Tableaux for Temporal Logics of Knowledge: Synchronous Systems of Perfect Recall or No Learning

Clare Dixon, Cláudia Nalon, Michael Fisher

Department of Computer Science
University of Liverpool
Liverpool, UK

{clare,claudia,michael}@csc.liv.ac.uk
http://www.csc.liv.ac.uk/~{clare,claudia,michael}

Overview

- Temporal logics of knowledge are useful for specifying complex situations requiring dynamic and informational components, e.g. agent based systems.
- Often, we need to consider how knowledge evolves over time, i.e. to allow interaction between the time and knowledge components (this increases the complexity).
- There are few proof methods for combined logics without interactions and even fewer that include non-trivial interactions.
- Here we present tableaux based proof methods for a (linear time) temporal logic of knowledge allowing interactions, namely those of synchrony and perfect recall and synchrony and no learning.

Plan

- Temporal Logics of Knowledge (TLK).
- TLK with synchrony and perfect recall and synchrony and no learning.
- Tableaux for TLK with synchrony and perfect recall/no learning.
- Example.
- Soundness, completeness and termination.
- Related work.
- Conclusions.
- Future work.

Syntax

The formulae of *KL* (the fusion of PTL and S5) are constructed using:-

- a set $\mathcal{P}$ of proposition symbols and the constants **false** and **true**;
- the connectives $\neg$, $\vee$, $\wedge$, $\Rightarrow$, $\bigcirc$, $\Diamond$, $\Box$, $\mathcal{U}$, $\mathcal{W}$ and K

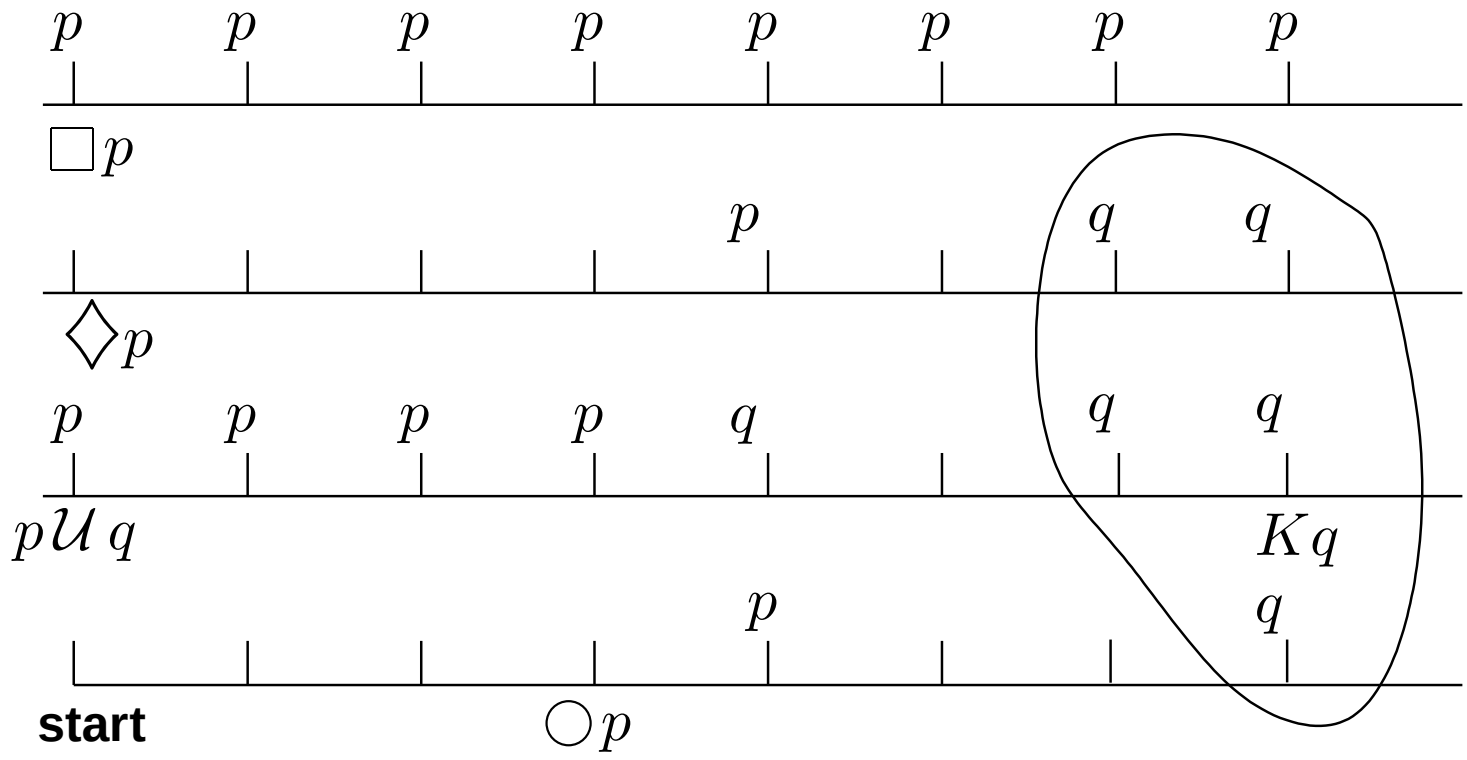
Well-formed formulae of *KL* (WFF_K) are:-

