Towards an Efficient Implementation of Tableaux for Reactive Safety Specifications *

Ander Alonso[1], Montserrat Hermo[2], and Josu Oca[2]

[1] Developair, San Sebastián, Spain
[2] University of the Basque Country, San Sebastián, Spain

Abstract. In this paper, we will show how to handle a new normal form called *terse normal form* (TNF), which is crucial to the development of a novel tableau method for deciding the realizability of specifications expressed in a safety fragment of LTL. The construction of these tableaux is based on the conversion of LTL formulas into TNF, the most computationally expensive part of the method. We will explain how to efficiently extract the relevant information required by the tableaux without having to compute the entire TNF of a safety formula.

Keywords: temporal logic · realizability · safety specifications · tableaux

1 Introduction

Linear Temporal Logic (LTL) [11] is a modal logic whose uses include expressing properties of reactive systems. The model checking problem decides, given a system S and an LTL specification φ, whether S models φ. Reactive synthesis is the problem of automatically producing S from φ with the guarantee that S models φ. On the other hand, the realizability problem decides whether S exists or not. In reactive synthesis specifications, the atomic variables are divided into variables controlled by the environment and the rest, controlled by the system. A specification is unrealizable if it constrains the environment, but when there is no environment, it is exactly the problem of deciding whether an LTL formula is satisfiable

In the last two decades, the problems of realizability and synthesis for reactive systems have received much attention (e.g., [2,3,4,5,9]), but to our knowledge, there was no tableaux technique that addresses the problems of realizability and synthesis until the publication of [7], where we introduced a tableau-based method for the realizability of a special type of reactive specifications called safety specifications [12].

Traditional tableau techniques for the satisfiability of LTL formulas do not directly work for realizability and synthesis. To illustrate the problem, consider

* This work was funded by the European Union (ERDF funds) under grant PID2020-112581GB-C22, European COST Action CA20111 EuroProofNet (European Research Network on Formal Proofs), and by the University of the Basque Country under project LoRea GIU21/044.

the following formula where p_e is an environment variable and s is a system variable: $\square\psi = \square(\bigcirc s \leftrightarrow \bigcirc p_e)$. The temporal operator $\bigcirc$ refers to the next time instant while the temporal operator $\square$ refers to all instants of time. In temporal logic, instants of time are usually called states.

The specification $\square\psi$ is realizable because the only thing the system has to do is to mimic in s the value observed in p_e. However, a traditional tableau for $\square\psi$ would work as shown in Fig. 1.

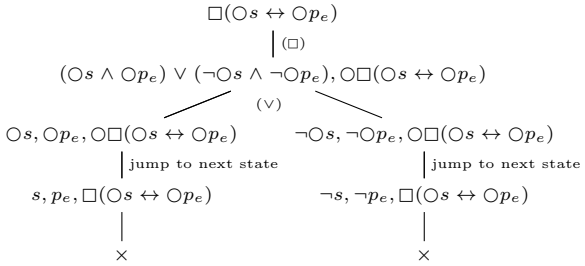

Fig. 1. Traditional tableau for $\square\psi$.

The left branch (respectively the right branch) in the tableau is closed because if the environment chooses $\neg p_e$ (respectively p_e) in the next state, the tableau incorrectly concludes that $\square\psi$ is unrealizable. The problem here is in the splitting of the two cases $\bigcirc s, \bigcirc p_e$ and $\bigcirc\neg s, \bigcirc\neg p_e$, which reveals too early the future choice of the system given the power to the environment to create a contradiction. To overcome this problem, in [7], we introduced the *terse normal form* (TNF) of formulas which prevents these incorrect splittings on formulas that reveal future choices too early. Figure 4 gives you a flavor of what a correct tableau would look like for $\square\psi = \square(\bigcirc s \leftrightarrow \bigcirc p_e)$.

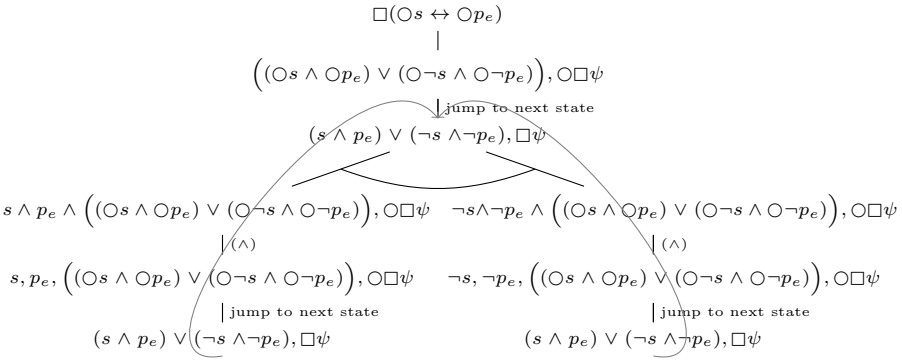

Fig. 2. Correct open tableau for $\square\psi = \square(\bigcirc s \leftrightarrow \bigcirc p_e)$ according the the tableau method presented in [7].

The construction of a TNF was explained very theoretically in [7] and in this paper we present an efficient way to implement it. Actually, the implementation we propose does not build the complete TNF, but only the formulas necessary for the tableau to decide correctly.

Structure of the paper The paper is divided into three main sections, which in turn contain subsections. Section 2 recalls definitions from temporal logic and realizabiblity, as well as the grammar of the safety specifications. The main concepts introduced in this section are the TNF and the set of minimal coverings that any TNF contains. This section is a summary of what was presented in [7].

Section 3 is entirely new and introduces the concept of *weaker minimal coverings*, which is ultimately what any tableau needs to decide correctly. In this section, we present an algorithm that constructs the set of all the *weakest* coverings for a safety specification (from which the minimal coverings can be extracted). The construction is performed without building the complete TNF. Finally, the correctness of the algorithm is proved.

2 Preliminaries

Given any set $\mathcal{V}$ of propositional variables, an assignment v is a map $\mathcal{V} \to \{true, false\}$. We denote by $\mathsf{Val}(\mathcal{V})$ the set of all assignments of $\mathcal{V}$. Given a particular formula φ on variables $\mathcal{V}$, we denote by $var(\varphi)$ the set of variables in $\mathcal{V}$ that occur in φ and by $\mathsf{Val}_\varphi(\mathcal{V})$ the maps $\mathcal{V} \to \{true, false\}$ that assign $true$ to φ.

Safety specifications. We deal with a fragment of LTL, which contains safety specifications [7]. These specifications are defined on two disjoint sets of variables: $\mathcal{E}$, environment variables, and $\mathcal{S}$, system variables. We use a subscript e (e.g., q_e) for the variables in $\mathcal{E}$. For us, the Boolean constants $\mathbf{T}$ and $\mathbf{F}$ are literals.

