# Knowledge Compilation in the Modal Logic S5

**Meghyn Bienvenu**
Universität Bremen
Bremen, Germany
meghyn@informatik.uni-bremen.de

**Hélène Fargier**
IRIT-CNRS,
Université Paul Sabatier,
Toulouse, France
fargier@irit.fr

**Pierre Marquis**
CRIL-CNRS, Université d'Artois
Lens, France
marquis@cril.univ-artois.fr

**Abstract**

In this paper, we study the knowledge compilation task for propositional epistemic logic S5. We first extend many of the queries and transformations considered in the classical knowledge compilation map to S5. We then show that the notion of disjunctive normal form (DNF) can be profitably extended to the epistemic case; we prove that the DNF fragment of S5, when appropriately defined, satisfies essentially the same queries and transformations as its classical counterpart.

## 1 Introduction

Propositional epistemic logic S5 is a well-known modal logic which is suitable for representing and reasoning about the knowledge of a single agent. For instance, in S5, one can represent the fact that an agent knows the truth value of a proposition $a$ without stating which one ($\mathbf{K}a \vee \mathbf{K}\neg a$); such a fact cannot be expressed in classical propositional logic. Reasoning from such pieces of knowledge is an important issue per se, but is also required when performing the progression of a knowledge state by an epistemic plan. Unfortunately, standard reasoning tasks in S5 are intractable; for instance, entailment in S5 is coNP-complete (Ladner 1977), like in classical propositional logic. Another example is the variable forgetting transformation, which can prove useful in epistemic planning but is intractable in the general case.

A standard approach to coping with the computational intractability of reasoning is to employ knowledge compilation (KC). The idea underlying KC is to pre-process parts of the available information (i.e., turning them into a compiled form) in order to improve on-line computational efficiency (see (Cadoli and Donini 1998) for a survey). A major issue in KC is the selection of an appropriate target language. In (Darwiche and Marquis 2002), the authors argue that the choice of a target language must be based both on the set of queries and transformations which can be achieved in polynomial time when the data are represented in the language, as well as the spatial efficiency of the language. They elaborate a KC map which can be viewed as a multi-criteria evaluation of a number of classical propositional languages, including DNF, CNF, DNNF, and OBDD. From there, the KC map has been extended to other propositional target languages. There has also been some work on knowledge compilation techniques suited to other formal settings, e.g. belief bases, Bayesian networks, description logics, etc.

In this paper, we study the knowledge compilation task for propositional epistemic logic S5. We introduce a family S5-DNF of subsets of S5 (called "fragments") which can be seen as the epistemic counterparts of DNF and evaluate the family with respect to several queries and transformations, including clausal entailment, variable forgetting, and conditioning by an (epistemic) term. Our evaluation shows one of the most promising languages of the family to be S5-DNF$_{\mathsf{DNF,CNF}}$, consisting of those S5 formulae in disjunctive normal form in which the formulae $\alpha$ in positive epistemic atoms ($\mathbf{K}\alpha$) are in DNF and those appearing in negative epistemic atoms ($\neg\mathbf{K}\alpha$) are in CNF.

The rest of the paper is organized as follows. Section 2 contains a brief refresher on classical propositional logic and the modal logic S5. Section 3 shows how many queries and transformations considered in the classical case can be extended to S5. In Section 4, we study the properties of the S5-DNF family of fragments of S5; we show in particular that the fragment S5-DNF$_{\mathsf{DNF,CNF}}$ satisfies essentially the same queries and transformations as its propositional counterpart. In Section 5, we sketch how our results can be exploited for epistemic planning. Finally, in Section 6 we conclude the paper with a discussion of our results.

## 2 Preliminaries

### Classical Propositional Logic

The propositional languages which have been considered as target languages for KC are typically subsets of the following PDAG language:

**Definition 1** (PDAG)**.** Let $PS$ be a finite set of propositional variables (atoms). A formula in PDAG is a rooted, directed acyclic graph (DAG) where each leaf node is labeled with $\top$, $\bot$, $x$ or $\neg x$, $x \in PS$; each internal node is labeled with $\wedge$ or $\vee$ and can have arbitrarily many children, or it is labeled with $\neg$ and has a single child.

For any subset $V$ of $PS$, $L_V$ denotes the subset of (classical) *literals* generated from the atoms of $V$, i.e., $L_V = \{x, \neg x \mid x \in V\}$. CL (resp. TE) is the set of (classical)

Copyright © 2010, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

*clauses* (resp. *terms*), namely disjunctions (resp. conjunctions) of literals. CNF (resp. DNF) is the subset of PDAG consisting of conjunctions of clauses (resp. disjunctions of terms). PDAG and its subsets are classically interpreted: the semantics of a formula is a Boolean function. A valuation (or world) $\omega$ is a mapping from $PS$ to $\{0,1\}$. The set of valuations is noted by $\Omega$.

Subsets $\mathsf{L}_1$ and $\mathsf{L}_2$ of PDAG are said to be *dual* iff there exists a polytime algorithm $f$ from $\mathsf{L}_1$ to $\mathsf{L}_2$ and a polytime algorithm $g$ from $\mathsf{L}_2$ to $\mathsf{L}_1$ such that for every $\alpha \in \mathsf{L}_1$, $f(\alpha) \equiv \neg\alpha$, and for every $\alpha \in \mathsf{L}_2$, $g(\alpha) \equiv \neg\alpha$.

In the following, we consider only subsets L of PDAG in which every classical clause (resp. term) has a polytime-computable representation in L. This is the case for all the languages considered in (Darwiche and Marquis 2002), except the explicit model representation MODS.

