Reasoning about Action in Polynomial Time

Thomas Drakengren and Marcus Bjareland
Department of Computer and Information Science
Link6ping University, S-581 83 Linkoping, Sweden
email: {thodr, marbj}@ida.liu.se

Abstract

Although many formalisms for reasoning about action exist, surprisingly few approaches have taken computational complexity into consideration. The contributions of this paper are the following: a temporal logic with a restriction for which deciding satisfiability is tractable, a tractable extension for reasoning about action, and NP-completeness results for the unrestricted problems. Many interesting reasoning problems can be modelled, involving nondeterminism, concurrency and memory of actions. The reasoning process is proved to be sound and complete.

1 Introduction

Although many formalisms for reasoning about action exist, surprisingly few approaches have taken computational complexity into consideration. One explanation for this might be that many interesting AI problems are (at least) NP-hard, and that tractable subproblems that are easily extracted, tend to lack expressiveness. This has led a large part of the AI community to rely on heuristics and incomplete systems to solve the problems (see e.g. [Ginsberg, 1996] for a discussion). This holds, in particular, for the area of reasoning about action, where the very expressive logical formalisms provide difficult obstacles when it comes to efficient implementation.

We feel, however, that the tractability boundary for sound and complete reasoning about action has not yet been satisfactorily investigated. We prove this by introducing a nontrivial subset of a logic with semantics closely related to the *trajectory semantics* of Sandewall [1994], for which satisfiability is tractable. Our logic can handle examples involving not only nondeterminism, but continuous time, concurrency and memory of actions as well, thus providing a conceptual extension of Sandewall's framework. The reader should note that our main concern is *computation,* as opposed to *modelling.*

This paper is organised as follows: Section 2 is an informal overview of the technical results, where also two examples are presented. In Section 3 we present the syntax and semantics of the basic temporal logic and the extension for reasoning about action, and in Section 4, we present the following: three intractability results for these formalisms, and the main results: the tractable subclasses of the temporal logic and its extension[1].

2 Overview

In Section 3 we develop a temporal logic, A, which is syntactically related to the propositional temporal logic TPTL [Alur and Henzinger, 1989]. The temporal domain is the set of real numbers and temporal expressions are based on relations $=, \leq, <, \geq$ and $>$ between linear polynomials with rational coefficients over a set of temporal variables. The semantics of this temporal logic is standard. The formalism for reasoning about action is *narrative* based, which means that *scenario descriptions* are used to model the real world. Scenario descriptions consist of formulae in the temporal logic *(observations)* and *action expressions* which are constructs that state that certain changes in values of the *features* (propositions, fluents) may occur. We write action expressions as $\pi \Rrightarrow [\alpha]\epsilon$ $\mathtt{Infl}$, where π is the precondition for the action, ϵ the effects, *a* a temporal expression denoting *when* the effects are taking place, and Inf 1 is the set of all features that are influenced by the action. The influenced features are not subject to the assumption of inertia, i.e. we allow them, and only them, to change during the execution of the action.

It turns out that deciding satisfiability is NP-complete, both for the temporal logic and the scenario descriptions. Interestingly, the problem is NP-complete for scenario descriptions that only include Horn clause observations, unconditional and unary action expressions (this terminology is explained later), and no stated relations between temporal expressions.

To extract a tractable subset from our formalism we rely on a recent result in *temporal constraint reasoning* by Jonsson and Backstrom in [1996] (also discovered independently by Koubarakis [1996]). They have identified a large tractable class of temporal constraint reasoning, using *Horn Disjunctive Linear Relations* (Horn DLR's)

[1] Due to lack of space, proofs of most theorems are omitted. However, they can all be obtained from the authors (until published).

which are relations between linear polynomials with rational coefficients. We make use of their result by restricting formulae in our scenario descriptions to be Horn and then by encoding scenario descriptions into Horn DLR's. For the temporal logic this is fairly straightforward. For the scenario descriptions, it turns out that we have to put some constraints on the temporal relations and actions in the scenario descriptions.

We will use the following two examples: Jump into a Lake with a Hat [Giunchiglia and Lifschitz, 1995] and Soup Bowl Lifting [Gelfond *et al.*, 1991]. Below we informally describe the examples.

Example 1 (Jump into a Lake with a Hat, JLH) If you jump into the lake you will get wet. If you have been in the water at some time point it is unclear if you still have your hat on. This is an example of *nondeterminism* and of *memory of actions.* □

Example 2 (Soup Bowl Lifting, SBL) If we lift either side of a soup bowl at some time points, the content will be spilled, unless we lift both sides at the same time point. This is an example of *simultaneous concurrency.* □

Both examples stated above can be handled by the tractable subset of our formalism. Note that we are not confined to simultaneous concurrency; actions may partly overlap.

