Taming the Complexity of Temporal Epistemic Reasoning

Clare Dixon, Michael Fisher, and Boris Konev

Department of Computer Science, University of Liverpool, Liverpool, U.K.
{CLDixon, MFisher, Konev}@liverpool.ac.uk

Abstract. Temporal logic of knowledge is a combination of temporal and epistemic logic that has been shown to be very useful in areas such as distributed systems, security, and multi-agent systems. However, the complexity of the logic can be prohibitive. We here develop a refined version of such a logic and associated tableau procedure with improved complexity but where important classes of specification can still be described. This new logic represents a combination of an "exactly one" temporal logic with an S5 multi-modal logic again restricted to the "exactly one" form.

1 Introduction

While temporal logic has been shown to be very useful, particularly in the areas of formal specification and verification [16, 14], there are two problems with its use:

1. *there are cases where* full *temporal logic is* too *expressive*
   — in particular, if we wish to describe the temporal properties of a restricted number of components, only one of which can occur at any moment in time, then the full temporal language forces us to describe the behaviour of all these components and their interactions explicitly;
2. *many applications require the extension of temporal logic with different modalities*
   — in particular, extensions with various modal logics (such as those describing knowledge or belief) are *very* useful.

Taking (2) first, there has been considerable work on defining, mechanising and applying combined temporal and modal logics.

A very popular modal approach is to use a logic of *knowledge*, i.e. an *epistemic* logic (typically S5 modal logic), in order to represent the knowledge that any player/agent/process has [7]. Involving multiple players/agents/processes leads us to *multi-dimensional* logics of knowledge [9] where multiple agents each have an associated notion of knowledge. We can then reason not only about the agent's knowledge of the situation, but also about the agent's knowledge of other agents, the agent's knowledge of other agents' situation, the agent's knowledge of other agents' knowledge about the situation, and so on.

This naturally leads on to *temporal logic of knowledge* [7], which is the combination of propositional discrete, linear temporal logic (LTL) with (S5) modal logic, which has been shown to be very useful in areas such as distributed systems [11, 8], security [4], and multi-agent systems [20, 17]. However, the complexity of such a combined logic is quite high, in particular even with a simple combination such as the fusion of LTL and multi-modal S5 the complexity of satisfiability is PSPACE [7].

In the meantime we have been working on (1). In [6], it was shown that, simply by incorporating *"exactly one" constraints* into a propositional temporal logic, much better computational complexity could be achieved. Essentially sets of propositions were allowed as part of the input (termed *"exactly one"* or *"constrained"* sets) where exactly one proposition from each set must hold at every moment in time. Normal unconstrained propositions were also allowed. Not only did this allow the concise specification of examples such as the representation of automata, basic planning problems, and agent negotiation protocols, but also greatly reduced the complexity of the associated decision procedure [5, 6]. Essentially, this is because (as the name suggests) *exactly* one element of each "exactly one" set must be satisfied at every temporal state. This allowed polynomial complexity concerning the constrained sets within the decision procedure.

In this paper we will now make the obvious connection between (1) and (2) above. Specifically, we here:

- define an "exactly-one" temporal logic of knowledge;
- define a complete tableau system for this new logic;
- explore the computational complexity of the tableau system; and
- explore potential applications of the approach.

Thus, the paper builds on our previous work on "exactly one" temporal logics [6] and tableaux for temporal logics of knowledge [26], but provides a new logic, new tableau system, and significant complexity improvements with potential applications. The tableau algorithm replaces traditional alpha and beta rules with a DPLL-like construction [2] which ensures the exactly one sets hold. The complexity results show how careful organisation of the problem can, in many cases, greatly reduce exponential bounds.

The paper is organised as follows. In Section 2 we introduce XL5, a constrained temporal logic of knowledge. The complexity of satisfiability for this logic is considered in Section 3. In Section 4 we present a tableau algorithm for XL5 and prove its completeness and refined complexity. In Sections 5 and Section 6, we demonstrate the tableau algorithm in practice and identify areas where we believe XL5 may be useful. Finally, in Section 7, we provide concluding remarks and discuss related and future work.

2 A Constrained Temporal Logic of Knowledge

The logic we consider is called "XL5", and its syntax and semantics are essentially that of a propositional temporal logic of knowledge [7], which is in turn

a *fusion* of propositional (linear, discrete) temporal logic [10] and an S5 modal logic of knowledge [12]. The models of such a logic are essentially a set of *timelines*, isomorphic to the Natural Numbers, at which modal relations (of the S5 variety) can link to points (a state occurring at a particular time in a timeline) in other timelines.

The main novelty in XL5 is that formulae of XL5($\mathcal{P}^1, \mathcal{P}^2, \ldots$) are constructed under the restrictions that *exactly* one proposition from every set $\mathcal{P}^i$ is true in every state. Note that propositions may appear in more than one set $\mathcal{P}^i$, i.e. $\mathcal{P}^i \cap \mathcal{P}^j$ may be non-empty for $i \neq j$. Furthermore, we assume that there exists a set of propositions, $\mathcal{A}$, in addition to those defined by the parameters $\mathcal{P}^i$, where for all i, $\mathcal{A} \cap \mathcal{P}^i = \emptyset$, and that these propositions are unconstrained as normal. Thus, XL5() is essentially a standard propositional, linear temporal logic of knowledge, while XL5($\mathcal{P}$,$\mathcal{Q}$,$\mathcal{R}$) is a temporal logic of knowledge containing at *least* the propositions $\mathcal{P} \cup \mathcal{Q} \cup \mathcal{R}$, where $\mathcal{P} = \{p_1, p_2, \ldots, p_l\}$, $\mathcal{Q} = \{q_1, q_2, \ldots, q_m\}$, and $\mathcal{R} = \{r_1, r_2, \ldots, r_n\}$ and any state in a XL5($\mathcal{P}$,$\mathcal{Q}$,$\mathcal{R}$) model must satisfy exactly one of $p_1, p_2, \ldots, p_l$; exactly one of $q_1, q_2, \ldots, q_m$; and exactly one of $r_1, r_2, \ldots, r_n$.

