Journal of Applied Non-Classical Logics
Vol. 00, No. 00, Month 201X, 1–22

RESEARCH ARTICLE

Temporal Abductive Reasoning about Biochemical Reactions

Serenella Cerrito,[†] Marta Cialdea Mayer,[‡] and Robert Demolombe[§]

[†]*IBISC, Université d'Evry Val d'Essonne and Université Paris-Saclay, France*
[‡]*Università degli Studi Roma Tre, Italy*
[§]*Institut de Recherche en Informatique de Toulouse, France*

(*Received 00 Month 201X; final version received 00 Month 201X*)

The interactions among the components of a biological system can be given a logical representation that is useful for reasoning about them. One of the relevant problems that may be raised in this context is finding what would explain a given behaviour of some component; in other terms, generating hypotheses that, when added to the logical theory modeling the system, imply that behaviour. Temporal aspects have to be taken into account, in order to model the causality relationship that may link the behaviour of a given component to that of another one.

This paper presents a hypothesis generation method for linear temporal logic theories whose formulae have a restricted syntactic form, which is however sufficient to model cellular and molecular interactions, as they are often represented by biologists. The method exploits the duality between hypothesis generation and consequence finding, and can therefore be also used to infer the consequences of a given fact. It is based on a resolution system proposed by Cavalli and Fariñas del Cerro in 1984.

Keywords: Linear Temporal Logic, Abduction, Hypothesis Generation, Consequence Finding, Reasoning about Biological Systems

1. Introduction

The biochemical properties of a cell depend on a series of intracellular and extracellular reactions that produce and consume proteins. Such reactions are in turn regulated by interactions with other proteins and enzymes which can either activate or inhibit them. Cellular and molecular interactions can be graphically represented by means of diagrams called *molecular interaction maps* (MIMs) (Kohn, Aladjem, Weinstein, & Pommier, 2006). Such networks may involve many proteins and enzymes, and are consequently very complex, so that reasoning by hand about them is very hard.

Several works, among which (Obeid, 2014; Demolombe, Fariñas del Cerro, & Obeid, 2013, 2014; Alliot et al., 2016), propose to give a logical representation of the series of biochemical reactions that may occur within a cell, in order to provide biologists with a useful support tool. A logical model of a MIM can be used to perform query answering by use of deduction, but also some form of abductive reasoning. For instance, since the maps used by biologists define causal relationships between different kinds of proteins, one may be interested in finding out which proteins should be activated or inhibited in order to obtain a given effect. Reasoning on such complex sequences of interactions may be of help, for instance, in the field of cancer research, when biologists have to find how the apoptosis (i.e. the "death") of a cell can be obtained by the activation or inhibition of some proteins in a cell. It is not too complex to check, by deductive reasoning, that the state of some proteins causes the apoptosis of a cell. However, it is much more difficult to find, by abductive reasoning, which proteins it is sufficient to activate or inhibit in order to obtain the apoptosis. The dual problem consists in finding out which would be the consequences of a given fact, knowing the overall behaviour of the network, in order to find out, for instance, the side effects of a given therapy. As a matter of fact, finding a solution to such problems cannot be reduced to finding a path in a graph, because, depending on the state of the proteins, a given edge may represent, or not, the capacity to change other proteins. Moreover, the status of many proteins can evolve in time, i.e., they can appear or disappear, according to the cell reactions.

A simple example of molecular interactions is represented in Figure 1. The graph represents a situation where the activation of a given protein A may activate the protein B. However, this activation is inhibited if either C or D are activated. Moreover, D is activated by F; if E is active, then C is inhibited, but this capacity to inhibit C is in turn inhibited when G is active. If, for instance, one is looking for the proteins whose activation produce B, there are two solutions: 1) A is active and neither C nor D are active, or 2) A and E are active, and both D and G are inactive. From this simple example one can imagine the difficulty in real cases where the number of proteins and edges increases.

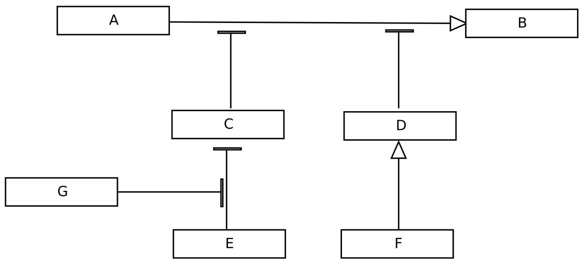

Figure 1. A simple molecular interaction map

The formalism used in (Demolombe et al., 2013, 2014; Obeid, 2014) to model metabolic networks is based on a fragment of first order logic. However, the status of the entities in a MIM may change in time. Consequently, reasoning about metabolic networks requires some form of temporal reasoning and, in fact, the logic used to model and reason about MIMs in (Alliot et al., 2016) takes time into account. In some cases, for instance, the reactants of a biochemical reaction can be consumed when producing their effect. In a MIM, this kind of productions are distinguished by using filled-in arrows, like in Figure 2, that represents the fact that B produces C, and is consumed in the reaction; on the contrary, A is still present and active after producing B. Let us assume that only A is initially present and active; since A produces B, both A and B will be present next. But, subsequently, B produces C and is consumed in the reaction, i.e. it disappears, and can no more been produced, since C inhibits the capacity of A to produce B. This shows that the status of the proteins represented in the simple MIM in Figure 2 evolves in time.

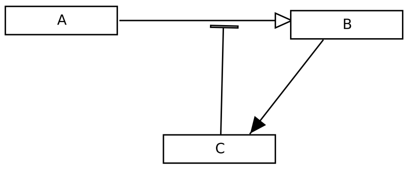

Figure 2. A resource consuming reaction

The proposal presented in (Alliot et al., 2016) consists in representing networks of molecular interactions in a particular logic, called Molecular Interaction Logic (MIL). The work provides a general methodology allowing for the representation of all the basic molecular interactions. In order to reason about MIMs, it is shown that such MIL logical representations can be translated into a Linear Temporal Logic (LTL) theory.

In this paper, we propose an hypothesis generation method for LTL theories consisting of formulae in a restricted syntactic form, which is however sufficient to model the metabolic interactions in biological systems. Like shown in (Alliot et al., 2016), in fact, MIMs can be represented by LTL theories using only the temporal operators $\bigcirc$ ("next") and $\square$ ("always"). Moreover, such theories are equivalent to sets of formulae where the $\square$ operator only occurs as the outermost logical operator. Let us consider, for instance, the network of Figure 1. With some simplification, one can use the atoms $A, B, C, \ldots$ to represent the fact that the proteins $A, B, C, \ldots$ are active, and represent the simple MIMs basically by the following set of formulae:

$$\begin{aligned}
&\square(A \wedge \neg C \wedge \neg D \rightarrow \bigcirc B),\\
&\square(E \wedge \neg G \rightarrow \bigcirc \neg C),\\
&\square(F \rightarrow \bigcirc D)
\end{aligned}$$

Actually, also inertia rules should be added to the representation, as well as completion axioms in order to take into account the underlying closed world assumption. We omit them here for simplicity, and the reader is referred to (Alliot et al., 2016) for a detailed account of the representation methodology.

The first works dealing with abduction in a temporal setting (Eshghi, 1988; Shanahan, 1989) were based on the Event Calculus (Kowalski & Sergot, 1986). In more recent years, several methods have been proposed for computing temporal explanations relying on constraint networks (see, for instance, (Bouzid & Ligeza, 2000; Brusoni, Console, Terenziani, & Dupré, 1997; Ribeiro & Porto, 1994)), and this kind of approach has also been applied in biomedical domains (Teijeiro, Félix, & Presedo, 2014). To the best of our knowledge, however, the problem of hypothesis generation has never been faced in the context of logics with a non-quantitative representation of time. The maybe most similar problem to hypothesis generation in LTL addressed in the literature is temporal query abduction in Description Logics. The language used in (Klarman & Meyer, 2014), which mainly addresses complexity issues, is, in many respects, more expressive than propositional future time LTL. However there are some limitations that make it unsuited for the application considered in the present work. In particular, the statements representing properties holding at all time points are the usual TBox statements, which do not contain the "next" operator.

In the rest of this paper, linear temporal logic is briefly presented (Section 2), followed by a description of hypothesis generation problems (Section 3). Section 4 is the core of this work, presenting the syntactic restrictions on temporal formulae dealt with and the hypothesis generation method for such formulae. Some properties of the underlying inference system are stated (and proved in the Appendix). The hypothesis generation method is based on the resolution system for LTL presented by (Cavalli & Fariñas del Cerro, 1984). This choice is justified in Section 5, where other existing resolution calculi for LTL are considered. The same section also addresses some modeling issues. Finally, Section 6 concludes this work.

2. Linear Temporal Logic

The language of propositional linear temporal logic (LTL) considered in this work contains only unary future time temporal operators: $\square$ (now and always in the future), $\Diamond$ (either now or sometimes in the future), and $\bigcirc$ (in the next state). Among the propositional connectives, $\neg$ (negation), $\wedge$ (conjunction) and $\vee$ (disjunction) are taken as primitive. The model of time underlying LTL is a countably infinite sequence of states (a *time frame*), that can be identified with $\mathbb{N}$. Its elements are called time points. An interpretation $\mathcal{M}$ is a function mapping each time point i to the set of propositional letters true at i. The satisfiability relation $\mathcal{M}_i \models A$, for

$i \in \mathbb{N}$ (A is true at time point i in the interpretation $\mathcal{M}$), is inductively defined as follows:

(1) $\mathcal{M}_i \models p$ iff $p \in \mathcal{M}(i)$, for any propositional letter p in the language.
(2) $\mathcal{M}_i \models \neg A$ iff $\mathcal{M}_i \not\models A$.
(3) $\mathcal{M}_i \models A \wedge B$ iff $\mathcal{M}_i \models A$ and $\mathcal{M}_i \models B$.
(4) $\mathcal{M}_i \models A \vee B$ iff either $\mathcal{M}_i \models A$ or $\mathcal{M}_i \models B$.
(5) $\mathcal{M}_i \models \Box A$ iff for all $j \geq i$, $\mathcal{M}_j \models A$.
(6) $\mathcal{M}_i \models \Diamond A$ iff there exists $j \geq i$ such that $\mathcal{M}_j \models A$.
(7) $\mathcal{M}_i \models \bigcirc A$ iff $\mathcal{M}_{i+1} \models A$.

Truth is satisfiability in the initial state: a formula A is true in $\mathcal{M}$ (and $\mathcal{M}$ is a model of A) iff $\mathcal{M}_0 \models A$. Truth of sets of formulae is defined as usual. If S is a set of formulae and A is a formula, A is a logical consequence of S ($S \models A$) iff for every interpretation $\mathcal{M}$ and $k \in \mathbb{N}$: if $\mathcal{M}_k \models S$ then $\mathcal{M}_k \models A$. Two formulae A and B are logically equivalent iff $A \models B$ and $B \models A$.