- **false** and **true** and any element of $\mathcal{P}$ is in WFF_K;
- if A and B are in WFF_K then so are

$$\neg A \qquad A \vee B \qquad A \wedge B \qquad A \Rightarrow B \qquad K\,A$$
$$\Diamond A \qquad \Box A \qquad A\,\mathcal{U}\,B \qquad A\,\mathcal{W}\,B \qquad \bigcirc A$$

Semantics

A *model*, M, for KL is a structure $M = \langle TL, R, \pi \rangle$.



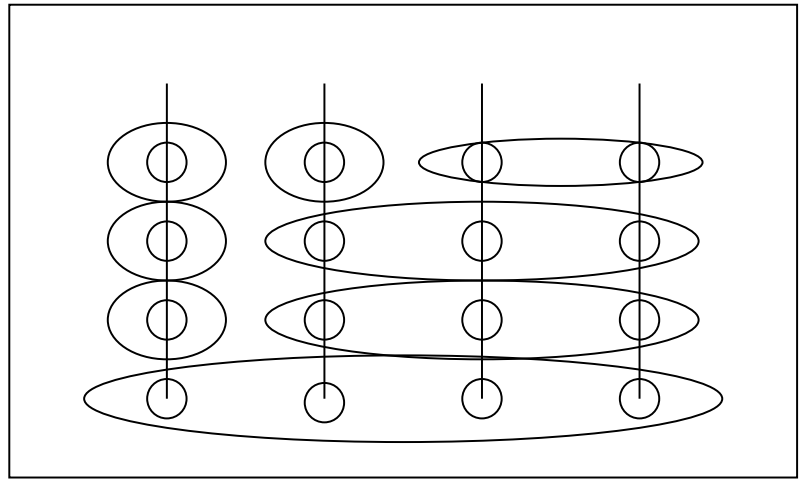
where

$$p \mathcal{W} q \equiv p \mathcal{U} q \vee \Box p$$

Synchrony and Perfect Recall (SPR)

For systems with *synchrony and perfect recall*, if $((s, m+1), (r, n+1)) \in R$ then $((s, m), (r, n)) \in R$ and $m = n$.

Typical model:



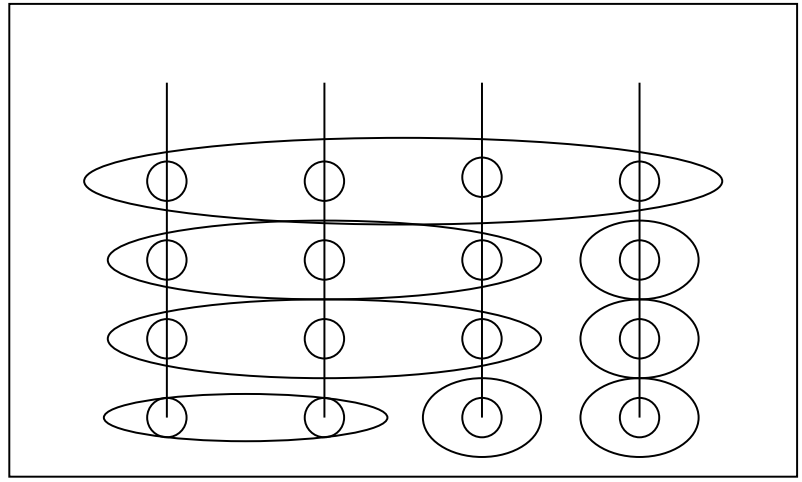
Synchrony and perfect recall (KL^{SPR}) can be axiomatised by adding the following axiom to the axioms of PTL and S5 (Halpern & Vardi).

$$K \bigcirc \varphi \Rightarrow \bigcirc K \varphi.$$

Synchrony and No Learning (SNL)

For systems with *synchrony and no learning*, if $((s,m),(r,n)) \in R$ then $((s,m+1),(r,n+1)) \in R$ and $m = n$.