## Modal Logic S5

Let us now consider a more general DAG-based language, suited to propositional epistemic logic:

**Definition 2 (S5).** Let $PS$ be a finite set of propositional variables (atoms). A formula in S5 is a rooted, directed acyclic graph (DAG) where each leaf node is labeled with $\top$, $\bot$, $x$ or $\neg x$, $x \in PS$; each internal node is labeled with $\wedge$ or $\vee$ and can have arbitrarily many children, or it is labeled with $\mathbf{K}$ or by $\neg$ and it has a single child; and each path contains at most one node $\mathbf{K}$. The size of a sentence $\Sigma$ in S5 denoted $|\Sigma|$, is the number of its DAG arcs plus the number of its DAG nodes. $Var(\Sigma)$ is the set of propositional symbols from $PS$ occurring in $\Sigma$.

The semantics of S5 can be defined in a standard way à la Kripke, but it is well-known (Fagin et al. 1995) that it can also be defined in an equivalent (yet simpler) way using pointed knowledge structures, of the form $M = \langle \omega, S\rangle$ where $\omega \in \Omega$ and $S$ is a (non-empty) subset of $\Omega$ containing $\omega$. The *satisfaction* of a formula $\varphi \in$ S5 by a structure $M = \langle \omega, S\rangle$ is defined inductively as follows:

- if $\varphi$ is a leaf node, then $\langle \omega, S\rangle \models \varphi$ iff $\omega \models \varphi$,
- if $\varphi$ is of the form $\neg\alpha$, then $\langle \omega, S\rangle \models \varphi$ iff $\langle \omega, S\rangle \not\models \alpha$,
- if $\varphi$ is of the form $\mathbf{K}\alpha$, then $\langle \omega, S\rangle \models \varphi$ iff for every $\omega' \in S$, $\langle \omega', S\rangle \models \alpha$,
- if $\varphi$ is of the form $\wedge(\alpha_1, \ldots, \alpha_k)$, then $\langle \omega, S\rangle \models \varphi$ iff for each $1 \leq i \leq k$, $\langle \omega, S\rangle \models \alpha_i$,
- if $\varphi$ is of the form $\vee(\alpha_1, \ldots, \alpha_k)$, then $\langle \omega, S\rangle \models \varphi$ iff there exists $1 \leq i \leq k$ such that $\langle \omega, S\rangle \models \alpha_i$.

$M$ is a model of $\varphi$ when it satisfies it. $\mathsf{Mod}(\varphi)$ denotes the set of models of $\varphi$. A formula $\varphi$ is said to be consistent (resp. valid) iff $\mathsf{Mod}(\varphi) \neq \emptyset$ (resp. $\mathsf{Mod}(\neg\varphi) = \emptyset$). Logical entailment and equivalence are defined as usual: $\varphi \models \psi$ iff $\mathsf{Mod}(\varphi) \subseteq \mathsf{Mod}(\psi)$, and $\varphi \equiv \psi$ iff $\mathsf{Mod}(\varphi) = \mathsf{Mod}(\psi)$.

In our definition of S5 we have chosen to disallow nested modalities. This is without any loss of generality since every formula from S5 can be turned in polynomial time into a flattened equivalent, thanks to the following "flattening equivalences": $\mathbf{K}\mathbf{K}\varphi \equiv \mathbf{K}\varphi$, $\mathbf{K}\neg\mathbf{K}\varphi \equiv \neg\mathbf{K}\varphi$, $\mathbf{K}(\varphi \wedge \psi) \equiv (\mathbf{K}\varphi) \wedge (\mathbf{K}\psi)$, $\mathbf{K}(\varphi \vee \mathbf{K}\psi) \equiv (\mathbf{K}\varphi) \vee (\mathbf{K}\psi)$ and $\mathbf{K}(\varphi \vee \neg(\mathbf{K}\psi)) \equiv (\mathbf{K}\varphi) \vee \neg(\mathbf{K}\psi)$.

PDAG is clearly a proper subset of S5. It contains the so-called *objective* formulae of S5. The *subjective fragment* of S5 (denoted s-S5) is defined as follows.

**Definition 3 (s-S5).** s-S5 consists of all S5 formulae $\varphi$ such that every path from the root of $\varphi$ to a node labeled by $l \in L_{PS}$ contains exactly one node labeled by $\mathbf{K}$.

s-S5 allows one to represent an agent's knowledge but not the actual state of the world (unless it is known); for instance, the S5 formula $a \wedge b \wedge \mathbf{K}a \wedge \neg\mathbf{K}b$ cannot be represented in s-S5.

A formula of the form $\mathbf{K}\varphi$ with $\varphi \in$ PDAG is called an *epistemic atom*. Each S5 formula $\Sigma$ can be viewed as a PDAG formula where the leaf nodes are Boolean constants, classical propositional literals or epistemic atoms.

As expected, an *epistemic literal* is an epistemic atom (also referred to as a positive epistemic literal) or a negated one (a negative epistemic literal). By *literal*, we will mean either a classical literal or an epistemic literal. Clearly enough, epistemic literals are already rather complex and some simple properties from the classical case cannot be extended to S5. Thus, since $\mathbf{K}\alpha$ is consistent (resp. valid) iff $\alpha$ is consistent (resp. valid), the consistency (resp. validity) problem for epistemic literals is NP-complete (resp. coNP-complete).

Now, taking advantage of the flattening equivalences, one can easily extend the notions of *clause* and *term*, and the languages CNF and DNF to S5 as follows:

**Definition 4 (S5-CL, S5-TE, S5-CNF, S5-DNF).** The set S5-CL (resp. S5-TE) of epistemic clauses (resp. terms) of S5 consists of all those disjunctions (resp. conjunctions) of literals containing *at most* one negative (resp. positive) epistemic literal. The language S5-CNF (resp. S5-DNF) is the subset of formulae from S5 consisting of conjunctions (resp. disjunctions) of epistemic clauses (resp. terms).

Note that the restriction to a single positive (resp. negative) epistemic literal per term (resp. clause) in Definition 4 is without loss of expressiveness because of the equivalence $\mathbf{K}(\varphi \wedge \psi) \equiv \mathbf{K}\varphi \wedge \mathbf{K}\psi$. Without this restriction, we cannot guarantee that the consistency of a positive epistemic term (resp. the validity of a negative epistemic clause) amounts to the consistency of each positive epistemic literal in it (resp. the validity of at least one negative epistemic literal in it). For instance, the positive epistemic term $\mathbf{K}a \wedge \mathbf{K}\neg a$ is inconsistent despite the fact that each of $\mathbf{K}a$ and $\mathbf{K}\neg a$ is consistent.