The standard axiomatic system for LTL is obtained by adding the following axioms A1-A5 and the inference rule R to any axiomatization of classical propositional logic:

$$
\begin{array}{ll}
A1. & \Box(A \to B) \to (\Box A \to \Box B) \\
A2. & \bigcirc(A \to B) \to (\bigcirc A \to \bigcirc B) \\
A3. & \bigcirc\neg A \equiv \neg\bigcirc A \\
A4. & \Box A \to (A \wedge \bigcirc\Box A) \\
A5. & \Box(A \to \bigcirc A) \to (A \to \Box A) \\
R. & \dfrac{A}{\Box A}
\end{array}
$$

3. Hypothesis Generation

Hypothesis generation consists in finding what, in the context of a given *background knowledge* T, would explain something that is not a consequence of T. Hypothesis generation is strictly related to abductive reasoning, a natural form of reasoning performed by an agent when new evidence informs her knowledge: an abductive inference is drawn when the truth of the sentence explaining the new evidence is "derived", i.e., it is added to the knowledge base. In general there may be different sentences explaining the same evidence, so some of them are rejected before the inference is performed. In other terms the inference is drawn when the agent is convinced that any other sentence is not plausible enough, compared with the chosen one, to be accepted as an explanation.

In the pure logical account of abduction, an abductive problem (in a given logic $\mathcal{L}$) is specified by a theory T and a sentence F to be explained (the *explanandum*); it is solved by finding a sentence E (among the best ones, according to some given preference criteria), such that $T \cup \{E\} \models_{\mathcal{L}} F$. It is moreover assumed that $T \not\models_{\mathcal{L}} F$ and $T \not\models_{\mathcal{L}} \neg F$, i.e., F is consistent with T. When viewed in these terms, there is an obvious duality between hypothesis generation and consequence finding (and between abduction and deduction): if T is the background theory, in the context of which a given fact F has to be explained, then the problem can be addressed by searching for a formula E such that $T \cup \{\neg F\} \models \neg E$. This amounts to $T \cup \{E\} \models F$, i.e, E explains F in the context of T. In other words, hypothesis generation can be reduced to consequence finding (up to a certain extent).

In both contexts, the formula looked for must satisfy additional conditions. In the logical approach to abduction, such conditions generally include *minimality* with respect to logical consequence: a formula E is a "relevant explanation" of F in the context of a theory T, only if $T \cup \{E\} \models F$ (i.e., E is an explanation of F) and there is no weaker explanation of F: for every formula E', if $T \cup \{E'\} \models F$ and $E \models E'$, then also $E' \models E$. In other terms, weaker explanations are preferred to stronger ones.

On the side of consequence finding, a stronger consequence is preferred to a weaker one: A is a "relevant consequence" of a set S of formulae only if for every consequence B of S, if $B \models A$, then also $A \models B$.

Though this may be seen as a "maximality" condition, it amounts to minimality w.r.t. *subsumption* when looking for clauses implied by a given set of clauses S, in classical logic: a "relevant consequence" of S is a clause C that is a consequence of S and is not subsumed by any other clause C' such that $S \models C'$.

Getting back to hypothesis generation, there are additional requirements to be fulfilled by relevant explanations, in order to rule out trivial ones: a relevant explanation E of F in the background theory T is a minimal explanation of F such that

A. $T \not\models \neg E$: if $T \cup \{E\}$ is inconsistent, E is a trivial explanation of any fact.
B. $E \not\models F$: since $T \cup \{F\} \models F$, F is an explanation of F itself in T. If E is a minimal explanation of F in T, then $E \models F$ implies $F \models E$, i.e., E and F are logically equivalent. This would also make E a trivial explanation.

Consequently, when looking for hypotheses explaining a given observation, the two conditions A and B must be additionally considered.

It is worth pointing out that, although the two above conditions have no meaning in a general consequence finding problem, they have to be considered when the problem is finding out which *new* consequences can be drawn when a given fact F' is added to the background theory T (as is the case in the biological setting considered in this work). In this case, a "relevant" consequence C must be such that $T \not\models C$ and $F' \not\models C$, i.e. the relevant new consequences must be derivable neither from the theory alone, nor from the fact F' alone.

4. The hypothesis generation method

The problem of computing abductive explanations for LTL theories is very hard to face. In fact, it is not easy to adopt methodologies based on resolution or tableau methods, like it can be done for classical logic (Cialdea Mayer & Pirri, 1993; Cox & Pietrzykowski, 1986; Demolombe & Fariñas del Cerro, 1991; Inoue, 1992) or even modal logics (Cialdea Mayer & Pirri, 1995), because both resolution and tableau-based proof systems for temporal logics are among the most complex ones for modal logics. The hypothesis generation method proposed in this work exploits the very simple form of formulae making up theories modeling the behaviour of biological systems, and both *explananda* and explanations. In this restricted context, the problem becomes much simpler.

The method exploits the duality between hypothesis generation and consequence finding. In order to properly define the minimality condition, a suitable notion of subsumption for temporal clauses (in the considered restricted syntactical form) is defined. Hypothesis generation derive the "relevant" consequences of the set of clauses obtained by adding the negation of the *explanandum* F to the background theory T. Relevant consequences are those which are not subsumed by any other consequence of $T \cup \{\neg F\}$ and are derived by using both $\neg F$ and some clause in T. The latter condition is required in order to satisfy the non triviality conditions A and B given in Section 3. Analogously, when the relevant consequences that can be drawn by the addition of a fact F to the background theory T are looked for, the solutions are the consequences of $T \cup \{F\}$ depending from both F and some clause in T, and are minimal w.r.t. subsumption.

4.1 *The syntactic restrictions*

The important restrictions in the LTL language used to state the background theory and the negation of the *explanandum* are the following, where, for the sake of simplicity, formulae are assumed to be in negation normal form:

(1) there are no occurrences of the $\Diamond$ operator; and
(2) the $\Box$ operator never occurs in the scope of a disjunction.

The next definition introduces the syntactical form allowed for clauses (*flat clauses*) occurring in a proof.[1] The notation $\bigcirc^n$ abbreviates a sequence of n occurrences of the $\bigcirc$ operator.

Definition 1 (Flat clauses)**.** A *modal literal* is a formula of the form $\bigcirc^n \ell$ where $n \geq 0$ and ℓ is a classical literal (i.e., either an atom or the negation of an atom).

A *flat clause* is either:

- an *initial clause*, of the form $L_1 \vee \cdots \vee L_k$ where $L_1, \ldots, L_k$ are modal literals and $k \geq 0$, or
- an *always clause*, of the form $\Box(L_1 \vee \cdots \vee L_k)$ where $L_1, \ldots, L_k$ are modal literals and $k \geq 0$.

Empty disjunctions will be denoted by $\bot$.

Calligraphic lowercase letters ($\ell, \mathcal{p}, \mathcal{q}$, possibly with subscripts) will be used to denote classical literals, while for modal literals the meta-symbols L and M (possibly with subscripts) will be used. As usual, disjunctions of literals are treated as sets, i.e., the order in which literals occur is irrelevant, and they are assumed to contain no repetitions. Flat clauses will sometimes be simply called clauses, when there is no risk of confusion.

It is worth pointing out that any LTL formula in negation normal form not containing $\Diamond$ and such that $\Box$ never occurs in the scope of a disjunction is logically equivalent to a set of flat clauses.

Facts to be explained are assumed to be either a conjunction of modal literals or formulae of the form $\Diamond(L_1 \wedge \cdots \wedge L_k)$, where $L_1, \ldots, L_k$ are modal literals. Consequently, the negation of an *explanandum* is equivalent to a flat clause. The generated hypotheses have the same form as the *explananda*, since they will be built as negations of flat clauses.

The subsumption relation for flat clauses is defined next.

Definition 2 (Subsumption)**.** Let C and C' be flat clauses. The clause C is subsumed by C' if one of the following cases holds (where disjunctions are treated as sets):

- $C = L_1 \vee \cdots \vee L_k \vee M_1 \vee \cdots \vee M_n$, for $n \geq 0$, and $C' = L_1 \vee \cdots \vee L_k$;
- $C = \bigcirc^m L_1 \vee \cdots \vee \bigcirc^m L_k \vee M_1 \vee \cdots \vee M_n$, for some $m, n \geq 0$, and $C' = \Box(L_1 \vee \cdots \vee L_k)$;
- $C = \Box(\bigcirc^m L_1 \vee \cdots \vee \bigcirc^m L_k \vee M_1 \vee \cdots \vee M_n)$, for some $m, n \geq 0$, and $C' = \Box(L_1 \vee \cdots \vee L_k)$.

For instance, both $\bigcirc p \vee \bigcirc^2 \neg p \vee q$ and $\Box(\bigcirc^2 p \vee \bigcirc^3 \neg p \vee q)$ are subsumed by $\Box(p \vee \bigcirc \neg p)$. Subsumption between two initial clauses amounts to the subset relationship, like in classical propositional logic. It is worth noting that the subsumption relation is transitive, and that, if C is subsumed by C', then $C' \to C$ is valid.[2]

4.2 *The basic inference rules for flat clauses*

In order to give a compact formulation of the inference rules of our formal system, it is useful to define the complement of a modal literal.

Definition 3 (Complement)**.** Let L be a modal literal. The *complement* of L, denoted by $\sim L$, is defined as follows:

[1]A flat clause corresponds to a clause in separated normal form, as defined by Fisher et al. (Fisher, Dixon, & Peim, 2001), but the syntactic form of flat clauses is more general. The definition of separated normal form can be found in Section 5.
[2]Like in classical logic, the converse does not hold. Consider, for instance $C = \Box(\neg p \vee \bigcirc^2 p)$ and $C' = \Box(\neg p \vee \bigcirc p)$. The flat clause C is not subsumed by C', though $C' \to C$ is valid.

- if $L = p$, where p is an atom, then $\sim L = \neg p$;
- if $L = \neg p$, where p is an atom, then $\sim L = p$;
- if $L = \bigcirc M$, where M is a modal literal, then $\sim L = \bigcirc{\sim}M$. In other terms, $\sim\bigcirc^n \ell = \bigcirc^n {\sim}\ell$, where ℓ is a classical literal.

The basic inference rules adopted in this work simplify, taking advantage of the flat clause form, the resolution rules introduced by Cavalli and Fariñas del Cerro (Cavalli & Fariñas del Cerro, 1984).

Definition 4 (Basic Inference Rules)**.** Let $L, L_1, \dots, L_n, M_1, \dots, M_m$ be modal literals and $k \geq 0$. The inference system includes the following rules:

$$\frac{\Box(L \vee L_1 \vee \cdots \vee L_n) \quad \Box(\bigcirc^k{\sim}L \vee M_1 \vee \cdots \vee M_m)}{\Box(\bigcirc^k L_1 \vee \cdots \vee \bigcirc^k L_n \vee M_1 \vee \cdots \vee M_m)} \ (R1)$$

$$\frac{\Box(L \vee L_1 \vee \cdots \vee L_n) \quad \bigcirc^k{\sim}L \vee M_1 \vee \cdots \vee M_m}{\bigcirc^k L_1 \vee \cdots \vee \bigcirc^k L_n \vee M_1 \vee \cdots \vee M_m} \ (R2)$$

$$\frac{L \vee L_1 \vee \cdots \vee L_n \quad {\sim}L \vee M_1 \vee \cdots \vee M_m}{L_1 \vee \cdots \vee L_n \vee M_1 \vee \cdots \vee M_m} \ (R3)$$

The two modal literals L and $\bigcirc^k{\sim}L$ in R1 and R2, and L and ${\sim}L$ in R3 are called the literals resolved upon in the inference.