3 Scenario Descriptions

We introduce a semantics that is a simpler variant of Sandewall's Features and Fluents Framework [Sandewall, 1994], in that the effects of an action can only occur at one and the same time point for a given action, and we use only propositional values of features (similar to the work of Doherty [1994]) However, in some respects this formalism is more flexible than Sandewall's: we use a *continuous* time domain, we allow *concurrently* executing actions, and effects of actions can depend on other states in the history than the state at the *starting time point* of the action (this implies *memory* of actions, in Sandewall's [1994] terminology). One example of a formalism having memory is that of Gustafsson and Doherty [1996].

Initially, a basic temporal logic is defined. The computational properties of this logic will be exploited by the scenario description logic, i.e. ultimately (in Section 4) the scenario descriptions will be transformed into formulae of the basic temporal logic.

3.1 Syntax

We begin by defining the basic temporal logic.

We assume that we have a set $\mathcal{T}$ of time point variables intended to take real values, and a set F of *features* intended to take propositional values.

Definition 3 A *signature* is a tuple $\sigma = \langle\mathcal{T},\mathcal{F}\rangle$, where T is a finite set of time point variables and T is a finite set of propositional features. A *time point expression* is a linear polynomial over T with rational coefficients. We denote the set of time point expressions over $\mathcal{T}$ by $\mathcal{T}^*$. □

Definition 4 Let $\sigma = \langle\mathcal{T},\mathcal{F}\rangle$ be a signature, let $\alpha,\beta \in \mathcal{T}^*$, $f \in \mathcal{F}$, $R \in \{=,\leq,<,\geq,>\}$, $\oplus \in \{\wedge,\vee,\rightarrow,\leftrightarrow\}$, and define the ***scenario description language*** Λ over σ by

$$\Lambda ::= \mathtt{T} \mid \mathtt{F} \mid f \mid \alpha R\beta \mid \neg\Lambda \mid \Lambda_1 \oplus \Lambda_2 \mid [\alpha]\Lambda.$$

A formula that does not contain any connectives (any of the constructs $\wedge,\vee,\rightarrow,\leftrightarrow,\neg$ and $[\cdot]$) is ***atomic***. If γ is atomic and $\alpha \in \mathcal{T}^*$, then the formulae γ, $[\alpha]\gamma$, $\neg\gamma$, $[\alpha]\neg\gamma$ and $\neg[\alpha]\gamma$ are ***literals***. (A formula $[\alpha]\gamma$ expresses that at time α, γ is true.) A literal l is ***negative*** iff it contains $\neg$ and its corresponding atomic formula γ is not of the form $\alpha R\beta$ for $R \in \{<,\leq,>,\geq\}$. A literal that is not negative is ***positive***. Disjunctions of literals are ***clauses***. A formula $\gamma \in \Lambda$ is in ***conjunctive normal form***, CNF, iff it is a conjunction of clauses. A formula γ is ***Horn*** iff it is a clause with at most one positive literal. A set Γ of formulae is ***Horn*** iff every $\gamma \in \Gamma$ is Horn. Syntactical identity between formulae is written $\equiv$, and when ambiguity is to be avoided, we denote formulae $\gamma \in \Lambda$ by $\ulcorner\gamma\urcorner$.

Let γ be a formula. A feature $f \in \mathcal{F}$ occurs ***free*** in γ iff it does not occur within the scope of a $[\alpha]$ expression in γ. $\alpha \in \mathcal{T}^*$ ***binds*** f in γ if a formula $[\alpha]\phi$ occurs as a subformula of γ, and f is free in ϕ. If no feature occurs free in γ, γ is ***closed***. If γ does not contain any occurrence of $[\alpha]$ for some $\alpha \in \mathcal{T}^*$, then γ is ***propositional***. □

Using Λ, we can thus express propositions being true at time points, and express relations between time points. Next we define the extension of the basic temporal logic by introducing ***action expressions***, i.e. constructs that enable modelling of change.

Definition 5 Let $\sigma = \langle\mathcal{T},\mathcal{F}\rangle$ be a signature. An ***action expression*** over σ is a tuple $A = \langle\alpha,\pi,\mathtt{Infl},\epsilon\rangle$, $\alpha \in \mathcal{T}^*$, π a closed formula in Λ, $\mathtt{Infl} \subseteq \mathcal{F}$, and ϵ a propositional formula, where all features occurring in ϵ are in $\mathtt{Infl}$. α is the ***result time point*** of A, π is the ***precondition*** of A, $\mathtt{Infl}$ is the set of ***influenced features*** of A, and ϵ is the ***effects*** of A. A is ***unconditional*** iff $\pi \equiv \mathtt{T}$, and ***unary*** iff $|\mathtt{Infl}| = 1$.

