Towards the Implementation of First-Order Temporal Resolution: the Expanding Domain Case

Boris Konev[1]*, Anatoli Degtyarev[2], Clare Dixon[1], Michael Fisher[1], and Ullrich Hustadt[1]

[1] Department of Computer Science, University of Liverpool, Liverpool L69 7ZF, U.K.
{B.Konev, C.Dixon, M.Fisher, U.Hustadt}@csc.liv.ac.uk
[2] Department of Computer Science, King's College London, Strand, London WC2R 2LS, U.K.
anatoli@dcs.kcl.ac.uk

Abstract

First-order temporal logic is a concise and powerful notation, with many potential applications in both Computer Science and Artificial Intelligence. While the full logic is highly complex, recent work on monodic first-order temporal logics has identified important enumerable and even decidable fragments. In this paper, we develop a clausal resolution method for the monodic fragment of first-order temporal logic over expanding domains. We first define a normal form for monodic formulae and show how arbitrary monodic formulae can be translated into the normal form, while preserving satisfiability. We then introduce novel resolution calculi that can be applied to formulae in this normal form and state correctness and completeness results for the method. We illustrate the method on a comprehensive example. The method is based on classical first-order resolution and can, thus, be efficiently implemented.

1 Introduction

In its propositional form, linear, discrete *temporal logic* has been widely used in the formal specification and verification of reactive systems [17, 14, 11]. Although recognised a powerful formalism, *first-order* temporal logic has generally been avoided due to complexity problems (e.g. there is no finite axiom system for general first-order temporal logic). However, recent work by Hodkinson *et al.* [10] has showed that a particular fragment of first-order temporal logic, termed the *monodic* fragment, has completeness (sometimes even decidability) properties. This breakthrough has led to considerable research activity examining the monodic fragment, in terms of decidable classes, extensions, applications and mechanisation, etc.

Concerning the mechanisation of monodic temporal logics, general tableau and resolution calculi have already been defined, in [12] and [4, 3], respectively. However, neither of these is particularly practical: the tableau method requires representation of *all* possible first-order models, while the resolution method requires the maximal combination of *all* temporal clauses. In this paper, we focus on an important subclass of temporal models, having a wide range of applications, for example in spatio-temporal logics [19, 9] and temporal description logics [1], namely those models that have *expanding domains*. In such models, the domains over which first-order terms range can increase at each temporal step. The focus on this class of models allows us to produce a simplified clausal resolution calculus, termed a *fine-grained* calculus, which is more amenable to efficient implementation.

Thus, we will define the expanding domain monodic fragment, the fine-grained resolution calculus, and provide completeness results for the fine-grained calculus relative to the completeness of the general resolution calculus [5]. A number of examples will be given, showing how the fine-grained calculus works in practice and, finally, conclusions and future work will be provided.

*On leave from Steklov Institute of Mathematics at St.Petersburg

2 First-Order Temporal Logic

First-Order (discrete linear time) Temporal Logic, FOTL, is an extension of classical first-order logic with operators that deal with a linear and discrete model of time (isomorphic to $\mathbb{N}$, and the most commonly used model of time). The first-order temporal language is constructed in a standard way [7, 10] from: *predicate symbols* $P_0, P_1, \ldots$ each of which is of some fixed arity (null-ary predicate symbols are called *propositions*); *individual variables* $x_0, x_1, \ldots$; *individual constants* $c_0, c_1, \ldots$; *booleans operators* $\wedge$, $\neg$, $\vee$, $\Rightarrow$, $\equiv$, **true** ('true'), **false** ('false'); together with *temporal operators* $\Box$ ('always in the future'), $\Diamond$ ('sometime in the future'), $\bigcirc$ ('at the next moment'), U (until), and W (weak until). There are no function symbols or equality in this FOTL language, but it does contain constants. For a given formula, ϕ, $const(\phi)$ denotes the set of constants occurring in ϕ. We write $\phi(x)$ to indicate that $\phi(x)$ *has at most one free variable* x (if not explicitly stated otherwise).

We consider temporal models over *expanding domains*, and formulae in FOTL are interpreted in *first-order temporal structures* of the form $\mathfrak{M} = \langle D_n, I_n \rangle$, $n \in \mathbb{N}$, where every D_n is a non-empty set such that whenever $n < m$, $D_n \subseteq D_m$, and I_n is an interpretation of predicate and constant symbols over D_n. We require that the interpretation of constants is *rigid*. Thus, for every constant c, and all moments of time $i, j \geq 0$, we have $I_i(c) = I_j(c)$.

A *(variable) assignment* $\mathfrak{a}$ is a function from the set of individual variables to $\cup_{n \in \mathbb{N}} D_n$. (This definition implies that variable assignments are rigid as well.) We denote the set of all assignments by $\mathfrak{V}$.

For every moment of time n, there is a corresponding *first-order* structure, $\mathfrak{M}_n = \langle D_n, I_n \rangle$; the corresponding set of variable assignments $\mathfrak{V}_n$ is a subset of the set of all assignments, $\mathfrak{V}_n = \{\mathfrak{a} \in \mathfrak{V} \mid \mathfrak{a}(x) \in D_n$ for every variable $x\}$; clearly, $\mathfrak{V}_n \subseteq \mathfrak{V}_m$ if $n < m$. Intuitively, FOTL formulae are interpreted in sequences of *worlds*, $\mathfrak{M}_0, \mathfrak{M}_1, \ldots$ with truth values in different worlds being connected via temporal operators.

The *truth* relation $\mathfrak{M}_n \models^{\mathfrak{a}} \phi$ in a structure $\mathfrak{M}$, *only for those assignments* $\mathfrak{a}$ *that satisfy the condition* $\mathfrak{a} \in \mathfrak{V}_n$, is defined inductively in the usual way under the following understanding of temporal operators:

$$
\begin{array}{lll}
\mathfrak{M}_n \models^{\mathfrak{a}} \bigcirc\phi & \text{iff} & \mathfrak{M}_{n+1} \models^{\mathfrak{a}} \phi; \\
\mathfrak{M}_n \models^{\mathfrak{a}} \Diamond\phi & \text{iff} & \text{there exists } m > n \text{ such that } \mathfrak{M}_m \models^{\mathfrak{a}} \phi; \\
\mathfrak{M}_n \models^{\mathfrak{a}} \Box\phi & \text{iff} & \text{for all } m \geq n,\ \mathfrak{M}_m \models^{\mathfrak{a}} \phi; \\
\mathfrak{M}_n \models^{\mathfrak{a}} (\phi \mathsf{U} \psi) & \text{iff} & \text{there exists } m \geq n, \text{ such that } \mathfrak{M}_m \models^{\mathfrak{a}} \psi, \\
 & & \quad \text{and for all } i \in \mathbb{N},\ n \leq i < m \text{ implies } \mathfrak{M}_m \models^{\mathfrak{a}} \phi; \\
\mathfrak{M}_n \models^{\mathfrak{a}} (\phi \mathsf{W} \psi) & \text{iff} & \mathfrak{M}_n \models^{\mathfrak{a}} (\phi \mathsf{U} \psi) \text{ or } \mathfrak{M}_n \models^{\mathfrak{a}} \Box\phi.
\end{array}
$$

$\mathfrak{M}$ is a *model* for a formula ϕ (or ϕ is *true* in $\mathfrak{M}$) if there exists an assignment $\mathfrak{a}$ such that $\mathfrak{M}_0 \models^{\mathfrak{a}} \phi$. A formula is *satisfiable* if it has a model. A formula is *valid* if it is satisfiable in any temporal structure under any assignment.

This logic is complex. It is known that even "small" fragments of FOTL, such as the *two-variable monadic* fragment (all predicates are unary), are not recursively enumerable [15, 10]. However, the set of valid *monodic* formulae is known to be finitely axiomatisable [18] (an FOTL-formula ϕ is called monodic if any subformulae of the form $\mathcal{T}\psi$, where $\mathcal{T}$ is one of $\bigcirc$, $\Box$, $\Diamond$ (or $\psi_1 \mathcal{T} \psi_2$, where $\mathcal{T}$ is one of U, W), contains at most one free variable.

3 Divided Separated Normal Form (DSNF)

A *temporal step clause* is a formula either of the form $p \Rightarrow \bigcirc l$, where p is a proposition and l is a propositional literal, or $(P(x) \Rightarrow \bigcirc M(x))$, where P is a unary predicate and M is a unary literal. We call a clause of the the first type an (original) *ground* step clause, and of the second type an (original) *non-ground* step clause.