The fragment of safety LTL specifications consists of formulas $\alpha \wedge \Box\psi$ defined on variables $\mathcal{E} \cup \mathcal{S}$, where α is a Boolean formula that expresses a constraint at the initial state. The formula $\Box\psi$, called the safety formula[3], restricts the behavior in all states by means of the following temporal operators:

$$\gamma ::= p \mid \neg\gamma \mid \bigcirc\gamma \mid \gamma \vee \gamma \mid \gamma \wedge \gamma \qquad \text{where } p \in \mathcal{E} \cup \mathcal{S} \cup \{\mathbf{T}, \mathbf{F}\}$$

Any safety specification is logically equivalent to a formula that only applies negation to variables (negation normal form) and only applies the $\bigcirc$-operator to Boolean literals by pushing the negation and the $\bigcirc$-operator to the propositional level . This can be done using propositional equivalences and the equivalences $\neg\bigcirc\gamma \equiv \bigcirc\neg\gamma$, $\bigcirc(\gamma_1 \wedge \gamma_2) \equiv \bigcirc\gamma_1 \wedge \bigcirc\gamma_2$, and $\bigcirc(\gamma_1 \vee \gamma_2) \equiv \bigcirc\gamma_1 \vee \bigcirc\gamma_2$. For technical reasons that improve the efficiency of the tableau method, our safety specifications are of the form $\alpha \wedge \Box\gamma$, where γ is as follows.

$$\ell ::= p \mid \neg p \qquad \gamma ::= \ell \mid \bigcirc^i \ell \mid \gamma \vee \gamma \mid \gamma \wedge \gamma \quad \text{where } i \geq 1 \text{ and } p \in \mathcal{E} \cup \mathcal{S} \cup \{\mathbf{T}, \mathbf{F}\}$$

Realizability of safety specifications. When $\mathcal{E}$ is empty, i.e., when we only have system variables, deciding whether a safety specification is realizable or not is equivalent to deciding whether is satisfiable or not. The interpretation of safety formulas only defined on system variables $\mathcal{S}$ is over traces. An infinite *trace* σ

[3] Throughout the paper and abusing notation, we also call ψ a safety formula.

is a denumerable sequence of states $\sigma_0, \sigma_1, \sigma_2, \ldots$ where each state $\sigma_i \in 2^{\mathcal{S}}$. Intuitively, σ_i represents the propositions that are satisfied in state i. Given a trace $\sigma = \sigma_0, \sigma_1, \sigma_2, \ldots$, by σ^i we denote the trace $\sigma_i, \sigma_{i+1}, \ldots$, where $i \geq 0$. Given a trace σ and a safety formula φ on $\mathcal{S}$, the meaning of $\sigma \models \varphi$ is the following.

$\sigma \models \Box\varphi$	iff $\sigma^j \models \varphi$ for all $j \geq 0$.	$\sigma \models \mathbf{T}$.	$\sigma \not\models \mathbf{F}$.
$\sigma \models p$	iff $p \in \sigma_0$.	$\sigma \models \neg p$	iff $p \notin \sigma_0$.
$\sigma \models \bigcirc^i p$	iff $\sigma^i \models p$.	$\sigma \models \bigcirc^i \neg p$	iff $\sigma^i \models \neg p$.
$\sigma \models \varphi \wedge \psi$	iff $\sigma \models \varphi$ and $\sigma \models \psi$.	$\sigma \models \varphi \vee \psi$	iff $\sigma \models \varphi$ or $\sigma \models \psi$.

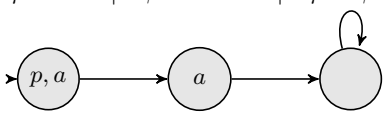

Fig. 3. Representations of a trace that is a model of $a \wedge \Box((p \rightarrow \bigcirc a \wedge (\neg p \rightarrow \bigcirc \neg a))$.

Example 1. Let $\mathcal{S} = \{p, a\}$, and the formula $\varphi = a \wedge \Box((p \rightarrow \bigcirc a \wedge (\neg p \rightarrow \bigcirc \neg a))$. This formula is satisfiable because the trace $\sigma_0 = \{p, a\}$, $\sigma_1 = \{a\}$, $\sigma_2 = \emptyset$, and $\sigma_i = \sigma_2$ for all $i > 2$. is a model of φ. Figure 3 represents this trace.

Now, we formally introduce the realizability problem.

Definition 1 (Realizability). *Let $\alpha \wedge \Box\psi$ be a safety specification on variables from $\mathcal{E} \cup \mathcal{S}$. We say that $\alpha \wedge \Box\psi$ is realizable if there is a function $\sigma : (2^{\mathcal{E}})^* \rightarrow 2^{\mathcal{S}}$, called a winning strategy for the system, such that for every infinite environment-string $E = E_0 \cdot E_1 \cdot E_2 \cdots \in (2^{\mathcal{E}})^{\omega}$, called an environment strategy, it holds that $E_0 \cup \sigma(E_0)$ is a model of α and the infinite trace $E_0 \cup \sigma(E_0), E_1 \cup \sigma(E_0 \cdot E_1), E_2 \cup \sigma(E_0 \cdot E_1 \cdot E_2), \cdots$ is a model of $\Box\psi$.*

Example 2. Let $\mathcal{E} = \{p_e\}$, $\mathcal{S} = \{a\}$, and the safety specification $a \wedge \Box((p_e \rightarrow \bigcirc a \wedge (\neg p_e \rightarrow \bigcirc \neg a))$. This specification is realizable because the system chooses a (i.e., it assigns *true* to the variable a) at the first state, independently of the environment play at this state. Next, if the environment chose p_e (i.e., it played assigning *true* to p_e) in the previous state, the system chooses a in the current state. On the contrary, if the environment chose $\neg p_e$, then the system chooses $\neg a$ in the current state. Figure 4 represents the game between the system and the environment.

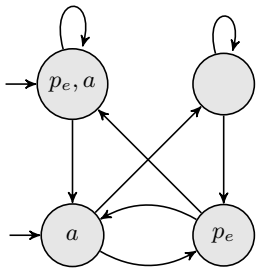

Fig. 4. Representations of the system strategy for any environment strategy in the safety specification $a \wedge \Box((p_e \rightarrow \bigcirc a \wedge (\neg p_e \rightarrow \bigcirc \neg a))$.

According to the semantics of the $\Box$ operator, any formula $\Box\psi$ is logically equivalent to $\psi \wedge \bigcirc\Box\psi$. Hence, the following proposition holds.

Proposition 1. *A safety specification $\alpha \wedge \Box\psi$ is realizable if, and only if, $(\alpha \wedge \psi) \wedge \bigcirc\Box\psi$ is realizable.*

Terse Normal Form. In [7], we introduced a tableau method capable of deciding the realizability of safety specifications. The tableaux, when receiving a safety specification $\alpha \wedge \Box\psi$ as input, have to find a DNF formula logically equivalent to the formula $(\alpha \wedge \psi)$. Next, our tableaux have to transform the DNF into a formula in *terse normal form* (TNF), which was defined to capture the conditions that any trace must satisfy in the (strict) future to be consistent with $\alpha \wedge \Box\psi$. From the TNF, the tableaux can explore all possible system strategies from a set of (what we call) *minimal coverings*. This section reviews all these concepts, which were already introduced in [7].

Basic (sub)formulas in the DNF of a safety formula are of the form ℓ and $\bigcirc^i\ell$ for $i \geq 1$. We classify them into *from-next* formulas: $\bigcirc\ell$ and *from-now* formulas: ℓ.

Definition 2 (Strict-future and moves). *[7] A strict-future formula is a* DNF *combination of from-next formulas. A move is the conjunction of a non-empty set of Boolean literals and (at most) one strict-future formula. If π is a move, then $\mathcal{L}(\pi)$ denotes the set of literals in π and $\mathcal{F}(\pi)$ denotes the strict-future formula in π.*

We can force the sets of Boolean literals to be non-empty w.l.o.g. because the empty set is logically equivalent to the set $\{\mathbf{T}\}$. Similarly, we consider that all moves have a strict-future formula. In the case where there is none, the strict-future formula is $\bigcirc\mathbf{T}$.

Definition 3 (TNF). *[7] A safety formula γ in Terse Normal Form* (TNF) *is a disjunction $\bigvee_{i=1}^{n} \pi_i$ of moves, and for all $1 \leq i \neq j \leq n$ there is at least one literal ℓ such that $\ell \in \mathcal{L}(\pi_i)$ and $\neg\ell \in \mathcal{L}(\pi_j)$.*

Proposition 2. *[7] For any safety formula γ there is a logically equivalent formula that is in* TNF.