The following (folklore) proposition, which characterizes the consistency of S5 terms, highlights the interaction between the classical formulae appearing in a term.

**Proposition 5.** *An epistemic term $\tau \wedge \mathbf{K}\alpha \wedge \neg\mathbf{K}\beta_1 \wedge \ldots \wedge \neg\mathbf{K}\beta_n$ (with $\tau$ an objective term) is consistent if and only if $\tau \wedge \alpha$ is consistent and $\alpha \wedge \neg\beta_i$ is consistent for every $1 \leq i \leq n$.*

In what follows, we will often need to refer to fragments of S5 in which the shape of formulae behind $\mathbf{K}$ has been restricted. If L, L′ $\subseteq$ PDAG and $\mathcal{L}$ is a subset of S5, then we denote by $\mathcal{L}_{\mathsf{L},\mathsf{L}'}$ the subset of $\mathcal{L}$ in which the formulae under the scope of $\mathbf{K}$ (resp. $\neg\mathbf{K}$) belong to L (resp. L′). For

example, S5-$\mathsf{CL}_{\mathsf{CNF,DNF}}$ consists of epistemic clauses with CNF formulae behind $\mathbf{K}$ and DNF formulae behind $\neg\mathbf{K}$. Finally, given a subset $\mathcal{L}$ of S5, we use s-$\mathcal{L}$ to refer to its restriction to s-S5, i.e., $\mathcal{L} \cap$ s-S5.

## 3 Towards a KC Map for S5

We show how the notions of expressiveness and succinctness (or spatial efficiency) as well as many queries and transformations considered in the classical KC map can be extended to S5.

### Expressiveness and Succinctness

Consider subsets $\mathcal{L}_1$ and $\mathcal{L}_2$ of S5.

**Definition 6 ($\leq_e$).** $\mathcal{L}_1$ is *at least as expressive as* $\mathcal{L}_2$, denoted $\mathcal{L}_1 \leq_e \mathcal{L}_2$, iff for every formula $\alpha \in \mathcal{L}_2$, there exists an equivalent formula $\beta \in \mathcal{L}_1$.

A subset $\mathcal{L}_1$ of a language $\mathcal{L}_2$ is said to be *complete w.r.t.* $\mathcal{L}_2$ when it is as expressive as $\mathcal{L}_2$. As in the classical case, succinctness is a refinement of expressiveness:

**Definition 7 ($\leq_s$).** $\mathcal{L}_1$ is *at least as succinct as* $\mathcal{L}_2$, denoted $\mathcal{L}_1 \leq_s \mathcal{L}_2$, iff there exists a polynomial $p$ such that for every formula $\alpha \in \mathcal{L}_2$, there exists an equivalent formula $\beta \in \mathcal{L}_1$ where $|\beta| \leq p(|\alpha|)$.

$=_s$ and $<_s$ are obtained as usual by taking the symmetric and asymmetric parts of $\leq_s$.

### Queries and Transformations

Most standard queries and transformations straightforwardly extend from classical logic to subsets $\mathcal{L}$ of S5. This is the case for CO (consistency), VA (validity), EQ (equivalence), SE (sentential entailment), and the (possibly bounded) closure transformations with respect to the propositional connectives: $\wedge$C, $\wedge$BC, $\vee$C, $\vee$BC, $\neg$C.

**Definition 8.**

- $\mathcal{L}$ satisfies CO (resp. VA) iff there exists a polytime algorithm that maps every formula $\alpha$ from $\mathcal{L}$ to 1 if $\alpha$ is consistent (resp. valid), and to 0 otherwise.
- $\mathcal{L}$ satisfies EQ (resp. SE) iff there exists a polytime algorithm that maps every pair of formulae $\alpha$, $\beta$ from $\mathcal{L}$ to 1 if $\alpha \equiv \beta$ (resp. $\alpha \models \beta$) holds, and to 0 otherwise.
- $\mathcal{L}$ satisfies $\wedge$C (resp. $\vee$C) iff there exists a polytime algorithm that maps every finite set of formulae $\alpha_1, \ldots, \alpha_n$ from $\mathcal{L}$ to a formula of $\mathcal{L}$ that is logically equivalent to $\wedge(\alpha_1 \ldots \alpha_n)$ (resp. $\vee(\alpha_1 \ldots \alpha_n)$).
- $\mathcal{L}$ satisfies $\wedge$BC (resp. $\vee$BC) iff there exists a polytime algorithm mapping every pair $\alpha, \beta$ of formulae from $\mathcal{L}$ to a formula of $\mathcal{L}$ logically equivalent to $\alpha \wedge \beta$ (resp. $\alpha \vee \beta$).
- $\mathcal{L}$ satisfies $\neg$C iff there exists a polytime algorithm that maps every formula $\alpha$ from $\mathcal{L}$ to a formula of $\mathcal{L}$ that is logically equivalent to $\neg\alpha$.

**CE and IM.** A straightforward generalization of clausal entailment (CE) would say that $\mathcal{L}$ satisfies CE iff there exists a polytime algorithm mapping every formula $\alpha$ from $\mathcal{L}$ and every clause $\gamma$ to 1 if $\alpha \models \gamma$ holds, and to 0 otherwise. As we saw earlier, the problems of deciding whether even a single epistemic atom is a tautology or a contradiction is intractable, which means that no non-trivial subset of S5 can exhibit polynomial behaviour for the query CE. A similar argument applies to implication of terms (IM). For this reason, we consider the following versions of CE (resp. IM), in which restrictions are placed also on the formulae appearing in the epistemic literals of the clause (resp. term):

**Definition 9 ($\mathsf{CE}_{\mathsf{CNF,DNF}}$, $\mathsf{IM}_{\mathsf{DNF,CNF}}$).** Let $\mathcal{L}$ denote any subset of S5. $\mathcal{L}$ satisfies $\mathsf{CE}_{\mathsf{CNF,DNF}}$ (resp. $\mathsf{IM}_{\mathsf{DNF,CNF}}$) iff there exists a polytime algorithm that maps every formula $\alpha$ from $\mathcal{L}$ and every epistemic clause $\gamma$ from S5-$\mathsf{CL}_{\mathsf{CNF,DNF}}$ (resp. term $\gamma$ from S5-$\mathsf{TE}_{\mathsf{DNF,CNF}}$) to 1 if $\alpha \models \gamma$ holds (resp. if $\gamma \models \alpha$), and to 0 otherwise.