Typical model:



Synchrony and no learning (KL^{SNL}) can be axiomatised by adding the following axiom to the axioms of PTL and S5 (Halpern & Vardi).

$$\bigcirc K\varphi \Rightarrow K \bigcirc \varphi.$$

Tableaux Systems

- To show a formula φ is valid it is negated and the tableau construction applied to $\neg\varphi$.
- Tableaux systems attempt to build a structure from which a model can be constructed.
- If no such structure can be constructed then $\neg\varphi$ is unsatisfiable and φ valid.
- The structure is constructed systematically by decomposing formulae dependent on their structure. Alpha rules add both consequences to the current state. Beta rules give two alternatives.
- Here we translate (efficiently) formulae first into a satisfiability preserving normal form and add clauses to account for SPR/SNL.

Definition of the Normal Form

Formulae in SNF$_K$ are of the general form $\Box^* \bigwedge_j T_j$ where each T_j, is of the form:-

$$
\begin{array}{rcll}
\textbf{start} & \Rightarrow & l & \text{(\textit{initial} clause)} \\
k & \Rightarrow & \bigcirc l & \text{(\textit{step} clause)} \\
k & \Rightarrow & \Diamond l & \text{(\textit{sometime} clause)} \\
k & \Rightarrow & m & \text{(\textit{modal} clause)} \\
\textbf{true} & \Rightarrow & \bigvee_{b=1}^{r} l_b & \text{(\textit{literal} clause)}
\end{array}
$$

where k, l and l_b are literals and m are Kl or $\neg Kl$ and $\Box^*$, meaning informally 'in every reachable state', can be defined in terms of K, $\Box$ and 'always in the past'.

SPR/SNL Step Clauses

We allow additional forms of step clauses.

$$K \bigvee_{a=1}^{g} k_a \Rightarrow \bigcirc K \left(\bigvee_{b=1}^{r} l_b \right) \quad \text{an SPR step clause}$$

$$\neg K \neg \bigwedge_{a=1}^{g} k_a \Rightarrow \bigcirc \neg K \neg \left(\bigwedge_{b=1}^{r} l_b \right) \quad \text{an SNL step clause}$$

- A formula is in ESNF$_{SPR}$ (ESNF$_{SNL}$), if it is in SNF$_K$ or it is an SPR (resp. SNL) step clause.
- They are formed by applying the SPR (SNL) axiom to subsets of step clauses.
- The idea is that these clauses capture the SPR/SNL requirements on sets of clauses.

The General Approach

- We try to apply the synchrony and perfect recall (or synchrony and no learning) constraint to step clauses before constructing the tableau.
- Recall the axiom that captures SPR is $K\bigcirc\varphi \Rightarrow \bigcirc K\varphi$.
- Consider a step clause $a \Rightarrow \bigcirc b$.
- Then we apply the following reasoning.

$$
\begin{array}{ll}
K(a \Rightarrow \bigcirc b) & \text{(due to the } \Box^* \text{ operator)} \\
Ka \Rightarrow K\bigcirc b & \text{(distributing the } K \text{ operator)} \\
Ka \Rightarrow \bigcirc Kb & \text{(applying the SPR axiom)}
\end{array}
$$

We add this last clause to the set of clauses.

The General Approach(Cont.)

- We do something similar for SNL (using the contrapositive of the step clauses and axiom).
- We must also take disjunctions of right and left sides of subsets of step rules, before we distribute the K operator and apply the SPR axiom.
- We must also take conjunctions of right and left sides of subsets of step rules, before we distribute the K operator and apply the SNL axiom.
- Because $\Diamond\varphi \equiv \varphi \vee \bigcirc\Diamond\varphi$ we add clauses to deal with distributing a K and applying the relevant axiom to the right hand side of sometime clauses.
- Two main parts: the internal construction of states; and adding new states to satisfy $\bigcirc\varphi$ or $\neg K\varphi$ formulae.