An *unconditional eventuality monodic temporal problem*, P, consists of:

1. a universal part, $\mathcal{U}$, given by a set of arbitrary closed first-order formulae;
2. an initial part, $\mathcal{I}$, again, a set of arbitrary closed first-order formulae;
3. a step part, $\mathcal{S}$, given by a set of original (ground and non-ground) temporal step clauses; and
4. an eventuality part, $\mathcal{E}$, given by a set of unconditional eventuality clauses of the form $\Diamond L(x)$ (a *non-ground* eventuality clause) and $\Diamond l$ (a *ground eventuality* clause), where l is a propositional literal and $L(x)$ is a unary non-ground literal.

Note that, in a monodic temporal problem, we do not allow two different temporal step clauses with the same left-hand sides. In what follows we will not distinguish between a finite set of formulae $\mathcal{X}$ and the conjunction $\bigwedge \mathcal{X}$ of formulae within the set. With each unconditional eventuality monodic temporal problem, we associate the formula

$$\mathcal{I} \wedge \Box\mathcal{U} \wedge \Box\forall x\mathcal{S} \wedge \Box\forall x\mathcal{E}.$$

Now, when we talk about particular properties of temporal problems (e.g., satisfiability, validity, logical consequences etc) we mean properties of the associated formula.

Any monodic first-order temporal formula can be transformed into a satisfiability-equivalent unconditional eventuality monodic temporal problem. We present the transformation as a two stage reduction.

Reduction to conditional DSNF. We first give a reduction from monodic FOTL to a normal form where, in addition to the parts above, *conditional* eventuality clauses of the form

$$P(x) \Rightarrow \Diamond L(x) \text{ and } p \Rightarrow \Diamond l$$

are allowed. The reduction is based on using a renaming technique to substitute non-atomic subformulae and replacing temporal operators by their fixed point definitions described e.g. in [8]. The translation can be described as a number of steps.

1. Translate a given monodic formula to negation normal form. (To assist understanding of the translation, we list here some equivalent FOTL formulae.)

   $$\forall x(\neg(\phi(x)\,\mathsf{U}\,\psi(x)) \equiv \neg\psi(x)\,\mathsf{W}\,(\neg\phi(x) \wedge \psi(x)))); \quad \forall x(\neg(\phi(x)\,\mathsf{W}\,\psi(x)) \equiv \neg\psi(x)\,\mathsf{U}\,(\neg\phi(x) \wedge \psi(x))).$$

2. Recursively rename innermost temporal subformulae, $\bigcirc\phi(x)$, $\Diamond\phi(x)$, $\Box\phi(x)$, $\phi(x)\,\mathsf{U}\,\psi(x)$, $\phi(x)\,\mathsf{W}\,\psi(x)$ by a new unary predicate $P(x)$. Renaming introduces formulae defining $P(x)$ of the following form:

   (a) $\Box\forall x(P(x) \Rightarrow \bigcirc\phi(x))$; (c) $\Box\forall x(P(x) \Rightarrow \Box\phi(x))$; (e) $\Box\forall x(P(x) \Rightarrow \phi(x)\,\mathsf{W}\,\psi(x))$.
   (b) $\Box\forall x(P(x) \Rightarrow \Diamond\phi(x))$; (d) $\Box\forall x(P(x) \Rightarrow \phi(x)\,\mathsf{U}\,\psi(x))$;

   Formulae of the form (a) and (b) are in the normal form[1], formulae of the form (c) and (d) require extra reduction by removing the temporal operators using their fixed point definitions; formulae of the last kind can be reduced by the semantics of the W operator.

[1] Possibly, after (first-order) renaming the complex expression $\phi(x)$; the formulae introduced by renaming are put in the universal part. This kind of first-order renaming is used implicitly further in this section.

3. Use fixed point definitions
   $\Box\forall x(P(x) \Rightarrow \Box\phi(x))$ is satisfiability equivalent to

$$\Box\forall x(P(x) \Rightarrow R(x)) \wedge \Box\forall x(R(x) \Rightarrow \bigcirc R(x)) \wedge \Box\forall x(R(x) \Rightarrow \phi(x)),$$

   and $\Box\forall x(P(x) \Rightarrow (\phi(x) \,\mathsf{U}\, \psi(x)))$ is equivalent (w.r.t. satisfiability) to

$$\begin{array}{l}\Box\forall x(P(x) \Rightarrow \Diamond\psi(x)) \wedge \Box\forall x(P(x) \Rightarrow \phi(x) \vee \psi(x)) \wedge \\ \Box\forall x(P(x) \Rightarrow S(x) \vee \psi(x)) \wedge \Box\forall x(S(x) \Rightarrow \bigcirc(\phi(x) \vee \psi(x))) \wedge \\ \Box\forall x(S(x) \Rightarrow \bigcirc(S(x) \vee \psi(x))),\end{array}$$

   where $R(x)$ and $S(x)$ are new unary predicates.

Conditional problems to unconditional problems. In the second stage, we replace any formula $\Box\forall x(P(x) \Rightarrow \Diamond L(x))$ by

$$\Box\forall x\,(((P(x) \wedge \neg L(x)) \Rightarrow \mathit{waitforL}(x)) \wedge ((\mathit{waitforL}(x) \wedge \bigcirc\neg L(x)) \Rightarrow \bigcirc\mathit{waitforL}(x)) \wedge \Diamond\neg\mathit{waitforL}(x))$$

where $\mathit{waitforL}(x)$ is a new unary predicate. The proof of satisfiability equivalence of these two expressions can be found in [4].

The described transformations yield the following theorem.

Theorem 1 (Transformation) Every monodic first-order temporal formula can be reduced, in a satisfiability equivalence preserving way, to an unconditional eventuality monodic temporal problem with at most a linear increase in the size of the problem.

Example 1 Let us consider the temporal formula $\exists x\Box\Diamond\forall y\forall z\exists u\Phi(x,y,z,u)$ where $\Phi(x,y,z,u)$ does not contain temporal operators. First, we rename the innermost temporal subformula by a new predicate,

$$\exists x\Box P_1(x) \wedge \Box\forall x[P_1(x) \Rightarrow \Diamond\forall y\forall z\exists u\Phi(x,y,z,u)].$$

Now, we rename the first '$\Box$'-formula and the formula under the '$\Diamond$' operator,

$$\exists x P_3(x) \wedge \Box\forall x[P_1(x) \Rightarrow \Diamond P_2(x)] \wedge \Box\forall x[P_2(x) \Rightarrow \forall y\forall z\exists u\Phi(x,y,z,u)] \wedge \Box\forall x[P_3(x) \Rightarrow \Box P_1(x)],$$

"unwind" (i.e. replace using its fixed point definition) the '$\Box$' operator

$$\begin{array}{rcl}\exists x P_3(x) & \wedge & \Box\forall x[P_1(x) \Rightarrow \Diamond P_2(x)] \wedge \Box\forall x[P_2(x) \Rightarrow \forall y\forall z\exists u\Phi(x,y,z,u)] \\ & \wedge & \Box\forall x[P_3(x) \Rightarrow P_4(x)] \wedge \Box\forall x[P_4(x) \Rightarrow \bigcirc P_4(x)] \wedge \Box\forall x[P_4(x) \Rightarrow P_1(x)],\end{array}$$

and, finally, remove the condition on the eventuality

$$\begin{array}{rcl}\exists x P_3(x) & \wedge & \Box\forall x[P_2(x) \Rightarrow \forall y\forall z\exists u\Phi(x,y,z,u)] \wedge \Box\forall x[P_3(x) \Rightarrow P_4(x)] \wedge \Box\forall x[P_4(x) \Rightarrow \bigcirc P_4(x)] \\ & \wedge & \Box\forall x[P_4(x) \Rightarrow P_1(x)] \wedge \Box\forall x[(P_1(x) \wedge \neg P_2(x)) \Rightarrow \mathit{waitforP}_2(x)] \wedge \Box\forall x\Diamond\neg\mathit{waitforP}_2(x) \\ & \wedge & \Box\forall x[(\mathit{waitforP}_2(x) \wedge \bigcirc\neg P_2(x)) \Rightarrow \bigcirc\mathit{waitforP}_2(x)].\end{array}$$

The parts of this formula form the following monodic temporal problem (we also rename the complex $P_2(x) \vee \mathit{waitforP}_2(x)$ expression by $P_5(x)$):

$$I = \{\exists x P_3(x)\}, \quad \mathcal{U} = \left\{\begin{array}{l}\forall x(P_2(x) \Rightarrow \forall y\forall z\exists u\Phi(x,y,z,u)),\ \forall x(P_3(x) \Rightarrow P_4(x)),\ \forall x(P_4(x) \Rightarrow P_1(x)), \\ \forall x((P_1(x) \wedge \neg P_2(x)) \Rightarrow \mathit{waitforP}_2(x)),\ \forall x(P_5(x) \Rightarrow P_2(x) \vee \mathit{waitforP}_2(x))\end{array}\right\},$$

$$\mathcal{S} = \{P_4(x) \Rightarrow \bigcirc P_4(x),\ \mathit{waitforP}_2(x) \Rightarrow \bigcirc P_5(x)\} \quad \mathcal{E} = \{\Diamond\neg\mathit{waitforP}_2(x)\}.$$

4 Completeness calculus

We include here the calculus from the paper on completeness of monodic temporal resolution for the expanding domain case [5] to make the paper self-contained. We use this 'completeness calculus' to show relative completeness of the calculus presented in the next section. More details on the completeness calculus, as well as the proofs of its properties stated below, can be found in [5].