For convenience, we write action expressions as $\pi \Rightarrow [\alpha]\epsilon\,\mathtt{Infl}$; for example we have $[3]\textit{loaded} \Rightarrow [4]\neg\textit{alive}\{\textit{alive}\}$ for an action shoot. If $\pi \equiv \mathtt{T}$, we remove it and the $\Rightarrow$ symbol, and if $\epsilon \equiv \mathtt{T}$, we remove it. An example is an unconditional loading action $[2]\textit{loaded}\{\textit{loaded}\}$, and the action of spinning the chamber of a gun $[3]\{\textit{loaded}\}$.

An ***observation*** over σ is a closed formula in Λ. □

Next, we combine the concepts defined so far into one.

Definition 6 A ***scenario description*** is a tuple $\Upsilon = \langle\sigma,\mathrm{SCD},\mathrm{OBS}\rangle$, where $\sigma = \langle\mathcal{T},\mathcal{F}\rangle$ is a signature, SCD (the ***schedule***) is a finite set of action expressions over σ, and OBS is a finite set of observations over σ. The ***size*** of a scenario description is defined as the sum of lengths of all formulae in SCD and OBS. □

Now, we formalise the examples from Section 2.

Example 7 (JLH) The intended conclusion of the following scenario is that the person is wet at time c_1, and we do not know if the hat is on at time point C_2, occurring after the person jumps.

OBS1	$[0]hat\_on \wedge dry \wedge on\_land$
SCD1	$[c_1]\neg on\_land\{on\_land\}$
SCD2	$[c_1]\neg on\_land \Rightarrow [c_1]\neg dry\{dry\}$
SCD3	$[c_1]\neg on\_land \Rightarrow [c_2]\{hat\_on\}$
OBS4	$c_1 \geq 0 \wedge c_2 \geq c_1$ □

Example 8 (SBL) We have two actions: one for lifting the left side of the soup bowl and one for lifting the right side. If the actions are not executed simultaneously, the table cloth will no longer be dry. The intended conclusion here, is that $C_2 = C_1$.

OBS1	$[0]dry$
SCD1	$[c_1]leftup\{leftup\}$
SCD2	$[c_1]\neg rightup \Rightarrow [c_1]\neg dry\{dry\}$
SCD3	$[c_2]rightup\{rightup\}$
SCD4	$[c_2]\neg leftup \Rightarrow [c_2]\neg dry\{dry\}$
OBS2	$[c_2]dry$
OBS3	$c_2 \geq 0 \wedge c_1 \geq 0$ □

3.2 Semantics

For the presentation of the semantics we proceed similarly to the presentation of the syntax. We begin by defining the semantics of the basic temporal logic.

Definition 9 Let $\sigma = \langle \mathcal{T}, \mathcal{F} \rangle$ be a signature. A *state* over is a function from T to the set {T, F} of truth values. A *history* over σ is a function h from R to the set of states. A *valuation* ϕ is a function from T to R. It is extended in a natural way (as a homomorphism from T^* to R), giving e.g. $\phi(3t + 4.3) = 3\phi(t) + 4.3$. A *development,* or *interpretation,* over σ is a tuple $\langle h, \phi \rangle$ where h is a history and ϕ is a valuation. □

Definition 10 Let $\gamma \in \Lambda$, and let $D = \langle h, \phi \rangle$ be a development. Define the *truth value* of γ in D for a time point $t \in \mathbf{R}$, denoted $D(\gamma, t)$, as follows (here we overload T and F to denote both formulae and truth values). Assume $f \in \mathcal{F}$, $R \in \{=, \leq, <, \geq, >\}$, $\alpha, \beta \in \mathcal{T}^*$, $\gamma, \delta \in \Lambda$, $\oplus \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$, and $\tau \in \{\mathrm{T}, \mathrm{F}\}$. Now define

$$D(\tau, t) = \tau \qquad D(f, t) = h(t)(f)$$
$$D(\alpha R \beta, t) = \phi(\alpha) R \phi(\beta) \qquad D(\neg\gamma, t) = \neg D(\gamma, t)$$
$$D(\gamma \oplus \delta, t) = D(\gamma, t) \oplus D(\delta, t) \qquad D([\alpha]\gamma, t) = D(\gamma, \phi(\alpha)).$$

Two formulae γ_1 and γ_2 are *equivalent* iff $D(\gamma_1, t) = D(\gamma_2, t)$ for all D and t. A set $\Gamma \subseteq \Lambda$ of formulae is *satisfiable* iff there exists a development D and a time point $t \in \mathbf{R}$ such that $D(\gamma, t)$ is true for every $\gamma \in \Gamma$. A development D is a *model* of a set $\Gamma \subseteq \Lambda$ of closed formulae iff $D(\gamma, t)$ is true for every $t \in \mathbf{R}$ and $\gamma \in \Gamma$. □

Fact 11 For $\gamma \in \Lambda$ and $\alpha \in \mathcal{T}^*$, $\neg[\alpha]\gamma$ is equivalent to $[\alpha]\neg\gamma$. For a closed formula γ, $D(\gamma, t) = D(\gamma, t')$ for any $t, t' \in \mathbf{R}$ and development D. □

Thus, if γ is closed, we can write $D(\gamma)$ instead of $D(\gamma, t)$.