In addition to the above rules, a simplification rule is added, allowing one to replace $\Box\bot$ with $\bot$. Simplification will be applied implicitly. In what follows, R1-R3 will denote the inference system consisting of the three rules of Definition 4 (and simplification), and the symbol $\vdash_{R1-R3}$ is used to denote derivability in such a system.

Example 1. Consider the example of the simple MIM of Figure 1, and its representation by means of the set of formulae $\{\Box(A \wedge \neg C \wedge \neg D \rightarrow \bigcirc B), \Box(E \wedge \neg G \rightarrow \bigcirc\neg C), \Box(F \rightarrow \bigcirc D)\}$. The flat clause form of such formulae constitute the background theory $T = \{\Box(\neg A \vee C \vee D \vee \bigcirc B), \Box(\neg E \vee G \vee \bigcirc\neg C), \Box(\neg F \vee \bigcirc D)\}$. In order to find an explanation for $\Diamond B$, its negation is added to T. Figure 3 contains a complete deduction from $T \cup \{\neg\Diamond B\}$. The derived clauses 5 and 6 depend from both some clause in T and from clause 4, so they are used to generate the two explanations $\Diamond(A \wedge \neg C \wedge \neg D)$ and $\Diamond(E \wedge \neg G \wedge \bigcirc A \wedge \bigcirc\neg D)$.

1) $\Box(\neg A \vee C \vee D \vee \bigcirc B)$	(in T)
2) $\Box(\neg E \vee G \vee \bigcirc\neg C)$	(in T)
3) $\Box(\neg F \vee \bigcirc D)$	(in T)
4) $\Box(\neg B)$	(negation of the *explanandum*)
5) $\Box(\neg A \vee C \vee D)$	(from 1 and 4)
6) $\Box(\neg E \vee G \vee \bigcirc\neg A \vee \bigcirc D)$	(from 2 and 5)

Figure 3. A deduction from the clauses of Example 1

Example 2. As another simple example, let us consider the problem of finding an explanation for $\Diamond p$ in the theory $T = \{\Box(\neg p \vee q \vee \bigcirc t), \Box(\neg p \vee e \vee \bigcirc t), \Box(q \vee \bigcirc p), \Box(e \vee \bigcirc p)\}$. A complete deduction from $T \cup \{\neg\Diamond p\}$ can be found in Figure 4. The four clauses of lines 8-11, beyond being consequences of T alone, are subsumed by other clauses in the derivation (8 and 11 are subsumed by both 7 and 6, 9 is subsumed by 7 and 10 by 6), so they are ignored for hypothesis generation and should not even be added to the derivation.

The derivation of 6 makes use both of some clause in T and of clause 5, and the same holds for 7, therefore they are used to generate the two explanations $\Diamond\neg q$ and $\Diamond\neg e$, i.e. the negations of clauses 6 and 7, respectively.

1)	$\Box(\neg p \vee q \vee \bigcirc t)$	(in T)
2)	$\Box(\neg p \vee e \vee \bigcirc t)$	(in T)
3)	$\Box(q \vee \bigcirc p)$	(in T)
4)	$\Box(e \vee \bigcirc p)$	(in T)
5)	$\Box\neg p$	(negation of the *explanandum*)
6)	$\Box q$	(from 3 and 5)
7)	$\Box e$	(from 4 and 5)
8)	$\Box(e \vee \bigcirc q \vee \bigcirc^2 t)$	(from 1 and 4)
9)	$\Box(e \vee \bigcirc e \vee \bigcirc^2 t)$	(from 2 and 4)
10)	$\Box(q \vee \bigcirc q \vee \bigcirc^2 t)$	(from 1 and 3)
11)	$\Box(q \vee \bigcirc e \vee \bigcirc^2 t)$	(from 2 and 3)

Figure 4. A deduction from the clauses of Example 2

Example 3. Consider the theory $T = \{\Box(\neg p \vee q \vee \bigcirc r), \Box(\neg s \vee \bigcirc p)\}$ and the *explanandum* $\Diamond r$. A (complete) deduction from T and the negation of the *explanandum* is illustrated in Figure 5. Clause 5 is a consequence of T, since in its derivation no use is made of clause 3, therefore its negation would be a trivial explanations. On the contrary, 4 and 6 depend both on T and the negation of the *explanandum*, therefore their negations constitute the two explanations found for $\Diamond r$, i.e., $\Diamond(\neg q \wedge p)$ and $\Diamond(s \wedge \bigcirc\neg q)$.

1)	$\Box(\neg p \vee q \vee \bigcirc r)$	(in T)
2)	$\Box(\neg s \vee \bigcirc p)$	(in T)
3)	$\Box\neg r$	(negation of the *explanandum*)
4)	$\Box(\neg p \vee q)$	(from 1 and 3)
5)	$\Box(\neg s \vee \bigcirc q \vee \bigcirc^2 r)$	(from 1 and 2)
6)	$\Box(\neg s \vee \bigcirc q)$	(from either 2 and 4, or 3 and 5)

Figure 5. A deduction from the clauses of Example 3

4.3 *Refutational completeness*

The rules R1-R3 are special cases of the general resolution rules defined in (Cavalli & Fariñas del Cerro, 1984). The latter are defined for formulae in a clausal form, that will be here called CF-clauses. CF-clauses include flat clauses, but not vice-versa (CF-clauses are expressively complete for LTL).

The problem of the refutational completeness for flat clauses of the system R1-R3 is addressed by reduction to the resolution calculus defined in (Cavalli & Fariñas del Cerro, 1984), that will henceforth be called CF and for which a brief description is given below, limited to what is relevant for the treatment of flat clauses. The resolution rule of CF, when restricted to act on to CF-clauses without any occurrence of the $\Diamond$ operator, can be reformulated as follows. If C_1 and C_2 are clauses, $\Sigma(C_1, C_2) \rhd C$ denotes a Σ-*reduction step* of C_1 and C_2 and is recursively defined by the following Σ-reduction rules:[1]

(a) $\Sigma(p, \neg p) \rhd \perp$;
(b) $\Sigma(D_1 \vee D_2, F) \rhd \Sigma(D_1, F) \vee D_2$;
(c) $\Sigma(\bigcirc E, \bigcirc F) \rhd \bigcirc\Sigma(E, F)$;
(d) $\Sigma(\Box E, \nabla F) \rhd \nabla\Sigma(E, F)$ where $\nabla \in \{\Box, \bigcirc\}$;
(e) $\Sigma(\Box E, F) \rhd \Sigma(E, F)$;
(f) $\Sigma(\Box E, F) \rhd \Sigma(\Box\Box E, F)$.

The reflexive and transitive closure of $\rhd$ is denoted by $\rhd^*$: $\Sigma(C_1, C_2) \rhd^* C$ if and only if C is a clause and there is a sequence of Σ-reduction steps $\Sigma(C_1, C_2) \rhd \cdots \rhd C$. Two clauses C_1 and C_2 are *resolvable* if $\Sigma(C_1, C_2) \rhd^* C$ for some clause C. The *simplification* of a clause C is obtained by recursively replacing F for every subformula of the form $\perp \vee F$, and $\perp$ for every subformula of the form $\Box\perp$ and $\bigcirc\perp$. If C_1 and C_2 are resolvable, then a *CF-resolvent* $R(C_1, C_2)$ of C_1 and C_2 is the simplification of some C such that $\Sigma(C_1, C_2) \rhd^* C$.

When restricted to act on flat clauses, the system CF and R1-R3 are equivalent. Below, the symbol $\vdash_{CF}$ denotes derivability in the calculus CF.

Theorem 1. *If $S \cup \{C\}$ is a set of flat clauses, then:*

(1) if $S \vdash_{CF} C$, then $S \vdash_{R1-R3} C'$ for some flat clause C' subsuming a clause that is logically equivalent to C;
(2) if $S \vdash_{R1-R3} C$, then $S \vdash_{CF} C'$ for some clause C' that is logically equivalent to C.

A direct consequence of Theorem 1 is the following:

Corollary 1. *The inference system consisting of the rules R1-R3 is sound and refutationally complete for flat clauses.*

The proof of these results and the others which follow can be found in the Appendix.

Example 4. Let us consider, for instance, the unsatisfiable set $S = \{p, \Box(\neg p \vee \bigcirc p), \bigcirc^2\neg p\}$. The following derivation shows that $S \vdash_{R1-R3} \perp$.

1) $\Box(\neg p \vee \bigcirc p)$	(in S)
2) p	(in S)
3) $\bigcirc^2\neg p$	(in S)
4) $\bigcirc p$	(from 1 and 2)
5) $\bigcirc^2 p$	(from 1 and 4)
6) $\perp$	(from 3 and 5)

In general, every clause of the form $\bigcirc^n p$ can be derived from $\{p, \Box(\neg p \vee \bigcirc p)\}$. Therefore any set of the form $\{p, \Box(\neg p \vee \bigcirc p), \bigcirc^n\neg p\}$ can be refuted.

4.4 *A weak form of implicational completeness*

In order to study the derivational strength of the inference system R1-R3, a translation into classical first-order logic is defined, mapping each flat clause into a monadic formula. It simplifies the translation used to give an arithmetical semantics to LTL (see, for instance, (Abadi, 1989)).[1] The translation takes again advantage of the syntactical restrictions, in particular of the fact that,

[1]In (Cavalli & Fariñas del Cerro, 1984), other reduction rules are included, but they all act on CF-clauses containing the $\Diamond$ operator, that is absent in flat clauses.
[1]In order to avoid misunderstandings, it is worth pointing out that, although syntactically the mapping τ is a simplification of Kamp's translation, the semantics of the target language is not First-Order Monadic Logic of Order, so that one cannot expect that the equivalence established by Kamp's theorem holds.

since the $\Box$ operator only occurs as the outermost logical symbol in a flat clause, it acts as a sort of global modality.

Definition 5 (Translation)**.** Let P be a set of propositional atoms, and $\mathcal{L}_P$ the first-order language containing a unary predicate symbol p for each $p \in P$, a constant a and a unary functional symbol f.

The (auxiliary) mapping τ^* maps a first-order term of $\mathcal{L}_P$ and a disjunction of modal literals into a formula of $\mathcal{L}_P$, and is defined as follows:

- $\tau^*(t, p) = p(t)$, if $p \in P$;
- $\tau^*(t, \neg p) = \neg p(t)$, if $p \in P$;
- $\tau^*(t, \bigcirc L) = \tau^*(f(t), L)$, if L is a modal literal;
- $\tau^*(t, A \vee B) = \tau^*(t, A) \vee \tau^*(t, B)$

Flat clauses are translated into classical first-order clauses of $\mathcal{L}_P$ by means of the mapping τ defined as follows:

- $\tau(L_1 \vee \cdots \vee L_k) = \tau^*(a, L_1 \vee \cdots \vee L_k)$;
- $\tau(\Box(L_1 \vee \cdots \vee L_k)) = \tau^*(x, L_1 \vee \cdots \vee L_k)$.

When translating a set S of flat clauses, a different variable is used for each clause in S.