Let P be a monodic temporal problem, and let

$$P_{i_1}(x) \Rightarrow \bigcirc M_{i_1}(x), \ldots, P_{i_k}(x) \Rightarrow \bigcirc M_{i_k}(x)$$

be a subset of the set of its original non-ground step clauses. Then

$$\exists x(P_{i_1}(x) \wedge \ldots \wedge P_{i_k}(x)) \Rightarrow \bigcirc \exists x(M_{i_1}(x) \wedge \ldots \wedge M_{i_k}(x)),$$
$$P_{i_j}(c) \Rightarrow \bigcirc M_{i_j}(c)$$

are *e-derived* step clauses, where $c \in const(\mathsf{P})$. Let $\{\Phi_1 \Rightarrow \bigcirc \Psi_1, \ldots, \Phi_n \Rightarrow \bigcirc \Psi_n\}$ be a set of e-derived step clauses or original *ground* step clauses. Then

$$\bigwedge_{i=1}^{n} \Phi_i \Rightarrow \bigcirc \bigwedge_{i=1}^{n} \Psi_i$$

is called a *merged e-derived step clause*.

Let $\mathcal{A} \Rightarrow \bigcirc \mathcal{B}$ be a merged e-derived step clause, possibly including the *degenerate* clause $\mathbf{true} \Rightarrow \bigcirc \mathbf{true}$, let $P_1(x) \Rightarrow \bigcirc M_1(x), \ldots, P_k(x) \Rightarrow \bigcirc M_k(x)$ be a subset of the original step clauses, and let $\mathcal{A}(x) \leftrightharpoons \mathcal{A} \wedge \bigwedge_{i=1}^{k} P_i(x)$, $\mathcal{B}(x) \leftrightharpoons \mathcal{B} \wedge \bigwedge_{i=1}^{k} M_i(x)$. Then

$$\forall x(\mathcal{A}(x) \Rightarrow \bigcirc \mathcal{B}(x))$$

is called a *full merged step clause*. (In the case $k = 0$, the truth value of the conjunctions is **true**, and the merged step clause is just a merged e-derived step clause.)

Let P be a monodic temporal problem. Now, denote by P^c the set $\mathsf{P} \cup \{\Diamond L(c) \mid \Diamond L(x) \in \mathcal{E}, c \in const(\mathsf{P})\}$. If $\mathsf{P} = \mathsf{P}^c$, then we say that the temporal problem is *constant flooded*[2] (or P is in *constant flooded form*). Evidently, P^c is satisfiability equivalent to P. We apply our resolution procedure to constant flooded temporal problems.

The inference system we use consists of the following inference rules[3]. (Recall that the premises and conclusion of these rules are (implicitly) closed under the $\Box$ operator.)

- *Step resolution rule w.r.t. $\mathcal{U}$*: $\dfrac{\mathcal{A} \Rightarrow \bigcirc \mathcal{B}}{\neg \mathcal{A}}\ (\bigcirc^{\mathcal{U}}_{res})$, where $\mathcal{U} \cup \{\mathcal{B}\} \vdash \perp$.

- *Initial termination rule w.r.t. $\mathcal{U}$:* The contradiction $\perp$ is derived and the derivation is (successfully) terminated if $\mathcal{U} \cup I \vdash \perp$.

- *Eventuality resolution rule w.r.t. $\mathcal{U}$:*

$$\frac{\forall x(\mathcal{A}_1(x) \Rightarrow \bigcirc(\mathcal{B}_1(x))) \ldots \forall x(\mathcal{A}_n(x) \Rightarrow \bigcirc(\mathcal{B}_n(x))) \qquad \Diamond L(x)}{\forall x \bigwedge_{i=1}^{n} \neg \mathcal{A}_i(x)}\ (\Diamond^{\mathcal{U}}_{res}),$$

[2]Strictly speaking, P^c is not in DSNF: we have to rename ground eventualities by propositions. Rather than 'flooding', we could have introduced special inference rules to deal with constants.

[3]In what follows, $\mathcal{A} \Rightarrow \bigcirc \mathcal{B}$ and $\mathcal{A}_i \Rightarrow \bigcirc \mathcal{B}_i$ denote merged e-derived step clauses, $\forall x(\mathcal{A}(x) \Rightarrow \bigcirc(\mathcal{B}(x)))$ and $\forall x(\mathcal{A}_i(x) \Rightarrow \bigcirc(\mathcal{B}_i(x)))$ denote full merged step clauses, and $\mathcal{U}$ denotes the (current) universal part of the problem.

where $\forall x(\mathcal{A}_i(x) \Rightarrow \bigcirc\mathcal{B}_i(x))$ are full merged step clauses such that for all $i \in \{1,\ldots,n\}$, the *loop* side conditions

$$\forall x(\mathcal{U} \wedge \mathcal{B}_i(x) \Rightarrow \neg L(x)) \quad \text{and} \quad \forall x(\mathcal{U} \wedge \mathcal{B}_i(x) \Rightarrow \bigvee_{j=1}^{n} (\mathcal{A}_j(x)))$$

are both valid[4].

The set of merged step clauses, satisfying the loop side conditions, is called a *loop in* $\Diamond L(x)$ and the formula $\bigvee_{j=1}^{n} \mathcal{A}_j(x)$ is called a *loop formula*.

- *Ground eventuality resolution rule w.r.t.* $\mathcal{U}$*:*

$$\frac{\mathcal{A}_1 \Rightarrow \bigcirc\mathcal{B}_1 \quad \ldots \quad \mathcal{A}_n \Rightarrow \bigcirc\mathcal{B}_n \qquad \Diamond l}{\bigwedge_{i=1}^{n} \neg\mathcal{A}_i}\ (\Diamond^{\mathcal{U}}_{res}),$$

where $\mathcal{A}_i \Rightarrow \bigcirc\mathcal{B}_i$ are merged e-derived step clauses such that the *loop* side conditions $\mathcal{U} \wedge \mathcal{B}_i \vdash \neg l$ and $\mathcal{U} \wedge \mathcal{B}_i \vdash \bigvee_{j=1}^{n} \mathcal{A}_j$ for all $i \in \{1,\ldots,n\}$ are satisfied. *Ground loop* and *ground loop formula* are defined similarly to the case above.

Note 1 A *derivation* in the described inference system is a sequence of universal parts, extended bit by bit by the conclusions of the inference rules. Successful termination means that the given problem is unsatisfiable. Note that the I, $\mathcal{S}$ and $\mathcal{E}$ parts of the temporal problem are not changed.

Note 2 The conclusion of an application of all temporal resolution rules is a universally quantified formula.

Theorem 2 (Soundness) The temporal resolution rules preserve satisfiability.

Theorem 3 (Completeness of Temporal Resolution) If P is an unsatisfiable unconditional eventuality constant flooded monodic temporal problem, then there exists a successfully terminating derivation from P.

Example 2 The need of constant flooding can be demonstrated by the following example. None of the rules of temporal resolution can be applied directly to the (unsatisfiable) temporal problem given by

$$\begin{array}{ll} I = \{P(c)\} & \mathcal{S} = \{q \Rightarrow \bigcirc q\} \\ \mathcal{U} = \{q \equiv P(c)\} & \mathcal{E} = \{\Diamond\neg P(x)\}. \end{array}$$

If, however, we add to the problem an eventuality clause $\Diamond l$ and a universal clause $l \Rightarrow \neg P(c)$,

$$q \Rightarrow \bigcirc q$$

will be a loop in l; and the eventuality resolution rule would derive $\neg\mathbf{true}$.

[4] In the case $\mathcal{U} \vdash \forall x \neg L(x)$, the degenerate clause, $\mathbf{true} \Rightarrow \bigcirc\mathbf{true}$, can be considered as a premise of this rule; the conclusion of the rule is then $\neg\mathbf{true}$ and the derivation successfully terminates.

5 Fine grained resolution

The notion of a (full) merged step clause of the completeness calculus is quite involved and the search for appropriate merging of simpler clauses is computationally hard. Finding *sets* of such full merged step clauses needed for the temporal resolution rule is even more difficult. In this section we introduce a calculus where the inference rules are "milled" into small steps more suitable for effective implementation. First, we concentrate on the implementation of the step resolution inference rule; then, we show how to effectively find premises for the eventuality resolution rule by means of step resolution.