Proof. Any safety formula γ can be converted into a disjunctive normal form-like formula, that we call $\mathsf{DNF}(\gamma)$. For constructing $\mathsf{DNF}(\gamma)$, it suffices to use classical logical equivalences on Boolean connectives and equivalences on temporal connectives.

Then, we transform each pair of moves in $\mathsf{DNF}(\gamma)$ with indexes $1 \leq i \neq j \leq n$ such that for all literal $\ell \in \mathcal{L}(\pi_i)$ it holds that $\neg\ell \notin \mathcal{L}(\pi_j)$ as follows. Let $\delta = \mathcal{L}(\pi_i) \cap \mathcal{L}(\pi_j)$, $\delta_1 = \mathcal{L}(\pi_i) \setminus \delta$ and $\delta_2 = \mathcal{L}(\pi_j) \setminus \delta$. Then, we apply

$$\begin{aligned}(\delta \wedge \delta_1 \wedge \eta_1) \vee (\delta \wedge \delta_2 \wedge \eta_2) \equiv\ & (\delta \wedge \delta_1 \wedge \delta_2 \wedge (\eta_1 \vee \eta_2)) \\ & \vee\ \mathsf{DNF}(\delta \wedge \delta_1 \wedge \neg\delta_2 \wedge \eta_1) \\ & \vee\ \mathsf{DNF}(\delta \wedge \neg\delta_1 \wedge \delta_2 \wedge \eta_2)\end{aligned} \tag{1}$$

where $\mathcal{F}(\pi_1) = \eta_1$ and $\mathcal{F}(\pi_j) = \eta_2$.

This transformation is repeatedly applied until every pair (π_i, π_j) satisfies the required condition. It is easy to see that the above process produces a formula in TNF for γ. Moreover, since we only apply logical equivalences to subformulas, by substitutivity, the resulting formula is logically equivalent to γ. □

This proof appeared in [7], but we reproduce it here because it provides a method for converting any LTL formula into TNF (see Example 3). The equivalence in Equation (1) follows from the next equivalence, which is easy to check.

$$(\delta \wedge \delta_1 \wedge \eta_1) \vee (\delta \wedge \delta_2 \wedge \eta_2) \equiv$$
$$(\delta \wedge \delta_1 \wedge \delta_2 \wedge (\eta_1 \vee \eta_2)) \vee (\delta \wedge \neg\delta_1 \wedge \delta_2 \wedge \eta_2) \vee (\delta \wedge \delta_1 \wedge \neg\delta_2 \wedge \eta_1)$$

Example 3. Let $\Box\psi$ be a safety specification, where $\mathcal{E} = \{p_e\}$, and

$$\mathsf{DNF}(\psi) = (p_e \wedge a \wedge \bigcirc b) \vee (p_e \wedge a \wedge \bigcirc^2 c) \vee (\bigcirc \neg c)$$

The process of building a TNF for ψ could be as follows.

$$\begin{aligned}
\mathsf{DNF}(\psi) &= \underline{(p_e \wedge a \wedge \bigcirc b)} \vee \underline{(p_e \wedge a \wedge \bigcirc^2 c)} \vee (\bigcirc \neg c) \\
&= (p_e \wedge a \wedge (\bigcirc b \vee \bigcirc^2 c)) \vee \mathsf{DNF}(p_e \wedge a \wedge \mathbf{F} \wedge \bigcirc b) \vee \\
&\quad \vee \mathsf{DNF}(p_e \wedge a \wedge \mathbf{F} \wedge \bigcirc^2 c) \vee (\bigcirc \neg c) \\
&= \underline{(p_e \wedge a \wedge (\bigcirc b \vee \bigcirc^2 c))} \vee \underline{(\bigcirc \neg c)} \\
&= (p_e \wedge a \wedge \mathbf{T} \wedge (\bigcirc b \vee \bigcirc^2 c \vee \bigcirc \neg c)) \vee \mathsf{DNF}(p_e \wedge a \wedge \mathbf{F} \wedge (\bigcirc b \vee \bigcirc^2 c)) \vee \\
&\quad \vee \mathsf{DNF}(\mathbf{T} \wedge \neg(p_e \wedge a) \wedge \bigcirc \neg c) \\
&= (p_e \wedge a \wedge (\bigcirc b \vee \bigcirc^2 c \vee \bigcirc \neg c)) \vee \underline{(\neg p_e \wedge \bigcirc \neg c)} \vee \underline{(\neg a \wedge \bigcirc \neg c)} \\
&= (p_e \wedge a \wedge (\bigcirc b \vee \bigcirc^2 c \vee \bigcirc \neg c)) \vee (\neg p_e \wedge \neg a \wedge \bigcirc \neg c) \vee \\
&\quad \vee (\neg p_e \wedge a \wedge \bigcirc \neg c), \vee (p_e \wedge \neg a \wedge \bigcirc \neg c) \\
&= \mathsf{TNF}(\psi)
\end{aligned}$$

Remark 1. The TNF computed from an initial DNF is not unique and the number of its moves could be exponential in the number of literals of the DNF.

In the safety specification $\Box\psi$ of the introduction, $\mathsf{TNF}(\bigcirc p_e \leftrightarrow \bigcirc s)$ has only one move: $((\bigcirc p_e \wedge \bigcirc s) \vee (\bigcirc \neg p_e \wedge \bigcirc \neg s))$, with the empty set of literals and with one future-strict formula, which is a disjunction.

We define now a special subset of moves in a TNF that are called coverings.

Definition 4. *[7] A formula $\bigvee_{i=1}^{n} \pi_i$ in TNF with $\cup_{i=1}^{n} \mathsf{Val}_{\pi_i}(\mathcal{E}) = \mathsf{Val}(\mathcal{E})$ is called a covering. A covering is minimal if $\bigvee_{i=1, i\neq j}^{n} \pi_i$ is not a covering for any $1 \leq j \leq n$.*

Intuitively, a minimal covering represents a system strategy from the current state, since it explains what the system does for any possible environment play.

Therefore, the collection of all minimal coverings represents all possible strategies. Moreover, each move in a strategy contains all the strict-future possibilities for this move.

Example 4. Let $\mathsf{TNF}(\psi) = (p_e \wedge c \wedge \eta_1) \vee (\neg p_e \wedge c \wedge \eta_2) \vee (\neg c \wedge \eta_3)$, where η_1, η_2, η_3 are strict-future formulas and $\mathcal{E} = \{p_e\}$. It is a non-minimal covering, but the third move $(\neg c \wedge \eta_3)$ is a minimal one. The two first moves together also provide a minimal covering.

Given a safety specification $(\alpha \wedge \Box\psi)$, Our tableau-method computes a *terse normal form* for $(\alpha \wedge \psi)$, $\mathsf{TNF}(\alpha \wedge \psi)$, from which the set of minimal coverings, $\{C_1, \cdots, C_n\}$, is extracted. We proved the next statement.

Proposition 3. *[7] Let $\alpha \wedge \Box\psi$ be a safety specification. Let $\mathsf{TNF}(\alpha \wedge \psi)$ be a terse normal form for $(\alpha \wedge \psi)$. Let $\{C_1, \cdots, C_n\}$ be the set of all minimal coverings included in $\mathsf{TNF}(\alpha \wedge \psi)$. The specification $\alpha \wedge \Box\psi$ is realizable, if, and only if, there exists $1 \leq i \leq n$ such that for all move π in C_i it holds that $(\pi \wedge \bigcirc\Box\psi)$ is realizable.*

We abuse notation and simply say that $(C_i \wedge \bigcirc\Box\psi)$ is realizable when for all move π in C_i, $(\pi \wedge \bigcirc\Box\psi)$ is realizable. Note also that $\{C_1, \cdots, C_n\}$ must contain at least one minimal covering, i.e, $n > 0$. Otherwise, $\alpha \wedge \Box\psi$ is not realizable.

