# 'Knowing Whether' in Proper Epistemic Knowledge Bases

**Tim Miller, Paolo Felli, Christian Muise, Adrian R. Pearce, Liz Sonenberg**
Department of Computing and Information Systems, University of Melbourne
{tmiller,paolo.felli,christian.muise,adrianrp,l.sonenberg}@unimelb.edu.au

## Abstract

Proper epistemic knowledge bases (PEKBs) are syntactic knowledge bases that use multi-agent epistemic logic to represent nested multi-agent knowledge and belief. PEKBs have certain syntactic restrictions that lead to desirable computational properties; primarily, a PEKB is a conjunction of modal literals, and therefore contains no disjunction. Sound entailment can be checked in polynomial time, and is complete for a large set of arbitrary formulae in logics $\mathrm{K}_n$ and $\mathrm{KD}_n$. In this paper, we extend PEKBs to deal with a restricted form of disjunction: 'knowing whether'. An agent $i$ knows whether $\varphi$ iff agent $i$ knows $\varphi$ or knows $\neg\varphi$; that is, $\Box_i\varphi \vee \Box_i\neg\varphi$. In our experience, the ability to represent that an agent knows whether something holds is useful in many multi-agent domains. We represent knowing whether with a modal operator, $\Delta_i$, and present sound polynomial-time entailment algorithms on PEKBs with $\Delta_i$ in $\mathrm{K}_n$ and $\mathrm{KD}_n$, but which are complete for a smaller class of queries than standard PEKBs.

## Introduction

Reasoning about the nested beliefs or knowledge of other agents is essential for many collaborative and competitive tasks. While highly expressive doxastic and epistemic logics exist for this (Fagin et al. 1995), such logics are computationally expensive. To implement agents that can efficiently reason about other agents in complex scenarios, approximations of these logics are necessary in many cases.

*Proper epistemic knowledge bases* (PEKBs), proposed by Lakemeyer and Lespérance (2012), are one such approximation. PEKBs are syntactic knowledge bases that contain sets of *restricted modal literals* (RMLs), which are modal literals that contain no conjunction or disjunction. Lakemeyer and Lespérance (2012) show that, for the epistemic logic $\mathrm{K}_n$, a PEKB can be compiled in exponential time into a special normal form, called *prime implicate normal form* (PINF), which is logically equivalent to the original PEKB, and database-like entailment queries can be made on these compiled form formulae in polynomial time. Muise et al. (2015b) extended this work to show that, if the knowledge base is known to be consistent, for example due to the use of a sound belief update operator, then entailment queries can be done in polynomial time *without* the expensive precompilation step. In both cases, entailment queries are complete for a reasonably large set of arbitrary modal logic formula, and sound for any query. Muise et al. (2015a) used this to compile a multi-agent epistemic planning problem into a classical planning problem, allowing plan and policy generation over multi-agent epistemic states.

The lack of disjunction is a restriction that is acceptable for some scenarios, but clearly not all; however extending PEKBs to include disjunction would eliminate the desirable computational properties. We believe the next best opportunity is to include a restricted form of disjunction: 'knowing whether'. An agent *knows whether* $\varphi$ is true — or in the case of belief logics, is *opinionated* as to whether $\varphi$ is true — if they know $\varphi$ or they know $\neg\varphi$. Formally, $\Box_i\varphi \vee \Box_i\neg\varphi$. The ability to model 'knowing whether' is useful in many domains, such as for action preconditions in planning (Scherl and Levesque 1993), diagnostic planning applications (Baier, Mombourquette, and McIlraith 2014), and in scenarios such as gossip (van Ditmarsch and Kooi 2015), muddy children (Fagin et al. 1995), and questioning (Aloni, Egré, and De Jager 2013). In particular, 'knowing whether' is useful in multi-agent domains in which agents know (or observe) that events occur without observing the outcome of the event itself. Consider the following scenario involving multi-vehicle search and rescue. If one unmanned vehicle (vehicle 1) has surveyed a sequence of points looking for survivors, we know that the vehicle knows whether there are survivors at surveyed points. If another vehicle (vehicle 2) observes a survivor at point A in the sequence, it can then infer that the first vehicle knows that there is a survivor ($(\Box_1 survivor \vee \Box_1\neg survivor) \wedge \Box_2 survivor \supset \Box_1 survivor$) and may assume that vehicle 1 has returned to base to report without finishing its survey, as pre-scripted. Vehicle 2 can now re-plan that: (a) it should survey the remaining points; and (b) it need not enact a rescue plan. Without being able to represent that vehicle 1 knows whether there is a survivor at point A, vehicle 2 cannot infer this without regressing "into the past", because it does not have $\Box_1 survivor \vee \Box_1\neg survivor$ in its knowledge base, so cannot infer that $\Box_1 survivor$. In our experience modelling real applications, the lack of disjunction in knowledge bases has only been problematic for these 'knowing whether' cases.

In this paper, we extend PEKBs to include knowing whether. Rather than model this as an explicit, yet restricted disjunction, we follow Fan et al. (2015) by modelling knowing whether as its own modal operator, $\Delta_i$, defined as $\Delta_i\phi \equiv \Box_i\phi \vee \Box_i\neg\phi$. After providing the necessary background for the paper in the following section, we show how to compile an extended PEKB into a prime implicate normal form in exponential time and space, and how to perform entailment queries for logic $\mathrm{K}_n$ in polynomial time, preserving the nice computational properties from Lakemeyer and Lespérance (2012). The cost is a less expressive set of complete queries. We then show that, as with Muise et al. (2015b), a consistent knowledge base in logic $\mathrm{KD}_n$ can be queried (for a smaller class of queries) in polynomial time without any prior compilation. We conclude with a discussion and future work.

The end result is a model for efficiently reasoning about nested beliefs in multi-agent environments that would be suitable for many scenarios.

## Background

### Epistemic Modal Logics

Let $\mathcal{P}$ and $Ag$ respectively be finite sets of propositions and agents. The set of well-formed formulae $\mathcal{L}$ for epistemic logic is obtained from the following grammar:

$$\varphi ::= p \mid \varphi \wedge \varphi \mid \neg\varphi \mid \Box_i\varphi$$

in which $p \in \mathcal{P}$ and $i \in Ag$. Informally, $\Box_i\varphi$ means that agent $i$ knows (or believes[1]) $\varphi$. The shorthand $\Diamond_i\varphi$ is the dual of $\Box_i\varphi$, defined as $\Diamond_i\varphi \equiv \neg\Box_i\neg\varphi$. Operators for $\vee$, $\supset$, and $\equiv$ can be derived in the usual way.