Now we define the semantics of the action expressions based on models for the basic temporal logic. Inertia (the frame problem) is handled by identifying all time points where a feature f possibly can change its value. Then during every interval where no such change time point exists, f has to have the same value throughout the interval.

Definition 12 Let $D = \langle h, \phi \rangle$ be a development. An action expression $A = \langle \alpha, \pi, \mathtt{Infl}, \epsilon \rangle$ *has effects* in D iff $D(\pi) = \mathrm{T}$. Let $f \in \mathcal{F}$, and define $Chg(D, \textsc{scd}, f, t)$ to be true for a time point $t \in \mathbf{R}$ iff $f \in \mathtt{Infl}$ for some action expression $A = \langle \alpha, \pi, \mathtt{Infl}, \epsilon \rangle \in \textsc{scd}$ that has effects in D, with $\phi(\alpha) = t$. Note that we can only have $Chg(D, \textsc{scd}, f, t)$ true for a finite number of time points for fixed SCD and f.

Let $\Upsilon = \langle \sigma, \textsc{scd}, \textsc{obs} \rangle$ be a scenario description. An *intended model* of Υ is a development $D = \langle h, \phi \rangle$ where

- D is a model of OBS
- For each $A = \langle \alpha, \pi, \mathtt{Infl}, \epsilon \rangle \in \textsc{scd}$ that has effects in D, $D(\epsilon, \phi(\alpha)) = \mathrm{T}$
- For each $f \in \mathcal{F}$ and $s, t \in \mathbf{R}$ with $s < t$ such that for no $t' \in (s, t)$ (open interval), $Chg(D, \textsc{scd}, f, t')$ holds, we have $h(t')(f) = h(s)(f)$ for every $t' \in (s, t)$.

Denote by $Mod(\Upsilon)$ the set of all intended models for a scenario description Υ.

A formula $\gamma \in \Lambda$ is *entailed* by a scenario description Υ, denoted $\Upsilon \models \gamma$, iff γ is true in all intended models of Υ. Υ is *satisfiable* iff $Mod(\Upsilon) \neq \emptyset$. □

Fact 13 If $\Upsilon = \langle \sigma, \textsc{scd}, \textsc{obs} \rangle$ is a scenario description and $\gamma \in \Lambda$ a formula, then $\Upsilon \models \gamma$ iff $\langle \sigma, \textsc{scd}, \textsc{obs} \cup \{\neg\gamma\} \rangle$ is unsatisfiable. □

We comment how this is used in our two examples:

- It is intended that JLH is unsatisfiable adding the observation $[c_1]dry$, and that neither adding $[c_2]hat\_on$ nor $[c_2]\neg hat\_on$ will make it unsatisfiable.
- For SBL, adding $c_2 \neq c_1$ as an observation will make the scenario unsatisfiable.

4 Complexity Results

4.1 Basic Results

It is no surprise that deciding satisfiability for the basic temporal logic is NP-hard. Proofs of NP-completeness, on the other hand, depend on the tractability results.

Proposition 14 Deciding satisfiability of a set $\Gamma \subseteq \Lambda$ is NP-hard.

Proof: Propositional logic is a subset of Λ. □

Corollary 15 Deciding whether a scenario description is satisfiable is NP-hard. □

That these problems are in NP, and thus are NP-complete, is proved in Theorem 22 and Theorem 23.

Interestingly, we can strengthen the result considerably.

Theorem 16 Deciding whether a scenario description is satisfiable is NP-hard, even if action expressions are

unconditional and unary, only Horn observations are allowed, and no relations between time points may be stated.
Proof (sketch): A reduction from 3SAT. □

We now present the key to tractability, which is a linear-programming approach to temporal constraint reasoning, by Jonsson and Backstrom [1996].

Definition 17 Let α and β be linear polynomials with rational coefficients over some set X of variables. Then a *disjunctive linear relation,* DLR, is a disjunction of one or more expressions of the form $\alpha = \beta$, $\alpha \neq \beta$, $\alpha \leq \beta$, $\alpha < \beta$. A DLR is *Horn* iff it contains at most one disjunct with the relation =, < or <.