**Forgetting.** Variable forgetting is a fundamental operation from a knowledge representation point of view with a number of applications, including reasoning under inconsistency and planning (cf. Section 5). Forgetting can be used to simplify an agent's epistemic state by discarding information concerning propositional variables which are no longer relevant. Forgetting in S5 can be defined as follows (Zhang and Zhou 2009):

**Definition 10 (Forgetting).** We say that a formula $\Psi$ is a *forgetting* of $\mathcal{V} \subseteq PS$ from $\Phi$ if

**(i)** $\Phi \models \Psi$,

**(ii)** $Var(\Psi) \cap \mathcal{V} = \emptyset$,

**(iii)** for every formula $\Psi'$ such that $\Phi \models \Psi'$ and $Var(\Psi') \cap \mathcal{V} = \emptyset$, we have $\Psi \models \Psi'$.

A subset $\mathcal{L}$ of S5 satisfies FO iff there exists a polytime algorithm that maps every formula $\Phi \in \mathcal{L}$ and every $\mathcal{V} \subseteq PS$ to a formula $\Psi \in \mathcal{L}$ which is a forgetting of $\mathcal{V}$ from $\Phi$.

The result of forgetting $\mathcal{V}$ from a formula $\Phi$ is the logically strongest consequence of $\Phi$ which does not mention variables from $\mathcal{V}$, i.e., it is the projection of $\Phi$ onto $PS \setminus \mathcal{V}$. One should note that in some modal logics, like S4 (Ghilardi and Zawadowski 1995), for some choices of $\Phi$ and $\mathcal{V}$, no formula satisfying the above three conditions can be found. Luckily, such a formula always exists in S5, and item (iii) ensures it is unique up to logical equivalence. We henceforth denote this formula by $\mathsf{forget}(\Phi, \mathcal{V})$.

**Conditioning.** Conditioning is a key transformation in a number of representation settings as it permits the incorporation of pieces of evidence into a representation. In many settings (including classical propositional logic), conditioning is defined as a composition of conjunction with a term and forgetting. In S5, we can proceed similarly provided we restrict the type of terms we allow:

**Definition 11 (Epistemic conditioning).** A subset $\mathcal{L}$ of S5 satisfies eCD iff there exists a polytime algorithm that maps every formula $\Phi \in \mathcal{L}$ and every satisfiable epistemic term $\Upsilon$ from S5-$\mathsf{TE}_{\mathsf{DNF,CNF}}$ to a formula $\Phi|\Upsilon \in \mathcal{L}$ such that

$$\Phi|\Upsilon \equiv \mathsf{forget}(\Phi \wedge \Upsilon, Var(\Upsilon)).$$

Note that the restriction to terms from S5-$\mathsf{TE}_{\mathsf{DNF,CNF}}$ is harmless from an expressiveness point of view; without it, the conditioning of formulae from quite trivial fragments of S5 can be shown to be coNP-hard using the following reduction: a classical formula $\varphi$ is unsatisfiable iff

$\mathsf{forget}(\mathbf{K}a \wedge \mathbf{K}(\neg a \vee \varphi), Var(\mathbf{K}(\neg a \vee \varphi)))$ is unsatisfiable. Observe that the latter satisfiability check can be done in polytime since it involves a S5 formula without propositional variables.

With our definition, the relationship with clausal entailment satisfied in the classical case continues to hold[1]:

**Proposition 12.** *Let $\mathcal{L}$ be a subset of S5. If $\mathcal{L}$ satisfies both eCD and CO, then it also satisfies $\mathsf{CE}_{\mathsf{CNF,DNF}}$.*

In the classical case, the conditioning of a formula $\varphi$ by a (classical) term $\tau$ is equivalent to the formula resulting from replacing all variables $v$ in $\varphi$ by $\top$ (resp. $\bot$) if $v$ appears positively (resp. negatively) in $\tau$. This "syntactic" conditioning is straightforwardly extended to S5:

**Definition 13 (Syntactic conditioning).** A subset $\mathcal{L}$ of S5 satisfies CD iff there exists a polytime algorithm that maps every formula $\Phi \in \mathcal{L}$ and every satisfiable objective term $\tau = v_1 \wedge \ldots \wedge v_n \wedge \neg v_{n+1} \wedge \ldots \wedge \neg v_m$ to a formula $\Phi|\mathbf{K}\tau \in \mathcal{L}$ such that $\Phi|\mathbf{K}\tau \equiv \Phi[v_1 \leftarrow \top, \ldots, v_n \leftarrow \top, v_{n+1} \leftarrow \bot, \ldots, v_m \leftarrow \bot]$.

Using the same notations for the two versions of conditioning is harmless as syntactic conditioning of a S5 formula is just regular conditioning by an epistemic atom:

**Proposition 14.** *The following are equivalent:*

1. $\Phi[v_1 \leftarrow \top, \ldots, v_n \leftarrow \top, v_{n+1} \leftarrow \bot, \ldots, v_m \leftarrow \bot]$.
2. $\mathsf{forget}(\Phi \wedge \mathbf{K}(v_1 \wedge \cdots \wedge v_n \wedge \neg v_{n+1} \wedge \ldots \wedge \neg v_m), \{v_1, \ldots, v_m\})$.