The semantics of this logic are given using *Kripke structures*. A Kripke structure is a tuple $M = (\mathcal{W}, \pi, R_1, \ldots, R_n)$, in which $W$ is the set of all possible worlds, $\pi \in \mathcal{W} \rightarrow 2^{\mathcal{P}}$ is a function that maps each world $w$ to the set of all propositions that hold in $w$, and $R_i \subseteq \mathcal{W} \times \mathcal{W}$ (for each $i \in Ag$) is an accessibility relation which captures each agent's *uncertainty* about the world —see (Fagin et al. 1995) for details. The formula $\Box_i\varphi$ holds in $w$ iff $\varphi$ is true in all worlds that the agent $i$ considers possible at $w$, while $\Diamond_i\varphi$ means that the agent considers $\varphi$ possible, thus holding in at least one possible world. The satisfaction of a formula $\varphi$ in a structure $M$ and a world $w$, denoted as $M, w \vDash \varphi$, is defined inductively over the structure of $\varphi$:

$$\begin{array}{lll} M, w \vDash p & \text{iff} & p \in \pi(w) \\ M, w \vDash \varphi \wedge \psi & \text{iff} & M, w \vDash \varphi \text{ and } M, w \vDash \psi \\ M, w \vDash \neg\varphi & \text{iff} & M, w \nvDash \varphi \\ M, w \vDash \Box_i\varphi & \text{iff} & \text{for all } v \in R_i(w),\ M, v \vDash \varphi \end{array}$$

We say that $\varphi$ entails $\psi$, written $\varphi \models \psi$ iff for every model $M$ and world $w$ such that $M, w \vDash \varphi$, we have $M, w \vDash \psi$.

Kripke structures with specific constraints lead to specific properties (or *axioms*) of knowledge or belief (Fagin et al. 1995). For example, axiom K holds for any Kripke frame, while the axiom D holds on *serial* Kripke frames, resulting in the axioms:

$$\begin{array}{llll} K & \Box_i(\varphi \supset \psi) \supset (\Box_i\varphi \supset \Box_i\psi) & \text{(Distribution)} \\ D & \Box_i\varphi \supset \neg\Box_i\neg\varphi & \text{(Consistency)} \end{array}$$

[1]For the remainder of the paper, we will use the term *knowledge* to mean both knowledge and belief.

Combinations of axioms result in different logics. Here, we consider logics $\mathrm{K}_n$ (axiom $K$ only) and $\mathrm{KD}_n$ (K and D), in which $_n$ specifies that there are multiple agents in the model.

**Example 1.** Consider an example, taken from Muise et al. (2015b) about two agents, Bob and Alice, who are coworkers. Bob knows Alice has applied for a promotion. Bob sees an envelope containing the outcome of the promotion, but has no information about whether Alice has opened the envelope. However, Bob knows that if Alice has opened the envelope, then Alice will know whether she has gained her promotion, and that if Alice has not opened the letter, she will not know whether she has gained her promotion. Assuming that '$opened$' and '$promoted$' represent that Alice has opened the letter and has been promoted respectively, we can represent the above using modal logic as follows:

$$\Box_B(\mathit{opened} \supset (\Box_A\mathit{promoted} \vee \Box_A\neg\mathit{promoted})) \wedge$$
$$\Box_B(\neg\mathit{opened} \supset \neg(\Box_A\mathit{promoted} \vee \Box_A\neg\mathit{promoted}))$$

Such logics are expressive, but computationally present several challenges. In the single agent case, Ladner (1977) showed that satisfiability is NP-complete, while Halpern and Moses (1985) demonstrated satisfiability is PSPACE-complete for multiple agents, but only NP-complete if the nesting of modal formulae is bounded (Halpern 1995). As such, syntactic approaches to knowledge bases have been investigated, with Eberle (1974) one of the first to consider syntactic knowledge bases, and Konolige (1983) the first to consider syntactic belief bases with *nested* belief.

### Knowledge compilation and prime implicates

*Prime implicates* have been used to mitigate the complexity of checking satisfiability and entailment in syntactic knowledge bases. A formula $\varphi$ is a prime implicate of a knowledge base $KB$ iff $KB \models \varphi$ (it is an *implicate*) and for all $\varphi'$ such that $KB \models \varphi'$, $\varphi' \models \varphi$ implies $\varphi \models \varphi'$ (it is *prime*).

The challenge is to calculate exactly the set of prime implicates of a knowledge base, and check entailment of a formula against the prime implicates instead of the original knowledge base. Bienvenu (2008; 2009) presents an algorithm that takes a logic $\mathrm{K}_n$ knowledge base and compiles it into *prime implicate normal form* (PINF), which is a DNF-like normal form in which all clauses are prime implicates. Essentially, a formula in PINF contains exactly the formulae required to check entailment of another formula, and nothing more. Entailment of a formula on the PINF formula can be checked in polynomial time, but the cost of generating the PINF is double-exponential in time and space.

### Proper epistemic knowledge bases

Lakemeyer and Lespérance (2012) define a *proper epistemic knowledge base* (PEKB) as a set of restricted formulae, called *restricted modal literals* (RMLs), of the form:

$$\alpha ::= p \mid \neg p \mid \Box_i\alpha \mid \Diamond_i\alpha$$

Thus, a PEKB contains no disjunctive formulae. Note that RMLs are in *negation normal form* (NNF), i.e. negation appears only in front of propositional variables. Any $\mathrm{K}_n$ modal literal can be re-written into NNF using $\neg\Box_i\varphi \equiv \Diamond_i\neg\varphi$, $\neg\Diamond_i\varphi \equiv \Box_i\neg\varphi$, and $\neg\neg p \equiv p$.

Following Bienvenu, Lakemeyer and Lespérance (2012) show how to compile a PEKB into PINF formula in single-exponential time and space, and how to check entailment of this PINF in polynomial time. Thus, by sacrificing expressiveness, some computational cost can be reduced. Their entailment algorithm is sound for arbitrary $\mathrm{K}_n$ queries, and complete for PINF formulae and for formulae in a specific normal form $\mathcal{NF}$, in which the semantic relationship between formulae is restricted to a certain class. Muise et al. (2015b) showed that if a PEKB is *consistent*, then for a smaller class of queries, compilation to PINF is not required: the PEKB can be queried directly in polynomial time. Thus, if one uses a sound belief update mechanism on a PEKB, then the expensive compilation to PINF can be avoided.

### 'Knowing whether' in epistemic logics

Fan et al. (2015) define knowing whether $\varphi$ such that $M, w \models \Delta_i\varphi$ iff:

$$\text{for all } v_1, v_2 \in R_i(w) \quad M, v_1 \vDash \varphi \Leftrightarrow M, v_2 \vDash \varphi$$

That is, at a world $w$, agent $i$ knows whether $\varphi$ if and only if all reachable worlds agree on the truth of $\varphi$. They also define an operator $\nabla_i\varphi$ as a shorthand for $\neg\Delta_i\varphi$. Note that this is not a dual operator, but a negation operator, read as: $i$ does not know whether $\varphi$. They show that knowing whether is in fact equivalent to contingency logic (Humberstone 1995; Kuhn 1995) and the negation of 'ignorance' in ignorance logic (van der Hoek and Lomuscio 2003; Van Der Hoek and Lomuscio 2004). The following properties hold for $\Delta_i$:

$$\begin{array}{ll} \Delta_i\varphi \equiv \Delta_i\neg\varphi & \nabla_i\varphi \equiv \Diamond_i\varphi \wedge \Diamond_i\neg\varphi \\ \Box_i\varphi \supset \Delta_i\varphi & \Box_i\varphi \equiv \Delta_i\varphi \wedge \Diamond_i\varphi \text{ (with axiom D)} \end{array}$$

PEKBs are not able to express the notion of knowing whether, due to the fact that it discusses uncertainty via disjunction: $\Delta_i\varphi \equiv (\Box_i\varphi \vee \Box_i\neg\varphi)$. As such, a class of problems that require knowing whether are out of scope. In the remainder of this paper, we extend the notion of PEKBs to include the modal operators $\Delta_i$ and $\nabla_i$ into PEKBs for logics $\mathrm{K}_n$ and $\mathrm{KD}_n$, without sacrificing the desirable computational properties of PEKBs.