SAT-based TNF computation. As we have previously explained, a tableau for a safety specification, $\alpha \wedge \Box\psi$, starts obtaining a DNF for $(\alpha \wedge \psi)$. The first step in doing so is to interpret each *from-next* subformula in ψ as a Boolean literal and to build a DNF as short as possible.

There are automatic tools that find DNFs for propositional formulas reasonably well, but this task is not easy: the problem of deciding whether a propositional formula has a DNF of size $\leq k$ or not is EXPTIME complete (when k is binary) and Σ_2^P-complete (when k is unary) [6].

We are using a tool called BICA [8], which using a SAT-solver, given a propositional formula in an arbitrary form, is able to compute a smallest size set of prime implicants equivalent to the formula. Let us remember what a prime implicant is.

Definition 5 (Implicants and prime implicants). *[10] Any model μ of φ, seeing μ as a set of literals, is an implicant of φ. If no subset of μ is an implicant of φ, then μ is said to be a prime implicant.*

Example 5. The propositional formula $p \rightarrow q$ has two prime implicants: $\{q\}$ and $\{\neg p\}$. Furthermore, $p \rightarrow q$ is logically equivalent to $q \vee \neg p$.

Proposition 4. *Let φ be a propositional formula. The disjunction of all prime implicants of φ is a DNF logically equivalent to φ.*

In the worst case, the number of prime implicants of a propositional formula is exponential with respect to the number of variables of the formula.

Example 6. Let φ be the following propositional formula [1]:

$$(x_1 \wedge x_2 \wedge \cdots \wedge x_n \wedge y_0) \vee (\neg x_1 \wedge x_2 \wedge \cdots \wedge x_n \wedge y_1)$$
$$\vee\ (\neg x_2 \wedge \cdots \wedge x_n \wedge y_2) \vee \cdots \vee (\neg x_n \wedge y_n)$$

This formula has at least 2^n prime implicants corresponding to:

$(b_1 \wedge b_2 \wedge \cdots \wedge b_n \wedge y_0)$ where b_i can be either x_i or y_i.

In addition to the $n+1$ prime implicants:

$$(x_1 \wedge x_2 \wedge \cdots \wedge x_n \wedge y_0) \vee (\neg x_1 \wedge x_2 \wedge \cdots \wedge x_n \wedge y_1)$$
$$\vee\ (\neg x_2 \wedge \cdots \wedge x_n \wedge y_2) \vee \cdots \vee (\neg x_n \wedge y_n)$$

From DNFs to TNFs and minimal coverings. It is worth noting that the DNF for $(\alpha \wedge \psi)$ fulfills the following properties.

Remark 2. Each prime implicant in the DNF for $(\alpha \wedge \psi)$ is a move. If the DNF for $(\alpha \wedge \psi)$ has no models for an environment play, then we immediately know that $(\alpha \wedge \Box\psi)$ is unrealizable, i.e., the realizability of $(\alpha \wedge \Box\psi)$ requires that $\mathsf{Val}_{\mathsf{DNF}(\alpha\wedge\psi)}(\mathcal{E}) = \mathsf{Val}(\mathcal{E})$.

Once we have a DNF for $(\alpha \wedge \psi)$ such that $\mathsf{Val}_{\mathsf{DNF}(\alpha\wedge\psi)}(\mathcal{E}) = \mathsf{Val}(\mathcal{E})$, the next step is to construct a TNF for $(\alpha \wedge \psi)$ (following the steps explained in Proposition 2) and extract the set of minimal coverings. As this process is very inefficient, in the following section we present a correct and more efficient alternative.

3 Implementation of weaker minimal coverings

Given a safety specification $\alpha \wedge \Box\psi$, suppose that $\mathsf{Val}_{\mathsf{DNF}(\alpha\wedge\psi)}(\mathcal{E}) = \mathsf{Val}(\mathcal{E})$ and we have constructed a TNF for $(\alpha \wedge \psi)$.

Weaker minimal coverings. Let us start by introducing a preorder among sets of minimal coverings in the TNF.

Definition 6 (weaker moves). *Let π_1, π_2 be two moves fulfilling that $\mathsf{Val}_{\pi_2}(\mathcal{E}) \cap \mathsf{Val}_{\pi_1}(\mathcal{E}) \neq \emptyset$. Let η_1 and η_2 be their respective strict-future formulas. If η_2 is a logical consequence of η_1, we say that π_2 is weaker than π_1.*

The first requirement guarantees that both π_1 and π_2 have a choice at the current state for any environment play in $\mathsf{Val}_{\pi_2}(\mathcal{E}) \cap \mathsf{Val}_{\pi_1}(\mathcal{E})$. The second requirement guarantees that if η_1 is a winning strategy for the system, then η_2 is as well and, of course, the other way around: if there is no a winning strategy for η_2, there is no winning strategy for η_1 either.

Definition 7 (weaker minimal coverings). *Let $\widehat{C} = (\widehat{\pi}_1 \vee \cdots \vee \widehat{\pi}_m)$ and $C = (\pi_1 \vee \cdots \vee \pi_n)$[4] be two minimal coverings in the TNF. We say that $\widehat{C}$ is weaker than C, denoted as $\widehat{C} \leq C$, if for all $1 \leq i \leq m$ exists $1 \leq j \leq n$ such that $\widehat{\pi}_i$ is weaker than π_j.*

[4] Whenever convenient, we identify the minimal coverings with set of moves.

Example 7. Let $\mathcal{E} = \{p_e\}$ and $\mathcal{S} = \{a, b\}$. Let η_1 and η_2 two from-next formulas and the safety specification $\Box\psi = C_1 \vee C_2$ where C_1 and C_2 are as follows.

$$C_1 = (\neg a \wedge b \wedge \eta_1) \qquad C_2 = (p_e \wedge \neg a \wedge \neg b) \vee (\neg p_e \wedge a \wedge (\eta_1 \vee \eta_2))$$

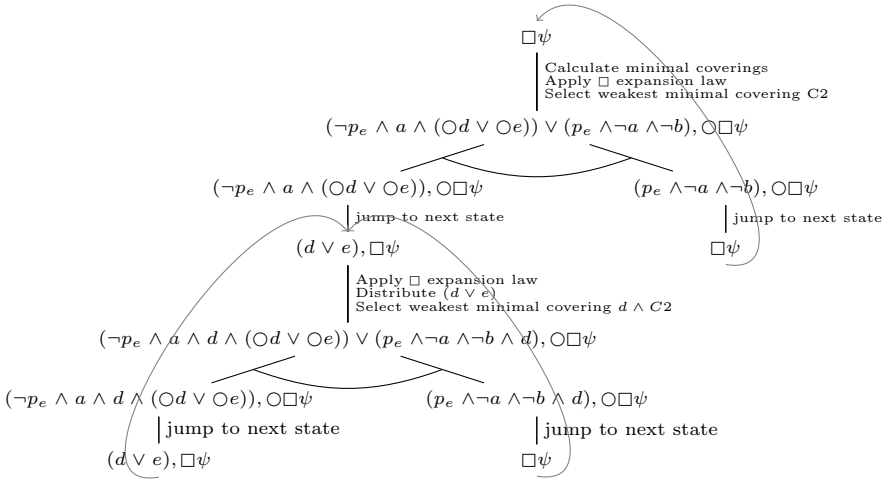

Fig. 5. Open tableau for $\Box\psi$.