Notice that if we use the more limited, syntactic variant of conditioning, we lose the fact that clausal entailment can be performed by conditioning followed by consistency-checking, a key property from the classical setting. However, syntactic conditioning by a propositional term $\tau$ is sufficient for modelling the change in an agent's knowledge when learning that the real world satisfies $\tau$.

Now that a notion of epistemic conditioning is available, we can investigate whether fragments like OBDD and DNNF which proved very interesting target languages for KC (Darwiche and Marquis 2002) can be extended to S5. Indeed, Shannon expansion, which is based on case analysis and conditioning is key to compiling classical propositional formulae into such fragments (Huang and Darwiche 2007). Basically, for any chosen variable $x$, the idea is to turn a formula $\alpha$ into the equivalent formula $(x \wedge (\alpha \mid x)) \vee (\neg x \wedge (\alpha \mid \neg x))$; interestingly, both $x \wedge (\alpha \mid x)$ and $\neg x \wedge (\alpha \mid \neg x)$ are decomposable conjunctive formulae since $x$ does not occur in $\alpha \mid x$ or in $\alpha \mid \neg x$. The fact that a classical formula $\alpha$ is equivalent to its Shannon expansion $(x \wedge (\alpha \mid x)) \vee (\neg x \wedge (\alpha \mid \neg x))$ over $x$ follows easily from the fact that $x \wedge (\alpha \mid x) \equiv x \wedge \alpha$ and $\neg x \wedge (\alpha \mid \neg x) \equiv \neg x \wedge \alpha$. Unfortunately, such a property does not hold in S5, as the following example demonstrates:

**Example 15.** Consider $\alpha = \neg\mathbf{K}(\neg a \vee \neg b)$ and $l = \neg\mathbf{K}\neg a$. It is easily verified that $l \wedge \alpha \equiv \alpha$, but $l \wedge (\alpha|l) \equiv (\neg\mathbf{K}\neg a) \wedge (\neg\mathbf{K}\neg b)$, and hence $l \wedge (\alpha \mid l) \not\equiv l \wedge \alpha$.

[1]Proofs have been omitted for lack of space but can be found in the appendix of a long version available at `ftp://ftp.irit.fr/IRIT/RPDMP/PapersFargier/aaai10.pdf`.

This prevents the extension of OBDD, DNNF and related fragments to S5.

# 4 The S5-DNF Family

In this section, we investigate the KC properties of the $\mathsf{S5\text{-}DNF}_{\mathsf{L},\mathsf{L}'}$ fragments of S5, as well as the corresponding $\mathsf{s\text{-}S5\text{-}DNF}_{\mathsf{L},\mathsf{L}'}$ fragments of s-S5. We pay particular attention to the fragment $\mathsf{S5\text{-}DNF}_{\mathsf{DNF},\mathsf{CNF}}$, in which every positive (resp. negative) epistemic atom $\mathbf{K}\alpha$ (resp. $\neg\mathbf{K}\alpha$) is such that $\alpha \in$ DNF (resp. $\alpha \in$ CNF).

## Expressiveness and Succinctness

The completeness of $\mathsf{S5\text{-}DNF}_{\mathsf{L},\mathsf{L}'}$ w.r.t. S5 whenever L and L′ are complete w.r.t. PDAG follows from:

**Proposition 16.** *If each* PDAG *formula admits an at most single-exponential representation in* L *and in* L′*, then every S5 formula can be transformed into an equivalent formula in $\mathsf{S5\text{-}DNF}_{\mathsf{L},\mathsf{L}'}$ which is at most single-exponentially larger.*

The proof of Proposition 16 yields a procedure for compiling arbitrary S5 formulae into $\mathsf{S5\text{-}DNF}_{\mathsf{L},\mathsf{L}'}$, consisting of a first step in which we leverage the distributivity property of $\wedge$ over $\vee$ to obtain an equivalent formula in S5-DNF, a second step in which we apply the equivalence $\mathbf{K}(\varphi \wedge \psi) \equiv (\mathbf{K}\varphi) \wedge (\mathbf{K}\psi)$ to group $\mathbf{K}$-literals together in every conjunction, and a final step in which we put the propositional formulae behind $\mathbf{K}$ into the required form.

The following proposition shows how one can take advantage of succinctness results given in the classical KC map to derive succinctness results for fragments of S5-DNF.

**Proposition 17.** *Let* L, L′*, and* L′′ *be complete subsets w.r.t.* PDAG*. We have:*

$$\mathsf{L} \leq_s \mathsf{L}' \text{ iff } \mathsf{S5\text{-}DNF}_{\mathsf{L},\mathsf{L}''} \leq_s \mathsf{S5\text{-}DNF}_{\mathsf{L}',\mathsf{L}''}$$

*The same holds if we replace S5-DNF by s-S5-DNF.*

## Queries and Transformations

Negative results about queries are easily transferred from DNF to $\mathsf{S5\text{-}DNF}_{\mathsf{L},\mathsf{L}'}$ (even when restricted to s-S5):

**Proposition 18.** *Assuming P≠NP, $\mathsf{S5\text{-}DNF}_{\mathsf{L},\mathsf{L}'}$ does not satisfy VA, SE, EQ, or $\mathsf{IM}_{\mathsf{DNF,CNF}}$. The same is true for $\mathsf{s\text{-}S5\text{-}DNF}_{\mathsf{L},\mathsf{L}'}$.*

The remaining two queries, CO and $\mathsf{CE}_{\mathsf{CNF,DNF}}$, are known to be feasible for DNF. The following proposition shows that under certain conditions, these results can be lifted to $\mathsf{S5\text{-}DNF}_{\mathsf{L},\mathsf{L}'}$.