An assigment m of variables in X to real numbers is a *model* of a set T of DLR's iff all formulae in T are true when taking the values of variables in the DLR's. A set of DLR's is *satisfiable* iff it has a model. □

The following result is the main result of Jonsson and Backstrom [1996].

Proposition 18 Deciding satisfiability of a set of Horn DLR's is polynomial. □

Now we restrict the scenario description language and the form of actions. Furthermore, a structural restriction on scenario descriptions, verifiable in polynomial time, is imposed. We shall define an encoding function that takes a Horn scenario description Υ and returns a set Γ of Horn DLR's such that Γ is satisfiable iff Υ is satisfiable.

4.2 Satisfiability of Horn Formulae is Tractable

First, we code Horn formulae as Horn DLR's.

Definition 19 Let l be a closed literal, and assume the existence of fresh, unique time point variables t_f^α for each $f \in \mathcal{F}$ and $\alpha \in \mathcal{T}^*$. Then $C(l)$ is defined as follows, assuming $R \in \{=, \leq, <, \geq, >\}$ and $f \in \mathcal{F}$.

$$\begin{array}{ll}
C(\mathrm{T}) = \ulcorner 0 = 0 \urcorner & C(\mathrm{F}) = \ulcorner 0 \neq 0 \urcorner \\
C(\alpha R \beta) = \ulcorner \alpha R \beta \urcorner & C(\neg \alpha = \beta) = \ulcorner \alpha \neq \beta \urcorner \\
C(\neg \alpha \leq \beta) = \ulcorner \alpha > \beta \urcorner & C(\neg \alpha < \beta) = \ulcorner \alpha \geq \beta \urcorner \\
C(\neg \alpha \geq \beta) = \ulcorner \alpha < \beta \urcorner & C(\neg \alpha > \beta) = \ulcorner \alpha \leq \beta \urcorner \\
C([\eta]\alpha R \beta) = C(\alpha R \beta) & C([\alpha] f) = \ulcorner t_f^\alpha = 0 \urcorner \\
C(\neg[\alpha]\gamma) = C([\alpha]\neg\gamma) & C([\alpha]\neg f) = \ulcorner t_f^\alpha \neq 0 \urcorner
\end{array}$$

Let $\gamma \in \Lambda$ be a closed Horn formula, and let γ' be obtained from γ by simplifying away occurrences of T and F. Now $\gamma' = \bigvee_i l_i$. Then define $C(\gamma)$ to be the DLR $\delta = \bigvee_i C(l_i)$. Note that δ is always a Horn DLR.

Let $\Gamma \subsetneq \Lambda$ be a set of closed Horn formulae, and $\mathcal{T}$ the set of all time point expressions occurring in Γ. Then $C(\Gamma)$ is defined by

$$\begin{aligned} C(\Gamma) = \{C(\gamma) | \gamma \in \Gamma\} \cup \\ \{C(\neg[\alpha] f \vee \beta \neq \alpha \vee [\beta] f) | f \in \mathcal{F}, \alpha, \beta \in \mathcal{T}\}. \end{aligned}$$

The second set is called the *correspondence equations.* Note that the argument of C in a correspondence equation is equivalent to $[\alpha] f \wedge \beta = \alpha \rightarrow [\beta] f$. □

The following result is crucial.

Theorem 20 Let $T \subsetneq \Lambda$ be a set of closed Horn formulae. Then T is satisfiable iff $C(T)$ is satisfiable.
Proof (sketch): Straightforward: the key is to use the correspondence equations. □

Corollary 21 Deciding satisfiability of sets of closed Horn formulae is polynomial.
Proof: It is clear that the transformation C is polynomial. The result follows from Proposition 18. □

Now we have the results for the proofs of membership in NP for the satisfiability problems of Λ and of scenario descriptions.

Theorem 22 Deciding satisfiability of a set $\Gamma \subsetneq \Lambda$ is NP-complete.
Proof (sketch): By Proposition 14, it remains to prove that the problem is in NP. This is done by letting the set of all literals used in T which are true in a model serve as a polynomial representation of the model. Furthermore, satisfiability of such a set is polynomial, by Corollary 21. □

Theorem 23 Deciding whether a scenario description is satisfiable is NP-complete.
Proof (sketch): By Corollary 15, it remains to prove that the problem is in NP. As a polynomial representation of a model, we use the set of literals used in the scenario description that are true in it, and additional atomic formulae expressing temporal relations. Then checking satisfiability of such sets will be polynomial, like in Theorem 22. □

4.3 Tractable Scenario Descriptions

Using Corollary 21, we see that if we can code scenario descriptions into sets of Horn formulae, we will have a polynomial algorithm for reasoning with scenario descriptions. In order to obtain such a result, we need to restrict what scenario descriptions are allowed.