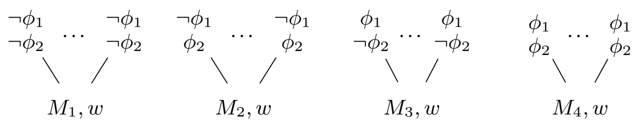
## Extending PEKBs with 'Knowing whether'

Rather than deal with knowing whether as the disjunction of $\Box_i\varphi \vee \Box_i\neg\varphi$, we use the $\Delta_i$ and $\nabla_i$ operators, in which $\Delta_i\varphi$ specifies that agent $i$ knows whether $\varphi$ holds, and $\nabla_i\varphi$ is defined as $\neg\Delta_i\varphi$; that is, agent $i$ does not know whether $\varphi$ holds. First, we extend a definition of RMLs to include the $\Delta_i$ operator:

$$\begin{array}{lll} \alpha & ::= & p \mid \neg p \mid \Box_i\alpha \mid \Diamond_i\alpha \mid \Delta_i\beta \\ \beta & ::= & p \mid \Box_i\beta \mid \Diamond_i\beta \mid \Delta_i\beta \end{array}$$

Thus, the modal literals are in NNF, which includes the restriction that a negation never occurs after a $\Delta_i$. As with RMLs, any modal literal, including those with $\Delta_i$ operators, can be written into NNF, and $\nabla_i$ can be removed completely:

$$\begin{array}{lllllll} \neg\Delta_i\varphi & \equiv & \nabla_i\varphi & \quad & \Delta_i\neg\varphi & \equiv & \Delta_i\varphi \\ \nabla_i\neg\varphi & \equiv & \nabla_i\varphi & \quad & \nabla_i\varphi & \equiv & \Diamond_i\varphi \wedge \Diamond_i\neg\varphi \end{array}$$



$\neg\phi_1$ $\neg\phi_2$ … $\neg\phi_1$ $\neg\phi_2$
$M_1, w$
$\neg\phi_1$ $\phi_2$ … $\neg\phi_1$ $\phi_2$
$M_2, w$
$\phi_1$ $\neg\phi_2$ … $\phi_1$ $\neg\phi_2$
$M_3, w$
$\phi_1$ $\phi_2$ … $\phi_1$ $\phi_2$
$M_4, w$

Figure 1: Set of models corresponding to the four $\Delta$-combinations reachable from world $w$ for $\Delta_i\phi_1, \Delta_i\phi_2$.

In the remainder of this paper, we will use the symbol P for PEKBs. Also, to simplify the notation, we will consider a conjunction of $\mathrm{K}_n$ formulae $\gamma \wedge \Delta_i\phi_1 \wedge \ldots \wedge \Delta_i\phi_l \wedge \Diamond_i\psi_1 \wedge \ldots \wedge \Diamond_i\psi_m \wedge \Box_i\chi_1 \wedge \ldots \wedge \Box_i\chi_n$, in which $\gamma$ is a propositional formula, $\phi_1, \ldots, \phi_l, \psi_1, \ldots, \psi_m, \chi_1, \ldots, \chi_n$ are formulae in $\mathrm{K}_n$ ($\mathrm{KD}_n$), so that each symbol can be used to disambiguate the outermost operator of each formula. We use $\alpha$ to represent RMLs, and $\varphi$ and $\psi$ as general $\mathrm{K}_n$ ($\mathrm{KD}_n$) formulae. We use $\varphi' \in \varphi$ to access conjuncts of $\varphi$, when $\varphi$ is a conjunction of formulae.

The introduction of $\Delta_i$ formulae introduces a restricted form of disjunction. For example, given the set of formulae $\{\Delta_i\phi_1, \Delta_i\phi_2\}$, at least four models satisfy this: one for each combinations of $[\neg]\phi_1 \wedge [\neg]\phi_2$ holding in all possible worlds. This is shown in Figure 1. Note that for each model, exactly one combination of $[\neg]\phi_1 \wedge [\neg]\phi_2$ holds at all possible worlds at $w$. We will refer to such combinations as $\Delta_i$-combinations. The set of all $\Delta_i$-combinations for agent $i$ in PEKB P is defined formally as:

$$\Delta_i\text{-combs}(\mathrm{P}) = \{ \bigwedge_{\Delta_i\psi \in \mathrm{P}'} \neg\psi \quad \wedge \bigwedge_{\Delta_i\psi \in (\mathrm{P}\setminus\mathrm{P}')} \psi \mid \mathrm{P}' \subseteq \mathrm{P}\}$$

**Theorem 1.** Given a PEKB $\varphi = \gamma \wedge \Delta_i\phi_1 \wedge \ldots \wedge \Delta_i\phi_l \wedge \Diamond_i\psi_1 \wedge \ldots \wedge \Diamond_i\psi_m \wedge \Box_i\chi_1 \wedge \ldots \wedge \Box_i\chi_n$, we have that $\varphi \models \bot$ *iff* at least one of the following:

(a) $\gamma \models \bot$;
(b) $\psi_j \wedge \chi_1 \wedge \ldots \wedge \chi_n \models \bot$ (for some $j$); or
(c) for all $\Delta_i$-combinations $\phi$ of $\varphi$, for some $j$, $\phi \wedge \psi_j \models \bot$

The final point means all $\Delta_i$-combinations conflict with at least one $\psi_j$.

*Proof.* For the right-to-left case, assume that the left side is false, and the formula is instead satisfiable. Then there must be some $M, w$ such that $M, w \models \gamma \wedge \Delta_i\phi_1 \wedge \ldots \wedge \Delta_i\phi_l \wedge \Diamond_i\psi_1 \wedge \ldots \wedge \Diamond_i\psi_m \wedge \Box_i\chi_1 \wedge \ldots \wedge \Box_i\chi_n$. From this, $M, w \models \gamma$ holds trivially, therefore $\gamma$ is satisfiable. If $M, w \models \Diamond_i\psi_1 \wedge \ldots \wedge \Diamond_i\psi_m \wedge \Box_i\chi_1 \wedge \ldots \wedge \Box_i\chi_n$, then there exists a world $w'$ reachable from $w$ such that $M, w' \models \psi_j \wedge \chi_1 \wedge \ldots \wedge \chi_n$ for all $j$, therefore, it must be that $\psi_j \wedge \chi_1 \wedge \ldots \wedge \chi_n$ is satisfiable, for all $j$. For part (c), consider some $\Delta_i\phi_u$. If $M, w \models \Delta_i\phi_u$, then $M, w \models \Box_i\phi_u$ or $M, w \models \Box_i\neg\phi_u$. If there are several $\Delta_i$ formulae, $M, w \models \Delta_i\phi_1 \wedge \ldots \wedge \Delta_i\phi_l \wedge \Diamond_i\psi_1 \wedge \ldots \wedge \Diamond_i\psi_m$, it must be that for *some* $\Delta_i$-combination $\phi$, we have that $M, w \models \Box_i\phi$. Therefore, for all $j$, there exists a world $w'$ such that $M, w' \models \phi \wedge \psi_j$, i.e. $\phi \wedge \psi_j$ is satisfiable.