2.1 Syntax

The alphabet of XL5($\mathcal{P}^1, \mathcal{P}^2, \ldots, \mathcal{P}^m$) contains the following symbols:

1. the "exactly one" sets $\mathcal{P}^i = \{p_1^i, p_2^i, \ldots\}$ of atomic propositions and a set of unconstrained propositions $\mathcal{A}$ such that $\mathcal{P}^1 \cup \mathcal{P}^2 \cup \ldots \cup \mathcal{P}^m \cup \mathcal{A} = \textsc{Prop}$;
2. basic classical connectives, $\wedge$, $\vee$, $\Rightarrow$, $\neg$, F and T;
3. a set $\mathsf{Ag} = \{1, \ldots, n\}$ of agents;
4. the unary modal connectives K_i, where $i \in \mathsf{Ag}$; and
5. the temporal connectives, $\bigcirc$ (next), $\Box$ (always), $\diamondsuit$ (sometime), and $\mathcal{U}$ (until).

2.2 Semantics

The semantics of XL5 is based upon *timelines* which are themselves composed of *points*. These are defined as follows. A *timeline*, l, is an infinitely long, linear, discrete sequence of states, indexed by the Natural Numbers. We assume that $\mathcal{TL}$ is the set of all timelines. A *point*, p, is a pair $p = (l, u)$, where $l \in \mathcal{TL}$ is a timeline and $u \in \mathbb{N}$ is a temporal index into l.

Any point (l, u) will uniquely identify a state $l(u)$. Let the set of all points be $Points$. We then let an agent's knowledge accessibility relation R_i hold over $Points$, i.e., $R_i \subseteq Points \times Points$, for all $i \in Ag$. This captures the idea of an agent being uncertain both about which timeline it is in, and how far along that timeline it is. A *valuation* for XL5 is a function that takes a point and a proposition, and says whether that proposition is true (T) or false (F) at that point.

We can now define model structures for XL5. A *model structure*, M, for XL5 is a structure $M = \langle TL, R_1, \ldots, R_n, \pi \rangle$, where:

$$
\begin{array}{lll}
\langle M,(l,u)\rangle \models T & & \\
\langle M,(l,u)\rangle \models p & \text{iff} & \pi((l,u),p) = T \text{ (where } p \in \textsc{Prop}) \\
\langle M,(l,u)\rangle \models \neg\varphi & \text{iff} & \langle M,(l,u)\rangle \not\models \varphi \\
\langle M,(l,u)\rangle \models \varphi \vee \psi & \text{iff} & \langle M,(l,u)\rangle \models \varphi \text{ or } \langle M,(l,u)\rangle \models \psi \\
\langle M,(l,u)\rangle \models K_i\varphi & \text{iff} & \forall l' \in TL, \forall v \in \mathbb{N}, \text{ if } ((l,u),(l',v)) \in R_i \text{ then } \langle M,(l',v)\rangle \models \varphi \\
\langle M,(l,u)\rangle \models \bigcirc\varphi & \text{iff} & \langle M,(l,u+1)\rangle \models \varphi \\
\langle M,(l,u)\rangle \models \varphi \mathcal{U} \psi & \text{iff} & \exists v \in \mathbb{N} \text{ such that } (v \geq u) \text{ and } \langle M,(l,v)\rangle \models \psi, \\
 & & \quad \text{and } \forall w \in \mathbb{N}, \text{ if } (u \leq w < v) \text{ then } \langle M,(l,w)\rangle \models \varphi
\end{array}
$$

Fig. 1. Semantics of XL5

- $TL \subseteq \mathcal{TL}$ is a set of timelines;
- R_i, for all $i \in Ag$, is an agent accessibility relation over $Points$, i.e., $R_i \subseteq Points \times Points$ such that R_i is an equivalence relation; and
- $\pi : Points \times \textsc{Prop} \to \{T, F\}$ is a valuation which satisfies the "exactly one" sets, i.e. makes exactly one element of each set $\mathcal{P}^i$ true.

As usual, we define the semantics of the language via the satisfaction relation '$\models$'. For XL5, this relation holds between pairs of the form $\langle M,(l,u)\rangle$ (where M is a model structure and $(l,u) \in Points$), and XL5 formulae. The rules defining the satisfaction relation for selected operators (others can be defined from these) are given in Fig. 1. Other Boolean and temporal operators can be obtained with the usual equivalences. For the operators '$\diamondsuit$' and '$\Box$' these are $\diamondsuit\psi \equiv T\,\mathcal{U}\,\psi$ and $\Box\psi \equiv \neg\diamondsuit\neg\psi$.

We assume that for each $(l,u) \in Points$ the valuation π satisfies the "exactly one" sets, i.e. makes exactly one element of each set $\mathcal{P}^i$ true. Notice that the logical symbols of XL5 do not allow us to express such global requirements (eg everywhere exactly one of p, q, and r hold), and to represent such constraints *within* the logic, the set of logical operators would have to be extended with universal modalities. In other words, global restrictions on models give more expressivity to the logic.

A formula φ is *satisfied* in a model M if there exist a timeline l such that $\langle M,(l,0)\rangle \models \varphi$. A formula is satisfiable if there exists a model in which it is satisfied. A formula is valid if its negation is unsatisfiable.

3 Complexity of XL5

When it comes to the complexity of reasoning, if only the length of a given formula is taken into account, XL5 does not have any advantages over an unrestricted fusion of $S5_n$ and LTL.

Theorem 1. *The satisfiability problem for XL5($\mathcal{P}$), even if all variables belong to the single constrained set $\mathcal{P}$, is PSPACE-complete.*

Proof. The PSPACE upper bound can be obtained from the complexity of temporal epistemic logic [7]. To prove the lower bound, we reduce the PSPACE-complete satisfiability of multi-modal $S5_n$ to satisfiability of XL5. Let φ be a multi-modal $S5_n$-formula constructed over $\{p_0, \ldots, p_m\}$. Consider a XL5($\{s, p'_0, \ldots, p'_m\}$)-formula φ' obtained from φ by replacing every occurrence of a proposition p_i with $\Diamond p'_i$ for all p_i, $0 \leq i \leq m$, where s and $p'_0, \ldots, p'_m$ are new propositions, which do not occur to φ. Notice that the size of φ' is linear in the size of φ. We show that φ is satisfiable if, and only if, φ' is satisfiable.