The strategy can briefly be described as follows: we identify all observation time points which bind a feature value and all time points where an action expression possibly can change a feature value. Then we connect bound literals with biconditionals, between time points where the literal value should not change. E.g. if some action expression changes the value of the feature / at time point α, there exists a $\gamma \in$ OBS which binds / at a time point β, $\alpha < \beta$, and no changes of the value of / occurs between a and β, then $[\alpha] f \leftrightarrow [\beta] f$ should be added to the theory. This formula can be rewritten in Horn form. The example represents one of the six cases (case 3). The other cases are similar.

The restrictions are basically two: First we will have to represent action expressions as Horn formulae *(restricted* action expressions). Second, the scenario descriptions must be *ordered,* we could, e.g., not remove the restriction $\alpha < \beta$ in the example above.

Definition 24 Let $\Upsilon = \langle \sigma, \mathrm{SCD}, \mathrm{OBS} \rangle$ be a scenario description. For each $f \in \mathcal{F}$, define $E_f = \{\alpha | \langle \alpha, \pi, \mathtt{Infl}, \epsilon \rangle \in \mathrm{SCD} \wedge f \in \mathtt{Infl}\}$ and $C_f = \{\alpha | \alpha \textit{ binds } f \textit{ in } \gamma | \gamma \in O\}$, for $O = \mathrm{OBS} \cup$

$\{\pi | \langle \alpha', \pi, \mathtt{Infl}, \epsilon \rangle \in \textsc{scd}\}$. E_f is *ordered* iff for $\alpha, \beta \in E_f$, exactly one of $\alpha < \beta$, $\alpha = \beta$ and $\alpha > \beta$ is consistent with[2] OBS. For $\alpha, \beta \in E_f$, E_f ordered, we define $\alpha \prec_f \beta$ iff $\alpha < \beta$ is consistent with OBS, $-\infty \prec_f \alpha$ iff for no $\beta \in E_f$, $\beta < \alpha$ is consistent with OBS, and $\alpha \prec_f \infty$ iff for no $\beta \in E_f$, $\alpha < \beta$ is consistent with OBS.

Let $\alpha \in E_f$, $\beta \in \mathcal{T}^*$, and define $\alpha \lll_f \beta$ iff $\alpha \leq \beta$ is consistent with OBS, $\alpha > \beta$ is inconsistent with OBS, and for every $\gamma \in E_f$, $\alpha < \gamma \leq \beta$ is inconsistent with OBS. Also define $-\infty \lll_f \beta$ iff for every $\alpha \in E_f$ $\alpha \leq \beta$ is inconsistent with OBS.

If for all $f \in \mathcal{F}$, E_f is ordered, and for all $\omega \in C_f$, $\alpha \lll_f \omega$ for some $\alpha \in E_f \cup \{-\infty\}$, then Υ is *ordered*. The last condition says that for each observation of a feature f, there is a unique change of f which sets its value, or it precedes all changes of f. □

Proposition 25 Testing if a scenario description Υ is ordered is polynomial, if OBS is Horn. □

Definition 26 Let $A = \langle \alpha, \pi, \mathtt{Infl}, \epsilon \rangle$ be an action expression. Then A is *restricted* iff either of the following holds:

- π is T and ϵ is a conjunction of propositional Horn formulae
- π is a disjunction of negative literals, and ϵ is either T, or a conjunction of negative literals.

An ordered scenario description $\Upsilon = \langle \sigma, \textsc{scd}, \textsc{obs} \rangle$ is *restricted* iff every action expression in SCD is restricted and OBS is Horn. A restricted scenario description is *normal* iff it is ordered, and for every action expression $A \in \textsc{scd}$, the following holds:

- π is T and ϵ is a propositional Horn formula
- π is a negative literal and ϵ is either T or a negative literal.

□

Both the examples previously stated (Example 7 and Example 8) are restricted, which is easy to verify.

The following result will make the forthcoming proofs easier.

Proposition 27 Let Υ be a restricted scenario description. Then we can in polynomial time construct an equivalent normal scenario description Υ'.

Proof (sketch): It is easy to replace the action expressions of Υ by normal action expressions which are equivalent. □

Thus, we can assume that our restricted scenario descriptions are normal. Next, we define the function Φ which transforms scenario descriptions into sets of Horn formulae.

Definition 28 First let $A = \langle \alpha, \pi, \mathtt{Infl}, \epsilon \rangle$ be a normal action expression. We have three cases for Φ:

- If $\pi \equiv$ T and $\epsilon \equiv h$ for $h = \bigvee_j l_j$ propositional Horn, then define $\Phi(A) = \{\bigvee_j [\alpha] l_j\}$
- If $\pi \equiv \neg l$ and $\epsilon \equiv$ T, then define $\Phi(A) = \emptyset$
- If $\pi \equiv \neg l$ and $\epsilon \equiv \neg m$, then define $\Phi(A) = \{l \vee [\alpha] \neg m\}$.