The calculus is inspired by the following consideration: Suppose that the completeness calculus applies the step resolution rule to a merged e-derived step clause $\mathcal{A} \Rightarrow \bigcirc\mathcal{B}$. The rule can be applied if $\mathcal{B} \cup \mathcal{U} \vdash \perp$ and this fact can be established by a first-order resolution procedure (that would skolemise the universal part). Then the conclusion of the rule, $\neg\mathcal{A}$, is added to $\mathcal{U}$ resulting in a new universal part $\mathcal{U}'$. Suppose that the step resolution rule is applied to another merged e-derived step clause, $\mathcal{A}' \Rightarrow \bigcirc\mathcal{B}'$. The side condition, $\mathcal{B}' \cup \mathcal{U}' \vdash \perp$, again can be checked by a first-order resolution procedure. Since we never add new existential formulae (Note 2), $\mathcal{U}'$ can be skolemised in exactly the same way as $\mathcal{U}$. Therefore, we can actually keep $\mathcal{U}$ in clausal form.

Note further that we are not only going to check side conditions for the rules of the completeness calculus by means of first-order resolution but also *search for clauses to merge* in the same time.

Fine-grained resolution might generate additional step clauses of the form

$$C \Rightarrow \bigcirc D, \tag{1}$$

where C is a *conjunction* of propositions, unary predicates of the form $P(x)$, and ground formulae of the form $P(c)$, where P is a unary predicate symbol and c is a constant occurring in a *originally given problem*; D is a *disjunction* of arbitrary literals.

Before any inference rules are applied to a given problem, we do some preprocessing.

- Translate a given problem P into constant flooded form.
- For every original non-ground step clause

  $$P(x) \Rightarrow \bigcirc Q(x)$$

  and every constant $c \in const(\mathsf{P})$, add the clause

  $$P(c) \Rightarrow \bigcirc Q(c) \tag{2}$$

  to the set of original step clauses.
- Clausify the universal part as if there is no connection with temporal logic at all. The resulting clauses are called *universal clauses*. Let us denote the set of universal clauses by $\mathcal{U}^s$. Analogously, clausify the initial part. The resulting clauses are called *initial clauses*. We denote the set of initial clauses by I^s. Originally, the sets $\mathcal{U}^s$ and I^s do not have common Skolem constants and functions. Initial and universal clauses are kept separately.

In sections 5.1 and 5.2, we assume that a given problem is preprocessed.

5.1 Fine-grained step resolution

Fine-grained step resolution consists of the following inference rules. We implicitly assume that different premises and conclusion have no variables in common; variables are renamed if necessary.

1. *Arbitrary (first-order) resolution between universal clauses*

2. *Fine-grained (restricted) step resolution*

$$\frac{C_1 \Rightarrow \bigcirc(D_1 \vee L) \quad C_2 \Rightarrow \bigcirc(D_2 \vee \neg M)}{(C_1 \wedge C_2)\sigma \Rightarrow \bigcirc(D_1 \vee D_2)\sigma},$$

   where $C_1 \Rightarrow \bigcirc(D_1 \vee L)$ and $C_2 \Rightarrow \bigcirc(D_2 \vee \neg M)$ are step clauses and σ is an mgu of the literals L and M *such that* σ *does not map variables from* C_1 *or* C_2 *into a constant or a functional term.*[5]

$$\frac{C_1 \Rightarrow \bigcirc(D_1 \vee L) \quad D_2 \vee \neg N}{C_1\sigma \Rightarrow \bigcirc(D_1 \vee D_2)\sigma},$$

   where $C_1 \Rightarrow \bigcirc(D_1 \vee L)$ is an step clause, $D_2 \vee \neg N$ is a universal clause, and σ is an mgu of the literals L and N *such that* σ *does not map variables from* C_1 *into a constant or a functional term.*

3. *Right factor*

$$\frac{C \Rightarrow \bigcirc(D \vee L \vee M)}{C\sigma \Rightarrow \bigcirc(D \vee L)\sigma},$$

   where σ is an mgu of the literals L and M *such that* σ *does not map variables from* C *into a constant or a functional term.*

4. *Left factor*

$$\frac{(C \wedge L \wedge M) \Rightarrow \bigcirc D}{(C \wedge L)\sigma \Rightarrow \bigcirc D\sigma},$$

   where σ is an mgu of the literals L and M[6].

5. *Subsumption by a universal clause*
   a universal clause D_2 *subsumes* a step clause $C_1 \Rightarrow \bigcirc D_1$ if D_2 subsumes D_1 or D_2 subsumes[7] $\neg C_1$.

6. *Subsumption by a step clause*
   a step clause $C_1 \Rightarrow \bigcirc D_1$ *subsumes* a step clause $C_2 \Rightarrow \bigcirc D_2$ if there exists a substitution σ such that $D_1\sigma \subseteq D_2$ and $\neg C_1\sigma \subseteq \neg C_2$.

7. *Clause conversion*
   a step clause of the form $C \Rightarrow \bigcirc$**false** is rewritten into a *universal clause* $\neg C$.

8. *Initial resolution*
   we allow arbitrary inferences between initial and universal clauses, the result is an initial clause. Universal clauses may subsume initial clauses but not vice versa.

We define the notions of a *derivation* by fine-grained resolution and a *saturation procedure* in a way similar to, e.g., [2].

[5] This restriction justifies skolemisation: Skolem constants and functions do not 'sneak' in the left-hand side of step rules, and, hence, Skolem constants from different moments of time do not mix.

[6] We do not need to restrict the unifier σ in this case since the restriction holds automatically.

[7] Here, and further, $\neg(L_1(x) \wedge \ldots \wedge L_k(x))$ abbreviates $(\neg L_1(x) \vee \ldots \vee \neg L_k(x))$.

Note 3 All the terms substituted to the left hand sides are free variables.

Note 4 Fine-grained step resolution without the restriction on substitutions would, certainly, lead to unsoundness: It would prove, e.g., "unsatisfiability" of the monodic problem given by $\mathcal{U} = \{u1 : \exists x \neg Q(x), u2 : \forall x(P(x) \vee Q(x))\}$ and $\mathcal{S} = \{s1 : P(x) \Rightarrow \bigcirc Q(x)\}$. (After skolemisation, $\mathcal{U}^s = \{us1 : \neg Q(c),\ us2 : P(x) \vee Q(x)\}$, then unrestricted resolution would derive $us3 : \neg P(c)$ from $us1$ and $s1$, and then the contradiction from $us1$, $us2$, and $us3$.)

A clause of the form $C \Rightarrow \bigcirc\textbf{false}$, where C is of the same form as in (1), is called *a final clause*.

The proof of the following two lemmas can be found in the appendix.

Lemma 4 Let $C \Rightarrow \bigcirc\textbf{false}$ be an arbitrary final clause derived by fine-grained step resolution. Then there exists a merged e-derived step rule $\mathcal{A} \Rightarrow \bigcirc\mathcal{B}$ such that $\mathcal{B} \cup \mathcal{U} \vdash \perp$ and $\mathcal{A} = \widetilde{\exists} C$, where $\widetilde{\exists}$ means existential quantification over all free variables.

Lemma 5 Let $\mathcal{A} \Rightarrow \bigcirc\mathcal{B}$ be a merged e-derived step rule such that $\mathcal{B} \cup \mathcal{U} \vdash \perp$. Then there exists a derivation by fine-grained step resolution of a final clause $C \Rightarrow \bigcirc\textbf{false}$ such that $\mathcal{A} \Rightarrow \widetilde{\exists} C$.

Lemma 4 ensures soundness of fine-grained step resolution. Lemma 5 says that the conclusion of an application of the clause conversion rule, $\neg C$, subsumes the conclusion of an application of the step resolution rule of the completeness calculus, $\neg\mathcal{A}$.

Theorem 6 The calculus consisting of the rules of fine-grained step resolution, together with the (both ground and non-ground) eventuality resolution rule, is sound and complete for the monodic fragment over expanding domains.

Proof Soundness follows from Lemma 4. The proof of the completeness theorem [5] goes by showing that there exists a conclusion of an application of an inference rule, Φ, leading to the deletion of some nodes from the *eventuality graph* (a graph representing possible temporal models). A node $\mathcal{C}$ is deleted from the graph if the categorical formula, $\mathcal{F}_\mathcal{C}$, together with the universal part, $\mathcal{U}$, is satisfiable, but $\mathcal{F}_\mathcal{C} \wedge \Phi \wedge \mathcal{U}$ is unsatisfiable.