For instance, if $C = p \vee \bigcirc q \vee \bigcirc^2 \neg r$ (an initial clause), $\tau(C) = p(a) \vee q(f(a)) \vee \neg r(f(f(a)))$, while $\tau(C') = p(x) \vee q(f(x)) \vee \neg r(f(f(x)))$ for the always clause $C' = \Box(p \vee \bigcirc q \vee \bigcirc^2 \neg r)$.

By use of the above defined translation, the relation between classical and temporal subsumption can be established. We recall that, in classical logic, a clause C subsumes a clause C' if there exists a substitution θ such that $C\theta \subseteq C'$.

Theorem 2. *A flat clause C subsumes a flat clause C' if and only if $\tau(C)$ (classically) subsumes $\tau(C')$.*

The correspondence between the temporal and classical settings established by the translation τ also applies to the inference system consisting of the three rules R1-R3. As a matter of fact, such rules are just a rewriting of the classical resolution rule. In what follows, the symbol $\vdash_{FOL}$ denotes derivability by classical resolution in first-order logic. Analogously, while $\models$ denotes logical consequence in LTL, logical consequence in first-order logic is denoted by $\models_{FOL}$. As usual, a classical clause is intended to be universally closed; in particular, $\tau(C_1), \dots, \tau(C_n) \models_{FOL} \tau(C)$ stands for $\forall\tau(C_1), \dots, \forall\tau(C_n) \models_{FOL} \forall\tau(C)$, where $\forall A$ is the universal closure of A.

Theorem 3. *If $C_1, \dots, C_n, C$ are flat clauses, then $C_1, \dots, C_n \vdash_{R1-R3} C$ if and only if $\tau(C_1), \dots, \tau(C_n) \vdash_{FOL} \tau(C)$.*

Theorem 3 allows one to exploit results holding for classical logic, such as the *implicational completeness* of resolution:

Theorem 4 ((Lee, 1967))**.** *Let S be a set of classical clauses and C a non valid clause. If $S \models_{FOL} C$, there is a clause C' subsuming C such that C' is derivable from S by (classical) resolution.*

The strict correspondence between classical resolution and the temporal rules R1-R3 implies a weak form of completeness w.r.t. consequence finding:

Theorem 5. *If $C_1, \dots, C_n, C$ are flat clauses, C is not valid and $\tau(C_1), \dots, \tau(C_n) \models_{FOL} \tau(C)$, then there exists a clause C' subsuming C such that $C_1, \dots, C_n \vdash_{R1-R3} C'$.*

The above results allows one to prove that the calculus R1-R3 is implicationally complete w.r.t. initial clauses:

Theorem 6. *If $C_1, \dots, C_n$ are flat clauses, C is a non valid initial clause and $C_1, \dots, C_n \models C$, then there exists a clause C' subsuming C such that $C_1, \dots, C_n \vdash_{R1-R3} C'$.*

Example 5. Let S be the set

$$\{\bigcirc^2 p \vee \bigcirc^3 q,\ \Box(\bigcirc\neg p \vee \bigcirc^2 p \vee \bigcirc^2 q),\ \Box(\bigcirc^2\neg q \vee \bigcirc^3 p \vee \bigcirc^3 q)\}$$

For all $n \geq 3$ and any disjunction of modal literals D, it holds that $S \models \bigcirc^n p \vee \bigcirc^n q \vee D$.

The following derivation shows that the logical consequences of S of the form $\bigcirc^n p \vee \bigcirc^n q$ for $n \geq 3$ are derivable from S, hence for every initial clause $C = \bigcirc^n p \vee \bigcirc^n q \vee D$, with $n \geq 3$, there is a clause C' subsuming C such that $S \vdash_{R1-R3} C'$. The clauses of the form $\bigcirc^n p \vee \bigcirc^n q$ with $n \geq 3$ are all subsumed by $\Box(\bigcirc^3 p \vee \bigcirc^3 q)$, which, however, is not derivable from S.

1)	$\Box(\bigcirc\neg p \vee \bigcirc^2 p \vee \bigcirc^2 q)$	(in S)
2)	$\Box(\bigcirc^2\neg q \vee \bigcirc^3 p \vee \bigcirc^3 q)$	(in S)
3)	$\bigcirc^2 p \vee \bigcirc^3 q$	(in S)
4)	$\bigcirc^3 p \vee \bigcirc^3 q$	(from 1 and 3)
5)	$\bigcirc^3 p \vee \bigcirc^4 p \vee \bigcirc^4 q$	(from 2 and 4)
6)	$\bigcirc^4 p \vee \bigcirc^4 q$	(from 1 and 5)
7)	$\bigcirc^4 p \vee \bigcirc^5 p \vee \bigcirc^5 q$	(from 2 and 6)
8)	$\bigcirc^5 p \vee \bigcirc^5 q$	(from 1 and 7)
…		

4.5 *Coping with the induction axiom*

As a consequence of Theorem 5 and Corollary 1, the translation given in Definition 5 enjoys the following general property: if $\tau(C_1), \ldots, \tau(C_n) \models_{FOL} \tau(C)$, then $C_1, \ldots, C_n \models C$. The converse, obviously, does not hold: for instance, $p, \Box(\neg p \vee \bigcirc p) \models \Box p$, but $p(a), \forall x(\neg p(x) \vee p(f(x)) \not\models_{FOL} \forall x p(x)$. As a matter of fact, the rules R1-R3 do not allow one to derive the flat clause $\Box p$ from p and $\Box(\neg p \vee \bigcirc p)$ (though all the subsumed clauses of the form $\bigcirc^n p$ are derivable, by Theorem 6). So, the system whose only rules are R1-R3 is implicationally incomplete. Such rules in fact do not take into account what makes LTL different from FOL, i.e. the induction axiom (axiom A5 of Section 2): $\Box(A \rightarrow \bigcirc A) \rightarrow (A \rightarrow \Box A)$.

It is worth pointing out that there is no contradiction with Corollary 1: the negation of the induction axiom cannot be refuted in R1-R3 just because it cannot be expressed as a set of flat clauses (i.e. $\{A, \Box(\neg A \vee \bigcirc A), \Diamond\neg A\}$ is not a set of set of flat clauses).

To the aim of gaining implicational completeness, the restricted syntax of flat clauses can be exploited again. Here, the $\Box$ operator only occurs as the outermost logical symbol, and consequence finding is also restricted to flat clauses. In this context, the induction axiom can be taken into account by adding the following induction rule:

$$\frac{L_1 \vee \cdots \vee L_k \quad C_1 \quad \ldots \quad C_k}{\Box(L_1 \vee \cdots \vee L_k)}\ \text{(Ind)}$$

where, for $i = 1, \ldots, k$, C_i is any clause subsuming $\Box({\sim}L_i \vee \bigcirc L_1 \vee \cdots \vee \bigcirc L_k)$.

It is worth observing that the following inference would also be correct:

$$\frac{L_1 \vee \cdots \vee L_k \vee Q \quad \Box({\sim}L_1 \vee \bigcirc L_1 \vee \cdots \vee \bigcirc L_k) \quad \ldots \quad \Box({\sim}L_k \vee \bigcirc L_1 \vee \cdots \vee \bigcirc L_k)}{Q \vee \Box(L_1 \vee \cdots \vee L_k)}$$

but its conclusion is not a flat clause. Analogously, although $\Box(p \vee q), \Box(\neg q \vee \bigcirc q) \models \Box(p \vee \Box q)$, the conclusion could not be derived just because it is not a flat clause.

Example 6. By use of the induction rule, $\Box(\bigcirc^3 p \vee \bigcirc^3 q)$ can be derived from the set of clauses $S = \{\bigcirc^2 p \vee \bigcirc^3 q,\ \Box(\bigcirc\neg p \vee \bigcirc^2 p \vee \bigcirc^2 q),\ \Box(\bigcirc^2\neg q \vee \bigcirc^3 p \vee \bigcirc^3 q)\}$. In fact, $S \vdash_{R1-R3} \bigcirc^3 p \vee$

$\bigcirc^3 q$, as shown in Example 5, $\Box(\bigcirc\neg p \vee \bigcirc^2 p \vee \bigcirc^2 q)$ subsumes $\Box(\bigcirc^3\neg p \vee \bigcirc^4 p \vee \bigcirc^4 q)$ and $\Box(\bigcirc^2\neg q \vee \bigcirc^3 p \vee \bigcirc^3 q)$ subsumes $\Box(\bigcirc^3\neg q \vee \bigcirc^4 p \vee \bigcirc^4 q)$. Consequently, the induction rule can be applied to obtain $\Box(\bigcirc^3 p \vee \bigcirc^3 q)$.

Let FlaTL be the proof system consisting of the rules R1-R3 and Ind, and let $\vdash_{FlaTL}$ denote derivability in FlaTL. It may be hypothesized that FlaTL is complete for LTL w.r.t. consequence finding restricted to flat clauses.

Conjecture. If $C_1, \dots, C_n, C$ are flat clauses, C is not valid and $C_1, \dots, C_n \models C$, then there exists a clause C' subsuming C such that $C_1, \dots, C_n \vdash_{FlaTL} C'$.

4.6 *Non termination*

The calculus FlaTL does not enjoy the termination property, even if the generation of clauses subsumed by other clauses in the proof is blocked. A simple example of non-terminating derivation can be extracted from Example 4: if $T = \{\Box(p \rightarrow \bigcirc p), p\}$:

$$\dfrac{\Box(\neg p \vee \bigcirc p) \qquad \dfrac{\Box(\neg p \vee \bigcirc p) \qquad \dfrac{\Box(\neg p \vee \bigcirc p) \quad p}{\bigcirc p}\,(R2)}{\bigcirc\bigcirc p}\,(R2)}{\dfrac{\bigcirc\bigcirc\bigcirc p}{\vdots}}\,(R2)$$

Derivations may not terminate even if the application of rule R2 is blocked when the induction rule Ind can be applied. For instance, from $T = \Box(\neg p \vee \bigcirc p)$ every clause of the form $\Box(\neg p \vee \bigcirc^n p)$, for $n \geq 1$, can be generated, and none of them is subsumed by the others:

$$\dfrac{\Box(\neg p \vee \bigcirc p) \qquad \dfrac{\Box(\neg p \vee \bigcirc p) \qquad \dfrac{\Box(\neg p \vee \bigcirc p) \quad \Box(\neg p \vee \bigcirc p)}{\Box(\neg p \vee \bigcirc^2 p)}\,(R1)}{\Box(\neg p \vee \bigcirc^3 p)}\,(R1)}{\dfrac{\Box(\neg p \vee \bigcirc^4 p)}{\vdots}}\,(R1)$$

As a matter of fact, $\Box(\neg p \vee \bigcirc p)$ has an infinite number of logical consequences. They are all implied by $\Box(\neg p \vee \Box p)$, but the latter is not a flat clause.

5. Discussion

In this section two points are addressed: the first one concerns modeling issues in relation to the completeness of the hypothesis generation method; the second one regards the choice of the resolution calculus on which it is based.