Tableau Construction Rules

Alpha Rules

α	α_1	α_2
$\phi \wedge \psi$	ϕ	ψ
$\neg(\phi \vee \psi)$	$\neg\phi$	$\neg\psi$

Beta Rules

β	β_1	β_2
$\phi \vee \psi$	ϕ	ψ
$\neg(\phi \wedge \psi)$	$\neg\phi$	$\neg\psi$
$\Diamond\phi$	ϕ	$\neg\phi \wedge \bigcirc\Diamond\phi$

Delta Rules

δ	δ_1
$K\phi$	ϕ
$\neg\neg\phi$	ϕ
$\neg$**true**	**false**
$\neg$**false**	**true**

Construction of Propositional Tableaux

Given a set of formulae Δ and ESNF$_{SPR/SNL}$ clauses T:

1. If **true** $\Rightarrow \bigvee_{b=1}^{r} l_b$ in T add $\bigvee_{b=1}^{r} l_b$ to Δ.
2. (a) For each $k \Rightarrow \bigcirc l$ in T replace Δ by $\Delta \cup \{\neg k\}$ and $\Delta \cup \{k, \bigcirc l\}$.
   (b) For each $k \Rightarrow Kl$ in T replace Δ by $\Delta \cup \{Kl\}$ and $\Delta \cup \{\neg k, \neg Kl\}$.
   (c) Similar rules for other modal, SPR, SNL clauses.
3. (a) If $\varphi \in \Delta$ is an α formula replace Δ by $\Delta \cup \{\alpha_1, \alpha_2\}$
   (b) If $\varphi \in \Delta$ is an β formula replace Δ by $\Delta \cup \{\beta_1\}$ and $\Delta \cup \{\beta_2\}$
   (c) If $\varphi \in \Delta$ is an δ formula replace Δ by $\Delta \cup \{\delta_1\}$
4. Delete any Δ that contains φ and $\neg\varphi$ or **false**.

Tableau for SPR/SNL (Outline)

1. Translate into normal form and add the new SPR/SNL clauses. Call this set T.
2. Construct the set of propositional tableaux for the right hand side of initial clauses and T. Use as initial states.
3. Create new states in the same equivalence class (to satisfy $\neg K\varphi$ formulae).
4. Create new next-time states (to satisfy $\bigcirc\varphi$ formulae).
5. Deletion rules. Delete any state such that there is no state to satisfy:-
   (a) $\neg K\varphi$ in the same equivalence class;
   (b) $\bigcirc\varphi$ in the next-time relation;
   (c) $\diamondsuit\varphi$ reachable via the next-time relation.

Example

We show $K\bigcirc p \Rightarrow \bigcirc Kp$ is valid in KL^{SPR}. First we negate and show the tableau algorithm applied to $K\bigcirc p \wedge \neg\bigcirc Kp$ returns an empty structure.

$$\textbf{start} \Rightarrow x \qquad \begin{array}{lll} x \Rightarrow Ky & & x \Rightarrow \bigcirc z \\ y \Rightarrow \bigcirc p & & z \Rightarrow \neg Kp \end{array}$$