Let $\neg\mathcal{A}$ be a conclusion of an application of the step resolution rule leading to deletion of a node. By Lemma 5 there exists a derivation by fine-grained step resolution of a final clause $C \Rightarrow \bigcirc\textbf{false}$ such that $\mathcal{A} \Rightarrow \widetilde{\exists} C$. Under these conditions, $\mathcal{F}_\mathcal{C} \wedge \widetilde{\forall} \neg C \wedge \mathcal{U}$ is unsatisfiable. ❑

5.2 Loop search

Next we use fine grained step resolution to find the appropriate set of full merged clauses to apply the (ground or non-ground) eventuality resolution rule. It has been noticed in [4] that in order to effectively find a loop in $L(x) \in \mathcal{E}$, given a formula with one free variable $\Phi(x)$ we have to be able to find the set of all full merged clauses of the form $\forall x(\mathcal{A}(x) \Rightarrow \bigcirc\mathcal{B}(x))$ such that the formula

$$\forall x(\mathcal{B}(x) \wedge \mathcal{U} \Rightarrow \Phi(x))$$

is valid (where $\Phi(x) = H(x) \wedge \neg L(x)$ and $H(x)$ is a disjunction of the left-hand sides of some full merged step clauses).

Let $\forall x(\mathcal{A}(x) \Rightarrow \bigcirc\mathcal{B}(x))$ be a full merged step clause such that $\forall x(\mathcal{B}(x) \wedge \mathcal{U} \Rightarrow \Phi(x))$. Note that $\forall x(\mathcal{B}(x) \wedge \mathcal{U} \Rightarrow \Phi(x))$ is valid iff $\exists x(\mathcal{B}(x) \wedge \mathcal{U} \wedge \neg\Phi(x))$ is unsatisfiable. Let c^l be a distinguished constant to be used in loop search that we call the *loop constant*. We assume that the loop constant does not occur in a given problem and is not used for skolemisation.

Function BFS

Input: A set $\mathbf{S}$ of universal and step clauses, saturated by fine-grained resolution and an eventuality clause $\Diamond L(x) \in \mathcal{E}$.

Output: A formula $H(x)$ with at most one free variable.

Method:
1. Let $H_0(x) = \mathbf{true}$; $N_0 = \emptyset$; $i = 0$.
2. Let $\mathbf{S_{i+1}} = \mathrm{LT}(\mathbf{S}) \cup \{\mathbf{true} \Rightarrow \bigcirc(\neg H_i(c^l) \vee L(c^l))\}$. Apply the rules of fine-grained step resolution *except the clause conversion rule* to $\mathbf{S_{i+1}}$. If we obtain a contradiction, then return the loop **true** (in this case $\forall x \neg L(x)$ is implied by the universal part). Otherwise let $N_{i+1} = \{C_j \Rightarrow \bigcirc\mathbf{false}\}_{j=1}^{k}$ be the set of all *new* final clauses from $\mathbf{S_{i+1}}$.
3. If $N_{i+1} = \emptyset$, let $H_{i+1}(x) = \mathbf{false}$; else let $H_{i+1}(x) = \bigvee_{j=1}^{k} C_j\{c^l \to x\}$.
4. If $N_{i+1} = \emptyset$ return **false**.
5. If $\forall x(H_i(x) \Rightarrow H_{i+1}(x))$ return $H_{i+1}(x)$.
6. $i = i+1$; goto 2.

Figure 1: Breadth-first search using fine-grained step resolution.

Let us define a *transformation for loop search* on a set of universal and step clauses $\mathbf{S}$ as follows. $\mathrm{LT}(\mathbf{S})$ is the minimal set of clauses containing $\mathbf{S}$ such that for every *original* non-ground step clause $(P(x) \Rightarrow \bigcirc Q(x)) \in \mathbf{S}$, the set $\mathrm{LT}(\mathbf{S})$ contains the clause

$$P(c^l) \Rightarrow \bigcirc Q(c^l). \qquad (3)$$

We add the clause[8] $\mathbf{true} \Rightarrow \bigcirc\neg\Phi(c^l)$ to $\mathrm{LT}(\mathbf{S})$ and apply the rules of fine-grained step resolution *except the clause conversion rule* to it.

The proof of the following two lemmas can be found in the appendix.

Lemma 7 Let $\mathbf{S}$ be a set of universal and step clauses, and let $C \Rightarrow \bigcirc\mathbf{false}$ be a final clause derived by the rules of fine-grained step resolution *except the clause conversion rule* from $\mathrm{LT}(\mathbf{S}) \cup \{\mathbf{true} \Rightarrow \bigcirc\neg\Phi(c^l)\}$ such that at least one of the clauses originating from $\mathbf{true} \Rightarrow \bigcirc\neg\Phi(c^l)$ is involved in the derivation. Then there exists a full merged (from $\mathbf{S}$) clause $\forall x(\mathcal{A}(x) \Rightarrow \bigcirc\mathcal{B}(x))$ such that the formula $\forall x(B(x) \wedge \mathcal{U} \Rightarrow \Phi(x))$ is valid and $\mathcal{A}(x) = (\widetilde{\exists} C)\{c^l \to x\}$.

Lemma 8 Let $\mathbf{S}$ be a set of universal and step clauses, and let $\forall x(\mathcal{A}(x) \Rightarrow \bigcirc\mathcal{B}(x))$ be a full merged (from $\mathbf{S}$) step clause such that $\forall x(\mathcal{B}(x) \wedge \mathcal{U} \Rightarrow \Phi(x))$. Then there exists a derivation by the rules of fine-grained step resolution *except the clause conversion rule* from $\mathrm{LT}(\mathbf{S})$ of a final clause $C \Rightarrow \bigcirc\mathbf{false}$ such that $\forall x(\mathcal{A}(x) \Rightarrow (\widetilde{\exists} C\{c^l \to x\}))$.

Then the loop search algorithm from [4] can be reformulated as shown in Fig. 1. (This algorithm is essentially based on the BFS algorithm for propositional temporal resolution [6].)

The proof of the following two statements is a direct adaptation of the proofs from [4] taking into consideration arguments similar to those given in the proof of Theorem 6.

Lemma 9 The BFS algorithm terminates provided that all calls of saturation by step resolution terminate. Let $H(x)$ be a formula produced by the BFS algorithm. Then $H(x)$ is a loop formula in $L(x)$.

Theorem 10 Temporal resolution is complete if we restrict ourselves to loops found by the BFS algorithm.

[8] In fact, a set of clauses since $\neg H(x)$, and $\neg\Phi(x)$, is a set of first-order clauses.

5.3 Example

Let us consider a monodic temporal problem P given by $I = \emptyset$, $\mathcal{U} = \{\forall x(B(x) \Rightarrow A(x) \wedge \neg L(x)),\ l \Rightarrow \exists x A(x)\}$, $\mathcal{S} = \{s1 : A(x) \Rightarrow \bigcirc B(x)\}$, $\mathcal{E} = \{e1 : \Diamond L(x),\ e2 : \Diamond l\}$. We especially chose such a trivial example to be able to demonstrate thoroughly the steps of our proof search algorithm.
We clausify $\mathcal{U}$ resulting in $\mathcal{U}^s = \{u1 : (\neg B(x) \vee A(x)),\ u2 : (\neg B(x) \vee \neg L(x)),\ u3 : \neg l \vee A(c)\}$.

- Step resolution
  We can deduce the following clauses by fine-grained step resolution:

  $$s2 : A(x) \Rightarrow \bigcirc A(x) \quad (\, s1,\, u1) \qquad\qquad s3 : A(x) \Rightarrow \bigcirc \neg L(x) \quad (\, s1,\, u2)$$

  The set of clauses is saturated. Now we try finding a loop in $\Diamond L(x)$.
- Loop search
  The set $\mathbf{S} = \{u1, u2, u3, s1, s2, s3\}$; $H_0(x) = \textbf{true}$; $N_0 = \emptyset$; $i = 0$. $\mathrm{LT}(\mathbf{S}) = \{lt1 : A(c^l) \Rightarrow \bigcirc B(c^l)\}$.
  We deduce the following clauses by fine-grained step resolution (except clause conversion) from $\mathbf{S_1} = \mathbf{S} \cup \mathrm{LT}((S)) \cup \{l1 : \textbf{true} \Rightarrow \bigcirc L(c^l)\}$:

  $$\begin{array}{llll} l2 : A(c^l) \Rightarrow \bigcirc A(c^l) & (\, lt1,\, u1) & l4 : \textbf{true} \Rightarrow \bigcirc \neg B(c^l) & (\, u2,\, l1) \\ l3 : A(c^l) \Rightarrow \bigcirc \neg L(c^l) & (\, lt1,\, u2) & l5 : A(c^l) \Rightarrow \bigcirc \textbf{false} & (\, l3,\, l1) \end{array}$$

  The set of clauses is saturated. Then $N_1 = \{A(c^l) \Rightarrow \bigcirc\textbf{false}\}$, $H_1(x) = A(x)$. Obviously, $\forall x(H_0(x) \Rightarrow H_1(x))$ is not true.