For the left-to-right case, suppose that the right side is false. Construct a Kripke structure and a set of worlds such that $M, w \models \gamma$, and $M, w_j \models \psi_j \wedge \chi_1 \wedge \ldots \chi_n$ for all

$j$, and where all $w_j$ are reachable from $w$. It is clear that $M, w \models \gamma \wedge \Diamond_i\psi_1 \wedge \ldots \wedge \Diamond_i\psi_m \wedge \Box_i\chi_1 \wedge \ldots \wedge \Box_i\chi_n$. Further, we know that for some $\Delta_i$-combination $\phi$, we have $\phi \wedge \psi_j$ is satisfiable for all $j$. Take this $\phi$ and add it to each reachable world $w'$. $\phi$ is equivalent to some $\Delta_i$-combination of P. Therefore, for each $\phi_u$, we have that $M, w_j \models \phi_u$ for all $w_j$ or $M, w_j \models \neg\phi_u$ for all $w_j$, and therefore, $M, w \models \Delta_i\phi_1 \wedge \ldots \wedge \Delta_i\phi_l$. Combining with the above, $M, w \models \gamma \wedge \Delta_i\phi_1 \wedge \ldots \wedge \Delta_i\phi_l \wedge \Diamond_i\psi_1 \wedge \ldots \wedge \Diamond_i\psi_m \wedge \Box_i\chi_1 \wedge \ldots \wedge \Box_i\chi_n$, so the formula is satisfiable. □

This theorem provides us with a way to determine satisfiability of K$_n$ PEKBs, but also with a starting point for pre-compiling PEKBs into prime implicate normal form.

For the remainder of the paper, we use $\Delta_i$-combs$^{\subseteq}$(P) to denote the $\Delta_i$-combinations of all subsets of PEKB P, defined as:

$$\Delta_i\text{-combs}^{\subseteq}(\mathrm{P}) = \bigcup_{P' \subseteq P} \Delta_i\text{-combs}(\mathrm{P}') \setminus \{\neg\psi \mid \Delta_i\psi \in \mathrm{P}\}$$

A formula $\phi \in \Delta_i$-combs$^{\subseteq}$(P) is thus a conjunction of positive and negated elements in $\Delta_i$ formulae. E.g., if $\mathrm{P} = \{\Delta_i\phi_1, \Delta_i\phi_2\}$, then $\Delta_i$-combs$^{\subseteq}(\mathrm{P}) = \{\phi_1, \phi_2, \phi_1 \wedge \phi_2, \phi_1 \wedge \neg\phi_2, \neg\phi_1 \wedge \phi_2, \neg\phi_1 \wedge \neg\phi_2\}$. Note that $\neg\phi_1, \neg\phi_2 \notin \Delta_i$-combs$^{\subseteq}$(P) as $\Delta_i\phi_1 \equiv \Delta_i\neg\phi_1$.

**Definition 1** (Prime implicate normal form (PINF)). A set of formulae $\Sigma$, is in *prime implicate normal form* (PINF) iff every formula in $\Sigma$ is a prime implicate of $\Sigma$ (as defined in the background section), and every formula is a *term*, where a term is defined by:

$$T \quad ::= p \mid \neg p \mid \Box_i T \mid \Diamond_i T \mid \Delta_i T \mid T \wedge T$$

Note that this definition is different from Bienvenu (2009) (subsequently adopted by Lakemeyer and Lespérance (2012)) as it pertains only to terms, not arbitrary K$_n$ formula.

For example, the set of formulae $\{\Box_i\Box_j(p \wedge q), \Delta_i\Diamond_j p\}$ is not in PINF because for any model in which $\Box_i\Box_j p$ holds, it must be that $\Delta_i\Diamond_j p$ also holds.

Next, we introduce a function *PEKB2PINF* that converts PEKBs into a PINF formula such that the PINF formula is logically equivalent to the original PEKB, but contains only the prime implicates of PEKB. Step 1 first determines potential prime implicates that result from combining multiple RMLs. Steps 2–4 then eliminate non-prime implicates, such as the example above. Finally, step 5 determines if there are contradictions in the PEKB.

**Definition 2** (*PEKB2PINF*(P)). The function *PEKB2PINF* takes a conjunction of terms[2] P, and returns a PINF formula $\Sigma$. First, we define the following:

$$\begin{array}{lcl}
B_i(\mathrm{P}) & = & \{\varphi \mid \Box_i\varphi \in \mathrm{P}\} \\
D_i(\mathrm{P}) & = & \{\varphi \mid \Diamond_i\varphi \in \mathrm{P}\} \\
W_i(\mathrm{P}) & = & \{\varphi \mid \Delta_i\varphi \in \mathrm{P}\} \\
\mathit{Prop}(\mathrm{P}) & = & \{l \mid l \text{ is a non-modal literal} \wedge l \in \mathrm{P}\}
\end{array}$$

[2]Note that a PEKB is just a restricted form of this where the terms are all restricted modal literals.

The *PEKB2PINF* algorithm is then defined as follows:

1. Let $F_i(\mathrm{P}) = \bigwedge_{\varphi \in B_i(\mathrm{P})} \varphi$.
   Let $\Gamma(\mathrm{P}) = \mathit{Prop}(\mathrm{P}) \cup$
   $\{\Diamond_i(\varphi \wedge F_i(\mathrm{P})) \mid \varphi \in D_i(\mathrm{P}) \wedge B_i(\mathrm{P}) \neq \emptyset\} \cup$
   $\{\Diamond_i(\varphi) \mid \varphi \in D_i(\mathrm{P}) \wedge B_i(\mathrm{P}) = \emptyset\} \cup$
   $\{\Box_i(F_i(\mathrm{P})) \mid B_i(\mathrm{P}) \neq \emptyset\} \cup$
   $\{\Delta_i(\Psi) \mid \Psi \in \Delta_i\text{-combs}^{\subseteq}(W_i(\mathrm{P}))\}$.
2. For each $M(\varphi) \in \Gamma(\mathrm{P})$, where $M$ is either $\Diamond_i$ or $\Box_i$, replace it by $M(\mathit{PEKB2PINF}(\varphi))$.
3. For all $\Diamond_i\varphi_1, \Delta_i\varphi_2 \in \Gamma(\mathrm{P})$, if $\varphi_1 \models \varphi_2$, remove $\Delta_i\varphi_2$ from $\Gamma(\mathrm{P})$, and replace every $\Box_i\chi \in \Gamma(\mathrm{P})$ with $\Box_i(\chi \wedge \varphi_2)$.
4. For all $\Box_i\varphi_1, \Delta_i\varphi_2 \in \Gamma(\mathrm{P})$, if $\varphi_1 \models \varphi_2$, remove $\Delta_i\varphi_2$.
5. If any of the following hold, return $\bot$ :
   (a) $\Diamond_i\bot$ is in $\Gamma(\mathrm{P})$
   (b) both $p$ and $\neg p$ are in $\Gamma(\mathrm{P})$,
   (c) if $\Delta_i\varphi_1$, $\Diamond_i\varphi_2$, and $\Diamond_i\varphi_3$ are in $\Gamma(\mathrm{P})$ and $\varphi_1 \wedge \varphi_2 \models \bot$ and $\neg\varphi_1 \wedge \varphi_3 \models \bot$.
   Otherwise return $\Sigma = \bigwedge_{\psi \in \Gamma(\mathrm{P})} \psi$.

