Parameterized Complexity of Dynamic Belief Updates: A Complete Map

Thomas Bolander[1] and Arnaud Lequen[2]

[1] DTU Compute, Technical University of Denmark, tobo@dtu.dk
[2] Univ Rennes, ENS Rennes, France, arnaud.lequen@ens-rennes.fr

Abstract. Dynamic Belief Update (DBU) is a model checking problem in Dynamic Epistemic Logic (DEL) concerning the effect of applying a number of epistemic actions on an initial epistemic model. It can also be considered as a plan verification problem in epistemic planning. The problem is known to be PSPACE-hard. To better understand the source of complexity of the problem, previous research has investigated the complexity of 128 parameterized versions of the problem with parameters such as number of agents and size of epistemic actions. The complexity of many parameter combinations has been determined, but previous research left 14 parameter combinations open. In this paper, we solve all of these open problems. Most of the parameter combinations turns out to be fixed-parameter intractable, except 3 that are fixed-parameter tractable.

Keywords: Parameterized Complexity · Model Checking · Dynamic Epistemic Logic · Plan Verification

Disclaimer

This version is a revised version of the original article, which mistakenly states that acopu-DBU is fixed-parameter intractable, while it is actually fixed-parameter tractable.

In our original article, Theorem 3 erroneously characterizes the problem acopu-DBU as FP-intractable, while we only show the FP-intractability of acpu-DBU[3]. In this version, the statement of the theorem has been corrected accordingly. Nonetheless, further investigation showed that acopu-DBU is actually FP-tractable. This is shown by Theorem 4, which is only a slightly stronger version of the theorem presented in our original paper.

The proofs of both theorems remain very close to the original proofs. In addition, their corollaries are left untouched, and every other result remains valid.

[3] The authors would like to sincerely thank Marco Montali for noticing this mistake

1 Introduction

In the fields of psychology, ecology, economy, and various areas of computer science like automated planning and distributed systems, the need often arises to model multi-agent systems and reason about the knowledge of the involved agents. Indeed, situations where multiple human or artificial agents interact with their environment, and have to update their knowledge accordingly, are ubiquitous. Our cognitive ability to attribute mental states to ourselves and to others has been previously studied under the name of *Theory of Mind reasoning* [20,22].

There is a consensus across all fields that the problem posed by Theory of Mind reasoning is a computationally intractable task. Nonetheless, the exact aspects that are responsible for its hardness are still open to discussion. Indeed, in many real-life situations, humans manage to reason fairly efficiently about the knowledge of themselves and other agents. A conjecture is that higher-order thinking is the most prominent source of intractability, and situations requiring modest depth of reasoning are the easiest to untangle. However, this claim is still at the heart of a decades-old debate, where different explanations collide [16,24].

As an illustration of some of the potential complications, let us consider the *coordinated attack problem* [13], which is famous in the area of distributed systems. In this problem, two generals, i and j, are stationed in separate camps and plan to capture a fortified place. They will only succeed by attacking simultaneously, and thus have to agree on a common time to begin the assault. In addition, they will only attack if they are certain that the other general will follow through. However, they can only communicate by means of a messenger, who might be captured while travelling between camps. In this setting, each general has to reason about the knowledge of the other, and its evolution after each message that is sent. Suppose general i decides to attack at dawn. General i will need to ensure that general j knows, and will hence send a messenger to let j know. A while after having dispatched the messenger, general i might ask himself: "*Does general j now know* that I will attack at dawn?". Suppose the message was successfully delivered. Upon reception of the message, general j might also reflect on general i's knowledge, and ask himself: "*Does general i know* that *I know* that he will attack at dawn?". As general i has no way of knowing whether the message arrived safely or not (the messenger might have been caught), general j can acknowledge the message by sending another message back to general i's camp, who will then wonder: "Does general j know that I know that he knows that...?".

The coordinated attack problem is famous for the fact that, no matter how many messages are being sent back and forth, the two generals can never achieve common knowledge that they will attack at dawn, and can hence never safely coordinate an attack [13]. What *will* happen, though, is that their level of shared knowledge will grow higher and higher as more and more messages are successfully delivered. After the successful delivery of the first message, we will have that "general j knows that i will attack at dawn". After the successful delivery of 2 messages (one from i to j and an acknowledgement from j to i), we will have that "general i knows that general j knows that i will attack at dawn".