Clearly, if φ' is satisfiable then φ is satisfiable. Suppose now that φ has a model. Since φ does not contain temporal operators, its model $M = \langle TL, R_1, \ldots, R_n, \pi\rangle$ is such that, for any j, we have $((l_1, u_1), (l_2, u_2)) \in R_j$ implies $u_1 = u_2 = 0$. A model for φ' is $M' = \langle TL, R_1, \ldots, R_n, \pi'\rangle$, where $\pi'((l, u), p'_i) = T$ if, and only if, $u = i$ and $\pi((l, 0), p_i) = T$; and $\pi'((l, u), s) = T$ if, and only if, $u > n$ or $\pi((l, 0), p_u) = F$ (that is, we set in M' the i-th proposition p'_i true in the i-th moment of time whenever in M the proposition p_i is true in the beginning of time; if p'_i is set to be false in M' in the i-th moment of time, s is true in the same moment). Clearly, M' satisfies the exactly one restriction $\{s, p_0, \ldots, p_m\}$ and $\langle M, (l, 0)\rangle \models p_i$ if, and only if, $\langle M', (l, 0)\rangle \models \Diamond p'_i$. Thus, $M' \models \varphi'$. □

Theorem 2. *The satisfiability problem for two fragments of XL5($\mathcal{P}$)*

- *all variables belong to the single constrained set $\mathcal{P}$, there is one agent and temporal operators are not used; and*
- *all variables belong to the single constrained set $\mathcal{P}$ and modal operators are not used*

is NP-hard.

Proof. To prove the lower NP bound for the class of single-agent formulae without temporal operators, we reduce the Boolean satisfiability problem to satisfiability of XL5 formulae. Let φ be a Boolean formula over variables $p_1, \ldots, p_n$. Let $\psi = s \wedge \varphi'$, where s is a new proposition and φ' is obtained from φ by replacing every occurrence of a proposition p_i with the expression $K\neg p'_i$, where p'_i is a new proposition not used in φ, and let $\mathcal{X} = \{s, p'_1, \ldots, p'_n\}$ be the 'exactly one' constraint.

Suppose that an assignment $\mathcal{I}$ satisfies φ. We show how to construct a model $\mathcal{M} = \langle TL, R, \pi\rangle$. The set of timelines $TL = \{l_0, l_1, \ldots, l_n\}$ (recall that n is the number of variables in φ), the relation R is the full relation of the set of points, and the valuation π is defined as follows

- for all $u \in \mathbb{N}$, $\pi((l_0, u), s) = T$ and for all $i, 1 \leq i \leq n$, $\pi((l_0, u), p'_i) = F$.
- for every $i, 1 \leq i \leq n$ and every $u \in \mathbb{N}$ we have
  - $\pi((l_i, u), p'_j) = F$ for all $j \neq i$
  - $\pi((l_i, u), p'_i) = T$ if, and only if, $\mathcal{I}(p_i) = F$
  - $\pi((l_i, u), s) = T$ if, and only if, $\mathcal{I}(p_i) = T$.

Notice that $\langle \mathcal{M}, (l_0, 0)\rangle \models K\neg p'_i$ if, and only if, $\mathcal{I}(p_i) = T$. Since φ' is obtained from φ by renaming every occurrence of x_i with $K\neg x'_i$ we have $\langle \mathcal{M}, (l_0, 0)\rangle \models \psi$.

The NP lower bound for the subclass of XL5 formulae, in which modal operators are not used, can be obtained from the PSPACE lower bound above by considering propositional formulae instead of $S5_n$ formulae. □

Theorem 3 later demonstrates, however, that XL5 reasoning is tractable if the number of occurrences of temporal and modal operators is bounded.

4 Tableau for XL5

Consider an XL5 formula φ that is to be shown to be satisfiable. The tableau algorithm constructs sets of *extended assignments* of propositions and modal subformulae i.e. a mapping to true or false, that satisfy both the "exactly one" sets and φ. However, rather than using the standard alpha and beta rules (see for example the modal tableau in [12, 26]) these are constructed using a DPLL-based expansion [2]. Next the algorithm attempts to satisfy modal formulae, of the form $\neg K_i\psi$, and temporal formulae, of the form $\bigcirc\psi$ and $\psi_1 \mathcal{U} \psi_2$ (or their negations), made true in such an extended assignment by constructing R_i and "next time" successors which are themselves extended assignments which must satisfy particular subformulae (and the exactly one sets). We begin with some definitions.

Definition 1. *If φ is an XL5 formula, then $sub(\varphi)$ is the set of all subformulae of φ:*

$$sub(\varphi) = \begin{cases} \{\varphi\} & \text{if } \varphi \in \textsc{Prop} \text{ or } \varphi = T \text{ or } \varphi = F \\ \{\neg\psi\} \cup sub(\psi) & \text{if } \varphi = \neg\psi \\ \{\psi * \chi\} \cup sub(\psi) \cup sub(\chi) & \text{if } \varphi = \psi * \chi \text{ where } * \text{ is } \vee, \wedge \text{ or } \Rightarrow \\ \{K_i\psi\} \cup sub(\psi) & \text{if } \varphi = K_i\psi \\ \{\bigcirc\psi\} \cup sub(\psi) & \text{if } \varphi = \bigcirc\psi \\ \{\psi_1 \mathcal{U} \psi_2\} \cup sub(\psi_1) \cup sub(\psi_2) & \text{if } \varphi = \psi_1 \mathcal{U} \psi_2 \end{cases}$$

A formula $\psi \in sub(\varphi)$ is a modal subformula *of φ if, and only if, ψ is of the form $K_i\psi'$ for some ψ'. A formula $\psi \in sub(\varphi)$ is a* temporal subformula *of φ if, and only if, ψ is of the form $\bigcirc\psi'$ or $\psi'_1 \mathcal{U} \psi'_2$ for some ψ', ψ'_1, ψ'_2.*

Definition 2. *Let φ be an XL5 formula, $\textsc{Prop}(\varphi)$ be the set of all propositions occurring in φ, $\textsc{Mod}(\varphi)$ be the set of all modal subformulae of φ, and $\textsc{Temp}(\varphi)$ be the set of all temporal subformulae of φ. We assume, without loss of generality, that $\mathcal{P}^i \subseteq \textsc{Prop}(\varphi)$ for $i : 1 \leq i \leq n$. An* extended assignment *ν for φ is a mapping from $\Sigma(\varphi) = \textsc{Prop}(\varphi) \cup \textsc{Mod}(\varphi) \cup \textsc{Temp}(\varphi)$ to $\{T, F\}$.*