Note that each PINF formula will contain at most one $\Box_i\chi$ for each $i$: the prime implicate formed from all $\Box_i$ formulae. This definition of *PEKB2PINF* is similar to Lakemeyer and Lespérance's algorithm, except that it handles $\Delta_i$ formulae.

The prime implicates of $\Delta_i$ formulae deserve discussion. Given $\Delta_i\phi_u$ and $\Delta_i\phi_v$, six prime implicates result: $\Delta_i\phi_u$, $\Delta_i\phi_v$, $\Delta_i(\phi_u \wedge \phi_v)$, $\Delta_i(\phi_u \wedge \neg\phi_v)$, $\Delta_i(\neg\phi_u \wedge \phi_v)$, and $\Delta_i(\neg\phi_u \wedge \neg\phi_v)$. That is, all conjunctions of $[\neg]\phi_u$ and $[\neg]\phi_v$ are 'known whether', and all are prime implicates. While $\Delta_i\phi_u, \Delta_i\phi_v \models \Delta_i([\neg]\phi_u \wedge [\neg]\phi_v)$, the reverse does not hold. Consider if agent $i$ knows whether $\phi_u$ and knows whether $\phi_v$, then clearly agent $i$ knows whether $\phi_u \wedge \phi_v$. However, if agent $i$ knows whether $\phi_u \wedge \phi_v$ because it knows that $\neg(\phi_u \wedge \phi_v)$, this does not imply that it knows whether $\phi_u$ (nor $\phi_v$), because it may only know that their conjunction is false without knowing their individual truth values.

**Lemma 1.** Given a PINF formula $\Sigma$ such that $\Sigma \equiv \mathrm{P}$ for a given PEKB P, then $\Delta_i(\phi_u \wedge \phi_v) \in \Sigma$ iff both $\Delta_i\phi_u$ and $\Delta_i\phi_v$ are in $\Sigma$ (for $\phi_u \not\equiv \phi_v$).

*Proof.* As discussed above, $\Delta_i(\phi_u \wedge \phi_v) \not\models \Delta_i\phi_u$ (nor $\phi_v$). However, $\Delta_i(\phi_u \wedge \phi_v)$ is a prime implicate of P iff both $\Delta_i\phi_u$ and $\Delta_i\phi_v$ are in P, because conjunction is not allowed in any $\Delta_i\phi_j \in \mathrm{P}$, and $\Delta_i\phi_u \wedge \Delta_i\phi_v \models \Delta_i(\phi_u \wedge \phi_v)$. □

**Lemma 2.** Given a PINF formula $\Sigma = \gamma \wedge \Delta_i\phi_1 \wedge \ldots \wedge \Delta_i\phi_l \wedge \Diamond_i\psi_1 \wedge \ldots \wedge \Diamond_i\psi_m \wedge \Box_i\chi$ such that $\Sigma \equiv \mathrm{P}$ for some PEKB P, and a formula $\epsilon$ in NNF:

$$\begin{array}{llll}
(a) & \Sigma \models l & \text{iff} & l \in \gamma \quad (l \text{ is a non-modal literal}) \\
(b) & \Sigma \models \Delta_i\epsilon & \text{iff} & \Delta_i\Sigma' \in \Sigma \text{ s.t. } \Sigma' \equiv \epsilon \text{ or} \\
 & & & \Sigma \models \Box_i\epsilon \vee \Box_i\neg\epsilon; \\
(c) & \Sigma \models \Diamond_i\epsilon & \text{iff} & \Diamond_i\Sigma' \in \Sigma \text{ s.t. } \Diamond_i\Sigma' \models \Diamond_i\epsilon; \\
(d) & \Sigma \models \Box_i\epsilon & \text{iff} & \chi \models \epsilon
\end{array}$$

*Proof.* The case in which $\Sigma \models \bot$ is trivial, and we omit it. Consider $(a)$. By Theorem 1, $\Sigma \wedge \neg l \models \bot$ iff $\gamma \models l$.

Consider $(b)$. $\Sigma \models \Delta_i \epsilon$ is equivalent to $\Sigma \wedge \Diamond_i \epsilon \wedge \Diamond_i \neg\epsilon \models \bot$ and, by Theorem 1, either: (i) $\chi \models \epsilon$ or $\chi \models \neg\epsilon$; or (ii) $\phi \models \epsilon$ or $\phi \models \neg\epsilon$ for all $\Delta_i$-combinations $\phi$. If (i) holds then $\Sigma \models \Box_i \epsilon$ or $\Sigma \models \Box_i \neg\epsilon$, and therefore $\Sigma \models \Box_i \epsilon \vee \Box_i \neg\epsilon$. If (ii) holds then since any $\Delta_i \phi_u \in \Sigma$ either occurs in original PEKB, or is a combination in $\Delta_i$-combs$^{\subseteq}$(P) of the original PEKB (or both), then by Lemma 1, we know that every such $\Delta_i \phi_u$ must be in $\Sigma$. So there is some $\Delta_i \Sigma' \in \Sigma$ such that $\Sigma' \equiv \epsilon$. Note that the case where $\epsilon$ is of the form $\epsilon_1 \vee \epsilon_2$ is not considered because $\Delta_i \epsilon$ is in NNF, and $\Delta_i(\epsilon_1 \vee \epsilon_2) \equiv \Delta_i \neg(\epsilon_1 \vee \epsilon_2) \equiv \Delta_i(\neg\epsilon_1 \wedge \neg\epsilon_2)$.

Consider $(c)$. If $\Sigma \models \Diamond_i \epsilon$ then $\Sigma \wedge \Box_i \neg\epsilon \models \bot$, and from Theorem 1, there must be a $\psi_j$ such that $\psi_j \wedge \chi \models \epsilon$ and therefore, $\Diamond_i(\psi_j \wedge \chi) \models \Diamond_i \epsilon$. Because $\Sigma$ is in PINF, then either $\Diamond_i(\psi_j \wedge \chi) \in \Sigma$ or for some other $\Diamond_i \Sigma' \in \Sigma$, and $\Diamond_i \Sigma' \models \Diamond_i(\psi_j \wedge \chi)$. Therefore, $\Diamond_i \Sigma' \models \Diamond_i \epsilon$.