  Now the set $\mathbf{S_2} = \mathbf{S} \cup \mathrm{LT}(\mathrm{S}) \cup \{l6 : \textbf{true} \Rightarrow \bigcirc(\neg A(c^l) \vee L(c^l))\}$ and we deduce from it the following:

  $$\begin{array}{llll} l7 : A(c^l) \Rightarrow \bigcirc A(c^l) & (\, lt1,\, u1) & l12 : A(c^l) \Rightarrow \bigcirc \neg A(c^l) & (\, l8,\, l6) \\ l8 : A(c^l) \Rightarrow \bigcirc \neg L(c^l) & (\, lt1,\, u2) & l13 : \textbf{true}(c^l) \Rightarrow \bigcirc \neg B(c^l) & (\, u2,\, l9) \\ l9 : \textbf{true} \Rightarrow \bigcirc(\neg B(c^l) \vee L(c^l)) & (\, u1,\, l6) & l14 : A(c^l) \Rightarrow \bigcirc \neg B(c^l) & (\, l8,\, l9) \\ l10 : \textbf{true} \Rightarrow \bigcirc(\neg B(c^l) \vee \neg A(c^l)) & (\, u2,\, l6) & l15 : A(c^l) \Rightarrow \bigcirc \textbf{false} & (\, l8,\, l11) \\ l11 : A(c^l) \Rightarrow \bigcirc L(c^l) & (\, l7,\, l6) & & \end{array}$$

  The set of clauses is saturated. $N_2 = \{A(c^l) \Rightarrow \bigcirc\textbf{false}\}$, $H_2(x) = A(x)$.
  As $\forall x(H_1(x) \Rightarrow H_2(x))$, the loop is $A(x)$.
- Eventuality resolution
  We can apply now the eventuality resolution rule whose conclusion is

  $$u4 : \neg A(x).$$

- Step resolution

  $$u5 : \neg l \quad (\, u3,\, u4)$$

- Loop search
  $\mathbf{S} = \{u1, u2, u3, u4, u5, s1, s2, s3\}$; $H_0(x) = \textbf{true}$; $N_0 = \emptyset$; $i = 0$; $\mathrm{LT}(\mathbf{S}) = \{lt1 : A(c^l) \Rightarrow \bigcirc B(c^l)\}$; $\mathbf{S_1} = \mathbf{S} \cup \mathrm{LT}(\mathbf{S}) \cup \{l16 : \textbf{true} \Rightarrow \bigcirc l\}$; and we can deduce:

  $$l17 : \textbf{true} \Rightarrow \bigcirc\textbf{false} \quad (\, l16,\, u5)$$

  that is, a contradiction. The loop is **true**.
- Eventuality resolution
  We can apply now the eventuality resolution rule whose conclusion is $\neg\textbf{true}$. The problem is unsatisfiable.

Note 5 As the example shows, the presence of clauses of the form (2), introduced by the preprocessing, and (3), introduced by the transformation for loop search, might lead to repeated derivations (with free variables and with constants). This can be avoided, however, if instead of generating these clauses, we relax the conditions on substitutions in the definition of rules of fine-grain resolution by allowing original constants and the loop constant to be substituted to variables occurring to the left-hand side of a step clause. It can be seen that the set of derived final clauses would be the same.

Taking into consideration this note, we do not use the reduction for loop search, and clauses $l2$, $l3$, $l7$, $l8$ would not be derived. Instead, at the first iteration of BFS on $L(x)$, we would deduce the following clauses from $\mathbf{S_1} = \mathbf{S} \cup \{l1 : \mathbf{true} \Rightarrow \bigcirc L(c^l)\}$:

$$l4' : \mathbf{true} \Rightarrow \bigcirc\neg B(c^l) \quad (u2, l1) \qquad\qquad l5' : A(c^l) \Rightarrow \bigcirc\mathbf{false} \quad (s3, l1);$$

and at the second iteration from $\mathbf{S_2} = \mathbf{S} \cup \mathrm{LT(S)} \cup \{l6 : \mathbf{true} \Rightarrow \bigcirc(\neg A(c^l) \vee L(c^l))\}$:

$$\begin{array}{lll@{\qquad}ll}
l9' : \mathbf{true} \Rightarrow \bigcirc(\neg B(c^l) \vee L(c^l)) & (u1, l6) & & l13' : \mathbf{true}(c^l) \Rightarrow \bigcirc\neg B(c^l) & (u2, l9') \\
l10' : \mathbf{true} \Rightarrow \bigcirc(\neg B(c^l) \vee \neg A(c^l)) & (u2, l6) & & l14' : A(c^l) \Rightarrow \bigcirc\neg B(c^l) & (s3, l9') \\
l11' : A(c^l) \Rightarrow \bigcirc L(c^l) & (s2, l6) & & l15' : A(c^l) \Rightarrow \bigcirc\mathbf{false} & (s3, l11'). \\
l12' : A(c^l) \Rightarrow \bigcirc\neg A(c^l) & (s3, l6) & & &
\end{array}$$

6 Conclusion

We have described a fine-grained resolution calculus for monodic first order temporal logics over expanding domains. Soundness of the fine grained inference steps is easy to prove and completeness is shown relative to the completeness proof for the expanding domain for the non-fine grained version [5]. While implementation based on the general calculus would involve generating all subsets of the step clauses with which to apply the step and eventuality resolution rules, the fine-grained resolution inference rules can be implemented directly using any appropriate first-order theorem prover for classical logics. This makes the new calculus presented here particularly amenable to efficient implementation.

As part of our future work, we will examine extension of this approach to the case of temporal models with constant domains. We also aim to implement and test the calculus defined here.

Finally, we wish to acknowledge support for this work from EPSRC via research grant GR/R45376/01.

References

[1] A. Artale, E. Franconi, F. Wolter, and M. Zakharyaschev. A temporal description logic for reasoning over conceptual schemas and queries. In *Proceedings of JELIA'02*, volume 2424 of *Lecture Notes in Computer Science*, pages 98–110. Springer, 2002.

[2] L. Bachmair and H. Ganzinger. Resolution theorem proving. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, chapter 2, pages 19–99. Elsevier, 2001.

[3] A. Degtyarev and M. Fisher. Towards first-order temporal resolution. In *KI 2001, Proceedings*, volume 2174 of *Lecture Notes in Computer Science*, pages 18–32. Springer, 2001.

[4] A. Degtyarev, M. Fisher, and B. Konev. Monodic temporal resolution. Submitted. Available as Technical report ULCS-03-001 from `http://www.csc.liv.ac.uk/research/techreports`, 2003.

[5] A. Degtyarev, M. Fisher, and B. Konev. Monodic temporal resolution: the expanding domain case. Technical Report ULCS-03-004, University of Liverpool, Department of Computer Science, 2003. `http://www.csc.liv.ac.uk/research/techreports`.

[6] C. Dixon. Temporal resolution using a breadth-first search algorithm. *Annals of Mathematics and Artificial Intelligence*, 22:87–115, 1998.

[7] M. Fisher. A normal form for first-order temporal formulae. In *Proc. CADE-11*, volume 607 of *LNCS*, pages 370–376. Springer, 1992.

[8] M. Fisher, C. Dixon, and M. Peim. Clausal temporal resolution. *ACM Transactions on Computational Logic*, 2(1):12–56, 2001.

[9] D. Gabelaia, R. Kontchakov, A. Kurucz, F. Wolter, and M. Zakharyaschev. On the computational complexity of spatio-temporal logics. Accepted to *FLAIRS 2003*, 2003.

[10] I. Hodkinson, F. Wolter, and M. Zakharyaschev. Decidable fragments of first-order temporal logics. *Annals of Pure and Applied Logic*, 106:85–134, 2000.

[11] G. J. Holzmann. The model checker Spin. *IEEE Trans. on Software Engineering*, 23(5):279–295, 1997.

[12] R. Kontchakov, C. Lutz, F. Wolter, and M. Zakharyaschev. Temporalizing tableaux. Accepted to *Studia Logica*.

[13] Alexander Leitsch. *The Resolution Calculus*. Springer, 1997.

[14] Z. Manna and A. Pnueli. *The Temporal Logic of Reactive and Concurrent Systems: Specification*. Springer, 1992.

[15] S. Merz. Decidability and incompleteness results for first-order temporal logic of linear time. *Journal of Applied Non-Classical Logics*, 2:139–156, 1992.

[16] R. Nieuwenhuis and A. Rubio. Paramodulation-based theorem proving. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, chapter 7, pages 371–443. Elsevier, 2001.