C_1 contains a single move $\pi_1 = (\neg a \wedge b \wedge \eta_1)$ and C_2 contains two moves $\pi_{2,1} = (p_e \wedge \neg a \wedge \neg b)$ and $\pi_{2,2} = (\neg p_e \wedge a \wedge (\eta_1 \vee \eta_2))$. Obviously, $\mathsf{TNF}(\psi) = \psi$ and its set of minimal coverings is $\{C_1, C_2, C_3, C_4\}$ where C_3 and C_4 are the following minimal coverings.

$$C_3 = (\neg p_e \wedge \neg a \wedge b \wedge \eta_1) \vee (p_e \wedge \neg a \wedge \neg b)$$
$$C_4 = (p_e \wedge \neg a \wedge b \wedge \eta_1) \vee (\neg p_e \wedge a \wedge (\eta_1 \vee \eta_2))$$

It holds that $C_2 \leq C_1$ because $\mathsf{Val}_{\pi_{2,1}}(\mathcal{E}) \cap \mathsf{Val}_{\pi_1}(\mathcal{E}) \neq \emptyset$ and $\mathbf{T}$ is a logical consequence of η_1, as well as, $\mathsf{Val}_{\pi_{2,2}}(\mathcal{E}) \cap \mathsf{Val}_{\pi_1(\mathcal{E})} \neq \emptyset$ and $(\eta_1 \vee \eta_2)$ is a logical consequence of η_1. Moreover, it is easy to see that $C_2 \leq C_3$ and $C_2 \leq C_4$.

According to Proposition 3, our tableau has to choose one of the minimal covering to start with. The best action is to start with C_2. Suppose the tableau is closed because the branch corresponding to $(\neg p_e \wedge a \wedge (\eta_1 \vee \eta_2)) \wedge \bigcirc\Box\psi$ is not realizable. Since the system can choose to assign *true* to the literal a in the current state, the non-realizability of this move is due to a future issue. Then, when the environment chooses $\neg p_e$, we are sure that none of the other minimal coverings C_1, C_3 and C_4 has a winning strategy for the system. The system would choose $(\neg a \wedge b \wedge \eta_1)$ in C_1 and C_3 and would choose $(a \wedge (\eta_1 \vee \eta_2))$ in C_4. All these strategies lose. Therefore, by Proposition 3, $\Box\psi$ is not realizable. Consequently, we do not need to check the minimal coverings C_1, C_3, and C_4. Furthermore, by Proposition 3 if the tableau for $C_2 \wedge \bigcirc\Box\psi$ is open, we immediately know that $\Box\psi$ is realizable. Figure 5 briefly shows how to construct an open tableau for $\Box\psi$ with $\eta_1 = \bigcirc d$ and $\eta_2 = \bigcirc e$ using the weaker minimal covering C2.

We formalize previous ideas in the next propositions.

Proposition 5. *Let* $\widehat{C} = \{\widehat{\pi}_1, \cdots \widehat{\pi}_m\}$ *and* $C = \{\pi_1, \cdots \pi_n\}$ *be two minimal coverings in the* TNF*. Let* $\widehat{C} \leq C$*. If* C *is realizable, then* $\widehat{C}$ *is realizable.*

Proof. Since $\widehat{C}$ and C are minimal coverings, by Remark 2, we know that $\mathsf{Val}_{\pi_j}(\mathcal{E}) \neq \emptyset$ for all $1 \leq j \leq n$ and $\mathsf{Val}_{\widehat{\pi}_i}(\mathcal{E}) \neq \emptyset$ for all $1 \leq i \leq m$. Suppose C is realizable. That means,

$$\text{for all } 1 \leq j \leq n : \mathcal{F}(\pi_j) \text{ is realizable.} \tag{2}$$

Since $\widehat{C} \leq C$, according to Definition 7, for any $1 \leq i \leq m$, there must exists $1 \leq j \leq n$ such that $\widehat{\pi}_i$ is weaker than π_j, which means that $\mathcal{F}(\widehat{\pi}_i)$ is a logical consequence of $\mathcal{F}(\pi_j)$. Equation (2) ensures that

$$\text{for all } 1 \leq i \leq m : \mathcal{F}(\widehat{\pi}_i) \text{ is realizable.}$$

In addition, for all $1 \leq i \leq m : \pi_i$ has a system strategy for the current state. Hence, $\widehat{C}$ is realizable. □

By the previous proposition, it is enough to consider the *weaker* minimal coverings included in the TNFs. That is, those covers $\widehat{C} \subseteq$ TNF for which there is not any other $C \subseteq$ TNF such that $\widehat{C} \leq C$.

Proposition 6. *Let* $\alpha \wedge \Box\psi$ *be a safety specification. Let* $\widehat{C}_1, \cdots, \widehat{C}_m$ *be all the weaker minimal coverings included in a* TNF *for* $(\alpha \wedge \psi)$*. The specification* $\alpha \wedge \Box\psi$ *is realizable, if, and only if, there exists* $1 \leq i \leq m$ *such that* $(\widehat{C}_i \wedge \bigcirc\Box\psi)$ *is realizable.*

Proof. The backward direction is straightforward by Proposition 3. If there exists a weaker minimal covering such that $(\widehat{C}_i \wedge \bigcirc\Box\psi)$ is realizable, in particular, exists a minimal covering.

The forward direction. Suppose that $\alpha \wedge \Box\psi$ is realizable. By Proposition 3, there exists $C = \{\pi_1, \cdots \pi_n\}$ a minimal covering included in a TNF$(\alpha \wedge \psi)$, such that $(C \wedge \bigcirc\Box\psi)$ is realizable. If C is a weaker minimal covering we are done. Otherwise, there exists $1 \leq i \leq m$ such that $\widehat{C}_i < C$. Now, by Proposition 5, $\widehat{C}_i$ is realizable. □

Let us see another example of TNF. In this example, we use sets of moves instead of disjunctions of moves.

Example 8. Let $\Box\psi = \Box((p_e \wedge ((\bigcirc b \wedge (a_1 \vee a_2)) \vee (\neg a_1 \wedge \bigcirc^2 c))) \vee (\neg p_e \wedge \bigcirc\neg b))$ be a safety specification where $\mathcal{E} = \{p_e\}$. Bica returns the next DNF for ψ: $(p_e \wedge a_1 \wedge \bigcirc b) \vee (p_e \wedge a_2 \wedge \bigcirc b) \vee (\neg a_1 \wedge \bigcirc^2 c) \vee (\neg p_e \wedge \bigcirc\neg b)$. The computation of a TNF for ψ according to Proposition 2, could start with I_1.

$I_1 = \{\underline{(p_e \wedge a_1 \wedge \bigcirc b)}, \underline{(p_e \wedge a_2 \wedge \bigcirc b)}, (\neg a_1 \wedge \bigcirc^2 c), (\neg p_e \wedge \bigcirc\neg b)\}$