Every extended assignment ν can be represented by a set of formulae

$$\Delta_\nu = \bigcup_{\substack{\psi \in \Sigma(\varphi) \\ \nu(\psi) = T}} \{\psi\} \quad \cup \quad \bigcup_{\substack{\psi \in \Sigma(\varphi) \\ \nu(\psi) = F}} \{\neg\psi\}$$

Let ψ be a XL5 formula such that $\textsc{Prop}(\psi) \subseteq \textsc{Prop}(\varphi)$, $\textsc{Mod}(\psi) \subseteq \textsc{Mod}(\varphi)$ *and* $\textsc{Temp}(\psi) \subseteq \textsc{Temp}(\varphi)$. *An extended assignment ν for φ is* compatible *with ψ if, and only if, the following conditions hold.*

- *For every set $\mathcal{P}^i$, there exists exactly one proposition $p \in \mathcal{P}^i$ such that $\nu(p) = T$ (and so $\nu(q) = F$ for all $q \in \mathcal{P}^i, q \neq p$).*
- *The result of replacing every occurrence of a proposition $p \in \textsc{Prop}(\psi)$ in ψ with $\nu(p)$, every occurrence of a modal subformula $\psi' \in \textsc{Mod}(\psi)$, such that ψ' is not in the scope of another modal or temporal operator in ψ, with $\nu(\psi')$, and every occurrence of a temporal subformula $\chi \in \textsc{Temp}(\psi)$, such that χ is not in the scope of another modal or temporal operator in ψ, with $\nu(\chi)$, evaluates to T.*
- *If $\nu(K_j\chi) = T$, for some modal subformula $K_j\chi$ of ψ, then ν is compatible with χ.*
- *If $\nu(\chi_1 \mathcal{U} \chi_2) = T$, for some temporal subformula $\chi_1 \mathcal{U} \chi_2$ of ψ, then ν is compatible with χ_1 or χ_2.*
- *If $\nu(\chi_1 \mathcal{U} \chi_2) = F$, for some temporal subformula $\chi_1 \mathcal{U} \chi_2$ of ψ, then ν is compatible with $\neg\chi_2$.*

We denote by $\mathcal{N}(\varphi)$ the set of all extended assignments of φ.

Example 1. Within XL5($\{p, q\}$), let $\varphi = \neg K_1(p \wedge \bigcirc K_2 \neg p)$. Suppose ψ is φ itself. Consider the extended assignment ν_1 represented by the set $\Delta_{\nu_1} = \{p, \neg q, \neg K_1(p \wedge K_2\neg p), \bigcirc K_2\neg p, K_2\neg p\}$. Then the first two conditions of compatibility with ψ hold true. Notice, however, that ν_1 is not compatible with ψ since $\nu_1(K_2\neg p) = T$ but $\neg p$ evaluates to F under ν_1. The extended assignment ν_2 represented by the set $\Delta_{\nu_2} = \{p, \neg q, \neg K_1(p \wedge K_2\neg p), \bigcirc K_2\neg p, \neg K_2\neg p\}$ is compatible with ψ.

Lemma 1. *Let φ be an XL5 formula and ψ be its subformula. Then the set of all extended assignments for φ compatible with ψ can be computed in $O\left(|\mathcal{P}^1| \times \ldots \times |\mathcal{P}^n| \times 2^{|\mathcal{A}|} \times 2^k \times 2^t\right)$ time, where $|\mathcal{P}^i|$ is the size of the set $\mathcal{P}^i$ of constrained propositions, $|\mathcal{A}|$ is the size of the set $\mathcal{A}$ of non-constrained propositions, k is the number of modal operators in φ, and t is the number of temporal operators in φ.*

Proof. The set of all extended assignments compatible with ψ can be constructed by the DPLL algorithm, where we first split on elements of $\mathcal{P}^i$ (that requires $O(|\mathcal{P}^1| \times \ldots \times |\mathcal{P}^n|)$ time) and then on elements of $\mathcal{A}$, $\textsc{Mod}(\varphi)$, and $\textsc{Temp}(\varphi)$. □

Note 1. The complexity of TLX (linear time logic parametrised by exactly one sets) given in [6] is polynomial in the size of the number of constrained propositions (that is, propositions belonging to an exactly one set) and does not depend on the number of temporal operators occurring to the formula. This is because [6] only considers formulae in a normal form where temporal operators only occur in the form $\Box(\varphi_1 \Rightarrow \bigcirc\varphi_2)$, and $\Box\diamondsuit\varphi_2$, where φ_1 is a conjunction of literals

and φ_2 is a disjunction of literals. Reduction to the normal form introduces unconstrained propositions, therefore, the complexity result for XL5 is not worse than for TLX. Notice further that in many practical applications (for example, when modelling transition systems), parts of the formula are in the normal form and no reduction is necessary. Thus, in practice, the bound in Lemma 1 may not be reached. Indeed, if x and y belong to the same exactly one set, $\bigcirc x$ and $\bigcirc y$ are also mutually exclusive.

The tableau algorithm constructs a *structure* $H = (S, \eta, R_1, \ldots, R_n, L)$, where:

- S is a set of states;
- $\eta \subseteq S \times S$ is a binary *next-time* relation on S;
- $R_i \subseteq S \times S$ represents an accessibility relation over S for agent $i \in Ag$;
- $L : S \to \mathcal{N}(\varphi)$ labels each state with an extended assignment for φ.

We try to construct a structure from which a model may be extracted, and then delete states in this structure that are labelled with formulae such as $\neg K_i p$, $p \,\mathcal{U}\, q$, which are not satisfied in the structure. Expansion uses the formulae in the labels of each state to build R_i successors and η successors.

Given the XL5 formula φ to be shown unsatisfiable, we now perform the following tableau construction steps.