[17] A Pnueli. The Temporal Logic of Programs. In *Proceedings of the Eighteenth Symposium on the Foundations of Computer Science*, 1977.

[18] F. Wolter and M. Zakharyaschev. Axiomatizing the monodic fragment of first-order temporal logic. To appear in *Annals of Pure and Applied logic*, 2001.

[19] F. Wolter and M. Zakharyaschev. Qualitative spatio-temporal representation and reasoning: a computational perspective. In G. Lakemeyer and B. Nebel, editors, *Exploring Artifitial Intelligence in the New Millenium*, pages 175–216. Morgan Kaufmann, 2002.

7 Appendix

Proof [Of Lemma 4] Let Δ be a derivation of $C \Rightarrow \bigcirc\textbf{false}$ by fine-grained resolution; we may assume[9] that Δ is a derivation from the set of original step clauses $\mathcal{S}$ and the (current) set of universal clauses $\mathcal{U}^s$. Let

$$\begin{array}{rcl} P_1(x_1) & \Rightarrow & \bigcirc Q_1(x_1) \\ & \vdots & \\ P_k(x_k) & \Rightarrow & \bigcirc Q_k(x_k) \\ \\ p_1 & \Rightarrow & \bigcirc q_1 \\ & \vdots & \\ p_l & \Rightarrow & \bigcirc q_l \end{array}$$

be the set of all original clauses involved in Δ where $p_i \Rightarrow \bigcirc q_i$ denotes either an original step clause, or an e-derived step-clause of the form (2) added by the preprocessing (w.l.o.g., we assumed that all the variables $x_1,..,x_k$ are pairwise distinct). Note that (by accumulating the mgus used in the derivation) it is possible to construct a finite set of instances of these clauses (and universal clauses) such that $C \Rightarrow \bigcirc\textbf{false}$ is derivable from this new set of clauses and all mgus used in the derivation are empty[10]. That is, there are substitutions $\sigma_{1,1},\ldots,\sigma_{1,s_1},\sigma_{2,1},\ldots,\sigma_{2,s_2},\ldots\sigma_{k,s_k}$ such that

$$\begin{array}{rcl} P_1(x_1)\sigma_{1,1} & \Rightarrow & \bigcirc Q_1(x_1)\sigma_{1,1} \\ & \vdots & \\ P_1(x_1)\sigma_{1,s_1} & \Rightarrow & \bigcirc Q_1(x_1)\sigma_{1,s_1} \\ & \vdots & \\ P_k(x_k)\sigma_{k,1} & \Rightarrow & \bigcirc Q_k(x_k)\sigma_{k,1} \\ & \vdots & \\ P_k(x_k)\sigma_{k,s_k} & \Rightarrow & \bigcirc Q_k(x_k)\sigma_{k,s_k} \\ \\ p_1 & \Rightarrow & \bigcirc q_1 \\ & \vdots & \\ p_l & \Rightarrow & \bigcirc q_l \end{array} \tag{4}$$

(together with some instances of universal clauses) contribute to the derivation of $C \Rightarrow \bigcirc\textbf{false}$ where all mgus used in the derivation are empty, and, furthermore, $C = P_1(x_1)\sigma_{1,1} \wedge \ldots \wedge P_1(x_1)\sigma_{1,s_1} \wedge \ldots \wedge P_k(x_k)\sigma_{k,1} \wedge \ldots \wedge P_k(x_k)\sigma_{k,s_k} \wedge \ldots \wedge p_1 \wedge \ldots \wedge p_l$.

Note further (induction) that due to our restriction on the step resolution rule, for any i, j, the substitution $\sigma_{i,j}$ maps x_i into a free variable.

Let us group the instances of the original step clauses according to the value of the substitutions. We introduce an equivalence relation Σ on the clauses from (4) as follows: For every i, j, i', j' we have $\left(P_i(x_i)\sigma_{i,j} \Rightarrow \bigcirc Q_i(x_i)\sigma_{i,j},\ P_{i'}(x_{i'})\sigma_{i',j'} \Rightarrow \bigcirc Q_{i'}(x_{i'})\sigma_{i',j'}\right) \in \Sigma$ iff $x_i\sigma_{i,j} = x_{i'}\sigma_{i',j'}$ (it can be easily checked that Σ is an equivalence relation indeed). The relation Σ partitions (4) into K equivalence classes; let I_l be

[9] Indeed, let $\mathcal{U}'$ be $\mathcal{U}^s$ *together with all universal clauses subsumed at earlier steps of the derivation*. Then we can "repeat" derivations of all already derived intermediate clauses from $\mathcal{S} \cup \mathcal{U}'$. It suffices to note that if there exists a merged e-derived step clause $\mathcal{A} \Rightarrow \bigcirc\mathcal{B}$ such that $\mathcal{B} \cup \mathcal{U}' \vdash \perp$, we also have $\mathcal{B} \cup \mathcal{U}^s \vdash \perp$ (since "extra" clauses from $\mathcal{U}'$ are subsumed by other universal clauses, and universal clauses do not take part in the merging).

[10] The condition that premises of the non-ground binary resolution rule should be variable disjoint may be violated here; note, however, that this condition is needed for *completeness* of the proof system, not its *correctness*.

the set of indexes of the l-th equivalence class (we refer to clauses from (4) by indexes of the corresponding substitutions).

Let us for every $l : 1 \leq l \leq K$ define $C_l = \bigwedge_{(i,j)\in I_l} P_i(x_i)\sigma_{i,j}$, let $C_0 = \bigwedge p_i$. Note that $C = \bigwedge_{l=1}^{K} C_l \wedge C_0$ and this partition of C is disjoint. Let $D_l = \bigwedge_{(i,j)\in I_l} Q_i(x_i)\sigma_{i,j}$, let $D_0 = \bigwedge q_i$, let $D = \bigwedge_{l=1}^{K} D_l \wedge D_0$. Note that $\widetilde{\forall} D \wedge \mathcal{U} \vdash \perp$. Note further that if we replace the free variable of D_l with a fresh constant, c_l, there still exists a derivation of **false** from $\bigwedge D_{l=1}^{K}(c_l) \wedge D_0$ and universal clauses (with unifiers applied to universal and intermediate clauses only). It follows that $\bigwedge \exists x D_{l=1}^{K}(x) \wedge D_0 \wedge \mathcal{U} \vdash \perp$.

It suffices to note that $(\bigwedge_{l=1}^{K} \exists x C_l(x) \wedge C_0) \Rightarrow \bigcirc(\bigwedge_{l=1}^{K} \exists x D_l(x) \wedge D_0)$ is a merged e-derived step clause.
❑

Proof [Of Lemma 5] Since $\mathcal{B} \cup \mathcal{U} \vdash \perp$, by the definition of a merged e-derived step clause, there exists a set of instances of original step clauses

$$
\begin{array}{rcl}
P_{i_{1,1}}(c_1) & \Rightarrow & \bigcirc Q_{i_{1,1}}(c_1) \\
 & \vdots & \\
P_{i_{1,s_1}}(c_1) & \Rightarrow & \bigcirc Q_{i_{1,s_1}}(c_1) \\
P_{i_{2,1}}(c_2) & \Rightarrow & \bigcirc Q_{i_{2,1}}(c_2) \\
 & \vdots & \\
P_{i_{2,s_2}}(c_2) & \Rightarrow & \bigcirc Q_{i_{2,s_2}}(c_2) \\
 & \vdots & \\
P_{i_{k,1}}(c_k) & \Rightarrow & \bigcirc Q_{i_{k,1}}(c_k) \\
 & \vdots & \\
P_{i_{k,s_k}}(c_k) & \Rightarrow & \bigcirc Q_{i_{k,s_k}}(c_k) \\
p_1 & \Rightarrow & \bigcirc q_1 \\
 & \vdots & \\
p_l & \Rightarrow & \bigcirc q_l,
\end{array}
$$

where $c_1, \ldots, c_k$ are new (Skolem) constants, such that $Q_{i_{1,1}}(c_1) \wedge \ldots \wedge Q_{i_{1,s_1}}(c_1) \wedge Q_{i_{2,1}}(c_2) \wedge \ldots \wedge Q_{i_{2,s_2}}(c_2) \wedge \ldots \wedge Q_{i_{k,1}}(c_k) \wedge \ldots \wedge Q_{i_{k,s_k}}(c_k) \wedge q_1 \wedge \ldots \wedge q_l \wedge \mathcal{U}^s \vdash \perp$ (again, as in the proof of Lemma 4, $p_i \Rightarrow \bigcirc q_i$ denotes either an original ground step clause or a clause of the form (2) added by the preprocessing).

