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

Boris Konev†, Anatoli Degtyarev‡, Clare Dixon†, Michael Fisher†, Ullrich Hustadt†

† Department of Computer Science, University of Liverpool, UK
{konev, clare, michael, ullrich}@csc.liv.ac.uk

‡ Department of Computer Science, Kings College, London, UK
anatoli@dcs.kcl.ac.uk
Overview

First-order temporal logic is very expressive but has no finite axiom system.

The *monodic* fragment of FOTL has been shown to have completeness and sometimes even decidability properties (Hodkinson et al.).

Tableau and resolution calculi have been defined for the monodic fragment (Kontchakov et al., Degtyarev et al.), however neither are very practical.

Here we focus on expanding domains i.e. the domain which first-order terms can range over can increase at each temporal step.

We present a fine grained calculus with steps amenable to implementation.
Plan

- First-Order Temporal Logic (FOTL)
- A normal form for FOTL
- Step Resolution
- Eventuality Resolution
- Example
- Conclusions
Motivation

The monodic fragment of first-order temporal logic is useful:

- for spatio-temporal reasoning;
- for reasoning about temporal entity relation models;
- as a unifying framework for:
  - temporal logics of knowledge/belief;
  - spatio-temporal logics;
  - temporal description logics.
First-Order Temporal Logic

- First-order language without functional symbols extended with temporal connectives: $\bigcirc$, $\Diamond$, $\Box$, $\mathcal{U}$, $\mathcal{W}$.

- Formulae are interpreted in a sequence of first-order models, $\mathcal{M}_n = \langle D_n, I_n \rangle$, where $D_n \subseteq D_m$ (expanding domain) for $n < m$ and $I_n$ is an interpretation of constant and predicate symbols over $D_n$.

- The interpretation of constants and variables is rigid.

**Definition 1** An FOTL-formula $\phi$ is called monodic if any subformulae of the form $\mathcal{T} \psi$, where $\mathcal{T}$ is $\bigcirc$, $\Box$, or $\Diamond$ (or $\psi_1 \mathcal{T} \psi_2$, where $\mathcal{T}$ is $\mathcal{U}$ or $\mathcal{W}$), contains at most one free variable.

The set of valid monodic formulae is finitely axiomatisable.
Semantics of Temporal Operators

Usual semantics for the temporal operators.

\[ p \]

\[ \square p \]

\[ \Diamond p \]

\[ p \]

\[ p \quad p \quad p \quad p \quad p \quad p \quad p \quad p \quad p \quad p \]

\[ p \sqcup q \]

\[ p \]

\[ \Diamond p \]

\[ p \quad p \quad p \quad p \quad p \quad p \quad p \quad q \]

\[ p \sqcup q \]

where

\[ p \triangledown q \equiv p \sqcup q \lor \square p \]
Overview of the Resolution Method

The resolution method consists of three main steps:-

- translation to normal form;

- classical style resolution between clauses involving the next-time operator, or between purely classical logic clauses or between these two sets;

- eventuality resolution between eventuality clauses $(\Diamond L(x))$ and complex combinations of clauses involving the next-time operator $A_i(x) \Rightarrow \Box B_i(x)$ which have the effect $\forall x(\exists_i A_i(x) \Rightarrow \Box \Box \neg L(x))$ (and ground versions of these).

For simplicity, in the talk, we consider the case where there are no constants in the original problem.
Translation to Normal Form

We translate formulae into a satisfiability preserving normal form known as *Divided Separated Normal Form*.

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 $P(x)$ is a unary predicate and $M(x)$ 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.
A monodic temporal problem in Divided Separated Normal Form (DSNF) is a quadruple $\langle \mathcal{U}, \mathcal{I}, \mathcal{S}, \mathcal{E} \rangle$, where

- the universal part, $\mathcal{U}$, is a set of first-order formulae;
- the initial part, $\mathcal{I}$, is a set of first-order formulae;
- the step part, $\mathcal{S}$, is a set of original (ground and non-ground) temporal step clauses; and
- the eventuality part, $\mathcal{E}$, is a set of 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.

$\mathcal{U}$, $\mathcal{I}$, $\mathcal{S}$, and $\mathcal{S}$ are finite. With each monodic temporal problem, we associate $\mathcal{I} \land \Box \mathcal{U} \land \Box \forall x \mathcal{S} \land \Box \forall x \mathcal{E}$.
The original calculus required complex combinations of the ground and non-ground step clauses to apply such rules as the following.

\[ A \Rightarrow \bigcirc B \quad \frac{A \Rightarrow B}{\neg A} (\bigcirc U_{res}) \]

where \( U \cup \{ B \} \vdash \bot \).