Consider $(d)$. By Theorem 1, $\Sigma \models \Box_i \epsilon$ iff $\Sigma \wedge \Diamond_i \neg\epsilon \models \bot$ iff either: (i) $\chi \models \epsilon$; or (ii) for all $\phi$, either there exists a $\psi_j$ such that $\phi \wedge \psi_j$ is unsatisfiable or $\phi \wedge \neg\epsilon$ is unsatisfiable. Now consider any model $M, w$ such that, $M, w \models \Box_i \phi \wedge \Diamond_i \psi_1 \wedge \ldots \wedge \Diamond_i \psi_m$. It must be that $M, w \models \Box_i \epsilon$, because there is no $\psi_j$ such that $\phi \wedge \psi_j$ is unsatisfiable. Similarly, for any model $M, w$ such that $M, w \models \Box_i \neg\phi \wedge \Diamond_i \psi_1 \wedge \ldots \wedge \Diamond_i \psi_m$, it must be that $M, w \models \Box_i \epsilon$ for the same reason. Therefore, $\Sigma \models \Box_i \epsilon$. However, if $\Sigma \models \Box_i \epsilon$, and $\Sigma$ is in PINF, then it must be that $\Box_i \chi \models \Box_i \epsilon$, and thus $\chi \models \epsilon$, which is the same as (i). □

**Theorem 2.** *PEKB2PINF* returns a PINF formula that is equivalent to the original PEKB P; that is $\Gamma(\text{P}) \equiv \text{P}$ and every formula in $\Gamma(\text{P})$ is a prime implicate.

*Proof.* The proof is an extension of the proof for Lemma 2. We show the least straightforward case: $\Delta_i \epsilon$. If $\Delta_i \epsilon$ is a prime implicate of a PEKB P, then $\text{P} \models \Delta_i \epsilon$. As argued in the proof of Lemma 2, this means that either: (i) $\text{P} \models \Box_i \epsilon \vee \Box_i \neg\epsilon$; or (ii) there is some combination $\phi \in \Delta_i$-combs$^{\subseteq}$(P) such that $\Delta_i \phi \equiv \Delta_i \epsilon$. Because $\Delta_i \epsilon$ is a prime implicate, (ii) means that $\Delta_i \phi$ should be in $\Gamma(\text{P})$ for all $\phi \in \Delta_i$-combs$^{\subseteq}$(P), which is the case. However, for case (i), if either $\Box_i \epsilon$ or $\Box_i \neg\epsilon$ holds, then $\Delta_i \epsilon$ cannot be a prime implicate, because $\Box_i \epsilon \models \Delta_i \epsilon$ and $\Box_i \neg\epsilon \models \Delta_i \epsilon$. Thus, if $\Box_i \varphi_1, \Delta_i \varphi_2 \in \Gamma(\text{P})$ and $\varphi_1 \models \varphi_2$, then $\Delta_i \varphi_2$ should be excluded from $\Gamma(\text{P})$, which occurs in step 4 of *PEKB2PINF*. □

**Theorem 3.** The space and time complexity of *PEKB2PINF* is exponential in size and depth of P.

*Proof.* Checking entailment between two RMLs (steps 3 and 4) can be done in linear time in the depth of the shortest RML. The algorithm from Lakemeyer and Lespérance (2012) has worst-case time complexity of $O(|\text{P}|^{d+2})$, with $d$ being the modal depth of $\varphi$ (accounting for the recursive calls in step 2). Our algorithm must also derive $2^n$ $\Delta$ formulae in each recursive call, in which $n$ is the number of $\Delta$ formula in P. Therefore, the worst-case complexity is $O(|\text{P}|^{d+2+|\text{P}|})$. The proof of the space bound follows in a similar manner. □

## Query evaluation

In this section, we consider a querying mechanism, $V$, that takes a PINF formula $\Sigma$, and query $\varphi$ in NNF, and returns 1 or 0, in which 1 is interpreted as *true* and 0 as *unknown*. This query mechanism is exactly as Lakemeyer and Lespérance (2012), *except* that we need to define $V$ for $\Delta_i \varphi$.

**Definition 3** ( $V[\Sigma, \varphi]$). $V[\Sigma, \varphi] = 1$ if $\Sigma = \bot$, otherwise:

$$V[\Sigma, \top] = 1 \text{ and } V[\Sigma, \bot] = 0$$

$$V[\Sigma, l] = 1 \text{ if } l \in \Sigma, 0 \text{ otherwise}$$

$$V[\Sigma, \Box_i \varphi] = \begin{cases} 1 & \text{for some } \Box_i \Sigma' \in \Sigma,\ V[\Sigma', \varphi] = 1 \\ 0 & \text{otherwise} \end{cases}$$

$$V[\Sigma, \Diamond_i \varphi] = \begin{cases} 1 & \text{for some } \Diamond_i \Sigma' \in \Sigma,\ V[\Sigma', \varphi] = 1 \\ 0 & \text{otherwise} \end{cases}$$

$$V[\Sigma, \Delta_i \varphi] = \begin{cases} 1 & \text{if } V[\Sigma, \Box_i \varphi \vee \Box_i \neg\varphi] = 1 \text{ or} \\ & \text{if } \Delta_i \varphi \in \Sigma \\ 0 & \text{otherwise} \end{cases}$$

$$V[\Sigma, \varphi \vee \psi] = \max(V[\Sigma, \varphi], V[\Sigma, \psi])$$

$$V[\Sigma, \varphi \wedge \psi] = \min(V[\Sigma, \varphi], V[\Sigma, \psi])$$

Thus, $V$ is a simple database query on the PINF formula for a knowledge base using a structural subsumption similar to that of Bienvenu (2009), Lakemeyer and Lespérance (2012) and Muise et al. (2015b).

**Theorem 4** (Soundess of $V$). Let $\Sigma$ be a PEKB in PINF, and $\varphi$ a formula in NNF. If $V[\Sigma, \varphi] = 1$ then $\Sigma \models \varphi$.

*Proof.* Soundness follows directly from Lemma 2, except for the first three cases, and the conjunction and disjunction cases, which all hold trivially. □

**Comment.** To obtain a sound query mechanism, we do not actually need to consider all $\phi \in \Delta_i$-combs$^{\subseteq}$(P). *PEKB2PINF* could simply keep only those $\Delta_i \alpha$ formulae from the original PEKB, and re-write queries of the form $\Delta_i(\phi_u \wedge \phi_v)$ to $\Delta_i \phi_u \wedge \Delta_i \phi_v$. It is easy to show that this mechanism would be sound, but that the formula produced by *PEKB2PINF* would not be in PINF, because $\Delta_i(\phi_u \wedge \phi_v)$ is a prime implicate. However, the complexity of this compilation approach would be $O(|P|^{d+2})$ instead of $O(|P|^{d+2+|P|})$.

While $V$ is sound, it is quite clearly incomplete. For example, consider the simple query $(p \wedge p \supset q) \supset q$, which is a tautology, but for which $V[\Sigma, \varphi] = 0$.

**Definition 4** (Query normal form). We define a normal form for queries, called *query normal form* (QNF):

$$\begin{aligned} T &::= p \mid \neg p \mid \Box_i T \mid \Diamond_i T \mid \Delta_i T \mid T \wedge T \\ C &::= T \mid C \vee C \end{aligned}$$

A query is therefore a disjunction of terms. A formula in QNF is clearly in DNF (and therefore NNF).

**Definition 5** (Logical Separability)**.** From Lakemeyer and Lespérance (2012): the set of formulae $P$ is *logically separable* iff for every consistent set of formulae $P'$ this holds:

$$\text{if} \quad P \cup P' \models \bot \quad \text{then} \quad \exists \varphi \in P, \text{ s.t. } P' \cup \{\varphi\} \models \bot$$

Logical separability captures the property that there are 'no logical puzzles hidden' within $P$. For example, the set $\{\Box_i p, \Box_i(p \supset q)\}$ is not logically separable, because we can infer $\Box_i q$ from the combination of the two, yet the formula $\Diamond_i \neg q$ is consistent with each individual formula.