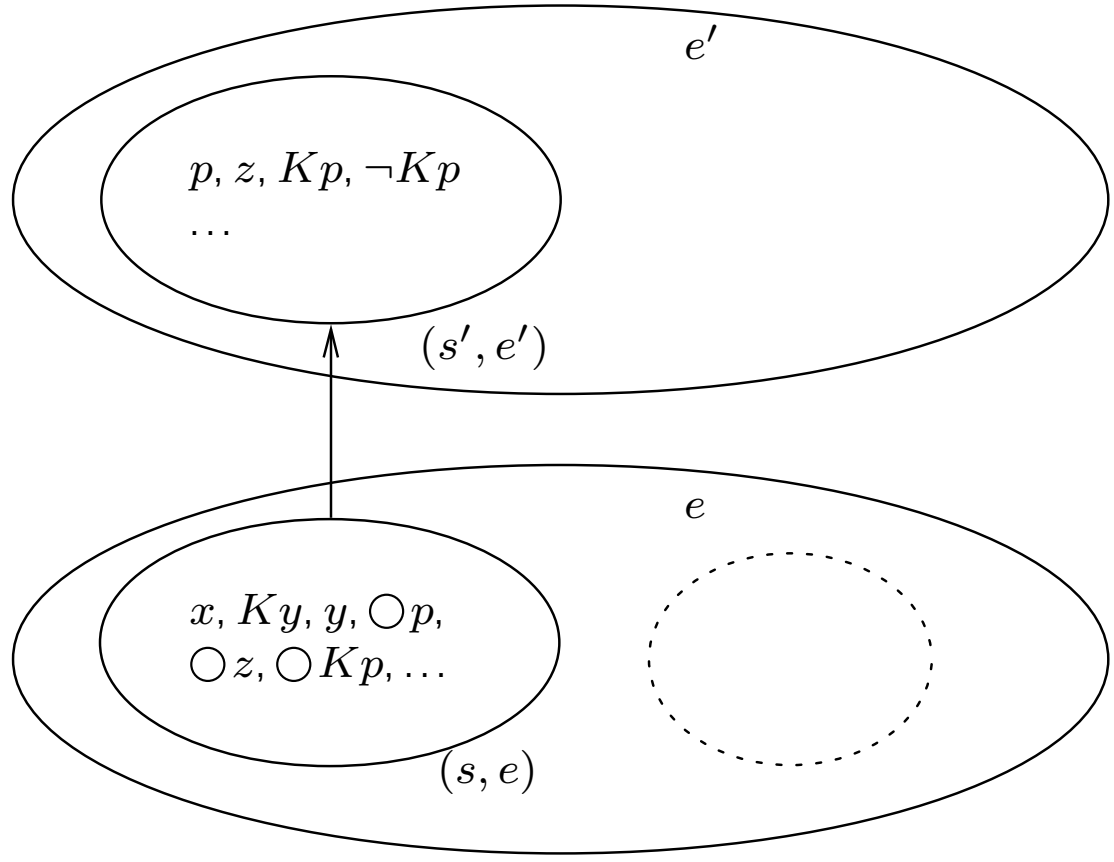
We add the following clauses (due to the SPR constraint).

$$\begin{array}{lll} Ky \Rightarrow \bigcirc Kp \\ Kx \Rightarrow \bigcirc Kz \end{array} \qquad K(x \vee y) \Rightarrow \bigcirc K(p \vee z)$$

Note, each initial state contains $\{x, Ky, y, \bigcirc p, \bigcirc z, \bigcirc Kp\}$

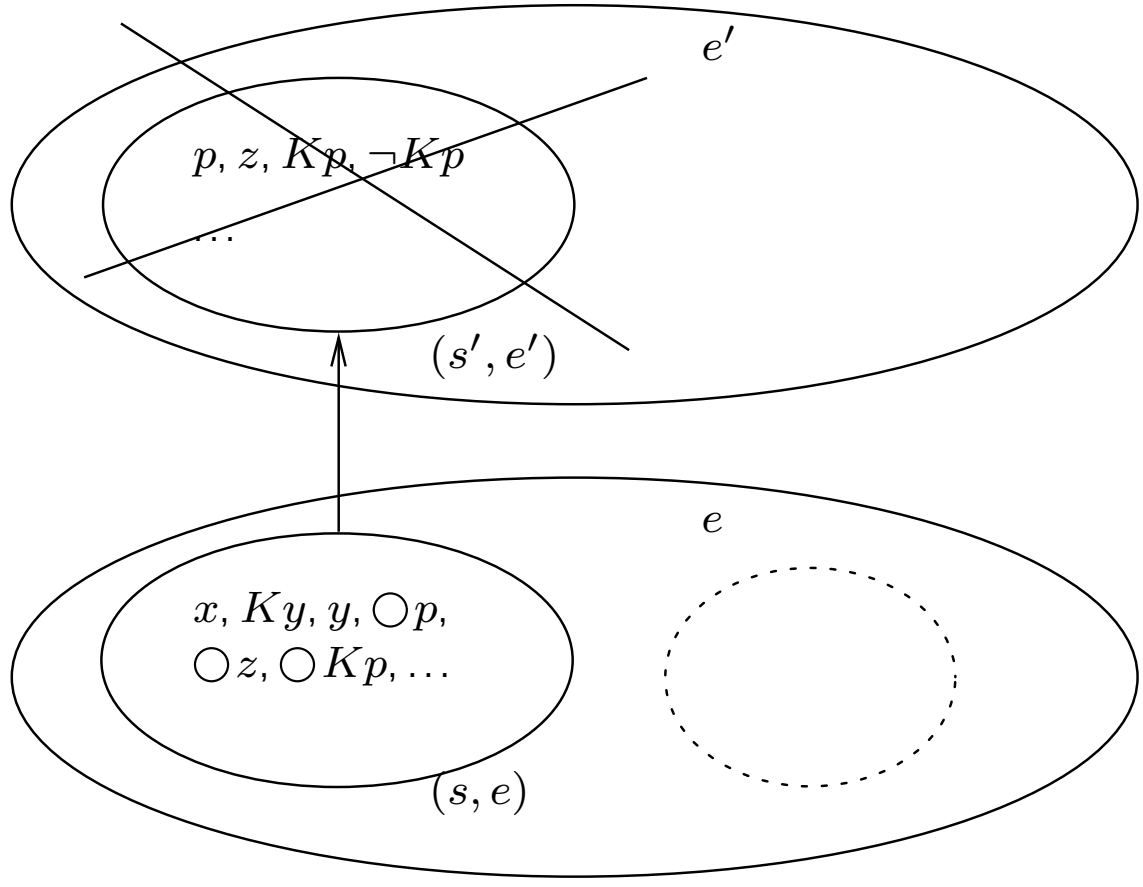
Example (Cont)