**Proposition 19.**

- *Let* L *and* L′ *be dual subsets of* PDAG*. If* L *satisfies CO and* ∧BC*, then $\mathsf{S5\text{-}DNF}_{\mathsf{L},\mathsf{L}'}$ satisfies CO. Conversely, if $\mathsf{S5\text{-}DNF}_{\mathsf{L},\mathsf{L}'}$ satisfies CO, then* L *satisfies CO.*
- *If $\mathsf{S5\text{-}DNF}_{\mathsf{L},\mathsf{L}'}$ satisfies CO and eCD, then it also satisfies $\mathsf{CE}_{\mathsf{CNF,DNF}}$.*

*Both statements also hold for $\mathsf{s\text{-}S5\text{-}DNF}_{\mathsf{L},\mathsf{L}'}$.*

We briefly explain the intuition behind the first statement. According to Proposition 5, to test if $\mathbf{K}\alpha \wedge \neg\mathbf{K}\beta$ is consistent, we must check whether the classical formula $\alpha \wedge \neg\beta$ is consistent. By requiring that L and L′ be dual, we can

transform in polytime $\neg\beta$ into an equivalent formula in L. We can then use bounded conditioning ($\wedge$BC) to conjoin the resulting formula with $\alpha$, and finally leverage CO to test if the resulting formula in L is consistent.

For the closure transformations, we have the following:

**Proposition 20.**

- *S5-DNF$_{\mathsf{L},\mathsf{L}'}$ satisfies $\vee$C and thus $\vee$BC.*
- *If* L *satisfies $\wedge$BC, then S5-DNF$_{\mathsf{L},\mathsf{L}'}$ satisfies $\wedge$BC.*
- *For* L $\subseteq$ PDAG*, let* L*[$\wedge$] denote the conjunctive closure of* L*, i.e., the language defined inductively by:* L $\subseteq$ L*[$\wedge$], and if $\alpha_1, \ldots, \alpha_n \in$ L[$\wedge$], then $\bigwedge_{i=1}^{n} \alpha_i \in$ L[$\wedge$].*
  *If* L*[$\wedge$] $<_s$* L*, S5-DNF$_{\mathsf{L},\mathsf{L}'}$ satisfies neither $\wedge$C nor $\neg$C.*

*The above results also hold for s-S5-DNF$_{\mathsf{L},\mathsf{L}'}$.*

Thus, we see that closure transformations which hold for DNF can be lifted to S5-DNF$_{\mathsf{L},\mathsf{L}'}$. Similarly, negative results for $\wedge$C and $\neg$C also extend to S5-DNF$_{\mathsf{L},\mathsf{L}'}$.

We can obtain polytime forgetting for S5-DNF$_{\mathsf{L},\mathsf{L}'}$ under certain restrictions on L and L$'$.

**Proposition 21.**

- *If* L *and* L$'$ *are dual,* L *satisfies CO, $\wedge$BC, and FO, and there is a polytime transformation from* L *to* DNF*, then S5-DNF$_{\mathsf{L},\mathsf{L}'}$ satisfies FO. We can drop the requirement of a polytime transformation from* L *to* DNF *for s-S5-DNF$_{\mathsf{L},\mathsf{L}'}$.*
- *If* DNF $\not\leq_s$ L*, then FO is not satisfied by S5-DNF$_{\mathsf{L},\mathsf{L}'}$.*

To understand the need for a polynomial translation from L to DNF in the first statement, consider an epistemic term $\tau \wedge \mathbf{K}\alpha$. The forgetting of $\mathcal{V}$ from this formula can be shown to be equivalent to $\mathsf{forget}(\tau \wedge \alpha, \mathcal{V}) \wedge \mathbf{K}\,\mathsf{forget}(\alpha, \mathcal{V})$. The first conjunct can be computed by putting $\tau \wedge \alpha$ into L (using $\wedge$BC), and then employing a polytime forgetting procedure for L. However, the result of this forgetting will be a formula from L, whereas we require the objective part of it to belong to DNF. This is where the polytime translation from L to DNF comes into play. Of course, if we are working with s-S5-DNF$_{\mathsf{L},\mathsf{L}'}$, then the component terms have no objective parts, so this concern does not apply.

The following proposition shows that syntactic conditioning is easily lifted to S5-DNF$_{\mathsf{L},\mathsf{L}'}$, but considerably more is needed to get the general form of conditioning.

**Proposition 22.**

- *S5-DNF$_{\mathsf{L},\mathsf{L}'}$ satisfies CD if both* L *and* L$'$ *satisfy CD.*
- *If* L *and* L$'$ *are dual,* L *satisfies CO, $\wedge$BC, and FO, and there are polynomial translations from* DNF *to* L *and back, then S5-DNF$_{\mathsf{L},\mathsf{L}'}$ satisfies eCD. We can drop the requirement of a polynomial translation from* L *to* DNF *for s-S5-DNF$_{\mathsf{L},\mathsf{L}'}$ when conditioning by terms from s-S5.*

Applying the preceding results to the specific case where L = DNF and L$'$ = CNF, we obtain the following:

**Corollary 23.** *S5-DNF$_{\mathsf{DNF},\mathsf{CNF}}$ satisfies CO, CE$_{\mathsf{CNF},\mathsf{DNF}}$, $\wedge$BC, $\vee$C, FO, and eCD, but not VA, SE, EQ, or IM (unless P=NP). It satisfies neither $\wedge$C nor $\neg$C.*

Thus, S5-DNF$_{\mathsf{DNF},\mathsf{CNF}}$ provides exactly the same polytime queries and transformations as DNF, a positive result. The second item of Proposition 21 implies that we cannot use a more succinct language than DNF for L if we want to satisfy FO. However, this is only true for full S5. If we work with the subjective fragment of S5, then there are other interesting choices for L. Suppose that L satisfies CO, $\wedge$BC, FO, there is a polytime translation from DNF to L, and we define L$' = \{\neg\varphi \mid \varphi \in \mathsf{L}\}$ (i.e., we force L and L$'$ to be dual). Then s-S5-DNF$_{\mathsf{L},\mathsf{L}'}$ will satisfy CO, CE$_{\mathsf{CNF},\mathsf{DNF}}$, $\wedge$BC, $\vee$C, FO, and eCD. Some recently introduced languages satisfy these conditions on L, e.g. AFF[$\vee$] and KROM$-$C[$\vee$] from (Fargier and Marquis 2008), or DNNF$_T$ from (Pipatsrisawat and Darwiche 2008). Moreover, these languages are all strictly more succinct than DNF. This means that the s-S5-DNF$_{\mathsf{L},\mathsf{L}'}$ fragments they induce are strictly more succinct than s-S5-DNF$_{\mathsf{DNF},\mathsf{CNF}}$ (following Proposition 17).