**Theorem 5** (Completeness of $V$ for QNF queries.)**.** Let $\Sigma$ be a PEKB in PINF and $\varphi$ a logically-separable formula in QNF. Then $V[\Sigma, \varphi] = 1$ iff $\Sigma \models \varphi$.

*Proof.* The left-to-right direction follows from Theorem 4. For the other direction, if $\Sigma \equiv \bot$, the proof is immediate. If $\Sigma \not\equiv \bot$, we prove by induction on the length of $\varphi$. The cases for propositions, $\Box_i\varphi$, $\Diamond_i\varphi$, and $\varphi \wedge \psi$ hold from Lakemeyer and Lespérance (2012) (Theorem 6). The $\varphi \vee \psi$ follows the same argument. Assume that the theorem holds for formula of length $n$, and that $\varphi$ is of length $n$+1. If $\Sigma \models \varphi \vee \psi$, then $\Sigma \cup \{\neg\varphi, \neg\psi\}$ is inconsistent. Because the query is logically separable, then $\Sigma \cup \{\neg\varphi\}$ or $\Sigma \cup \{\neg\psi\}$ is inconsistent. Thus $\Sigma \models \varphi$ or $\Sigma \models \psi$, and inductively, $V[\Sigma, \varphi \vee \psi]$. The $\Delta_i\varphi$ case holds from Lemma 2. ☐

## Consistent knowledge bases

In this section, we consider a natural extension for KD$_n$, namely the case when a KD$_n$ knowledge base is *consistent*; for example, a sound belief update mechanism is used to add new formulae to the PEKB. Working with consistent PEKBs, we can define a polynomial entailment mechanism that is complete for a restricted set of QNF formulae, and sound, but which can be performed on uncompiled PEKBs, thereby avoiding the exponential compilation step similar to *PEKB2PINF*. This is because we can re-write non-separable queries such as $\Delta_i\varphi, \Diamond_i\varphi \models_{\mathrm{KD}_n} \Box_i\varphi$ as $\Delta_i\varphi, \Diamond_i\varphi \models_{\mathrm{KD}_n} \Delta_i\varphi \wedge \Diamond_i\varphi$, which is separable. For the remainder of this section, we use $\models$ to mean $\models_{\mathrm{KD}_n}$.

**Lemma 3.** Given a PEKB $\mathrm{P} = \gamma \wedge \Delta_i\phi_1 \wedge \ldots \wedge \Delta_i\phi_l \wedge \Diamond_i\psi_1 \wedge \ldots \wedge \Diamond_i\psi_m \wedge \Box_i\chi_1 \wedge \ldots \wedge \Box_i\chi_n$ and an RML $\alpha$, then the following hold in logic KD$_n$:

| | | | |
|---|---|---|---|
| $(a)$ | $\mathrm{P} \models l$ | iff | $l \in \gamma$ ($l$ is a non-modal literal) |
| $(b)$ | $\mathrm{P} \models \Delta_i\alpha$ | iff | $\Delta_i\alpha \in \mathrm{P}$ or $\Box_i\alpha' \in \mathrm{P}$ s.t. $\alpha' \models \alpha$ or $\alpha' \models \neg\alpha$ |
| $(c)$ | $\mathrm{P} \models \Diamond_i\alpha$ | iff | $\Diamond_i\alpha' \in \mathrm{P}$ s.t. $\alpha' \models \alpha$ or $\Box_i\alpha' \in \mathrm{P}$ s.t. $\alpha' \models \alpha$ |
| $(d)$ | $\mathrm{P} \models \Box_i\alpha$ | iff | $\mathrm{P} \models \Delta_i\alpha \wedge \Diamond_i\alpha$ |

*Proof.* Case (a) is straightforward. For (b), $\mathrm{P} \models \Delta_i\alpha$ iff $\mathrm{P} \wedge \Diamond_i\alpha \wedge \Diamond_i\neg\alpha \models \bot$, and by Theorem 1, either: $\chi_1 \wedge \ldots \chi_n \models \alpha$ or $\chi_1 \wedge \ldots \chi_n \models \neg\alpha$; or (ii): $\phi \models \alpha$ or $\phi \models \neg\alpha$ for all $\Delta_i$-combinations $\phi$. If (i) holds, then $\mathrm{P} \models \Box_i\alpha \vee \Box_i\alpha$. If (ii) holds, then we know there is some $\phi$ such that $\phi \equiv \alpha$. Because $\alpha$ is an RML, then $\phi$ must also be an RML, and therefore $\Delta_i\phi \in \mathrm{P}$.

For case (c), if $\mathrm{P} \wedge \Box_i\neg\alpha \models \bot$ then by Theorem 1[3], there is a $\psi_v$ such that $\psi_v \wedge \chi_1 \wedge \ldots \wedge \chi_n \models \alpha$. Because P contains only RMLs, it cannot be that $\Diamond_i(\psi_v \wedge \chi_u) \in \mathrm{P}$ for any $\psi_v \wedge \chi_u$. Further, because $\alpha$ is an RML, it must be that $\psi_v \models \alpha$ or $\chi_u \models \alpha$ for some $u$, and therefore $\Diamond_i\psi_v \models \Diamond_i\alpha$ or $\Box_i\chi_u \models \Diamond_i\alpha$. Case (d) holds trivially from the equivalence $\Box_i\varphi \equiv \Delta_i\varphi \wedge \Diamond_i\varphi$ in logic KD$_n$. ☐

**Definition 6** (Reduced Query Normal Form)**.** We define *reduced query normal form* (reduced QNF) as:

$$\begin{array}{lcl} T & ::= & p \mid \neg p \mid \Box_i T \mid \Diamond_i T \mid \Delta_i T \\ C & ::= & T \mid C \vee C \mid C \wedge C \end{array}$$

Reduced QNF formulae are therefore Boolean combinations of RMLs. They differ from QNF formulae because conjunctions are not permitted inside modal operators.

**Definition 7** ($U[\mathrm{P}, \varphi]$)**.** Given a PEKB, P, and a query $\varphi$ in reduced QNF, the query mechanism $U[\mathrm{P}, \varphi]$ returns 1 for *true* and 0 for *unknown*: $U[\mathrm{P}, \top] = 1$, $U[\mathrm{P}, \bot] = 0$, or:

$U[\mathrm{P}, l] = 1$ if $l \in \mathrm{P}$, 0 otherwise

$$U[\mathrm{P}, \Diamond_i\alpha] = \begin{cases} 1 & \text{if } U[B_i(\mathrm{P}), \alpha] = 1 \text{ or} \\ & \text{for some } \Diamond_i\alpha' \in \mathrm{P},\ U[\{\alpha'\}, \alpha] = 1 \\ 0 & \text{otherwise} \end{cases}$$

$$U[\mathrm{P}, \Delta_i\alpha] = \begin{cases} 1 & \text{if } U[B_i(\mathrm{P}), \alpha \vee \neg\alpha] = 1 \text{ or} \\ & \text{if } \Delta_i\alpha \in \mathrm{P} \\ 0 & \text{otherwise} \end{cases}$$

$$U[\mathrm{P}, \Box_i\alpha] = U[\mathrm{P}, \Delta_i\alpha \wedge \Diamond_i\alpha]$$

$$U[\mathrm{P}, \varphi \wedge \psi] = \min(U[\mathrm{P}, \varphi], U[\mathrm{P}, \psi])$$

$$U[\mathrm{P}, \varphi \vee \psi] = \max(U[\mathrm{P}, \varphi], U[\mathrm{P}, \psi])$$

**Theorem 6.** Given a PEKB P and a query $\varphi$ in reduced QNF, the worst-case complexity of $U[\mathrm{P}, \varphi]$ is polynomial on the size of P.