1. *Initialisation.*
   First, set
   $$S = \eta = R_1 = \cdots = R_n = L = \emptyset.$$
   Construct $\mathcal{F}$, the set of all extended assignments for φ compatible with φ. For each $\nu_i \in \mathcal{F}$ create a new state s_i and let $L(s_i) = \nu_i$ and $S = S \cup \{s_i\}$.
2. *Creating R_i successors.* For any state s labelled by an extended assignment ν, i.e. $L(s) = \nu$ for each formula of the form $\neg K_i \psi \in \Delta_{L(s)}$ create a formula
   $$\psi' = \neg\psi \wedge \bigwedge_{K_i\chi \in \Delta_{L(s)}} \chi \quad \wedge \bigwedge_{K_i\chi \in \Delta_{L(s)}} K_i\chi \quad \wedge \bigwedge_{\neg K_i\chi \in \Delta_{L(s)}} \neg K_i\chi$$
   For each ψ' above construct $\mathcal{F}$, the set of all extended assignments for φ compatible with ψ' and for each member $\nu \in \mathcal{F}$ if there exists a state $s'' \in S$ such that $\nu = L(s'')$ then add (s, s'') to R_i, otherwise add a new state s' to S, labelled by $L(s') = \nu$, and add (s, s') to R_i.
3. *Creating η successors.* Let ν be an extended assignment for φ. Then $next(\nu)$ is the smallest subset of Δ_ν such that whenever:-
   - $\bigcirc\chi \in \Delta_\nu$ then $\chi \in next(\nu)$;
   - $\neg\bigcirc\chi \in \Delta_\nu$ then $\neg\chi \in next(\nu)$;
   - $\chi_1 \,\mathcal{U}\, \chi_2 \in \Delta_\nu$ but ν is not compatible with χ_2, then $\chi_1 \,\mathcal{U}\, \chi_2 \in next(\nu)$; and
   - $\neg(\chi_1 \,\mathcal{U}\, \chi_2) \in \Delta_\nu$ but ν is not compatible with $\neg\chi_1$, then $\neg(\chi_1 \,\mathcal{U}\, \chi_2) \in next(\nu)$.

For any state s labelled by an extended assignment ν, i.e. $L(s) = \nu$ create the set of formulae $\Delta = next(L(s))$. Let ψ' be the conjunction of formulae in Δ. For the formula ψ' above construct $\mathcal{F}$, the set of all extended assignments for φ compatible with ψ' and for each member $\nu \in \mathcal{F}$ if there exists a state $s'' \in S$ such that $\nu = L(s'')$ then add (s, s'') to η, otherwise add a new state s' to S, labelled by $L(s') = \nu$, and add (s, s') to η.

4. *Contraction.*
   Continue deleting any state s where
   - there exists a formula $\psi \in \Delta_{L(s)}$ such that ψ is of the form $\neg K_i \chi$ and there is no state $s' \in S$ such that $(s, s') \in R_i$ and $L(s')$ is compatible with $\neg\chi$,
   - $next(\nu)$ is not empty but there is no $s' \in S$ such that $(s, s') \in \eta$, or
   - there exists a formula $\psi \in \Delta_{L(s)}$ such that ψ is of the form $\chi_1 \mathcal{U} \chi_2$ and $\nexists s' \in S$ such that $(s, s') \in \eta^*$ and $L(s')$ is compatible with χ_2, where η^* is the transitive reflexive closure of η.

   until no further deletions are possible.

If φ is a formula then we say the tableau algorithm is *successful* if, and only if, the structure returned contains a state s such that φ is compatible with $L(s)$. We claim that a formula φ is XL5 satisfiable if, and only if, the tableau algorithm performed on φ is successful.

Theorem 3. *Let $\mathcal{P}^1, \ldots, \mathcal{P}^n$ be sets of constrained propositions, and φ be an XL5$(\mathcal{P}^1, \ldots, \mathcal{P}^n)$ formula such that $\bigcup_{i=1}^{n} \mathcal{P}^i \subseteq \textsc{Prop}(\varphi)$. Then*

- *φ is satisfiable if, and only if, the tableau algorithm applied to φ returns a structure $(S, \eta, R_1, \ldots, R_n, L)$ in which there exists a state $s \in S$ such that φ is compatible with $L(s)$.*
- *The tableau algorithm runs in time polynomial in $\big((k+t) \times |\mathcal{P}^1| \times \ldots \times |\mathcal{P}^n| \times 2^{|\mathcal{A}|+k+t}\big)$, where $|\mathcal{P}^i|$ is the size of the set $\mathcal{P}^i$ of constrained propositions, $|\mathcal{A}|$ is the size of the set $\mathcal{A}$ of non-constrained propositions, k is the number of modal operators in φ, and t is the number of temporal operators in φ.*

Proof. The correctness and completeness of the tableau algorithm can be proved by adapting the correctness and completeness proof given in [26]. The main difference between the two algorithms is that the algorithm in [26] applies propositional tableau expansion rules to formulae and the one given above uses DPLL-based expansion.

For the second part of the theorem, notice that the number of nodes in any structure does not exceed $\big(|\mathcal{P}^1| \times \ldots \times |\mathcal{P}^n| \times 2^{|\mathcal{A}|+k+t}\big)$. When creating R_i successors, we consider at most k formulae of the form $\neg K_i \psi \in \Delta_\nu$, and, by Lemma 1, the set of all extended assignments for φ compatible with ψ' can be computed in $O\big(|\mathcal{P}^1| \times \ldots \times |\mathcal{P}^n| \times 2^{|\mathcal{A}|+k+t}\big)$ time. Similarly, when creating η successors, we consider at most t formulae in $next(\nu)$. Building the structure and applying the contraction rule can be implemented in time polynomial in the structure size. □

5 Example

We now consider a longer example. We base this[1] on the simple card-playing scenario from [22].

Here, an agent (called Wiebe) can hold one of three cards. Each of these cards is in a different suit: *hearts*, *spades*, or *clubs*. The cards are dealt so that Wiebe holds one, one is on the table, and the final one is in a holder (*aka* deck). Following [22] we use simple propositions to represent the position of the cards. So, if $spades_w$ is true, then Wiebe holds a spade, if $clubs_t$ is true, then the clubs card is on the table, if $hearts_h$ is true, then the hearts card is in the holder, etc. Similarly, $K_w spades_w$ means that Wiebe *knows* he holds a spade. And so on.

Now, if we were to specify this scenario in a standard temporal logic of knowledge, we would be forced to specify much background information. For example:

- Wiebe's card is spades or hearts or clubs: $(spades_w \vee clubs_w \vee hearts_w)$
- but Wiebe cannot hold both spades and clubs, both spades and hearts, or both clubs and spades:

$$\neg(spades_w \wedge clubs_w) \wedge \neg(spades_w \wedge hearts_w) \wedge \neg(clubs_w \wedge hearts_w)$$

- And Wiebe knows both of the above, e.g: $K_w(spades_w \vee clubs_w \vee hearts_w)$
- Similarly for the *holder* and the *table*.
- The spades card must be either held by Wiebe or be in the holder or be on the table: $(spades_w \vee spades_h \vee spades_t)$
- but cannot be in more than one place:

$$\neg(spades_w \wedge spades_h) \wedge \neg(spades_w \wedge spades_t) \wedge \neg(spades_h \wedge spades_t)$$

- And again Wiebe knows the above, e.g: $K_w(spades_w \vee spades_h \vee spades_t)$
- Similarly for both the *hearts* and *clubs* cards.
- All the above statements hold globally.