Let Δ be a (first-order) resolution derivation of $\perp$ from $\mathcal{U}^s$ and the following set of clauses $Q_{i_{1,1}}(c_1), \ldots, Q_{i_{1,s_1}}(c_1), Q_{i_{2,1}}(c_2), \ldots, Q_{i_{2,s_2}}(c_2), \ldots, Q_{i_{k,1}}(c_k), \ldots, Q_{i_{k,s_k}}(c_k), q_1, \ldots, q_l$. Let $Q_{j_{1,1}}(c_1), \ldots, Q_{j_{1,t_1}}(c_1), Q_{j_{2,1}}(c_2), \ldots, Q_{j_{2,t_2}}(c_2), \ldots, Q_{j_{k,1}}(c_k), \ldots, Q_{j_{k,t_k}}(c_k), q_1, \ldots, q_k$ be its subset containing all the clauses involved in Δ.

We assume, for simplicity of the proof, that the lifting theorem (cf. e.g. [13]) holds for Δ, that is, there exists a non-ground (first-order) derivation Δ' of $\perp$ from $Q_{i_{1,1}}(x), \ldots, Q_{i_{1,s_1}}(x), Q_{i_{2,1}}(x), \ldots, Q_{i_{2,s_2}}(x), \ldots, Q_{i_{k,1}}(x), \ldots, Q_{i_{k,s_k}}(x), q_1, \ldots, q_l$ such that $\Delta \leq_s \Delta'$ in the terminology of [13]: Every clause C'_i of Δ' is a generalisation of the corresponding clause C_i of Δ.

Then there exists a derivation Γ by fine-grained step resolution from

$$
\begin{array}{rcl}
P_{j_{1,1}}(c_1) & \Rightarrow & \bigcirc Q_{j_{1,1}}(c_1) \\
 & \vdots & \\
P_{j_{1,t_1}}(c_1) & \Rightarrow & \bigcirc Q_{j_{1,t_1}}(c_1) \\
P_{j_{2,1}}(c_2) & \Rightarrow & \bigcirc Q_{j_{2,1}}(c_2) \\
 & \vdots & \\
P_{j_{2,t_2}}(c_2) & \Rightarrow & \bigcirc Q_{j_{2,t_2}}(c_2) \\
 & \vdots & \\
P_{j_{k,1}}(c_k) & \Rightarrow & \bigcirc Q_{j_{k,1}}(c_k) \\
 & \vdots & \\
P_{j_{k,t_k}}(c_k) & \Rightarrow & \bigcirc Q_{j_{k,t_k}}(c_k) \\
p_1 & \Rightarrow & \bigcirc q_1 \\
 & \vdots & \\
p_l & \Rightarrow & \bigcirc q_k,
\end{array}
$$

(and universal clauses) of a final clause $C \Rightarrow \bigcirc\mathbf{false}$, where $C = P_{j_{1,1}}(c_1) \wedge \ldots \wedge P_{j_{1,t_1}}(c_1) \wedge P_{j_{2,1}}(c_2) \wedge \ldots \wedge P_{j_{2,t_2}}(c_2) \wedge \ldots \wedge P_{j_{k,1}}(c_k) \wedge \ldots \wedge P_{j_{k,t_k}}(c_k) \wedge p_1 \wedge \ldots \wedge p_k$.

It is easy to see that the lifting theorem can be transfered to fine-grained inferences, and there exists a derivation Γ' from the set of original step clauses

$$
\begin{array}{rcl}
P_{j_{1,1}}(x_{j_{1,1}}) & \Rightarrow & \bigcirc Q_{j_{1,1}}(x_{j_{1,1}}) \\
 & \vdots & \\
P_{j_{1,t_1}}(x_{j_{1,t_1}}) & \Rightarrow & \bigcirc Q_{j_{1,t_1}}(x_{j_{1,t_1}}) \\
P_{j_{2,1}}(x_{2,1}) & \Rightarrow & \bigcirc Q_{j_{2,1}}(x_{j_{2,1}}) \\
 & \vdots & \\
P_{j_{2,t_2}}(x_{2,t_2}) & \Rightarrow & \bigcirc Q_{j_{2,t_2}}(x_{j_{2,t_2}}) \\
 & \vdots & \\
P_{j_{k,1}}(x_{j_{k,1}}) & \Rightarrow & \bigcirc Q_{j_{k,1}}(x_{j_{k,1}}) \\
 & \vdots & \\
P_{j_{k,t_k}}(x_{j_{k,t_k}}) & \Rightarrow & \bigcirc Q_{j_{k,t_k}}(x_{j_{k,t_k}}) \\
p_1 & \Rightarrow & \bigcirc q_1 \\
 & \vdots & \\
p_l & \Rightarrow & \bigcirc q_k,
\end{array}
$$

(and universal clauses) of a final clause $C' \Rightarrow \bigcirc\mathbf{false}$ such that $\Gamma' \geq_s \Gamma$, that is, every intermediate clause $C'_i \Rightarrow \bigcirc D'_i$ from Γ' is a generalisation of a corresponding clause from Γ. Note that none of the (Skolem) constants $c_1, \ldots, c_k$ occurs in Γ'. If, in the derivation Γ', a constant or a functional term was substituted into a variable occurring into the left-hand side of a clause, this clause would not be a generalisation of any clause from Γ. This implies the conclusion of the lemma. ❑

Note 6 This quite clear construction might be hard to fulfil in the presence of various *refinements* of resolution and/or *redundancy deletion*. As a remedy, we suggest considering *constrained calculi*, like e.g.

resolution over non-ground constrained clauses with constraint inheritance. It is known that such inference systems are complete and moreover compatible with redundancy elimination rules and many (liftable) refinements (see e.g. [16], theorems 5.11 and 5.12, subsections 5.4 and 5.5, resp.). Here we take into account that there are no clauses with equality, and therefore all sets are *well-constrained* in the terminology of [16].

Then instead of ground clauses of the form

$$P_{j_{l,t_l}}(c_l) \;\Rightarrow\; \bigcirc Q_{j_{l,t_l}}(c_l)$$

we consider their *constrained* representations

$$P_{j_{l,t_l}}(x_l) \;\Rightarrow\; \bigcirc Q_{j_{l,t_l}}(x_l) \cdot \{x_l = c_l\}.$$

Recall that in accordance with the semantics of constrained clauses, a clause $C \cdot T$ represents the set of all ground instances $C\sigma$ where σ is a solution of T. In our case, there is exactly one solution of $x_l = c_l$ given by the substitution $\{x \mapsto c_l\}$. So the semantics of

$$P_{j_{l,t_l}}(x_l) \;\Rightarrow\; \bigcirc Q_{j_{l,t_l}}(x_l) \cdot \{x_l = c_l\}$$

is just

$$P_{j_{l,t_l}}(c_l) \;\Rightarrow\; \bigcirc Q_{j_{l,t_l}}(c_l) \cdot \{\}.$$

So all clauses originated from the universal part have empty constraints and all temporal clauses have constraints defined above, and there exists a non-ground derivation of a constrained final clause with constraint inheritance. Note that the (Skolem) constants $c_1, \ldots, c_k$ may only occur in constraints but not in clauses themselves. It suffices to note that in this case inferences with constraint inheritance admit only two kinds of substitutions into x_l: either $\{x_l \mapsto c_l\}$ (however it is impossible because c_l occurs only in constraints), or $\{x_l \mapsto x_{l'}\}$ where $x_{l'}$ is bound by the same constraint $\{x_{l'} = c_l\}$. The case of matching x_l and y where y originates from the universal part is solved by the substitution $\{y \mapsto x_l\}$. A non-ground inference of a final clause, satisfying the conditions on substitutions in the fine-grained resolution rules, can be extracted from this constrained derivation implying, thus, the conclusion of the lemma 5.

Proof [Of Lemma 7] By Lemma 4, there exists a merged (from $LT(S)$) e-derived clause $\mathcal{A} \Rightarrow \bigcirc\mathcal{B}$ such that $\{\neg\Phi(c^l)\} \cup \mathcal{B} \cup \mathcal{U} \vdash \bot$ and $\mathcal{A} = \widetilde{\exists} C$. It suffices to notice that $\forall x((\mathcal{A} \Rightarrow \bigcirc\mathcal{B})\{c^l \to x\})$ is a full merged (from S) step rule and $\exists x(\Phi(x) \wedge \mathcal{B}\{c^l \to x\} \wedge \mathcal{U})$ is unsatisfiable. ❑

Proof [Of Lemma 8] The proof is analogous to the proof of Lemma 5. As we already noticed, $\exists x(\mathcal{B}(x) \wedge \mathcal{U} \wedge \neg\Phi(x))$ and this can be checked by a first-order resolution procedure. Since c^l does not occur in the problem, we can skolemise this existential quantifier with c^l. We lift now all Skolem constants but c^l. ❑