$I_2 = \{(p_e \wedge a_1 \wedge a_2 \wedge \bigcirc b), \underline{(p_e \wedge \neg a_1 \wedge a_2 \wedge \bigcirc b)},$

$$(p_e \wedge a_1 \wedge \neg a_2 \wedge \bigcirc b), \underline{(\neg a_1 \wedge \bigcirc^2 c)}, (\neg p_e \wedge \bigcirc \neg b)\}$$

$$\begin{aligned} I_3 = \{&(p_e \wedge a_1 \wedge a_2 \wedge \bigcirc b), (p_e \wedge \neg a_1 \wedge a_2 \wedge (\bigcirc b \vee \bigcirc^2 c))), \\ &\underline{(\neg p_e \wedge \neg a_1 \wedge \bigcirc^2 c)}, (\neg a_1 \wedge \neg a_2 \wedge \bigcirc^2 c), \\ &(p_e \wedge a_1 \wedge \neg a_2 \wedge \bigcirc b), \underline{(\neg p_e \wedge \bigcirc \neg b)}\} \end{aligned}$$

$$\begin{aligned} I_4 = \{&(p_e \wedge a_1 \wedge a_2 \wedge \bigcirc b), (p_e \wedge \neg a_1 \wedge a_2 \wedge (\bigcirc b \vee \bigcirc^2 c)), \\ &\underline{(\neg p_e \wedge \neg a_1 \wedge (\bigcirc^2 c \vee \bigcirc \neg b))}, \underline{(\neg a_1 \wedge \neg a_2 \wedge \bigcirc^2 c)}, \\ &(p_e \wedge a_1 \wedge \neg a_2 \wedge \bigcirc b), (\neg p_e \wedge a_1 \wedge \bigcirc \neg b)\} \end{aligned}$$

$$\begin{aligned} I_5 = \{&(p_e \wedge a_1 \wedge a_2 \wedge \bigcirc b), (p_e \wedge \neg a_1 \wedge a_2 \wedge (\bigcirc b \vee \bigcirc^2 c)), \\ &(\neg p_e \wedge \neg a_1 \wedge \neg a_2 \wedge (\bigcirc^2 c \vee \bigcirc \neg b)), (p_e \wedge \neg a_1 \wedge \neg a_2 \wedge \bigcirc^2 c), \\ &(\neg p_e \wedge \neg a_1 \wedge a_2 \wedge (\bigcirc^2 c \vee \bigcirc \neg b)), \\ &(p_e \wedge a_1 \wedge \neg a_2 \wedge \bigcirc b), (\neg p_e \wedge a_1 \wedge \bigcirc \neg b)\} \end{aligned}$$

$$\mathsf{TNF}(\psi) = \bigvee_{\pi \in I_5} \pi$$

Now we have to extract the set of all minimal coverings. According to Proposition 6, any tableau that decides the realizability of $\Box\psi$ should choose only the weaker ones. In this case,

$$\{(p_e \wedge \neg a_1 \wedge a_2 \wedge (\bigcirc b \vee \bigcirc^2 c)), (\neg p_e \wedge \neg a_1 \wedge \neg a_2 \wedge (\bigcirc^2 c \vee \bigcirc \neg b))\}$$

Consequently, moves $(p_e \wedge a_1 \wedge a_2 \wedge \bigcirc b), (p_e \wedge \neg a_1 \wedge \neg a_2 \wedge \bigcirc^2 c), (\neg p_e \wedge \neg a_1 \wedge a_2 \wedge (\bigcirc^2 c \vee \bigcirc \neg b)), (p_e \wedge a_1 \wedge \neg a_2 \wedge \bigcirc b), (\neg p_e \wedge a_1 \wedge \bigcirc \neg b)$ are irrelevant for the tableau.

The previous example clearly shows that we do not need all the information the TNF contains to identify the weaker minimal coverings. It suggests that a clever process could find such coverings without constructing the full TNF.

Weaker minimal coverings without constructing TNFs. We present an algorithm that given a safety specification $\alpha \wedge \Box\psi$, receives as input the $\mathsf{DNF}(\alpha \wedge \psi)$ such that $\mathsf{Val}_{\mathsf{DNF}(\alpha\wedge\psi)}(\mathcal{E}) = \mathsf{Val}(\mathcal{E})$, and returns the set of all weaker minimal coverings for $(\alpha \wedge \psi)$ without constructing the complete TNF. First, a definition to formalize the notion of being compatible.

Definition 8 (join operator and compatible sets). *Let $\Pi = \{\pi_1, \cdots, \pi_n\}$ a set of moves. The set Π is compatible if for all $1 \leq i \neq j \leq n$ and for all literal $\ell \in \mathcal{L}(\pi_i)$, it holds that $\neg\ell \notin \mathcal{L}(\pi_j)$.*
The join of moves in a compatible set Π, denoted as $join(\Pi)$, is a new move with $\mathcal{L}(join(\Pi)) = \mathcal{L}(\pi_1) \cup \cdots \cup \mathcal{L}(\pi_n)$ and $\mathcal{F}(join(\Pi)) = \mathcal{F}(\pi_1) \vee \mathcal{F}(\pi_2) \cdots \vee \mathcal{F}(\pi_n)$.

Proposition 7. *Let $\Pi = \{\pi_1, \cdots, \pi_n\}$ a compatible set of moves. It holds that $\{join(\Pi)\} \leq \Pi$.*

Proof. Since Π is compatible, for all $1 \leq i \leq n$, $\mathsf{Val}_{\pi_i}(\mathcal{E}) \cap \mathsf{Val}_{join(\Pi)}(\mathcal{E}) \neq \emptyset$. Moreover, $\mathcal{F}(\pi_1) \vee \mathcal{F}(\pi_2) \vee \cdots \vee \mathcal{F}(\pi_n)$ is a logical consequence of each strict-future formula of π_i. Hence, by Definition 7, $\{join(\Pi)\} \leq \Pi$. □

We now define the notion of extending a move with an environment literal.

Definition 9 (extension of moves with $\mathcal{E}$-literals). *Let ℓ_e be a literal of $\mathcal{E}$ and let π be a move. The extension of π with ℓ_e, $ext_{\mathcal{E}}(\pi, \ell_e)$, is a new move with $\mathcal{F}(ext_{\mathcal{E}}(\pi, \ell_e)) = \mathcal{F}(\pi)$ and $\mathcal{L}(ext_{\mathcal{E}}(\pi, \ell_e)) = \mathcal{L}(\pi) \cup \{\ell_e\}$.*

Note that when $\neg\ell_e \in \pi$, then $ext_{\mathcal{E}}(\pi, \ell_e) = \mathbf{F}$. We can extend the previous definition in two ways.

Definition 10 (extension of moves with sets of $\mathcal{E}$). *Let $\mathcal{R}$ be a set of $\mathcal{E}$ and let π be a move. The extension of π with $\mathcal{R}$, $set\_ext_{\mathcal{E}}(\pi, \mathcal{R})$ is the successive extensions of π with the variables of $\mathcal{R}$ and the negation of variables in $\mathcal{E} \setminus \mathcal{R}$.*

When M is a set of moves $M = \{\pi_1, \cdots, \pi_n\}$ and $\mathcal{R}$ is a set of $\mathcal{E}$, we define the extension of M with $\mathcal{R}$, denoted as $set\_ext_{\mathcal{E}}(M, \mathcal{R})$, is the set $\{set\_ext_{\mathcal{E}}(\pi_1, \mathcal{R}), \cdots, set\_ext(\pi_n, \mathcal{R})\}$.

Example 9. Let $\mathcal{E} = \{p_e, q_e, r_e\}$, $\mathcal{S} = \{a\}$. If $\mathcal{R} = \{p_e, q_e\}$ and $M = \{(p_e \wedge a \wedge \eta_1), (r_e \wedge \neg a \wedge \eta_2), (a \wedge \eta_4)\}$, the set of moves $set\_ext_{\mathcal{E}}(M, \mathcal{R}) = \{(p_e \wedge q_e \wedge \neg r_e \wedge a \wedge \eta_1), (p_e \wedge q_e \wedge \neg r_e \wedge a \wedge \eta_4)\}$.

When $\mathcal{R} = \emptyset$ and M is as before, the set of moves $set\_ext_{\mathcal{E}}(M, \emptyset) = \{(\neg p_e \wedge \neg q_e \wedge \neg r_e \wedge a \wedge \eta_4)\}$.

When $M = \emptyset$ and $\mathcal{R} = \{p_e, q_e\}$, the set of moves $set\_ext_{\mathcal{E}}(\emptyset, \mathcal{R}) = \{(p_e \wedge q_e \wedge \neg r_e)\}$.