However, we can model this scenario with six "exactly one" sets within XL5($\mathcal{P}^1$, $\mathcal{P}^2$, $\mathcal{P}^3$, $\mathcal{P}^4$, $\mathcal{P}^5$, $\mathcal{P}^6$) where

- $\mathcal{P}^1 = \{spades_w, clubs_w, hearts_w\}$ — Wiebe has exactly one card.
- $\mathcal{P}^2 = \{spades_h, clubs_h, hearts_h\}$ — exactly one card is in the holder.
- $\mathcal{P}^3 = \{spades_t, clubs_t, hearts_t\}$ — exactly one card is on the table.
- $\mathcal{P}^4 = \{spades_w, spades_h, spades_t\}$ — the spades card is in exactly one place.
- $\mathcal{P}^5 = \{clubs_w, clubs_h, clubs_t\}$ — the clubs card is in exactly one place.
- $\mathcal{P}^6 = \{hearts_w, hearts_h, hearts_t\}$ — the hearts card is in exactly one place.

[1] A version of this example was presented as part of *"WiebeFest 2009 — A Celebration of Prof. Wiebe van der Hoek's 50th Birthday"*; hence the principle agent is called 'Wiebe'!

Thus, all the formulae at the beginning of this section are unnecessary.

Now let us try to establish something using the tableau construction. Given the basic scenario, we add some temporal evolution. Specifically, we add the fact that Wiebe comes to know more about the scenario as time passes. Thus, we add:

- originally Wiebe has been dealt the clubs card (but has not looked at the card so doesn't know this yet), so $clubs_w$;
- at the next step Wiebe looks at his card so he knows that he has the *clubs* card, so $\bigcirc K_w clubs_w$.

So, one statement we may try to establish from this is that given the above, sometime Wiebe knows that either the hearts card or the spades card is in the holder.

$$(clubs_w \wedge \bigcirc K_w clubs_w) \Rightarrow \ \diamondsuit K_w(hearts_h \vee spades_h)\,.$$

We replace $\diamondsuit K_w(hearts_h \vee spades_h)$ by $T\,\mathcal{U}\,K_w(hearts_h \vee spades_h)$ and negate the above, giving

$$\varphi = \neg((clubs_w \wedge \bigcirc K_w clubs_w) \Rightarrow \ T\,\mathcal{U}\,K_w(hearts_h \vee spades_h))$$

and begin constructing the tableau for φ.

First we construct the set of extended assignments for φ compatible with φ. To save space we assume that any propositions not mentioned are false. Let

$$\mathcal{I}_0 = \{clubs_w, \bigcirc K_w clubs_w, \neg(T\,\mathcal{U}\,K_w(hearts_h \vee spades_h)), \neg K_w(hearts_h \vee spades_h)\}.$$

Notice that any extended assignment that does not contain $\mathcal{I}_0$ is not compatible with φ.

$$\begin{aligned}
\Delta_{L(s_0)} &= \mathcal{I}_0 \cup \{K_w clubs_w, hearts_h, spades_t\} \\
\Delta_{L(s_1)} &= \mathcal{I}_0 \cup \{K_w clubs_w, hearts_t, spades_h\} \\
\Delta_{L(s_2)} &= \mathcal{I}_0 \cup \{\neg K_w clubs_w, hearts_h, spades_t\} \\
\Delta_{L(s_3)} &= \mathcal{I}_0 \cup \{\neg K_w clubs_w, hearts_t, spades_h\}
\end{aligned}$$

Next we create R_w successors. For s_0 and s_1 let

$$\psi' = K_w clubs_w \wedge \neg K_w(hearts_h \vee spades_h) \wedge clubs_w \wedge \neg(hearts_h \vee spades_h).$$

There are no extended assignments of φ which are compatible with ψ' (essentially $\neg(hearts_h \vee spades_h)$ and $\mathcal{P}^2$ forces $clubs_h$ to hold which contradicts with $clubs_w$ and $\mathcal{P}^5$). Hence during the deletion phase s_0 and s_1 will be deleted as there are no R_w successors of s_0 and s_1 compatible with $\neg(hearts_h \vee spades_h)$. We can, however, construct a number of R_w successors for s_2 and s_3. When we attempt to construct η successors for s_2 and s_3 for $i = 2, 3$ we obtain

$$next(L(s_i)) = \{K_w clubs_w, \neg(T\,\mathcal{U}\,K_w(hearts_h \vee spades_h))\}$$

and

$$\psi'' = K_w clubs_w \wedge \neg(T\,\mathcal{U}\,K_w(hearts_h \vee spades_h))\}.$$

Let

$$\mathcal{I}_1 = \{\neg \bigcirc K_w \mathit{clubs}_w, \neg(T \,\mathcal{U}\, K_w(\mathit{hearts}_h \vee \mathit{spades}_h))\}.$$

We construct the extended assignments for φ which are compatible with ψ'' obtaining $L(s_0)$, $L(s_1)$ and

$$\Delta_{L(s_4)} = \mathcal{I}_1 \cup \{K_w \mathit{clubs}_w, \neg(K_w(\mathit{hearts}_h \vee \mathit{spades}_h)), \mathit{clubs}_w, \mathit{hearts}_h, \mathit{spades}_t\}$$
$$\Delta_{L(s_5)} = \mathcal{I}_1 \cup \{K_w \mathit{clubs}_w, \neg(K_w(\mathit{hearts}_h \vee \mathit{spades}_h)), \mathit{clubs}_w, \mathit{hearts}_t, \mathit{spades}_h\}$$

We add $(s_j, s_0), (s_j, s_1), (s_j, s_4), (s_j, s_5)$ to η for $j = 2, 3$.