The restriction to normal action expressions should be clear; it implies that $\Phi(A)$ is Horn. For a set $\mathcal{S}$ of action expressions, define $\Phi(\mathcal{S}) = \bigcup_{A \in \mathcal{S}} \Phi(A)$.

Let $\Upsilon = \langle \sigma, \textsc{scd}, \textsc{obs} \rangle$ be a restricted scenario description with $\sigma = \langle \mathcal{T}, \mathcal{F} \rangle$, and without loss of generality, assume that each feature in $\mathcal{F}$ occurs in SCD or OBS. Also let b be a fresh time point variable (b standing for "beginning"), and for each $f \in \mathcal{F}$, add a new feature f'. A few construction steps (basically corresponding to the possible relations between time points in E_f and C_f) are necessary. We provide the intuitions of the constructions in connection to them.

1. $\Gamma_1 = \textsc{obs} \cup \Phi(\textsc{scd}) \cup \{b < \alpha | \alpha \in E_f \cup C_f, f \in \mathcal{F}\}$.
The observations, the transformed action expressions, and an initial time point, are added.

2. $\Gamma_2 = \{\neg[\alpha]f \vee [b]f, [\alpha]f \vee \neg[b]f | f \in \mathcal{F}, \alpha \in C_f, -\infty \lll_f \alpha\}$.
No action expression influences f before α where it is bound by an observation, therefore f should have the same value at b as at α. Note that the members of Γ_2 can be rewritten to $[\alpha]f \leftrightarrow [b]f$.

3. $\Gamma_3 = \{\neg[\alpha]f \vee [\beta]f, [\alpha]f \vee \neg[\beta]f | f \in \mathcal{F}, \alpha \in E_f, \beta \in C_f, \alpha \lll_f \beta\}$.
f is influenced at α and bound by an observation at a later time point β. No actions have effects between α and β, therefore f should have the same value at β as it had at α.

4. $\Gamma_4 = \{\neg[\alpha]f' \vee [b]f, [\alpha]f' \vee \neg[b]f | f \in \mathcal{F}, \alpha \in E_f, -\infty \prec_f \alpha\}$.
This case resembles case 2, with the difference that f is influenced at α. Therefore, we introduce a new feature symbol f' which has the same value at α as f has at b. The new symbols will be handled properly below, in case 6.

5. $\Gamma_5 = \{\neg[\alpha]f \vee [\beta]f', [\alpha]f \vee \neg[\beta]f' | f \in \mathcal{F}, \alpha, \beta \in E_f, \alpha \prec_f \beta\}$.
This relates to case 3, as case 4 relates to case 2.

6. Since E_f is ordered, we can form equivalence classes T_f^α of time points for $\alpha \in E_f$ by
$T_f^\alpha = \{\beta \in E_f | \alpha = \beta \text{ is consistent with OBS}\}$.
For each $\alpha \in E_f$, define
$\mathcal{P}_f^\alpha = \{\pi | \langle \beta, \pi, \mathtt{Infl}, \epsilon \rangle \in \textsc{scd} \wedge f \in \mathtt{Infl} \wedge \beta \in T_f^\alpha\}$.
Now set $\Gamma_6 =$
$\{\bigvee \mathcal{P}_f^\alpha \vee \neg[\alpha]f \vee [\alpha]f', \bigvee \mathcal{P}_f^\alpha \vee \neg[\alpha]f' \vee [\alpha]f | \alpha \in E_f\}$.
First note that this set is equivalent to the set $\{\neg \bigvee \mathcal{P}_f^\alpha \to ([\alpha]f \leftrightarrow [\alpha]f') | \alpha \in E_f\}$.
Here we ensure that when actions do not have effects, f will have the same value as it had the last time it was changed. This value is held by the feature f'.

Now set $\Phi(\Upsilon) = \bigcup_i \Gamma_i$. It is clear that the transformation performed by Φ is polynomial. □

Theorem 29 Let Υ be a restricted scenario description, and set $\Gamma = C(\Phi(\Upsilon))$. Then Υ is satisfiable iff Γ is satisfiable.

[2]A formula γ is consistent with a set Γ iff $\Gamma \cup \{\gamma\}$ is satisfiable.

Proof (sketch): The key is that the sets T, force features to have values being satisfied in any intended model, and vice versa. □

Theorem 30 Deciding satisfiability (and entailment) for restricted scenario descriptions is polynomial. □

5 Discussion

One piece of related work is the approach by Schwalb et al. [1994] to reasoning about propositions being true at time points. Their choice for obtaining an algorithm is to code both propositions and temporal relations into propositional logic, whereas we do the opposite. However, their tractable inference algorithm is not complete, and they define no measure on when the correct inferences will be obtained, so it is very difficult to relate it to our approach. Furthermore they cannot handle inertia adequately: there, propositions may always change when actions are performed, but certainly this is undesirable if actions which do not affect all features are used.