The extension of moves with sets of environment variables allows us to group moves according to the environment plays. For instance, if $\mathcal{E} = \{p_e, q_e, r_e\}$ and $M = \{(q_e \wedge \eta_1), (q_e \wedge \neg p_e \wedge \eta_2)\}$, the extension of M with all possible environment plays is given by the extension with all possible subsets of $\{p_e, q_e\}$.

$$
\begin{aligned}
set\_ext_{\mathcal{E}}(M, \emptyset) &= \emptyset \\
set\_ext_{\mathcal{E}}(M, \{q_e\}) &= \{(q_e \wedge \neg p_e \wedge \eta_1), (q_e \wedge \neg p_e \wedge \eta_2)\} \\
set\_ext_{\mathcal{E}}(M, \{p_e\}) &= set\_ext_{\mathcal{E}}(M, \{p_e, q_e\}) = \{(q_e \wedge p_e \wedge \eta_1)\}
\end{aligned}
$$

As the next proposition states, it always happens that $\bigvee_{\pi \in M} \pi \equiv \bigvee_{\substack{\mathcal{R} \in 2^{\mathcal{E}} \\ \pi \in set\_ext_{\mathcal{E}}(M, \mathcal{R})}} \pi$.

In our example,

$$(q_e \wedge \eta_1) \bigvee (q_e \wedge \neg p_e \wedge \eta_2) \equiv (q_e \wedge \neg p_e \wedge \eta_1) \bigvee (q_e \wedge \neg p_e \wedge \eta_2) \bigvee (q_e \wedge p_e \wedge \eta_1)$$

Proposition 8. *Let $2^{\mathcal{E}}$ be the set of all subsets of $\mathcal{E}$. Let M be a set of moves. Define $\mathcal{E}'$ as $\mathcal{E} \cap var(M)$. It holds that $\bigvee_{\pi \in M} \pi \equiv \bigvee_{\substack{\mathcal{R} \in 2^{\mathcal{E}'} \\ \pi \in set\_ext_{\mathcal{E}}(M, \mathcal{R})}} \pi$*

Proof. The proof is based on two facts that are easy to see. The first one states that the formula $\bigvee_{\substack{\mathcal{R} \in 2^{\mathcal{E}'} \\ \pi \in set\_ext_{\mathcal{E}}(\emptyset, \mathcal{R})}} \pi$ is a tautology.

The second one is the fact that $\bigvee_{\substack{\mathcal{R} \in 2^{\mathcal{E}'} \\ \pi \in set\_ext_{\mathcal{E}}(M, \mathcal{R})}} \pi \equiv (\bigvee_{\pi \in M} \pi \wedge \bigvee_{\substack{\mathcal{R} \in 2^{\mathcal{E}'} \\ \pi \in set\_ext_{\mathcal{E}}(\emptyset, \mathcal{R})}} \pi)$. □

The algorithm. We explain the algorithm (see Algorithm 1) step by step using a running example. Let $\mathcal{E} = \{p_e, q_e\}$ and $\mathcal{S} = \{a, b\}$. Let $\Box\psi = \Box((a \wedge \bigcirc b) \vee (p_e \wedge \bigcirc\neg a))$.

Algorithm 1: weaker_moves($\mathsf{DNF}(\gamma)$) returns $\mathcal{T}$

```
1  % The formula γ is over variables E ∪ S and Val_{DNF(γ)}(E) = Val(E)
2  T := ∅;
3  M := {π : π is a move in DNF(γ)};
4  E' := E ∩ var(M);
5  for any set R ∈ 2^{E'} do
6  |  Calculate the largest compatible sets
7  |     Π_1^R, ··· Π_n^R in ext_{E'}(M, R);
8  |  J := {join(Π_1^R), ··· , join(Π_n^R)};
9  |  for 1 ≤ i ≠ j ≤ n do
10 |  |  if join(Π_i^R) is weaker than join(Π_j^R) then
   |  |     J := J \ {join(Π_j^R)};
11 |  end
12 |  T := T ∪ J;
13 end
14 return T;
```

Algorithm 1 returns a set $\mathcal{T}$ of the weaker moves included in a TNF for ψ. The input of the algorithm is $\mathsf{DNF}(\psi) = \psi$, which fulfills that $\mathsf{Val}_\psi(\mathcal{E}) = \mathsf{Val}(\mathcal{E})$. Initially, $\mathcal{T}$ is the empty set (line 2 of the algorithm) and M (line 3) contains all the moves of the input. In our example, $M = \{(a \wedge \bigcirc b), (p_e \wedge \bigcirc\neg a)\}$. Obviously. $\psi \equiv \bigvee_{\pi \in M} \pi$. In line 4, an iteration starts over all possible sets in $2^{\mathcal{E}'}$, where $\mathcal{E}' = \mathcal{E} \cap var(M)$, i.e, all possible environment play are examined. In our case, $\mathcal{E}' = \{p_e\}$ and there are two subsets in $2^{\mathcal{E}'}$: $\emptyset$ and $\{p_e\}$. For each of these subsets (line 5), the Algorithm 1 calculates the extension of M and the largest compatible sets in such extension (lines 6–7).

Following the example, the first iteration obtains $ext_{\mathcal{E}'}(M, \emptyset) = \{(\neg p_e \wedge a \wedge \bigcirc b)\}$ and the single compatible set $\Pi_1^\emptyset = \{(\neg p_e \wedge a \wedge \bigcirc b)\}$. Likewise, the second iteration calculates $ext_{\mathcal{E}'}(M, \{p_e\}) = \{(p_e \wedge a \wedge \bigcirc b), (p_e \wedge \bigcirc\neg a)\}$ and $\Pi_1^{\{p_e\}} = \{(p_e \wedge a \wedge \bigcirc b), (p_e \wedge \bigcirc\neg a)\}$.

Each iteration adds to $\mathcal{T}$ the set $\mathcal{J}$ formed by the *join* of each compatible set (line 8). The second iteration (lines 9–11) is executed to ensure that only the weaker *join*s remain in $\mathcal{T}$. In the example, the first iteration adds $join(\Pi_1^\emptyset) = \Pi_1^\emptyset$ to $\mathcal{T}$ while the second iteration adds $join(\Pi_1^{\{p_e\}}) = \{(p_e \wedge a \wedge (\bigcirc b \vee \bigcirc\neg a)\}$ to $\mathcal{T}$ (none of them is removed in the second iteration). Therefore, $\mathcal{T} = \{(\neg p_e \wedge a \wedge \bigcirc b), (p_e \wedge a \wedge (\bigcirc b \vee \bigcirc\neg a)\}$. At the end (line 10), $\mathcal{T}$ is returned.

Algorithm 1 shows the process of building the set of weaker moves without constructing the full TNF.

We review the two examples from the previous sections. We start with Example 3. There, $\mathcal{E} = \{p_e\}$ and $\mathsf{DNF}(\psi) = (p_e \wedge a \wedge \bigcirc b) \vee (p_e \wedge a \wedge \bigcirc^2 c) \vee (\bigcirc \neg c)$. The execution of Algorithm 1 calculates $\Pi_1^{\{p_e\}}$, the biggest compatible set in $ext(\mathsf{DNF}(\psi), \{p_e\})$, which is $\{(p_e \wedge a \wedge \bigcirc b), (p_e \wedge a \wedge \bigcirc^2 c), (p_e \wedge \bigcirc \neg c)\}$. Then, the TNF, $\mathcal{T}$ increases with the *join* of the three moves. $\mathcal{T} = \{(p_e \wedge a \wedge (\bigcirc b \vee \bigcirc^2 c \vee \bigcirc \neg c)\}$. In a second iteration, the biggest compatible set in $ext(\mathsf{DNF}(\psi), \emptyset)$ is calculated. $\Pi_1^{\emptyset} = \{(\neg p_e \wedge \bigcirc c)\}$. At the end of the process, $\mathcal{T} = \{(p_e \wedge a \wedge (\bigcirc b \vee \bigcirc^2 c \vee \bigcirc \neg c), (\neg p_e \wedge \bigcirc c)\}$. Note that $\mathcal{T}$ contains a single (weak) minimal covering.