### Zoom on the CE$_{\mathsf{CNF},\mathsf{DNF}}$ Query

Associated to each of the results of this section showing that a language satisfies a query or a transformation is a polytime procedure which achieves it. For space reasons, we cannot present all such procedures, so instead we focus on one particular query, epistemic clausal entailment CE$_{\mathsf{CNF},\mathsf{DNF}}$ since it is one of the more fundamental queries in reasoning.

Deciding whether an epistemic clause $\lambda$ from S5-CL$_{\mathsf{CNF},\mathsf{DNF}}$ is entailed by a S5-DNF$_{\mathsf{DNF},\mathsf{CNF}}$ formula $\alpha = \bigvee_{i=1}^{n} \gamma_i$ amounts to checking whether $\lambda$ is entailed by each epistemic term $\gamma_i$, i.e., whether $\gamma_i \wedge \neg\lambda \models \bot$. In order to perform such consistency tests, we transform each formula $\gamma_i \wedge \neg\lambda$ in polytime into an epistemic term from S5-TE$_{\mathsf{DNF},\mathsf{CNF}}$. To do so, we first turn $\neg\lambda$ into an equivalent term $\bar{\lambda}$ from S5-TE$_{\mathsf{DNF},\mathsf{CNF}}$ using classical transformations. Afterwards, we "group" the $\mathbf{K}$ literals from $\gamma_i$ and $\bar{\lambda}$ together using the equivalence $(\mathbf{K}\psi) \wedge (\mathbf{K}\varphi) \equiv \mathbf{K}(\psi \wedge \varphi)$ and the fact that DNF satisfies $\wedge$BC. All that remains then is to decide consistency of terms from S5-TE$_{\mathsf{DNF},\mathsf{CNF}}$. According to Proposition 5, an epistemic term $\tau \wedge \mathbf{K}\psi \wedge \neg\mathbf{K}\chi_1 \wedge \ldots \wedge \neg\mathbf{K}\chi_n$ is consistent if and only if $\tau \wedge \psi$ is consistent and each $\psi \wedge \neg\chi_i$ is consistent. For terms in S5-TE$_{\mathsf{DNF},\mathsf{CNF}}$, $\psi \in$ DNF, $\chi_i \in$ CNF and $\tau$ is a classical term. Using the duality of CNF and DNF, and the fact that DNF satisfies $\wedge$BC, we can put $\tau \wedge \psi$ and the $\psi \wedge \neg\chi_i$ into DNF, and then apply the linear time consistency procedure for DNF.

As a matter of illustration, suppose we want to show that $\mathbf{K}a$ entails $\mathbf{K}(a \wedge b) \vee \neg\mathbf{K}b$ from S5-CL$_{\mathsf{CNF},\mathsf{DNF}}$. We first conjoin $\mathbf{K}a$ with the negation of the clause, yielding $\mathbf{K}a \wedge \neg\mathbf{K}(a \wedge b) \wedge \mathbf{K}b$. Next we combine the two $\mathbf{K}$ atoms: $\mathbf{K}(a \wedge b) \wedge \neg\mathbf{K}(a \wedge b)$. We then combine the formulae in the positive atom with the negation of the formula in the negative atom: $(a \wedge b) \wedge \neg(a \wedge b)$. The resulting DNF $(a \wedge b \wedge \neg a) \vee (a \wedge b \wedge \neg b)$ is inconsistent, hence we obtain $\mathbf{K}a \models \mathbf{K}(a \wedge b) \vee \neg\mathbf{K}b$.

## 5 Application to Epistemic Planning

In this section, we briefly explain how our results can be applied to the problem of epistemic planning, see e.g. (Fagin et al. 1995; Brafman, Halpern, and Shoham 1998; Herzig, Lang, and Marquis 2003). Specifically, we show how progressing an epistemic state by an action can be done in polytime provided that the epistemic state is represented as a S5-DNF$_{\mathsf{DNF},\mathsf{CNF}}$ formula and the classical part of the action is represented as a DNF formula.

We recall that epistemic planning differs from traditional planning by allowing *epistemic actions*, which change the epistemic state of an agent, in addition to standard ontic (world-altering) actions. The distinction between a property holding and an agent *knowing* that a property holds is fundamental in this setting; indeed, the goal is often for an agent to determine whether or not a given property holds. As a matter of example, consider a simple circuit with two toggles $t_1$ and $t_2$ and two bulbs $b_1$ and $b_2$; the circuit is such that $b_1$ is lit iff $t_1$ and $t_2$ are both up, and $b_2$ is lit iff $t_1$ and $t_2$ are either both up or either both down. Suppose we have an agent who knows the circuit description (a static law $stat$ given by $stat = ((t_1 \wedge t_2) \Leftrightarrow b_1) \wedge ((t_1 \Leftrightarrow t_2) \Leftrightarrow b_2)$) and wants to determine the status of $t_2$ (up or down). If the agent is able to observe each bulb (thanks to the epistemic actions $o_1$, $o_2$) and to switch $t_1$ (thanks to the ontic action $s_1$), then she will be able to achieve her goal, using the following knowledge-based program: $\pi = o_2\,;$ if $\mathbf{K}b_2$ then $o_1$ else $(s_1\,; o_1)$. The validity of $\pi$ can be proved by showing that the epistemic state which results from the progression of the initial epistemic state represented by $\mathbf{K}stat$ by $\pi$ entails the goal $\mathbf{K}t_2 \vee \mathbf{K}\neg t_2$ (Herzig, Lang, and Marquis 2003).