In this paper our concern has been computational complexity for reasoning about action. It is important to note that although we have provided polynomial algorithms for the reasoning tasks, these can hardly be considered efficient. The important results, however, are that there *exist* polynomial algorithms; the next obvious step is to also make them *fast*. For efficient implementation, there is one direction we are particularly interested in investigating: since the technique used for achieving tractability can be described as an encoding of our logic as temporal constraints for which there is a tractable algorithm for determining satisfiability, it should be possible to do something similar for other tractable temporal algebras, for example those identified in the papers [Drakengren and Jonsson, 1996; 1997]. Also, an algorithm for a purely *qualitative* scenario description language (i.e. not involving metric time) would probably have a faster satisfiability-checker.

We have shown that satisfiability of scenario descriptions is NP-complete within our formalism. We feel that it would be a mistake to interpret this negatively. On the contrary, one could argue (in lines with [Gottlob, 1996]) that this would imply that many approximations, powerful heuristics and non-trivial tractable subsets of problems for reasoning about action remain to be found. This paper is a step on the way in this endeavour.

6 Conclusions

We have presented a temporal logic and an extension for reasoning about action from which tractable subsets have been extracted. This has been done with an encoding of the logic to Horn DLR's. The formalism is narrative based with continuous time, and the world is modelled with scenario descriptions consisting of action expressions and observations. It is possible to model nondeterminism, concurrency and memory of actions. Time is represented with linear polynomials with rational coefficients over real valued variables.

Acknowledgements

Thanks to Simin Nadjm-Tehrani, Peter Jonsson, Patrick Doherty, Lars Karlsson and Erik Sandewall for help and comments.

References

[AAAI, 1996] *Proc. 13th (US) Nat'l Conf. on Artif. Inteli (AAAI-96),* Portland, OR, USA, 1996.

[Alur and Henzinger, 1989] R. Alur and T.A. Henzinger. A really temporal logic. In *Proceedings of the 30th Annual Symposium on Foundations of Computer Science,* pages 164-169. IEEE Computer Society Press, 1989.

[Doherty, 1994] Patrick Doherty. Reasoning about action and change using occlusion. In Anthony G. Cohn, editor, *Proc. 11th Eur. Conf. on Artif. Intell. (ECAI '94),* Amsterdam, Netherlands, August 1994. Wiley.

[Drakengren and Jonsson, 1996] Thomas Drakengren and Peter Jonsson. Maximal tractable subclasses of Allen's interval algebra: Preliminary report. In AAAI [1996], pages 389-394.

[Drakengren and Jonsson, 1997] Thomas Drakengren and Peter Jonsson. Eight maximal tractable subclasses of Allen's algebra with metric time. *J. Artif. Intell. Res.,* 1997. To appear.

[Gelfond *et al.,* 1991] M. Gelfond, V. Lifschitz, and A. Rabinov. What are the limitations of the situation calculus? In R. Boyer, editor, *Essays in Honor of Woody Bledsoe,* pages 167-179. Kluwer Academic, 1991.

[Ginsberg, 1996] M. Ginsberg. Do computers need common sense? In *KR '96,* 1996.

[Giunchiglia and Lifschitz, 1995] E. Giunchiglia and V. Lifschitz. Dependent fluents. In *IJCAI '95,* 1995.

[Gottlob, 1996] G. Gottlob. Complexity and expressive power of KR formalisms. In *KR '96,* 1996.

[Gustafsson and Doherty, 1996] J. Gustafsson and P. Doherty. Embracing occlusion in specifying the indirect effects of actions. In *KR '96,* 1996.

[Jonsson and Backstrom, 1996] Peter Jonsson and Christer Backstrom. A linear-programming approach to temporal reasoning. In AAAI [1996], pages 1235-1240.

[Koubarakis, 1996] M. Koubarakis. Tractable disjunctions of linear constraints. In *Proc. of the 2nd Int'l Conf. on Princ. and Pract. for Constr. Prog.,* pages 297-307, Cambridge, MA, August 1996.

[Sandewall, 1994] Erik Sandewall. *Features and Fluents. The Representation of Knowledge about Dynamical Systems, volume I.* Oxford University Press, 1994.

[Schwalb *et al.,* 1994] E. Schwalb, K. Kask, and R. Dechter. Temporal reasoning with constraints on fluents and events. In *AAAI'94,* 1994.

TEMPORAL REASONING

Temporal Reasoning 3