*Proof.* The worst-case complexity is $O(|\mathrm{P}| \cdot d)$, in which $d$ is the depth of the knowledge base. In the worst case, for each sub-formula in $\varphi$, $U[\mathrm{P}, \varphi]$ would be required to iterate over each RML in P, recursively calling $U$ up to $d$ times. ☐

**Theorem 7** (Soundness and completeness of $U[\mathrm{P}, \varphi]$)**.** Let P be a set of RMLs and $\varphi$ a logically-separable reduced QNF formula. Then $U[\mathrm{P}, \varphi] = 1$ iff $\mathrm{P} \models_{\mathrm{KD}_n} \varphi$.

*Proof.* This follows immediately from Lemma 3, except the for simple $\top$ and $\bot$ cases, which are trivial, and the conjunction and disjunction cases, which can be proved in the same way as the proof for soundness of $V$ (Theorem 4). ☐

**Proposition 1.** For an arbitrary KD$_n$ formula $\varphi$, there is an equivalent formula $\varphi'$ in reduced QNF (and therefore in QNF) that is at worst exponentially larger than $\varphi$, and that for a given PEKB P, $\mathrm{P} \models \varphi$ iff $\mathrm{P} \models \varphi'$.

[3]In fact, the case for KD$_n$ is slightly different, but does not affect the proof for consistent knowledge bases.

For PEKBs, the three modal operators distribute over conjunction and disjunction (e.g. $\Box_i(\varphi \wedge \psi) \equiv \Box_i\varphi \wedge \Box_i\psi$ and P $\models \Delta_i(\varphi \wedge \psi)$ iff P $\models \Delta_i\varphi \wedge \Delta_i\psi$, in which P is a PEKB), and the resulting formula is at worst linear in the size of the original formula. The only exception is the rule rule P $\models \Diamond_i(\varphi \wedge \psi)$ iff P $\models (\Box_i\varphi \wedge \Diamond_i\psi) \vee (\Diamond_i\varphi \wedge \Box_i\varphi)$, which produces a formula that is at worst $2^n$ the size of the original formula, where $n$ is the nesting of $\Diamond$ operators. However, if the depth of the query formula is considerably smaller than the size of the knowledge base, compiling the query into reduced QNF is more attractive than compiling to PINF.

## Conclusions and future work

In this paper, we introduced the notion of 'knowing whether' into proper epistemic knowledge bases. The concept of knowing whether is not as expressive as disjunction, but adds a useful level of expressiveness into the PEKB framework. We show that for the logic $K_n$, we can maintain a polynomial entailment mechanism at the exponential cost of pre-compilation (as in the original PEKB definition (Lakemeyer and Lespérance 2012)), and for the logic $KD_n$, we can maintain the polynomial entailment query mechanism presented in more recent work (Muise et al. 2015b).

In future work, we will investigate notions of group belief and knowledge in PEKBs, such as common belief and common knowledge. We will also investigate how to perform sound and complete belief update on a PEKB to ensure that it remains consistent after an update.

**Acknowledgements** The authors thank Gerhard Lakemeyer for his insightful ideas and discussions on the 'knowing whether' operator.

This research is partially funded by Australian Research Council Discovery Grant DP130102825, *Foundations of Human-Agent Collaboration: Situation-Relevant Information Sharing*.

## References

Aloni, M.; Egré, P.; and De Jager, T. 2013. Knowing whether A or B. *Synthese* 190(14):2595–2621.

Baier, J. A.; Mombourquette, B.; and McIlraith, S. A. 2014. Diagnostic problem solving via planning with ontic and epistemic goals. In *Proceedings of the Fourteenth International Conference Principles of Knowledge Representation and Reasoning*, 388–397.

Bienvenu, M. 2008. Prime implicate normal form for ALC concepts. In *AAAI*, 412–417.

Bienvenu, M. 2009. Prime implicates and prime implicants: From propositional to modal logic. *Journal of Artificial Intelligence Research* 36(1):71–128.

Eberle, R. A. 1974. A logic of believing, knowing, and inferring. *Synthese* 26(3):356–382.

Fagin, R.; Halpern, J. Y.; Moses, Y.; and Vardi, M. Y. 1995. *Reasoning about knowledge*, volume 4. MIT press Cambridge.

Fan, J.; Wang, Y.; and van Ditmarsch, H. 2015. Contingency and knowing whether. *The Review of Symbolic Logic* 8:75–107.

Halpern, J. Y., and Moses, Y. 1985. A guide to the modal logics of knowledge and belief: Preliminary draft. In *Proceedings of the 9th international joint conference on Artificial intelligence*, 480–490. Morgan Kaufmann Publishers.

Halpern, J. Y. 1995. The effect of bounding the number of primitive propositions and the depth of nesting on the complexity of modal logic. *Artificial Intelligence* 75(2):361–372.

Humberstone, L. 1995. The logic of non-contingency. *Notre Dame Journal of Formal Logic* 36(2):214–229.

Konolige, K. 1983. A deductive model of belief. In *IJCAI*, 377–381.

Kuhn, S. 1995. Minimal non-contingency logic. *Notre Dame Journal of Formal Logic* 36(2):230–234.

Ladner, R. E. 1977. The computational complexity of provability in systems of modal propositional logic. *SIAM journal on computing* 6(3):467–480.

Lakemeyer, G., and Lespérance, Y. 2012. Efficient reasoning in multiagent epistemic logics. In *ECAI 2012 - 20th European Conference on Artificial Intelligence.*, 498–503.

Muise, C.; Belle, V.; Felli, P.; McIlraith, S.; Miller, T.; Pearce, A.; and Sonenberg, L. 2015a. Planning over multiagent epistemic states: A classical planning approach. In *The 29th AAAI Conference on Artificial Intelligence*, 3327–3334.

Muise, C.; Miller, T.; Felli, P.; Pearce, A.; and Sonenberg, L. 2015b. Efficient reasoning with consistent proper epistemic knowledge bases. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, 1461–1469. IFAAMAS.

Scherl, R. B., and Levesque, H. J. 1993. The frame problem and knowledge-producing actions. In *AAAI*, volume 93, 689–695. Citeseer.

van der Hoek, W., and Lomuscio, A. 2003. Ignore at your peril-towards a logic for ignorance. In *Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, 1148–1149. ACM.

Van Der Hoek, W., and Lomuscio, A. 2004. A logic for ignorance. In *Declarative Agent Languages and Technologies*. Springer. 97–108.

van Ditmarsch, H., and Kooi, B. 2015. *One Hundred Prisoners and a Light Bulb*. Springer.