We now discuss how off-line progression can be computed. First we consider the case where we want to progress an epistemic state by an epistemic action, which is typically represented by a formula of the form $\mathbf{K}\alpha_1 \vee \ldots \vee \mathbf{K}\alpha_k$, expressing the action's possible effects on what is known. In our example, $o_1 = \mathbf{K}b_1 \vee \mathbf{K}\neg b_1$ and $o_2 = \mathbf{K}b_2 \vee \mathbf{K}\neg b_2$. The fact that $\mathsf{S5\text{-}DNF}_{\mathsf{DNF,CNF}}$ satisfies $\wedge\mathsf{BC}$ shows that progressing an epistemic state (represented by a $\mathsf{S5\text{-}DNF}_{\mathsf{DNF,CNF}}$ formula) by an epistemic action into a new $\mathsf{S5\text{-}DNF}_{\mathsf{DNF,CNF}}$ formula can be done in polynomial time.

Progression of epistemic states by ontic actions can be carried out by considering two time-stamped copies $x_b, x_a$ of each propositional symbol $x$ ($b$ stands for "before" and $a$ for "after") and representing each ontic action $ont$ as an objective formula over the extended alphabet expressing how the action constrains the worlds before and after its execution. The progression of an epistemic state $\Phi$ by $ont$ is then given by the formula $\mathsf{forget}(\Phi_b \wedge \mathbf{K}ont, B)$ where each $a$ subscript has been removed (here, $\Phi_b$ is $\Phi$ with each symbol $x$ is replaced by $x_b$, and $B$ is the set of all symbols with a $b$ subscript). In our example, the action $s_1$ can be represented by $(t_{1_a} \Leftrightarrow \neg t_{1_b}) \wedge (t_{2_a} \Leftrightarrow t_{2_b}) \wedge stat_a$. Thanks to our results, once $ont$ has been turned into a DNF formula, the fact that $\mathsf{S5\text{-}DNF}_{\mathsf{DNF,CNF}}$ satisfies $\wedge\mathsf{BC}$ and FO ensures that computing this progression can be achieved in polynomial time when $\Phi$ is represented as a $\mathsf{S5\text{-}DNF}_{\mathsf{DNF,CNF}}$ formula.

Finally, in order to progress conditional actions of the form (if $\kappa$ then $a_1$ else $a_2$), one needs to progress each epistemic term of the epistemic state in S5-DNF by either action $a_1$ or $a_2$ (depending on whether the term entails the knowledge precondition $\kappa$), then take the disjunction of the resulting formulae. It follows from our results that the whole process can be done efficiently provided each epistemic state is represented as a $\mathsf{S5\text{-}DNF}_{\mathsf{DNF,CNF}}$ formula and each condition as a $\mathsf{S5\text{-}CNF}_{\mathsf{CNF,DNF}}$ formula. The final goal satisfaction test can also be done in polytime assuming the goal is represented by a $\mathsf{S5\text{-}CNF}_{\mathsf{CNF,DNF}}$ formula.

# 6 Conclusion

In this paper, we investigated the knowledge compilation task for propositional epistemic logic S5. Counterparts for the well-known classical fragment DNF have been studied, leading to the family of languages $\mathsf{S5\text{-}DNF}_{\mathsf{L,L'}}$. We identified a number of conditions on L, L′ for which queries and transformations (suitably extended to the epistemic case) are satisfied or not by the languages of the form $\mathsf{S5\text{-}DNF}_{\mathsf{L,L'}}$. The fragment $\mathsf{S5\text{-}DNF}_{\mathsf{DNF,CNF}}$ proved particularly interesting as it satisfies all of the queries and transformations that are satisfied by DNF. In particular, consistency-testing, forgetting, and suitably generalized forms of clausal entailment and conditioning are all feasible in polynomial time for $\mathsf{S5\text{-}DNF}_{\mathsf{DNF,CNF}}$ formulae, whereas none of these tasks is tractable for arbitrary S5 formulae. When only the subjective part of S5 is of interest, then other recently studied fragments, like $\mathrm{DNNF}_T$ or (disjunctive closure of) affine or Krom CNF, can be used in place of DNF in positive epistemic atoms, offering a gain in succinctness while allowing the same polytime queries and transformations. We also showed that no S5 counterparts for OBDD, DNNF, and other fragments based on Shannon expansion can be defined since this property does not hold in this logical setting.

# References


Brafman, R.; Halpern, J.; and Shoham, Y. 1998. On the knowledge requirements of tasks. *Artificial Intelligence* 98(1-2):317–349.

Cadoli, M., and Donini, F. 1998. A survey on knowledge compilation. *AI Communications* 10(3–4):137–150.

Darwiche, A., and Marquis, P. 2002. A knowledge compilation map. *Journal of Artificial Intelligence Research* 17:229–264.

Fagin, R.; Halpern, J.; Moses, Y.; and Vardi, M. 1995. *Reasoning about knowledge*. The MIT Press.

Fargier, H., and Marquis, P. 2008. Extending the knowledge compilation map: Krom, Horn, affine and beyond. In *Proc. of AAAI'08*, 442–447.

Ghilardi, S., and Zawadowski, M. W. 1995. Undefinability of propositional quantifiers in the modal system S4. *Studia Logica* 55(2):259–271.

Herzig, A.; Lang, J.; and Marquis, P. 2003. Action representation and partially observable planning using epistemic logic. In *Proc. of IJCAI'03*, 1067–1072.

Huang, J., and Darwiche, A. 2007. The language of search. *Journal of Artificial Intelligence Research* 29:191–219.

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

Pipatsrisawat, K., and Darwiche, A. 2008. New compilation languages based on structured decomposability. In *Proc. of AAAI'08*. 517-522.

Zhang, Y., and Zhou, Y. 2009. Knowledge forgetting: Properties and applications. *Artificial Intelligence* 173(16-17):1525–1537.