If we try to construct R_w successors of s_4 and s_5 we construct ψ' as previously and obtain no extended assignments of φ that satisfy ψ' from the reasons given before. Hence during deletions s_4 and s_5 will be deleted as they have no state s such that $(s_4, s) \in R_w$ or $(s_5, s) \in R_w$ such that $L(s)$ is compatible with $\neg(\mathit{hearts}_h \vee \mathit{spades}_h)$. Hence as s_0 and s_1 have already been deleted for similar reasons then s_2 and s_3 have no η successors and so are deleted. As there is no remaining state compatible with φ the tableau is unsuccessful and so φ is unsatisfiable and $(\mathit{clubs} \wedge \bigcirc K_w \mathit{clubs}_w) \Rightarrow \Diamond K_w(\mathit{hearts}_h \vee \mathit{spades}_h)$ is valid.

6 Potential Application Areas

Temporal logics of knowledge are important in both mainstream Computer Science and AI. Thus, with a variety of such logics with lower complexity, we can potentially target a number of areas.

6.1 Distributed Systems

Temporal logics of knowledge are typically used for the specification and verification of distributed systems and are also used in *knowledge-based protocols* [7]. Here, the idea is that when designing a distributed system, one often makes use of statements such as "if process a_1 knows that process a_2 has received message m_1, then a_1 should *eventually* send message m_2". Temporal logics of knowledge are used to formalise this kind of reasoning; knowledge is given a precise interpretation, in terms of the states of a process.

Thus, the "exactly one" variant can come into its own when we have epistemic or temporal states that are constrained in this way. Most obviously, if a process can be in only one particular mode (e.g. *running*, *suspended*, or *stopped*) then a logic describing this process can utilise the "exactly one" set $\{\mathit{running}, \mathit{suspended}, \mathit{stopped}\}$. Crucially, the process *knows* that its mode must be one of these, and every other process also knows this. For example, a process a in reasoning about its knowledge might construct formulae such as the following (about process b) $K_a K_b(\mathit{running}_a \vee \mathit{suspended}_a \vee \mathit{stopped}_a)$ and so on. Formulae such as these are implicit within the XL5 parametrised by appropriate "exactly one" sets.

6.2 Learning and Knowledge Evolution

As we saw in Section 5, temporal formulae can easily be used to describe how an agent's knowledge changes, for example due to learning, observation or announcements by other agents. Thus, many of the examples given in [23] in terms of *dynamic epistemic logic* [22] can be described in a more concise and efficient manner. We note that many of the examples given in [22] have quite low numbers of modal/dynamic operators and so we might expect a reasonable complexity within our logic. Similarly, the representation of the game Cluedo provided in [3] is concerned with how players' knowledge evolves over time and contains "exactly one" sets relating to the representation of the cards of each player and of the murderer.

6.3 Security

Not only do we gain similar advantages as in the area of distributed systems described above, but can also utilise the "exactly one" sets (and knowledge about them) to simplify security descriptions given in temporal logics of knowledge [4]. For example, reasoning about the fact that a message was sent by exactly one of "*expected_sender*", "*intruder*", or "*error_process*" can be simplified.

6.4 Robotics

Robot swarms are collections of, often simple, robots usually with some task to perform. Although the algorithm controlling each robot is fairly simple it is often challenging to prove desirable properties of the swarm, for example that the group of robots does stay in a connected group and some robot doesn't go off on its own and get lost, or that the task is completed successfully etc.

In [24] a swarm algorithm is specified using temporal logic. Underlying this algorithm we can see a number of "exactly one" sets relating to the robot direction (North, South, East, West); the location of the robot; and related to the robot's internal state. In [15] a swarm of foraging robots is defined. Underlying this description is a transition system that describes the different states a robot can pass through whilst foraging for food, for example leaving the nest, random walk, scanning for food, returning home, depositing food, resting etc. It is clear that, for each robot each of these will form an "exactly one" set. Knowledge may be used here to model the robot's awareness of nearby robots etc. Thus a logic such as XL5 may well help specify and prove properties of more complex robot swarms.

6.5 Planning and Knowledge Representation

The planning problem is given a set of initial conditions, a set of actions and a goal to find a suitable sequence of actions that can take us from the initial conditions to the goal. The planning problem can be represented as a temporal logic satisfiability problem, see for example [1]. Initial conditions can be represented

as formulae holding in the initial state, actions can be modelled by making a next step and recording that the action that has been taken, goals can be represented using eventualities ($\Diamond$-formulae) and invariants can be modelled using the $\Box$ operator. Given a representation of the initial conditions and consequences of actions in temporal logic ($SPEC$) and the representation of the goal in temporal logic ($GOAL$) if we can show that $SPEC \Rightarrow GOAL$ is satisfiable in a model then the model can be inspected to see what sequence of actions was used to reach the goal producing the required plan.

"Exactly one" sets seem to occur often in this domain, for example recording that exactly one action may occur at any moment. Further in the domains of interest, for example blocksworld, transportation and scheduling, exactly one sets also occur widely (for example that the location of transportation vehicles may be at one place at any moment, etc).

7 Concluding Remarks

We have taken recent ideas relating to temporal logics which allow the input of sets of proposition where exactly one from each set must hold and have adapted them to the framework of multi-modal temporal logics of knowledge. We have motivated the need for such constraints by considering a number of application areas. We have provided a tableau based algorithm to prove XL5 formulae which replaces the usual alpha and beta rules with a DPLL-based expansion and analysed its complexity. This shows that the tableau is useful when applied to problems with a large number of constrained propositions and a comparatively low number of unconstrained propositions, modal and temporal operators in the formula to be proved.

7.1 Related and Further Work

This paper defines a logic combining linear-time temporal logic and epistemic modal logic but allowing a number of constrained sets as input. In [21] the authors try and specify implicit facts and knowledge for (non-temporal) games of the form given in Section 5 for the modal logic $S5_n$. This is given in terms of a minimal (in some sense) set of formulae that are satisfied in every world for any model for the problem. However that work focuses on the specification of these facts and knowledge rather than looking at the complexity and decision procedure for a logic with constraints as input.

Here we define a tableau for a temporal logic of knowledge allowing the input of constraints. Tableau for modal logics have been given in [12] for example and tableau for propositional linear-time temporal logics have been given in [25, 19, 13]. The tableau we present here uses a DPLL-like construction to construct tableau states rather than the usual alpha and beta rules. The construction of modal and temporal successors follows what is usually done for modal and temporal tableau.

