1. If $(\alpha,\eta) \in \widehat{\Delta_0}(A)$ with $\vartheta\ell \in \alpha$, then η is substituted by
$$\begin{cases} \eta[\mathtt{Lit}(\ell,n) \cup G{\oplus}^n\ell{\uparrow}/\top, \mathtt{Lit}(\overline{\ell},n) \cup F{\oplus}^n\overline{\ell}{\downarrow}/\bot] & \text{if } \vartheta\ell = G{\oplus}^n\ell \\ \eta[\vartheta\ell{\uparrow}/\top, \overline{\vartheta\ell}{\downarrow}/\bot] & \text{if } \vartheta\ell \in \{GF\ell, FG\ell\} \\ \eta[(\vartheta\ell{\uparrow})^0/\top, (\overline{\vartheta\ell}{\downarrow})^0/\bot] & \text{otherwise} \end{cases}$$
2. If $(\alpha,\eta) \in \widehat{\Delta_1}(A)$ with $\vartheta\ell \in \alpha$, then η is substituted by $\top$.

Then, A is satisfiable if and only if B is satisfiable. Furthermore, if h is a model of B in t, then the interpretation h' such that $h'(\ell') = h(\ell')$ if $\ell' \neq \ell$ and $h'(\ell) = [t,\infty)$ is a model of A in t.

Proof. Let C be the formula obtained from A by applying the meaning-preserving substitutions below, given by Theorems 17, 19 and 20:

1. For all $(\alpha,\eta) \in \widehat{\Delta_0}(A)$ with $\vartheta\ell \in \alpha$, substitute η by
$$\begin{cases} \vartheta\ell \wedge \eta[\mathtt{Lit}(\ell,n) \cup G{\oplus}^n\ell{\uparrow}/\top, \mathtt{Lit}(\overline{\ell},n) \cup F{\oplus}^n\overline{\ell}{\downarrow}/\bot] & \text{if } \vartheta\ell = G{\oplus}^n\ell \\ \vartheta\ell \wedge \eta[\vartheta\ell{\uparrow}/\top, \overline{\vartheta\ell}{\downarrow}/\bot] & \text{if } \vartheta\ell \in \{GF\ell, FG\ell\} \\ \vartheta\ell \wedge \eta[(\vartheta\ell{\uparrow})^0/\top, (\overline{\vartheta\ell}{\downarrow})^0/\bot] & \text{otherwise} \end{cases}$$
2. For all $(\alpha,\eta) \in \widehat{\Delta_1}(A)$ with $\vartheta\ell \in \alpha$, substitute η by
$$\begin{cases} \vartheta\ell \vee \eta[\mathtt{Lit}(\ell,n) \cup F{\oplus}^n\ell{\downarrow}/\bot, \mathtt{Lit}(\overline{\ell},n) \cup G{\oplus}^n\overline{\ell}{\uparrow}/\top] & \text{if } \vartheta\ell = F{\oplus}^n\ell \\ \vartheta\ell \vee \eta[\vartheta\ell{\downarrow}/\bot, \overline{\vartheta\ell}{\uparrow}/\top] & \text{if } \vartheta\ell \in \{GF\ell, FG\ell\} \\ \vartheta\ell \vee \eta[(\vartheta\ell{\downarrow})^0/\bot, (\overline{\vartheta\ell}{\uparrow})^0/\top] & \text{otherwise} \end{cases}$$

As $A \equiv C$ it is enough to prove that B and C are equisatisfiable.

Note that, since ℓ is pure, there are no occurrences of $\overline{\ell}$ in formula C; also note that, by construction, neither ℓ nor $\overline{\ell}$ occur in B.

Any model h_B of B in t can be easily extended to a model h_C of C in t just by considering $h_C(\ell) = [t, \infty)$ and $h_C = h_B$ otherwise.