$$
\begin{array}{rcl}
\textbf{start} & \Rightarrow & x \\
x & \Rightarrow & Ky \\
y & \Rightarrow & \bigcirc p \\
x & \Rightarrow & \bigcirc z \\
z & \Rightarrow & \neg Kp \\
Ky & \Rightarrow & \bigcirc Kp \\
Kx & \Rightarrow & \bigcirc Kz \\
K(x \vee y) & \Rightarrow & \bigcirc K(p \vee z)
\end{array}
$$



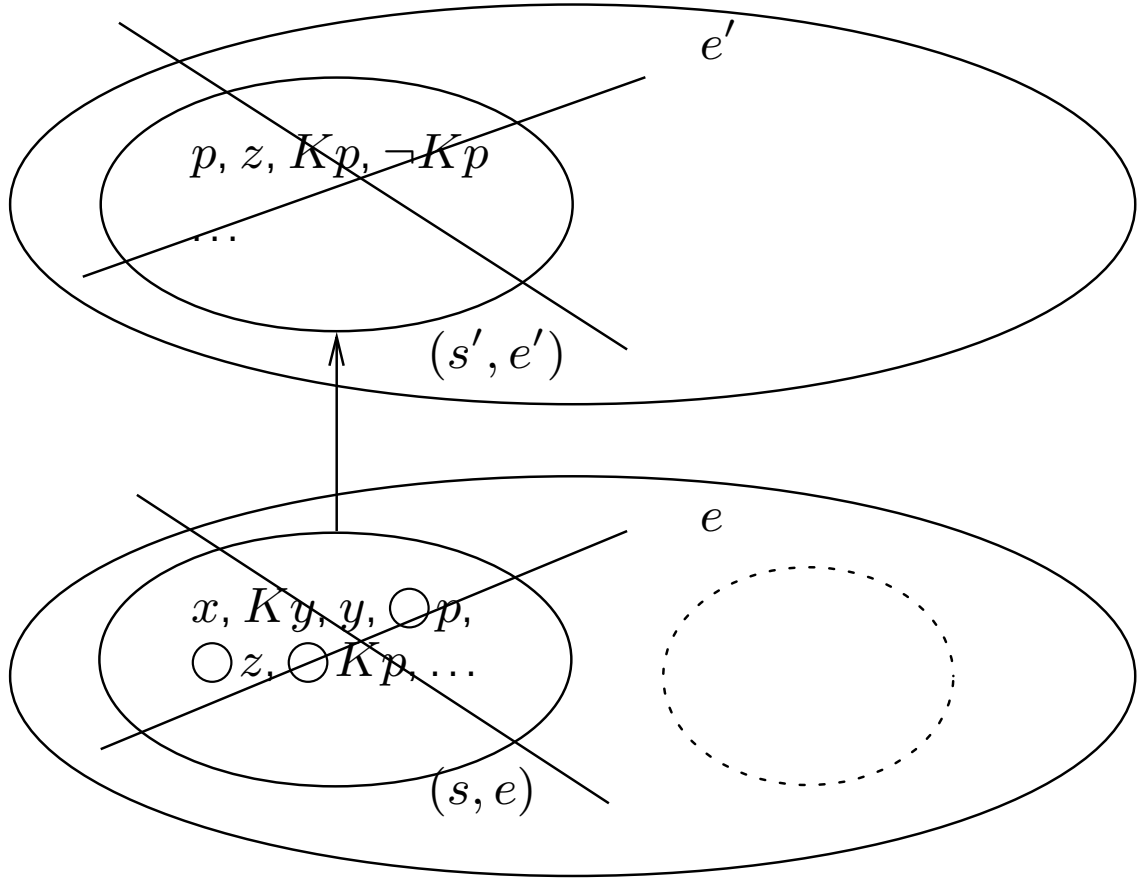
Example (Cont)