The authors of this paper have considered decision procedures for propositional linear-time temporal logics allowing the input of constrained sets in [5, 6]. Both assume the temporal formulae are in a normal form with clauses relating to the initial moment, conditions on the next moment given the current state, and eventualities. This is different than the approach here where, as usual, the tableau algorithm can be applied to any XL5 formula and does not require translation into normal form. In [5] a tableau-like structure (known as an Incremental Behaviour Graph) is constructed and allows more expressive constraints than we do here (eg exactly n from a set holding or less than n from a set holding). The paper [6] defines a resolution calculus. We note that with constrained logics just involving temporal logic we can express the constraints within the logic itself using the $\Box$-operator. However for XL5 we cannot express the constraints without adding a universal modality to the syntax, so we extend the expressivity of the logic.

Regarding further work, primarily we are interested in implementing the tableau system and applying the logical framework to applications outlined in Section 6. In addition, we would explore different combinations of logics. For example, tableau-based methods have been used for the Belief, Desire, Intention (BDI) logics of Rao and Georgeff in [18]. Whilst here we have considered propositional-linear time temporal logics combined with the modal logic of knowledge it would be fairly easy to amend this to combine with belief logics (KD45) or the logics KD for desire or intention.

Acknowledgements

The work of Fisher was partially supported by EPSRC grant EP/F033567 ("Verifying Interoperability Requirements in Pervasive Systems") and the work of Dixon was partially supported by EPSRC grant EP/D060451 ("Practical Reasoning Approaches for Web Ontologies and Multi-Agent Systems").

References

1. S. Cerrito and M. C. Mayer. Using linear temporal logic to model and solve planning problems. In *Proceedings of Artificial Intelligence : Methodology, Systems, and Applications (AIMSA)*, volume 1480 of *LNCS*, pages 141–152. Springer, 1998.
2. M. Davis, G. Logemann, and D. Loveland. A Machine Program for Theorem-Proving. *Commun. ACM*, 5(7):394–397, 1962.
3. C. Dixon. Using Temporal Logics of Knowledge for Specification and Verification–a Case Study. *Journal of Applied Logic*, 4(1):50–78, 2006.
4. C. Dixon, M. C. Fernández Gago, M. Fisher, and W. van der Hoek. Temporal Logics of Knowledge and their Applications in Security. *Electronic Notes in Theoretical Computer Science*, 186:27–42, 2007.
5. C. Dixon, M. Fisher, and B. Konev. Temporal Logic with Capacity Constraints. In *Proc. 6th Int. Symposium on Frontiers of Combining Systems*, volume 4720 of *LNCS*, pages 163–177. Springer, 2007.
6. C. Dixon, M. Fisher, and B. Konev. Tractable Temporal Reasoning. In *Proc. Int. Joint Conference on Artificial Intelligence (IJCAI)*. AAAI Press, 2007.

7. R. Fagin, J. Halpern, Y. Moses, and M. Vardi. *Reasoning About Knowledge*. MIT Press, 1995.
8. R. Fagin, J. Y. Halpern, and M. Y. Vardi. What Can Machines Know? On the Properties of Knowledge in Distributed Systems. *Journal of the ACM*, 39:328–376, 1996.
9. D. Gabbay, A. Kurucz, F. Wolter, and M. Zakharyaschev. *Many-Dimensional Modal Logics: Theory and Applications*. Number 148 in Studies in Logic and the Foundations of Mathematics. Elsevier Science, 2003.
10. D. Gabbay, A. Pnueli, S. Shelah, and J. Stavi. The Temporal Analysis of Fairness. In *Proc. Seventh ACM Symposium on the Principles of Programming Languages (POPL)*, pages 163–173, January 1980.
11. J. Y. Halpern and Y. Moses. Knowledge and Common Knowledge in a Distributed Environment. *J. ACM*, 37(3):549–587, 1990.
12. J. Y. Halpern and Y. Moses. A Guide to Completeness and Complexity for Modal Logics of Knowledge and Belief. *Artificial Intelligence*, 54:319–379, 1992.
13. G. Janssen. *Logics for Digital Circuit Verification: Theory, Algorithms, and Applications*. PhD thesis, Eindhoven University of Technology, Eindhoven, The Netherlands, 1999.
14. L. Lamport. *Specifying Systems: The TLA+ Language and Tools for Hardware and Software Engineers*. Addison Wesley Professional, 2003.
15. W. Liu, A. Winfield, J. Sa, J. Chen, and L. Dou. Strategies for energy optimisation in a swarm of foraging robots. In *SAB'06 Swarm Robotics Workshop*, 2006.
16. Z. Manna and A. Pnueli. *The Temporal Logic of Reactive and Concurrent Systems: Specification*. Springer-Verlag, New York, 1992.
17. R. C. Moore. *Logic and Representation*. Lecture Notes. Center for the Study of Language and Information (CSLI), 1994.
18. A. S. Rao and M. Georgeff. BDI Agents: from theory to practice. In *Proc. First Int. Conference on Multi-Agent Systems (ICMAS)*, pages 312–319, San Francisco, USA, 1995.
19. S. Schwendimann. *Aspects of Computational Logic*. PhD thesis, University of Bern, Switzerland, 1998.
20. W. van der Hoek. Systems for Knowledge and Beliefs. *Journal of Logic and Computation*, 3(2):173–195, 1993.
21. H. van Ditmarsch, W. van der Hoek, and B. Kooi. Descriptions of game states. In *Games, Logic, and Constructive Sets*, number 161 in CSLI Lecture Notes, pages 43–58. CSLI Publications, 2003.
22. H. van Ditmarsch, W. van der Hoek, and B. Kooi. Playing Cards with Hintikka — An Introduction to Dynamic Epistemic Logic. *Australasian Journal of Logic*, 3:108–134, 2005.
23. H. van Ditmarsch, W. van der Hoek, and B. Kooi. *Dynamic Epistemic Logic*, volume 337 of *Synthese Library Series*. Springer, 2007.
24. A. Winfield, J. Sa, M.-C. Fernández-Gago, C.Dixon, and M. Fisher. On Formal Specification of Emergent Behaviours in Swarm Robotic Systems. *Int. Journal of Advanced Robotic Systems*, 2(4):363–370, 2005.
25. P. Wolper. The Tableau Method for Temporal Logic: An Overview. *Logique et Analyse*, 110–111:119–136, June-Sept 1985.
26. M. Wooldridge, C. Dixon, and M. Fisher. A Tableau-Based Proof Method for Temporal Logics of Knowledge and Belief. *Journal of Applied Non-Classical Logics*, 8(3):225–258, 1998.