Although the implicational completeness of the calculus presented in this work w.r.t. always clauses is only conjectured at present, Theorem 6 establishes that the calculus is implicationally complete w.r.t. initial clauses. In the context of problems concerning biochemical reactions, implicational completeness for initial clauses must not be underestimated. In general both the fact to be explained and the found explanations can reasonably be represented by formulae of the form $\Diamond(L_1 \wedge \cdots \wedge L_k)$, so that one is interested in deriving always clauses. However, in some cases, counterintuitive explanations can be obtained. Consider, for instance, the case where one looks for causes for the inhibition of a given protein C ($\Diamond\neg C$) in the context of the background theory (equivalent to) $\{\Box(A \wedge \bigcirc B \rightarrow \bigcirc^2\neg C),\ \Box(C \rightarrow \bigcirc D)\}$ (activating A and then B causes the inhibition of C, and the activation of C causes the activation of D). Then, beyond the explanation

$\Diamond(A \wedge \bigcirc B)$, also the explanation $\Diamond\bigcirc\neg D$ would be obtained, as if a given fact could be explained by something happening in its future (see Figure 6, on the left). Logically, this is correct, but the inhibition of D cannot be considered a cause for the inhibition of C. If however the fact to be explained is expressed by an initial clause, for instance $\bigcirc^2\neg C$, then its explanations would be initial clauses too, and one could easily distinguish “good explanations” (causes) from the others by looking at whether the activation/inhibition of proteins should occur before or after the inhibition of C. In this simple example (see Figure 6, on the right), the only “good” explanation would be $A \wedge \bigcirc B$, while $\bigcirc^3\neg D$ would be recognized as a (logically good but) fake explanation.

1)$\Box(\neg A \vee \bigcirc\neg B \vee \bigcirc^2\neg C)$	(in T)	1)$\Box(\neg A \vee \bigcirc\neg B \vee \bigcirc^2\neg C)$	(in T)
2)$\Box(\neg C \vee \bigcirc D)$	(in T)	2)$\Box(\neg C \vee \bigcirc D)$	(in T)
3)$\Box C$	(negation of the *explanandum*)	3)$\bigcirc^2 C$	(negation of the *explanandum*)
4)$\Box\bigcirc D$	(from 2 and 3)	4)$\bigcirc^3 D$	(from 2 and 3)
5)$\Box(\neg A \vee \bigcirc\neg B)$	(from 1 and 3)	5)$\neg A \vee \bigcirc\neg B$	(from 1 and 3)

Explanations found:		Explanations found:	
1)$\Diamond(A \wedge \bigcirc B)$	(from row 5)	1)$A \wedge \bigcirc B$	(from row 5)
2)$\Diamond(\bigcirc\neg D)$	(from row 4)	2)$\bigcirc^3\neg D$	(from row 4)

Figure 6. The problem of fake explanations

A different way to solve the problem of “fake” explanations is by encoding rules as if the $\bigcirc$ operator meant “in the previous state”, and setting the fact to be explained in the present. In the above example, the background theory would be encoded by the set $S = \{\Box(\bigcirc^2 A \wedge \bigcirc B \rightarrow \neg C), \Box(\bigcirc C \rightarrow D)\}$ (C is inhibited whenever A is active two states before and B is active in the previous state, and D is active when C was previously active) and the fact to be explained would just be $\neg C$. In this case, explanations in the future are automatically ruled out: one obtains only the explanation $\bigcirc^2 A \wedge \bigcirc B$. In fact, the only clause that can be derived from $S \cup \{C\}$ is $\bigcirc^2\neg A \vee \bigcirc\neg B$, since $\Box(\bigcirc C \rightarrow D)$ cannot be resolved against C. Following this approach, explanations are again initial clauses, so that Theorem 6 guarantees that all the minimal ones can be found.

For what concerns the choice of the calculus proposed in (Cavalli & Fariñas del Cerro, 1984) as the starting point of the hypothesis generation method, a comparison with other resolution methods for LTL is in order. The two other main resolution systems for LTL proposed in the literature have been defined by Abadi and Manna (Abadi & Manna, 1985) and Fisher et al. (see, for instance, (Fisher et al., 2001)).

The first one, however, deals with formulae in non-clausal form, and this would raise unnecessary complications in the context of the application considered here. The second one, on the other hand, is not complete for consequence finding, even when deriving the consequence does not involve any use of the induction axiom.

In order to prove this claim, the system defined by Fisher et al. must be briefly presented. It deals with clauses in *separated normal form* (SNF), which have one of the following forms:

- initial clauses: $\Box(\mathbf{start} \rightarrow \ell_1 \vee \cdots \vee \ell_n)$ where **start** is a distinguished propositional letter whose semantics is given by the condition that $\mathcal{M}_i \models \mathbf{start}$ if and only if $i = 0$;
- step clauses: $\Box(\ell_1 \wedge \cdots \wedge \ell_n \rightarrow \bigcirc(p_1 \vee \cdots \vee p_k))$
- sometime clauses: $\Box(\ell_1 \wedge \cdots \wedge \ell_n \rightarrow \Diamond\ell)$

Here, ℓ_i, p_i and ℓ are classical literals, which may include $\top$ and $\bot$. Every LTL formula can be rewritten into a conjunction of SNF-clauses. The transformation involves the introduction of new propositional symbols; consequently, in order to use the calculus for hypothesis generation, the explanations extracted from a derivation should be re-converted back to formulae in the original language, and this may not be straightforward.

Assume, however, that the modal literals in the clauses of the background theory T and the negation of the *explanandum* have no nesting of the $\bigcirc$ operator. In this case, the only transformation needed to obtain SNF-clauses would be rewriting formulae of the form $\square D$, where D is a disjunction of classical literals (without any literal of the form $\bigcirc\ell$) into the conjunction of the two SNF-clauses $\square(\mathbf{start} \rightarrow D)$ and $\square(\top \rightarrow \bigcirc D)$. In particular, if the *explanandum* has the form $\Diamond(\ell_1 \wedge \cdots \wedge \ell_n)$, its negation is rewritten as the conjunction of the two SNF-clauses $\square(\mathbf{start} \rightarrow \overline{\ell_1} \vee \cdots \vee \overline{\ell_n})$ and $\square(\top \rightarrow \bigcirc(\overline{\ell_1} \vee \cdots \vee \overline{\ell_n}))$.

In the hypothesis generation problems considered in this work, the set of SNF-clauses to be considered are only initial and step clauses, since sometime clauses are absent. Therefore, it is sufficient to consider the so-called *step resolution rules* of the resolution calculus given by Fisher et al., i.e.:

$$\frac{\square(\mathbf{start} \rightarrow D_1 \vee p) \quad \square(\mathbf{start} \rightarrow D_2 \vee \neg p)}{\square(\mathbf{start} \rightarrow D_1 \vee D_2)}$$

$$\frac{\square(C_1 \rightarrow \bigcirc(D_1 \vee p)) \quad \square(C_2 \rightarrow \bigcirc(D_2 \vee \neg p))}{\square(C_1 \wedge C_2 \rightarrow \bigcirc(D_1 \vee D_2))}$$

where each C_i is a conjunction of literals and each D_i a disjunction of literals (beyond a merge rule that is not worth stating here).

This calculus is not complete for consequence finding. Consider, for instance the set $S = \{\square(q \rightarrow \bigcirc p), \square q\}$. The first formula, $C_1 = \square(q \rightarrow \bigcirc p)$, is an SNF-clause and the second one is rewritten into the conjunction of $C_2 = \square(\mathbf{start} \rightarrow q)$ and $C_3 = \square(\top \rightarrow \bigcirc q)$. From these SNF-clauses, no clause set equivalent to $\square\bigcirc p$ can be derived, although $S \models \square\bigcirc p$ (and, in fact, $S \vdash_{R1-R3} \square\bigcirc p$, by means of an application of the rule R1). This is due to the fact that literals resolved upon always occur in the right-hand side of SNF-clauses, so that the only negative occurrence of q (in C_1) cannot be resolved against any of the two positive occurrences of q (in C_2 and C_3). Other variants of the considered calculus, such as for instance (Konev, Degtyarev, Dixon, Fisher, & Hustadt, 2005), share the same feature, so that their implicational incompleteness can be shown by a similar reasoning.

Beyond general resolution systems for temporal logic, many logic programming approaches have been proposed. The most well known declarative ones are the programming languages Chronolog (firstly presented in (Wadge, 1988)), Templog (Abadi & Manna, 1989) and Gabbay's Temporal Prolog (Gabbay, 1987). The expressive power of Chronolog, however, is quite limited: the only allowed temporal operators are *first* (referring to the initial state) and *next*. Templog is restricted to act on temporal horn clauses, where only a positive literal may occur. Moreover, the system does not handle the induction axiom, for obvious efficiency reasons. Finally, the resolution-based procedure underlying Gabbay's Temporal Prolog lacks a completeness proofs. Though all these works, as well as other temporal logic programming approaches, deserve a deeper analysis, one can hardly expect that they enjoy the implicational completeness property that is essential in the context of the present work.

6. Concluding Remarks

This paper presents a hypothesis generation method for linear temporal logic, when formulae are restricted to clauses in a very simple form. The method is based on an inference system, called FlaTL, whose rules simplify those presented by (Cavalli & Fariñas del Cerro, 1984). The calculus FlaTL is refutationally complete and enjoys a weak form of implicational completeness. In particular, it is implicationally complete w.r.t. initial clauses. Its full implicational completeness is still an open question.

The method proposed in (Alliot et al., 2016) to reason on the LTL theories encoding MIMs relies on the encoding of LTL formulae into classical propositional ones, under a bounded time assumption: the truth of an atom p at each given time point n (up to the maximal time bound) is encoded by a distinguished classical atom p_n, and the behaviour of modal operators is simulated by finite conjunctions and disjunctions. This approach implies that, when the time bound is increased, the whole theory has to be encoded from scratch. Although in practice also our approach may be forced to consider a time bound in order to avoid infinite derivations (for instance, by limiting the maximal length of sequences of the *next* operator dominating classical literals), such a bound would not affect the theory itself, but only the inference mechanism.

Different resolution methods for LTL have been defined in the literature. The reasons for not chosing one of them, instead of the calculus defined in (Cavalli & Fariñas del Cerro, 1984), have been explained in Section 5. However, they still deserve a careful analysis to see whether new insights can be gained in order to refine the inference system FlaTL, as well as to devise possibly different implicational complete inference systems for flat clauses.

As far as future work is concerned, the problem of the full implicational completeness of the resolution calculus FlaTL defined in this paper should be addressed. Moreover, possible refinements of FlaTL can be studied, based on implicational complete resolution strategies. Such strategies can be identified by exploiting Theorem 3 and corresponding results in classical logic (for instance, (Demolombe & Fariñas del Cerro, 1991; Inoue, 1992; Slagle, Chang, & Lee, 1969)), as well as the fact that the derivation of any relevant consequence (whose negation is a relevant explanation) must make use of the negation of the *explanandum*.

Acknowledgments. The authors wish to thank Luis Fariñas del Cerro for having inspired this work, and the organizers of the 3rd Workshop on Logical Reasoning and Computation held in Toulouse in March 2016.

References

Abadi, M. (1989). The power of temporal proofs. *Theoretical Computer Science*, *65*(1), 35–83.

Abadi, M., & Manna, Z. (1985). Nonclausal temporal deduction. In *Proceedings of the conference on logic of programs* (pp. 1–15). Springer-Verlag.

Abadi, M., & Manna, Z. (1989). Temporal logic programming. *Journal of Symbolic Computation*, *8*, 277–295.