Here \( A \Rightarrow \bigcirc B \) denote complex combinations of step clauses (see the paper for their definition).
A Fine-Grained Calculus

- The idea of the new (fine-grained) calculus is to define small resolution steps, more like classical binary resolution, which are easier to implement.

- Several of the smaller steps may be required to mimic the original (complex) step resolution rule.

- Thus we may generate additional step clauses of the form

\[ C \Rightarrow \bigcirc D. \]

Here, \( C \) is a conjunction of propositions and unary predicates of the form \( P(x) \) and \( D \) is a disjunction of literals.

- First, rewrite the universal and initial parts into clausal form.
Deduction rules

1. Fine-grained (restricted) step resolution

\[
\begin{align*}
    (C_1 \land C_2) \sigma & \Rightarrow \bigcirc (D_1 \lor D_2) \sigma, \\
    C_1 \Rightarrow \bigcirc (D_1 \lor L) & \quad C_2 \Rightarrow \bigcirc (D_2 \lor \neg M), \\
    C_1 \Rightarrow \bigcirc (D_1 \lor L) & \quad D_2 \lor \neg M, \\
    C_1 \sigma & \Rightarrow \bigcirc (D_1 \lor D_2) \sigma,
\end{align*}
\]

where \( C_1 \Rightarrow \bigcirc (D_1 \lor L) \) and \( C_2 \Rightarrow \bigcirc (D_2 \lor \neg M) \) are step clauses, \( D_2 \lor \neg M \) is a universal clause, and \( \sigma \) is an mgu of the literals \( L \) and \( M \) such that \( \sigma \) does not map variables from \( C_1 \) or \( C_2 \) into a constant or a functional term.
2. Resolution (first-order) between universal clauses, resulting in a universal clause.

3. Resolution (first-order) between initial and universal clauses (or initial clauses), resulting in an initial clause.

4. Clause conversion. A step clause of the form $C \Rightarrow \bigcirc \text{false}$ is rewritten into the universal clause $\neg C$.

5. Right/left factor

$$
\frac{C \Rightarrow \bigcirc (D \lor L \lor M)}{C \sigma \Rightarrow \bigcirc (D \lor L)\sigma}, \quad \frac{(C \land L \land M) \Rightarrow \bigcirc D}{(C \land L)\sigma \Rightarrow \bigcirc D\sigma},
$$

where $\sigma$ is an mgu of $L$ and $M$ s.t. $\sigma$ does not map variables from $C$ into a constant or a functional term.
The Original Calculus (Eventualities)

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

$$\forall x (A_1(x) \Rightarrow \bigcirc (B_1(x)))$$

$$\vdots$$

$$\forall x (A_n(x) \Rightarrow \bigcirc (B_n(x)))$$

$$\forall x \bigwedge_{i=1}^{n} \neg A_i(x)$$

$$\diamond L(x)$$

$$(\diamond \mathcal{U}_{res})$$

where $\forall x (A_i(x) \Rightarrow \bigcirc B_i(x))$ are complex combinations of step clauses s.t. for all $i \in \{1, \ldots, n\}$, the side conditions $\forall x (\mathcal{U} \land B_i(x) \Rightarrow \neg L(x))$ and $\forall x (\mathcal{U} \land B_i(x) \Rightarrow \bigvee_{j=1}^{n} (A_j(x)))$ are valid. $\forall x (A_i(x) \Rightarrow \bigcirc (B_i(x)))$ are known as a loop in $\neg L(x)$. 
Loop Search and BFS

- Using fine-grained resolution for eventuality resolution is based on an extension to FOTL of the BFS algorithm for loop search in PTL.

- To find a loop in $\neg L(x)$ BFS finds a sequence of first-order formulae $H_0, \ldots H_n$ (where $H_0 = \text{true}$) s.t. for each $i \geq 0$ there are sets of complex combinations of step clause, $\forall x (A_{i+1}(x) \Rightarrow \bigcirc B_{i+1}(x))$, s.t.

$$\forall x ( (B_{i+1}(x) \land U) \Rightarrow (H_i(x) \land \neg L(x))) .$$

- The algorithm terminates when either $H_n(x) = \text{false}$ (no loop) or $\forall x (H_{n-1}(x) \Rightarrow H_n(x))$.

- Note $\forall x ( (B_{i+1}(x) \land U) \Rightarrow (H_i(x) \land \neg L(x)))$ is valid when $\exists x ((B_{i+1}(x) \land U) \land \neg (H_i(x) \land \neg L(x)))$ is unsatisfiable.
Introduce a new constant $c^l$ called the *loop constant*.