Conversely, given a model h_C of C in t, consider h_B to be defined by $h_B(\ell) = [t, \infty)$ and $h_B = h_C$ otherwise. Now, by monotonicity of $\vee$, $\wedge$, F and G, it is easy checked that h_B is a model of B in t. □

Theorem 32. Let A be a tnnf, ℓ a $\widehat{\Delta}$-k-pure literal in A, and B the formula obtained from A by the following substitutions

1. If $(\alpha, \eta) \in \widehat{\Delta_0}(A)$ with $\oplus^k \ell \in \alpha$ and $ord_A(\eta) = 0$, then η is substituted by $\eta[(\oplus^k \ell\uparrow)^0/\top, (\oplus^k \overline{\ell}\downarrow)^0/\bot]$
2. If $(\alpha, \eta) \in \widehat{\Delta_1}(A)$ with $\oplus^k \ell \in \alpha$, then η is substituted by $\top$

Then, A is satisfiable if and only if B is satisfiable. Furthermore, if h is a model of B in t, then the interpretation h' such that $h'(\ell') = h(\ell')$ if $\ell' \neq \ell$ and $h'(\ell) = h(\ell) \cup \{t + k\}$ is a model of A in t.

Proof. It is similar to the previous one, using only the reductions given by Theorem 17, this is obvious since Theorems 19 and 20 are not applicable to $\oplus^k \ell$. Also, note that the restriction $ord_A(\eta) = 0$ is necessary to be sure that we are really talking about the next k-th instant of time, in order to construct the models properly. □

Example 33. Following with the formula in Example 23, we had

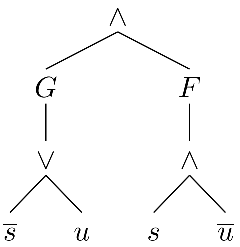
$$\widehat{\Delta}_0(A) = \{(pGq\overline{r}, \varepsilon), (FsF\overline{u}, 5), (s\overline{u}, 51)\}$$
$$\widehat{\Delta}_1(A) = \{(G\overline{s}Gu, 1)(G\overline{s}Gu, 14), (\overline{s}u, 141)\}$$

therefore

1. It is completely reducible: $Gq \in \alpha$ with $(\alpha, \varepsilon) \in \widehat{\Delta}_0(A)$.
2. literals p and $\overline{r}$ are 0-pure.

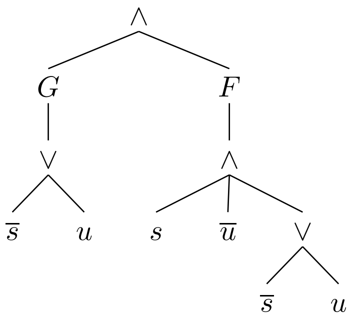
When applying the corresponding substitutions we get

$B =$



This formula cannot be reduced any longer. By applying a branching rule[3] we obtain

$B \equiv$



It is easy to check that node 21 is Δ_0-conclusive, by substituting this node by $\perp$ we get $\perp$ as a final result. Therefore the formula is unsatisfiable.

5. Conclusions and Future Work

We have introduced techniques for defining and manipulating lists of unitary implicants/implicates which can improve the performance of a given prover for temporal propositional logics by decreasing the size of the formulas to be branched. These strategies are interesting because can be used in *any* existing theorem prover, specially in non-clausal ones.

As future work, the information in the Δ-lists can be increased by refining the process of generation of temporal implicants/implicates. In addition, current work on G-clauses and F-cubes appears to be a new source of reduction results.

[3] Every prover for linear-time temporal logic has such rules, in the example we use just one of those in the literature.

References

[1] G. Aguilera, I. P. de Guzmán, and M. Ojeda. Increasing the efficiency of automated theorem proving. *Journal of Applied Non-Classical Logics*, 5(1):9–29, 1995.

[2] R. Ben-Eliyahu and M. Magidor. A temporal logic for proving properties of topologically general executions. *Information and Computation* 124(2):127–144, 1996.

[3] C. Courcoubetis and M. Yannakakis. The complexity of probabilistic verification. *Journal of the ACM* 42(4):857–907, 1995.

[4] L. Fix and O. Grumberg. Verification of temporal properties. *Journal of Logic and Computation* 6(3):343–362, 1996.

[5] Z. Manna and A. Pnueli. *The temporal logic of reactive and concurrent systems: specifications*. Springer-Verlag, 1992.

[6] Z. Manna and A. Pnueli. *Temporal verification of reactive systems: Safety*. Springer-Verlag, 1995.

[7] A. P. Sistla and E. M. Clarke. The complexity of propositional linear temporal logics. *Journal of the ACM*, 32(3):733-749, 1985.

[8] P. Wolper. The tableaux method for temporal logic: an overview. *Logique et Analyse* 28 année, 110-111:119–136, 1985.