Alliot, J.-M., Demolombe, R., Diéguez, M., Fariñas del Cerro, L., Favre, G., Faye, J.-C., & Obeid, N. (2016). Temporal logic modeling of biological systems. In S. Akama (Ed.), *Towards paraconsistent engineering* (pp. 205–226). Springer.

Bouzid, M., & Ligeza, A. (2000). Temporal causal abduction. *Constraints*, *5*(3), 303–319.

Brusoni, V., Console, L., Terenziani, P., & Dupré, D. T. (1997). An efficient algorithm for temporal abduction. In *Advances in artificial intelligence, proc. of the 5th congress of the italian association for artificial intelligence (ai*ia 97)* (pp. 195–206). Springer.

Cavalli, A. R., & Fariñas del Cerro, L. (1984). A decision method for linear temporal logic. In *7th international conference on automated deduction* (Vol. 170, pp. 113–127).

Cialdea Mayer, M., & Pirri, F. (1993). First order abduction via tableau and sequent calculi. *Bulletin of the IGPL*, *1*, 99–117.

Cialdea Mayer, M., & Pirri, F. (1995). Propositional abduction in modal logic. *Journal of the IGPL*, *3*(6), 907–919.

Cox, P. T., & Pietrzykowski, T. (1986). Causes for events: their computation and applications. In *Interanational conference on automated deduction (cade-86)* (pp. 608–621).

Demolombe, R., & Fariñas del Cerro, L. (1991). An inference rule for hypothesis generation. In *Proceedings of the 12th international joint conference on artificial intelligence (ijcai-91)* (pp. 152–157). Morgan Kaufmann Publishers Inc.

Demolombe, R., Fariñas del Cerro, L., & Obeid, N. (2013). Automated reasoning in metabolic networks with inhibition. In *Ai*ia: Advances in artificial intelligence. proc. of the xiiith international conference of the italian association for artificial intelligence* (Vol. 8249, pp. 37–47). Springer International Publishing.

Demolombe, R., Fariñas del Cerro, L., & Obeid, N. (2014). A logical model for molecular interaction maps. In *Logical modeling of biological systems* (pp. 93–123). John Wiley & Sons, Inc.

Eshghi, K. (1988). Abductive planning with event calculus. In *Proc. of the international conference on logic programming.* MIT Press.

Fisher, M., Dixon, C., & Peim, M. (2001). Clausal temporal resolution. *ACM Transactions on Computation Logic*, *2*(1), 12–56.

Gabbay, D. (1987). Modal and temporal logic programming. In A. Galton (Ed.), *Temporal logics and their applications* (pp. 197–237). Academic Press Professional, Inc.

Inoue, K. (1992). Linear resolution for consequence finding. *Artificial Intelligence*, *56*, 301–353.

Klarman, S., & Meyer, T. A. (2014). Complexity of temporal query abduction in DL-lite. In *Informal proceedings of the 27th international workshop on description logics (dl 2014)* (pp. 233–244).

Kohn, K. W., Aladjem, M. I., Weinstein, J. N., & Pommier, Y. (2006). Molecular interaction maps of bioregulatory networks: A general rubric for systems biology. *Molecular Biology of the Cell*, *17*(1), 1–13.

Konev, B., Degtyarev, A., Dixon, C., Fisher, M., & Hustadt, U. (2005). Mechanising first-order temporal resolution. *Information and Computation*, *199*(1-2), 55–86.

Kowalski, R., & Sergot, M. (1986). A logic-based calculus of events. *New Generation Computing*, *4*(1), 67–95.

Lee, R. C. T. (1967). *A completeness theorem and a computer program for finding theorems derivable from given axioms* (Unpublished doctoral dissertation). University of California, Berkley.

Obeid, N. (2014). *MIM-logic: A logic for reasoning about molecular interaction maps* (Unpublished doctoral dissertation). Université de Toulouse Paul Sabatier.

Ribeiro, C., & Porto, A. (1994). Abduction in temporal reasoning. In *Temporal logic, proc. of the first international conference (ictl'94)* (pp. 349–364). Springer.

Shanahan, M. (1989). Prediction is deduction but explanation is abduction. In *Proc. of the international joint conference on artificial intelligence (ijcai 89)* (pp. 1055–1060). Morgan Kaufmann.

Slagle, J. R., Chang, C. L., & Lee, R. C. T. (1969). Completeness theorems for semantic resolution in consequence-finding. In *Proceedings of the 1st international joint conference on artificial intelligence (ijcai'69)* (pp. 281–285). Morgan Kaufmann Publishers Inc.

Teijeiro, T., Félix, P., & Presedo, J. (2014). Using temporal abduction for biosignal interpretation: A case study on QRS detection. In *IEEE international conference on healthcare informatics (ICHI 2014)* (pp. 334–339).

Wadge, W. W. (1988). Tense logic programming: a respectable alternative. In *Proc. of the international symposium on lucid and intensional programming* (pp. 26–32).

Appendix A. Proofs

Proof of Theorem 1: If $S \cup \{C\}$ is a set of flat clauses, then: (1) if $S \vdash_{CF} C$, then $S \vdash_{R1-R3} C'$ for some flat clause C' subsuming a clause that is logically equivalent to C; (2) if $S \vdash_{R1-R3} C$, then $S \vdash_{CF} C'$ for some clause C' that is logically equivalent to C.

Proof. In order to ease readability, the Σ-reduction rules of the calculus CF are reported here:

(a) $\Sigma(p, \neg p) \rhd \bot$;
(b) $\Sigma(D_1 \vee D_2, F) \rhd \Sigma(D_1, F) \vee D_2$;
(c) $\Sigma(\bigcirc E, \bigcirc F) \rhd \bigcirc\Sigma(E, F)$;
(d) $\Sigma(\Box E, \nabla F) \rhd \nabla\Sigma(E, F)$ where $\nabla \in \{\Box, \bigcirc\}$;
(e) $\Sigma(\Box E, F) \rhd \Sigma(E, F)$;
(f) $\Sigma(\Box E, F) \rhd \Sigma(\Box\Box E, F)$.

Let $S \cup \{C\}$ be a set of flat clauses.

(1) We first prove that if $S \vdash_{CF} C$, then $S \vdash_{R1-R3} C'$ for some flat clause C' subsuming a clause that is logically equivalent to C. Observe, beforehand, that if $\Sigma(C_1, C_2) \rhd^* C$ for a given clause C, then the sequence of Σ-reduction steps leading to C necessarily ends with an application of the reduction rule (a) to a pair of complementary literals ℓ and ${\sim}\ell$ occurring in C_1 and C_2, respectively. The literals ℓ and ${\sim}\ell$ are called the literals resolved upon in the inference leading from C_1 and C_2 to the simplification of C.

The leading intuition of the proof can be explained as follows. It can be observed, beforehand, that every literal ℓ occurring in a flat clause occurs in one of the disjuncts of the clause, i.e. in a modal literal of the form $\bigcirc^n\ell$. Let us assume that $\Sigma(C_1, C_2)$ has to be reduced focusing on the literals ℓ, occurring in C_1, and ${\sim}\ell$, occurring in C_2. By the above observation, one of the disjuncts in C_1 has the form $\bigcirc^n\ell$ and one of the disjuncts in C_2 has the form $\bigcirc^m{\sim}\ell$. In order to reach the base step (a) of the reductions, reductions of the form (c) have to be applied, in order to factorize the sequences of $\bigcirc$ operators dominating ℓ and ${\sim}\ell$. And the base step $\Sigma(\ell, {\sim}\ell)$ can be reached by use of reductions (c) only when starting from $\Sigma(\bigcirc^k\ell, \bigcirc^k{\sim}\ell)$ for some $k \geq 0$. Intuitively, the literals resolved upon must be brought to the same time point. Therefore, every sequence of Σ-reduction steps leading to a clause ends with a (possibly empty) sequence of reductions of the form (c), applied to a pair of complementary literals dominated by the same number of $\bigcirc$, followed by the base step (a). Furthermore, if C_1 and C_2 are both initial clauses, it must necessarily be $n = m$ (i.e. the literals resolved upon are already at the same time point in the two clauses), because the only reduction rules that can be applied are (a)-(c). On the contrary, if, for instance, $C_1 = \Box(\bigcirc^n\ell \vee D)$ is an always clause, then it may be $n < m$, because reductions (f) may be used to increase the number of the outermost $\Box$ operator, before applying reductions (d). In the proof that follows, we assume that, when reducing $\Sigma(C_1, C_2)$, one of the shortest reduction paths is followed, i.e. that there are no interleaved applications of reductions (e) and (f) leading to the same result. Moreover, possible permutations in the order of application of the reduction rules, as well as different reduction sequences leading to the same result, will be ignored.

Assume now that C_1 and C_2 are resolvable flat clauses and that $R(C_1, C_2)$ is a CF-resolvent of C_1 and C_2, obtained by simplifying a clause C such that $\Sigma(C_1, C_2) \rhd^* C$. The following reasoning shows that a clause subsuming a clause logically equivalent to C can be derived from C_1 and C_2 by use of the rules R1-R3. Three cases are considered.

- C_1 and C_2 are both initial clauses. Then C can be obtained from C_1 and C_2 only by application of the Σ-reduction rules (a)-(c). In particular, rule (b) must be applied until a clause of the form $\Sigma(L, M) \vee D$ is obtained, where L and M are modal literals. Then rule (c) is applied until a clause of the form $\bigcirc^n\Sigma(p, \neg p) \vee D$ is obtained, which finally leads to D, by use of rule (a). Therefore, L and M have the forms $\bigcirc^n p$ and $\bigcirc^n \neg p$, respectively. In other terms, they are complementary modal literals, and the inference rule R3 can be applied to C_1 and C_2 to obtain D.

- C_1 and C_2 are both always clauses. It may be assumed, w.l.o.g., that $C_1 = \Box(\bigcirc^n \ell \vee D_1)$ and $C_2 = \Box(\bigcirc^k \bigcirc^n {\sim}\ell \vee D_2)$, where ℓ and ${\sim}\ell$ are the literals resolved upon. As observed above, the base step (a) of the Σ-reductions can only be reached by reducing $\Sigma(\bigcirc^m \ell, \bigcirc^m {\sim}\ell)$ for some m. Since the number of $\bigcirc$ operators in an argument of Σ cannot increase, it must be $m = n$, i.e. the reductions pass through a reduction of $\Sigma(\bigcirc^n \ell, \bigcirc^n {\sim}\ell)$. In order to bring ℓ and ${\sim}\ell$ to the same time point, the k more $\bigcirc$ operators dominating ${\sim}\ell$ in C_2 must be absorbed by the $\Box$ operator of C_1: reductions of the form (f) must be applied k times to obtain $\Box^k C_1$.