In Example 8, $\mathcal{E} = \{p_e\}$ and $\mathsf{DNF}(\psi) = (p_e \wedge a_1 \wedge \bigcirc b) \vee (p_e \wedge a_2 \wedge \bigcirc b) \vee (\neg a_1 \wedge \bigcirc^2 c) \vee (\neg p_e \wedge \bigcirc \neg b)$. The execution of Algorithm 1 calculates $\Pi_1^{\{p_e\}} = \{(p_e \wedge a_1 \wedge \bigcirc b), (p_e \wedge a_2 \wedge \bigcirc b)\}$; $\Pi_2^{\{p_e\}} = \{(p_e \wedge a_2 \wedge \bigcirc b), (p_e \wedge \neg a_1 \wedge \bigcirc^2 c)\}$; and $\Pi_1^{\emptyset} = \{(\neg p_e \wedge \neg a_1 \wedge \bigcirc^2 c), (\neg p_e \wedge \bigcirc b)\}$. For p_e, the move $join(\Pi_2^{\{p_e\}})$ is weaker than $join(\Pi_1^{\{p_e\}})$ Hence, $\mathcal{T} = \{(p_e \wedge a_2 \wedge \neg a_1 \wedge (\bigcirc b \vee \bigcirc^2 c), (\neg p_e \wedge \neg a_1 \wedge (\bigcirc^2 c \vee \bigcirc b))\}$. Here $\mathcal{T}$ also contains a single (weak) minimal covering.

3.1 Correction of the Algorithm

Intuitively, the algorithm returns all the weaker minimal coverings that can be extracted from the DNF of an LTL formula because it groups the moves of the DNF according to each environment play. For each of these plays, it gathers the sets of all moves that are compatible (capturing all the options in the strict future that are consistent with the environment play). Finally, by doing the *join* of each compatible set, the algorithm guarantees that returns all the weaker moves.

Proposition 9. *Let γ be a safety formula over variables $\mathcal{E} \cup \mathcal{S}$. Let $\mathcal{T}$ be the output of Algorithm 1 when γ is received as input. There exists a $\mathsf{TNF}(\gamma)$ logically equivalent to $\mathsf{DNF}(\gamma)$ fulfilling the following statements.*

1. *$\mathsf{TNF}(\gamma)$ contains all the moves in $\mathcal{T}$ (seeing them as sets).*
2. *For each minimal covering $C \subseteq \mathsf{TNF}(\gamma)$ there exists a minimal covering $\hat{C} \subseteq \mathcal{T}$ such that $\hat{C} \leq C$.*

Proof. First of all, it should be noted that $\mathsf{DNF}(\gamma) \equiv \bigvee_{\pi \in M} \pi$, which by Proposition 8 it is equivalent to $\bigvee_{\substack{\mathcal{R} \in 2^{\mathcal{E}'} \\ \pi \in set\_ext_{\mathcal{E}}(M,\mathcal{R})}} \pi$, where $\mathcal{E}' = \mathcal{E} \cap var(M)$. Then, we can construct a TNF from the latter formula by following the process explained in Proposition 2. There, Equation (1) can be seen in terms of the *join* operator.

$$\begin{aligned}(\delta \wedge \delta_1 \wedge \eta_1) \vee (\delta \wedge \delta_2 \wedge \eta_2) \equiv\ & join((\delta \wedge \delta_1 \wedge \eta_1), (\delta \wedge \delta_2 \wedge \eta_2)) \\ & \vee\ \mathsf{DNF}(\delta \wedge \delta_1 \wedge \neg \delta_2 \wedge \eta_1) \\ & \vee\ \mathsf{DNF}(\delta \wedge \neg \delta_1 \wedge \delta_2 \wedge \eta_2)\end{aligned}$$

Actually, joining the compatible moves in the same order as Algorithm 1 does results in such TNF. This TNF contains all moves included in $\mathcal{T}$, but also much more moves for each *join* just to preserve equivalence to the initial DNF(γ). Moreover, all extra moves added for each *join* are always *stronger* than the *join* itself (remember Proposition 7). This observation is the key to prove item 2., because the weaker minimal coverings in the TNF must contain those moves built by the successive application of the *join* operator (the weaker ones) and not the others. Therefore, the weaker minimal coverings in the TNF must be those contained in $\mathcal{T}$.

By the previous proposition and Proposition 6, we can ensure the correction of the algorithm.

Theorem 1. *Let $\alpha \wedge \Box\psi$ a safety specification. let $\mathcal{T}$ be the formula returned by Algorithm 1 when $\alpha \wedge \Box\psi$ is received as input. Assuming that the set of minimal coverings contained in $\mathcal{T}$ is the set $\{\widehat{C}_1, \cdots \widehat{C}_m\}$, the specification $\alpha \wedge \Box\psi$ is realizable, if, and only if, there exists $1 \leq i \leq m$ such that $(\widehat{C}_i \wedge \bigcirc\Box\psi)$ is realizable.*

References

1. Aizenstein, H.J.: On the Learnability of Disjunctive Normal Form Formulas and Decision Trees. Ph.D. thesis, USA (1993), uMI Order No. GAX93-28958
2. Bohy, A., Bruyère, V., Filiot, E., Jin, N., Jean-Fran c.R.: Acacia+, a tool for LTL synthesis. In: Proc. of CAV'12. LNCS, vol. 7358, pp. 652–657. Springer (2012)
3. Ehlers, R.: Unbeast: Symbolic bounded synthesis. In: Proc. of TACAS'11. LNCS, vol. 6605, pp. 272–275. Springer (2011)
4. Finkbeiner, B., Schewe, S.: Bounded synthesis. Int. J. Softw. Tools Technol. Transf. **15**(5-6), 519–539 (2013). `https://doi.org/10.1007/s10009-012-0228-z`, `https://doi.org/10.1007/s10009-012-0228-z`
5. Finkbeiner, B., Tentrup, L.: Detecting unrealizable specifications of distributed systems. In: Proc. of TACAS'14. LNCS, vol. 8413, pp. 78–92. Springer (2014)
6. Goldsmith, J., Hagen, M., Mundhenk, M.: Complexity of DNF minimization and isomorphism testing for monotone formulas. Inf. Comput. **206**, 760–775 (2008)
7. Hermo, M., Lucio, P., Sánchez, C.: Tableaux for realizability of safety specifications. In: Formal Methods - 25th International Symposium, FM 2023, Lübeck, Germany, March 6-10, 2023, Proceedings. Lecture Notes in Computer Science, vol. 14000, pp. 495–513. Springer (2023)
8. Ignatiev, A., Previti, A., Marques-Silva, J.: SAT-Based formula Simplification. In: SAT (2015)
9. Jobstmann, B., Galler, S., Weiglhofer, M., Bloem, R.: Anzu: A tool for property synthesis. In: Proc. of CAV'07. vol. 4590, pp. 258–262. Springer (2007)
10. Micheli, G.D.: Synthesis and Optimization of Digital Circuits. McGraw-Hill, Inc. (1994)
11. Pnueli, A.: The temporal logic of programs. In: Proc. of the 18th IEEE Symp. on Foundations of Computer Science (FOCS'77). pp. 46–67. IEEE CS Press (1977)
12. Zhu, S., Tabajara, L.M., Li, J., Pu, G., Vardi, M.Y.: A symbolic approach to Safety LTL synthesis. In: Hardware and Software: Verification and Testing. vol. 10629, pp. 147–162. Springer International Publishing (2017)