$$
\begin{array}{rcl}
\textbf{start} & \Rightarrow & x \\
x & \Rightarrow & Ky \\
y & \Rightarrow & \bigcirc p \\
x & \Rightarrow & \bigcirc z \\
z & \Rightarrow & \neg Kp \\
Ky & \Rightarrow & \bigcirc Kp \\
Kx & \Rightarrow & \bigcirc Kz \\
K(x \vee y) & \Rightarrow & \bigcirc K(p \vee z)
\end{array}
$$



Example (Cont)

$$
\begin{array}{rcl}
\textbf{start} & \Rightarrow & x \\
x & \Rightarrow & Ky \\
y & \Rightarrow & \bigcirc p \\
x & \Rightarrow & \bigcirc z \\
z & \Rightarrow & \neg Kp \\
Ky & \Rightarrow & \bigcirc Kp \\
Kx & \Rightarrow & \bigcirc Kz \\
K(x \vee y) & \Rightarrow & \bigcirc K(p \vee z)
\end{array}
$$



Hence, for each possibility we have an empty tableaux, hence $K\bigcirc\varphi \Rightarrow \bigcirc K\varphi$ is valid.

Theoretical Issues

- The normal form preserves satisfiability.

  Theorem 1 *Let ϕ be a well-formed formula in KL^{SPR} (KL^{SNL}) and $\tau(\phi) = \Box^* \bigwedge_i T_i$ where T_i is the set of clauses translated into ESNF$_{SPR}$ (ESNF$_{SNL}$). ϕ is satisfiable in KL^{SPR} (KL^{SNL}) if and only if $\tau(\phi)$ is satisfiable in KL^{SPR} (KL^{SNL}).*

- The tableau algorithm is sound and complete.

  Theorem 2 *The tableau algorithm applied to ϕ_0, a formula of KL^{SPR} (respectively KL^{SNL}), returns a non-empty structure, if and only if ϕ_0 is KL^{SPR} (respectively KL^{SNL}) satisfiable.*

- The tableau algorithm terminates.

Related Work

- Axioms, complexity for PTL + multi-modal S5 with interactions (Halpern, Vardi, Van der Meyden et al.).
- Resolution for PTL + S5 + SPR/SNL (Dixon, Fisher, Nalon).
- Resolution for PTL + (multi-modal) subsystems of S5 (Hustadt, Dixon, Schmidt, Fisher).
- Tableau for PTL, CTL, or CTL* + KD45, KD, KD (Rao, Georgeff).
- Tableau for PTL + multi-modal S5 or KD45 (Wooldridge, Dixon, Fisher).
- Tableau for temporal description logics (Lutz, Sturm, Wolter, Zakharyaschev, Günsel).

Summary

- Presented a tableau method for PTL and S5 allowing the interactions SPR and SNL.
- This uses a translation to a satisfiability preserving normal form to which clauses are added to account for SPR or SNL.
- The tableau construction is similar to that for temporal logics of knowledge (without interactions) except there are less tableau construction rules and we must make sure that the normal form clauses hold in all reachable states.
- Given an example of its usage.
- Stated soundness, completeness and termination results.

Current/Future Work

- Complexity analysis.
- Extend to more than one agent.
- These logics are complex so we need to develop strategies to guide the proof search, e.g. taking disjunctions/conjunctions of step clauses is expensive. Can this be carried out in a more controlled way?
- Adapt/extend to deal with other interactions e.g. perfect recall (without synchrony) etc.
- Apply the method to case studies, e.g. agent specification languages, knowledge games, or security protocols.
- Build a prototype implementation.