  More precisely, the base step (a) of the Σ-reductions, applied to the given literals, can be reached as follows (where the reduction symbol $\rhd$ is indexed by the applied Σ-reduction rule):

$$\begin{aligned}
\Sigma(C_1, C_2) &\rhd^*_{\mathrm{f}} \Sigma(\Box^k C_1, C_2) = \Sigma(\Box\Box^k(\bigcirc^n \ell \vee D_1), \Box(\bigcirc^k \bigcirc^n {\sim}\ell \vee D_2)) \\
&\rhd_{\mathrm{d}} \Box\Sigma(\Box^k(\bigcirc^n \ell \vee D_1), \bigcirc^k \bigcirc^n {\sim}\ell \vee D_2) \\
&\rhd_{\mathrm{b}} \Box(\Sigma(\Box^k(\bigcirc^n \ell \vee D_1), \bigcirc^k \bigcirc^n {\sim}\ell) \vee D_2) \\
&\rhd^*_{\mathrm{d}} \Box(\bigcirc^k \Sigma(\bigcirc^n \ell \vee D_1, \bigcirc^n {\sim}\ell) \vee D_2) \\
&\rhd_{\mathrm{b}} \Box(\bigcirc^k (\Sigma(\bigcirc^n \ell, \bigcirc^n {\sim}\ell) \vee D_1) \vee D_2) \\
&\rhd^*_{\mathrm{c}} \Box(\bigcirc^k (\bigcirc^n \Sigma(\ell, {\sim}\ell) \vee D_1) \vee D_2) \\
&\rhd_{\mathrm{a}} \Box(\bigcirc^k (\bigcirc^n \bot \vee D_1) \vee D_2)
\end{aligned}$$

  If $D_1 = L_1 \vee \cdots \vee L_m$, by applying the inference rule R1 to C_1 and C_2 the clause $\Box(\bigcirc^k L_1 \vee \cdots \vee \bigcirc^k L_m \vee D_2)$ is obtained, which is logically equivalent to the last CF-clause of the above Σ-reduction steps.

  A different clause can be obtained from $\Sigma(C_1, C_2)$ by dropping the outermost $\Box$ operator from C_2 (and, when $k = 0$, also from C_1; in this case, the first reduction step below uses rule (e)):

$$\begin{aligned}
\Sigma(C_1, C_2) &\rhd^*_{\mathrm{f}} \Sigma(\Box^k(\bigcirc^n \ell \vee D_1), \Box(\bigcirc^k \bigcirc^n {\sim}\ell \vee D_2)) \\
&\rhd_{\mathrm{e}} \Sigma(\Box^k(\bigcirc^n \ell \vee D_1), \bigcirc^k \bigcirc^n {\sim}\ell \vee D_2) \\
&\rhd_{\mathrm{b}} \Sigma(\Box^k(\bigcirc^n \ell \vee D_1), \bigcirc^k \bigcirc^n {\sim}\ell) \vee D_2 \\
&\rhd^*_{\mathrm{d}} \bigcirc^k \Sigma(\bigcirc^n \ell \vee D_1, \bigcirc^n {\sim}\ell) \vee D_2 \\
&\rhd_{\mathrm{b}} \bigcirc^k (\Sigma(\bigcirc^n \ell, \bigcirc^n {\sim}\ell) \vee D_1) \vee D_2 \\
&\rhd^*_{\mathrm{c}} \bigcirc^k (\bigcirc^n \Sigma(\ell, {\sim}\ell) \vee D_1) \vee D_2 \\
&\rhd_{\mathrm{a}} \bigcirc^k (\bigcirc^n \bot \vee D_1) \vee D_2
\end{aligned}$$

  If $D_1 = L_1 \vee \cdots \vee L_m$, the last CF-clause of the above Σ-reduction steps is logically equivalent to $\bigcirc^k L_1 \vee \cdots \vee \bigcirc^k L_m \vee D_2$, and this flat clause is subsumed by $\Box(\bigcirc^k L_1 \vee \cdots \vee \bigcirc^k L_m \vee D_2)$, that can be obtained from C_1 and C_2 by use of the inference rule R1.

- C_1 is an always clause and C_2 an initial one. Assume that the literals resolved upon are ℓ, occurring in C_1, and ${\sim}\ell$, occurring in C_2. Since the two literal resolved upon have to be brought to the same time point, and the Σ-reductions do not allow to increase the number of $\bigcirc$ in an initial clause (with no occurrences of the $\Box$ operator), it must be $C_1 = \Box(\bigcirc^n \ell \vee D_1)$ and $C_2 = \bigcirc^k \bigcirc^n {\sim}\ell \vee D_2$ for some $k \geq 0$. If $k > 0$ then the the base step (a) of the Σ-reductions, applied to the given literals, can be reached as follows:

$$\begin{aligned}\Sigma(C_1, C_2) &\rhd_{\mathrm{f}}^* \Sigma(\Box^k(\bigcirc^n \ell \vee D_1), \bigcirc^k \bigcirc^n {\sim}\ell \vee D_2)\\ &\rhd_{\mathrm{b}} \Sigma(\Box^k(\bigcirc^n \ell \vee D_1), \bigcirc^k \bigcirc^n {\sim}\ell) \vee D_2\\ &\rhd_{\mathrm{d}}^* \bigcirc^k \Sigma(\bigcirc^n \ell \vee D_1, \bigcirc^n {\sim}\ell) \vee D_2\\ &\rhd_{\mathrm{b}} \bigcirc^k(\Sigma(\bigcirc^n \ell, \bigcirc^n {\sim}\ell) \vee D_1) \vee D_2\\ &\rhd_{\mathrm{c}}^* \bigcirc^k(\bigcirc^n \Sigma(\ell, {\sim}\ell) \vee D_1) \vee D_2\\ &\rhd_{\mathrm{a}} \bigcirc^k(\bigcirc^n \bot \vee D_1) \vee D_2\end{aligned}$$

If $k = 0$, the first reduction step in the above sequence is by rule (e). If $D_1 = L_1 \vee \cdots \vee L_m$, by applying the inference rule R2 to C_1 and C_2, the clause $\bigcirc^k L_1 \vee \cdots \vee \bigcirc^k L_m \vee D_2$ is obtained, which is logically equivalent to the last clause of the above Σ-reductions.

In this case, no different clause can be obtained from $\Sigma(C_1, C_2)$ with the chosen literals to be resolved upon.

(2) For the other direction, let us assume that $C_1, C_2 \vdash_{R1-R3} C$. The reduction steps shown in the three cases considered above show that $C_1, C_2 \vdash_{CF} C'$ for some CF-clause C' logically equivalent to C. □

Proof of Corollary 1: The inference system consisting of the rules R1-R3 is sound and refutationally complete for flat clauses.

Proof. If $S \vdash_{R1-R3} C$, then by Theorem 1 $S \vdash_{CF} C$. Since CF is sound, $S \models C$, hence the system consisting of the rules R1-R3 is sound.

Since CF is refutationally complete, if S is unsatisfiable, then $S \vdash_{CF} \bot$. Therefore, if $S \cup \{C\}$ is a set of flat clauses, $S \vdash_{R1-R3} \bot$ follows from Theorem 1 and the fact that $\bot$ can be derived from any contradictory flat clause by use of the simplification rules. □

Proof of Theorem 2: A flat clause C subsumes a flat clause C' if and only if $\tau(C)$ (classically) subsumes $\tau(C')$.

Proof. Three cases are considered, according to the forms of C and C'. We consider here only the case when C and C' are both always clauses, the others being similar. In this case, if a flat clause C is subsumed by C', then they have the forms:

$$\begin{aligned}C &= \Box(\bigcirc^m \bigcirc^{p_1} \ell_1 \vee \cdots \vee \bigcirc^m \bigcirc^{p_k} \ell_k \vee M_1 \vee \cdots \vee M_n)\\ C' &= \Box(\bigcirc^{p_1} \ell_1 \vee \cdots \vee \bigcirc^{p_k} \ell_k)\end{aligned} \tag{A1}$$

where $m, p_1, \dots, p_k \geq 0$ and $\ell_1, \dots, \ell_k$ are classical.

In this case, $\tau(C)$ and $\tau(C')$ have the forms

$$\begin{aligned}\tau(C) &= \ell_1(f^m(f^{p_1}(x))) \vee \cdots \vee \ell_k(f^m(f^{p_k}(x))) \vee D\\ &= \ell_1(f^{p_1}(f^m(x))) \vee \cdots \vee \ell_k(f^{p_k}(f^m(x))) \vee D\\ \tau(C') &= \ell_1(f^{p_1}(y)) \vee \cdots \vee \ell_k(f^{p_k}(y))\end{aligned} \tag{A2}$$

where $f^n(t)$ stands for the term $f(f(\dots(f(t))))$ with n applications of the functional symbol f to the term t. If $\theta = \{f^m(x)/y\}$, then $\tau(C')\theta \subseteq \tau(C)$, therefore $\tau(C')$ classically subsumes $\tau(C)$.

For the converse, assume that $\tau(C')$ classically subsumes $\tau(C)$, i.e., $\tau(C')\theta \subseteq \tau(C)$ for some substitution θ. Since C and C' are always clauses, there is a single variable x occurring in C and a single variable y occurring in C'; moreover, every literal in $\tau(C)$ contains x and every literal in $\tau(C)$ contains y. Therefore, $\theta = \{f^m(x)/y\}$ for some m, and $\tau(C)$ and $\tau(C')$ have the forms shown in (A2). As a consequence C and C' have the forms given in (A1), i.e., C is subsumed by C'. □

Proof of Theorem 3: If $C_1, \dots, C_n, C$ are flat clauses, then $C_1, \dots, C_n \vdash_{R1-R3} C$ if and only if $\tau(C_1), \dots, \tau(C_n) \vdash_{FOL} \tau(C)$.

Proof. The proof is by induction on the length of the derivations. The base case is obvious in both directions. For the induction step, it must be proved that C is derivable from C_1 and C_2 by one of the rules R1-R3 if and only if $\tau(C)$ is a classical resolvent of $\tau(C_1)$ and $\tau(C_2)$.

(⇒) Three cases are considered, according to the applied rule. We only show here the treatment of the rule R1, the others being similar. In this case:

$$\begin{aligned} C_1 &= \Box(\bigcirc^p \ell \vee \bigcirc^{p_1} \mathcal{p}_1 \vee \cdots \vee \bigcirc^{p_n} \mathcal{p}_n) \\ C_2 &= \Box(\bigcirc^k \bigcirc^p {\sim}\ell \vee \bigcirc^{q_1} \mathcal{q}_1 \vee \cdots \vee \bigcirc^{q_m} \mathcal{q}_m) \\ C &= \Box(\bigcirc^k \bigcirc^{p_1} \mathcal{p}_1 \vee \cdots \vee \bigcirc^k \bigcirc^{p_n} \mathcal{p}_n \vee \bigcirc^{q_1} \mathcal{q}_1 \vee \cdots \vee \bigcirc^{q_m} \mathcal{q}_m) \end{aligned}$$

(where $\ell, \mathcal{p}_1, \dots, \mathcal{p}_n, \mathcal{q}_1, \dots, \mathcal{q}_m$ are classical literals) and

$$\begin{aligned} \tau(C_1) &= \ell(f^p(x)) \vee \mathcal{p}_1(f^{p_1}(x)) \vee \cdots \vee \mathcal{p}_n(f^{p_n}(x))) \\ \tau(C_2) &= {\sim}\ell(f^k(f^p(y))) \vee \mathcal{q}_1(f^{q_1}(y)) \vee \cdots \vee \mathcal{q}_m(f^{q_m}(y)) \\ &= {\sim}\ell(f^p(f^k(y))) \vee \mathcal{q}_1(f^{q_1}(y)) \vee \cdots \vee \mathcal{q}_m(f^{q_m}(y)) \end{aligned}$$

The two classical clauses generate the resolvent

$$\begin{aligned} A &= \mathcal{p}_1(f^{p_1}(f^k(y))) \vee \cdots \vee \mathcal{p}_n(f^{p_n}(f^k(y))) \vee \mathcal{q}_1(f^{q_1}(y)) \vee \cdots \vee \mathcal{q}_m(f^{q_m}(y)) \\ &= \mathcal{p}_1(f^k(f^{p_1}(y))) \vee \cdots \vee \mathcal{p}_n(f^k(f^{p_n}(y))) \vee \mathcal{q}_1(f^{q_1}(y)) \vee \cdots \vee \mathcal{q}_m(f^{q_m}(y)) \end{aligned}$$

by use of the mgu $\theta = \{f^k(y)/x\}$ of $\ell(f^p(x))$ and $\ell(f^p(f^k(y)))$. Since $A = \tau(C)$, we are done.