To detect each $H_{i+1}(x)$ we add the clause

$$\text{true} \Rightarrow \Box (\neg H_i(c^l) \lor L(c^l))$$

and resolve.

For all new clauses $C_j \Rightarrow \Box \text{false}$ generated let

$$H_{i+1}(x) = \bigvee_{j=1}^{k} C_j \{c^l \rightarrow x\}.$$  

We use all the rules for fine-grained resolution except the clause conversion rule.

We relax the restriction on substitutions and allow the loop constant to be substituted into the left hand side of a step clause.
Example

Let us consider a monodic temporal problem $P$ given by

$$I = \{i1 : l\}, \ U = \{\forall x (B(x) \Rightarrow A(x) \land \neg L(x)), \ l \Rightarrow \exists x A(x)\},$$

$$S = \{s1 : A(x) \Rightarrow \bigcirc B(x)\}, \ E = \{e1 : \Diamond L(x)\}.$$

Recall, this is short for $I \land \Box U \land \Box \forall x S \land \Box \forall x E$.

This is unsatisfiable which we will show by resolution.
Example (Cont.)

\[ I = \{ i1 : l \}, \quad U = \{ \forall x (B(x) \Rightarrow A(x) \land \neg L(x)), \ l \Rightarrow \exists x A(x) \} \]
\[ S = \{ s1 : A(x) \Rightarrow \Box B(x) \}, \quad E = \{ e1 : \Diamond L(x) \} \]

We clausify \( U \) resulting in
\[ U^s = \{ u1 : (\neg B(x) \lor A(x)), \ u2 : (\neg B(x) \lor \neg L(x)), \ u3 : \neg l \lor A(c) \}. \]

- **Step resolution**

\[ s2 : A(x) \Rightarrow \Box A(x) \quad (s1, u1) \]
\[ s3 : A(x) \Rightarrow \Box \neg L(x) \quad (s1, u2) \]

Now we try finding a loop in \( \Diamond L(x) \).
Example (Cont.)

\[ S = \{ s1 : A(x) \Rightarrow \Diamond B(x), \ s2 : A(x) \Rightarrow \Diamond A(x), \]
\[ s3 : A(x) \Rightarrow \Diamond \neg L(x) \} \]

Loop resolution: resolve \( \{ l1 : \text{true} \Rightarrow \Diamond L(c^l) \} \) with \( U, S \).

\[ l2 : A(c^l) \Rightarrow \Diamond \text{false} \quad (s3, l1) \]

\( H_1(x) = A(x) \). At the second iteration resolve

\( \{ l3 : \text{true} \Rightarrow \Diamond (\neg A(c^l) \lor L(c^l)) \} \) with \( U \) and \( S \).

\[ l4 : A(c^l) \Rightarrow \Diamond L(c^l) \quad (s2, l3) \]
\[ l5 : A(c^l) \Rightarrow \Diamond \text{false} \quad (s3, l4) \]

\( H_2(x) = A(x) \) and \( \forall x(H_1(x) \Rightarrow H_2(x)) \) so we terminate with the loop \( A(x) \).
Example (Cont.)

\[ \mathcal{I} = \{i1 : l\}, \mathcal{U}^s = \{u1 : (\neg B(x) \lor A(x)), u2 : (\neg B(x) \lor \neg L(x)), u3 : \neg l \lor A(c)\}, \mathcal{S} = \{s1 : A(x) \Rightarrow \Box B(x), s2 : A(x) \Rightarrow \Box A(x), s3 : A(x) \Rightarrow \Box \neg L(x)\}, \mathcal{E} = \{e1 : \Diamond L(x)\} \]

Eventuality resolution: we can apply now the eventuality resolution rule whose conclusion is

\[ u4 : \neg A(x). \]

Universal/initial resolution

\[ u5 : \neg l \quad (u3, u4) \quad i2 : \textbf{false} \quad (i1, u5) \]

The problem is unsatisfiable.
Theoretical Issues

The translation to the normal form is satisfiability preserving.

**Theorem 1** 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.

**Theorem 2** The calculus consisting of the rules of fine-grained step resolution, together with the (both ground and non-ground) eventuality resolution rule, is complete for the monodic fragment over expanding domains even if we restrict ourselves to loops found by the BFS algorithm.
Conclusions

- 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.

- Completeness is shown relative to the completeness proof for the expanding domain for the non-fine grained version.

- The fine-grained resolution inference rules are more amenable to efficient implementation and could be implemented directly using any appropriate first-order theorem prover for classical logics.