(⇐) First of all, we observe that factorization can never be applied to a clause $\tau(C)$. In fact, if C is an initial clause, $\tau(C)$ contains no variables, and if C is an always clause, all the literals in $\tau(C)$ contain the same variable, so no subset of its literals can be unified (unless it is a singleton).

So, it is sufficient to show that if $\tau(C)$ is a binary resolvent of $\tau(C_1)$ and $\tau(C_2)$, then $C_1, C_2 \vdash_{R1-R3} C$.

Assume that

$$\frac{\tau(C_1) \quad \tau(C_2)}{\tau(C)}$$

by use of classical resolution. Different cases are considered, according to whether both C_1 and C_2 are always clauses, or one or both of them are initial clauses. We deal here only with the first case, the others being similar.

If both C_1 and C_2 are always clauses, $\tau(C_1)$ and $\tau(C_2)$ have the forms

$$\begin{aligned} \tau(C_1) &= \mathcal{p}_1(f^{p_1}(x)) \vee \cdots \vee \mathcal{p}_n(f^{p_n}(x)) \\ \tau(C_2) &= \mathcal{q}_1(f^{q_1}(y)) \vee \cdots \vee \mathcal{q}_m(f^{q_m}(y)) \end{aligned}$$

where $\mathcal{p}_i$ and $\mathcal{q}_j$ are classical literals. We may assume, w.l.o.g., that $\mathcal{p}_1(f^{p_1}(x))$ and $\mathcal{q}_1(f^{q_1}(y))$ are the complementary literals resolved upon, and that $p_1 \leq q_1$. Consequently, $f^{q_1}(y) = f^{p_1}(f^{q_1-p_1}(y))$ and the m.g.u. of $\mathcal{p}_1(f^{p_1}(x))$ and the complement of $\mathcal{q}_1(f^{q_1}(y))$ is $\theta = \{f^{q_1-p_1}(y)/x\}$. Therefore, if $k = q_1 - p_1$, the binary resolvent of $\tau(C_1)$ and $\tau(C_2)$

is

$$\begin{aligned} A &= \mathscr{p}_2(f^{p_2}(f^k(y))) \vee \dots \mathscr{p}_n(f^{p_n}(f^k(y))) \vee \mathscr{q}_2(f^{q_2}(y)) \vee \dots \mathscr{q}_m(f^{q_m}(y)) \\ &= \mathscr{p}_2(f^k(f^{p_2}(y))) \vee \cdots \vee \mathscr{p}_n(f^k(f^{p_n}(y))) \vee \mathscr{q}_2(f^{q_2}(y)) \vee \cdots \vee \mathscr{q}_m(f^{q_m}(y)) \end{aligned}$$

Let now $C = \Box(\bigcirc^k\bigcirc^{p_2}\mathscr{p}_2 \vee \cdots \vee \bigcirc^k\bigcirc^{p_n}\mathscr{p}_n \vee \bigcirc^{q_1}\mathscr{q}_1 \vee \cdots \vee \bigcirc^{q_m}\mathscr{q}_m)$. Clearly, $A = \tau(C)$ and, since $q_1 = k + p_1$, C is derivable from C_1 and C_2 by application of the rule R1:

$$\frac{\Box(\bigcirc^{p_1}\mathscr{p}_1 \vee \cdots \vee \bigcirc^{p_n}\mathscr{p}_n) \quad \Box(\bigcirc^k\bigcirc^{p_1}\mathscr{q}_1 \vee \bigcirc^{q_2}\mathscr{q}_2 \vee \cdots \vee \bigcirc^{q_m}\mathscr{q}_m)}{\Box(\bigcirc^k\bigcirc^{p_2}\mathscr{p}_2 \vee \cdots \vee \bigcirc^k\bigcirc^{p_n}\mathscr{p}_n \vee \bigcirc^{q_2}\mathscr{q}_2 \vee \cdots \vee \bigcirc^{q_m}\mathscr{q}_m)}$$

□

The proof of Theorem 5 exploits the following intermediate result.

Lemma 1. *Let $C_1, \dots, C_n$ be flat clauses and A a first order clause. If $\tau(C_1), \dots, \tau(C_n) \vdash_{FOL} A$, then there exists a flat clause C such that $A = \tau(C)$.*

Proof. The proof is by induction on the length of the derivation of A from $\tau(C_1), \dots, \tau(C_n)$. If $A = \tau(C_i)$ for some $i = 1, \dots, n$, then the result trivially holds. For the induction step, it must be shown that for any (classical) resolvent A of two clauses $\tau(C_1)$ and $\tau(C_2)$ there exists a flat clause C such that $A = \tau(C)$.

By definition, any flat clause C has the following property:

(α) either (i) $\tau(C)$ is variable-free, or (ii) $\tau(C)$ contains a single variable x which occurs in every literal of $\tau(C)$.

In fact, when C is an initial clause, $\tau(C)$ is variable-free; if C is an always clause, then (ii) holds.

We first show that property α is preserved:

(1) if A is a classical resolvent of $\tau(C_1)$ and $\tau(C_2)$, then A enjoys α.

If either C_1 or C_2 (or both) are initial clauses, then either $\tau(C_1)$ or $\tau(C_2)$ (or both) are variable-free and so are also their resolvents.

Otherwise, let C_1 and C_2 be always clauses, where the variable x occurs in every literal of $\tau(C_1)$, y occurs in every literal of $\tau(C_2)$, and let $\mathscr{p}(f^n(x)) \in \tau(C_1)$ and $\mathscr{q}(f^k(y)) \in \tau(C_2)$ be the literals resolved upon. We may assume, w.l.g., that $n \leq k$. Then the mgu of the two literals is $\{f^m(y)/x\}$ for $m = k - n$. Consequently, y is the only variable occurring in the resolvent A of $\tau(C_1)$ and $\tau(C_2)$, and it occurs in every literal of A.

Then, we show that

(2) if A is a classical clause satisfying property α, then there exists a flat clause C such that $A = \tau(C)$.

If A is variable free, then it has the form

$$A = \ell_1(f^{p_1}(a)) \vee \cdots \vee \ell_n(f^{p_n}(a))$$

If $C = \bigcirc^{p_1}\ell_1 \vee \cdots \vee \bigcirc^{p_n}\ell_n$, then $A = \tau(C)$.

Otherwise, A satisfies (ii), hence it has the form

$$A = \ell_1(f^{p_1}(x)) \vee \cdots \vee \ell_n(f^{p_n}(x))$$

If $C = \Box(\bigcirc^{p_1}\ell_1 \vee \cdots \vee \bigcirc^{p_n}\ell_n)$, then $A = \tau(C)$.

Finally: if A is a resolvent of $\tau(C_1)$ and $\tau(C_2)$, then (1) implies that it satisfies α. Consequently, by (2), there exists a flat clause C such that $A = \tau(C)$. □

Proof of Theorem 5: If $C_1, \dots, C_n, C$ are flat clauses, C is not valid and $\tau(C_1), \dots, \tau(C_n) \vDash_{FOL} \tau(C)$, then there exists a clause C' subsuming C such that $C_1, \dots, C_n \vdash_{R1-R3} C'$.

Proof. If C is not valid, then $\tau(C)$ is not valid either. In fact, since either $\tau(C)$ is ground or it contains a single variable and no constants, it is valid only if it contains an atom and its negation. Such a pair of complementary classical literals corresponds to a a pair of complementary modal literals in C, that is therefore valid too.

Hence, if C is not valid and $\tau(C_1), \dots, \tau(C_n) \vDash_{FOL} \tau(C)$, then by Theorem 4, there exists a clause A subsuming $\tau(C)$ such that $\tau(C_1), \dots, \tau(C_n) \vdash_{FOL} A$. By Lemma 1, there exists a flat clause C' such that $A = \tau(C')$, i.e. $\tau(C_1), \dots, \tau(C_n) \vdash_{FOL} \tau(C')$. Since $\tau(C')$ subsumes $\tau(C)$, by Theorem 2, C' subsumes C. Finally, by Theorem 3, $C_1, \dots, C_n \vdash_{R1-R3} C'$. □

Proof of Theorem 6: If $C_1, \dots, C_n$ are flat clauses, C is a non valid initial clause and $C_1, \dots, C_n \vDash C$, then there exists a clause C' subsuming C such that $C_1, \dots, C_n \vdash_{R1-R3} C'$.

Proof. Let $C_1, \dots, C_n, C$ be flat clauses and $C = L_1 \vee ... \vee L_k$ a non valid initial clause. If $C_1, \dots, C_n \vDash C$, then $C_1, \dots, C_n, \neg C \vDash \bot$, i.e. $C_1, \dots, C_n, {\sim}L_1, \dots, {\sim}L_k \vDash \bot$, since $\neg C \equiv {\sim}L_1 \wedge \cdots \wedge {\sim}L_k$. Then by Corollary 1, $C_1, \dots, C_n, {\sim}L_1, \dots, {\sim}L_k \vdash_{R1-R3} \bot$, and, by Theorem 3: $\tau(C_1), \dots, \tau(C_n), \tau({\sim}L_1), \dots, \tau({\sim}L_k) \vdash_{FOL} \bot$. Since classical resolution is sound, $\tau(C_1), \dots, \tau(C_n), \tau({\sim}L_1), \dots, \tau({\sim}L_k) \vDash_{FOL} \bot$, which amounts to saying that $\tau(C_1), \dots, \tau(C_n), \neg(\neg\tau({\sim}L_1) \vee \cdots \vee \neg\tau({\sim}L_k)) \vDash_{FOL} \bot$, and: $\tau(C_1), \dots, \tau(C_n) \vDash_{FOL} \neg\tau({\sim}L_1) \vee \cdots \vee \neg\tau({\sim}L_k)$. Since clearly $\neg\tau({\sim}L_i)$ is logically equivalent to $\tau(L_i)$, it follows that $\tau(C_1), \dots, \tau(C_n) \vDash_{FOL} \tau(L_1) \vee ... \vee \tau(L_k)$. Moreover $\tau(L_1) \vee ... \vee \tau(L_k) = \tau(L_1 \vee \cdots \vee L_k)$, because every $\tau(L_i)$ is a ground literal. So, finally, by Theorem 5, there exists a clause C' subsuming C such that $C_1, \dots, C_n \vdash_{R1-R3} C'$. □