After 3 messages we have "general j knows that general i knows that general j knows that i will attack at dawn", etc. The potential source of computational complexity here is that each message delivery results in a higher level of Theory of Mind (higher level of shared knowledge), and this might require a larger model to represent. The coordinated attack problem actually doesn't at all exploit the worst-case computational behaviour of reasoning about the higher-order knowledge resulting from a sequence of actions. However, it still points to the fact that such problems can be non-trivial and it is not immediately obvious whether we can always ensure tractability (we can't).

Dynamic Epistemic Logic (DEL) is a well-suited framework to model situations like the coordinated attack problem, as it is a family of modal logics that allow not only to reason about (higher-order) knowledge, but also to represent how such knowledge is dynamically updated through the occurrence of events. Even if many decision problems associated with DEL are provably hard [11,20], it has been shown that simpler tasks involving DEL can be carried out fairly easily [20]. In this paper, we study the Dynamic Belief Update (DBU) problem, which boils down to verifying whether an epistemic formula holds in an epistemic model after a series of epistemic updates, *i.e.*, whether a certain epistemic fact holds after a sequence of (epistemic) events have occurred in an initial (epistemic) situation. In the coordinated attack problem, we could for instance ask about whether the two generals have shared knowledge to depth n about attacking at dawn after the successful delivery of n messages. This would be an instance of the DBU problem: The initial epistemic situation is the one where general i has decided to attack at dawn, the sequence of epistemic events is the n messages sent back and forth, and the epistemic fact to be check after the sequence of events is whether nth-order shared knowledge has been achieved.

In previous work, van de Pol *et al.* [20] identifies numerous dimensions of interest of a DBU instance. Those dimensions include the number of events that occur, the modal depth of the epistemic formula to be checked after the events have occurred, and the number of agents involved. The restriction of one or several dimensions of the problem forms a new, simpler problem, which is potentially easier. Using tools from parameterized complexity [12], it has been shown that some restricted problems are tractable, while others remain hard. Our work consists in extending the efforts of van de Pol *et al.* [20] to identify which aspects of DBU make it intractable. We manage to settle the tractability question of all problems left open by van de Pol *et al.*.

In section 2, we present the DEL framework of this paper, and after recalling notions of parameterized complexity, we present DBU and its parameters. In section 3, we prove our new fixed-parameter intractability results of DBU. In section 4, we settle the last remaining problems by proving their tractability.

2 Background

2.1 Dynamic Epistemic Logic

Dynamic Epistemic Logic (DEL) is a modal logic focused on reasoning about knowledge, which can be revised according to the evolution of the situation [11]. In this paper, we use a variant of DEL that allows multi-pointed epistemic models and has propositional postconditions [6]. While various other variants of DEL exist, we present here a simple version, that can be readily extended into a version of DEL with more general preconditions, postconditions, or frame conditions. The advantage of choosing a restricted version of DEL is that we mainly prove intractability results, and those intractability results will then of course immediately generalise from the restricted to the generalised versions of DEL.

The language $\mathcal{L}_K(P, \mathcal{A})$ of multi-agent epistemic logic is defined as follows, where p ranges over a finite set of propositional variables (atoms) P, and i over a finite set of agents $\mathcal{A}$:

$$\varphi := \top \mid p \mid \neg\varphi \mid \varphi \wedge \varphi \mid K_i\varphi,$$

The intended meaning of $K_i\varphi$ is "agent i knows φ". We will often use the abbreviated notation $\hat{K}_i\varphi = \neg K_i\neg\varphi$, which reads "agent i considers φ possible". Other symbols such as $\vee$ and $\rightarrow$ can be defined by abbreviation as usual. The semantics of the language is defined through *epistemic models* (Kripke models).

Example 1. Our running example will be the coordinated attack problem, informally introduced in Section 1. A propositional variable (atom) d is used to express that "general i attacks at dawn". For $k = i, j$, we use the atom m_k to denote that the messenger is currently at the camp of general k. We can then use the formula $\neg m_i \wedge \neg m_j$ to express that the messenger has been captured (he is no longer at any of the two camps). As mentioned in the introduction, after 2 messages successfully delivered, general i knows that general j knows that i will attack at dawn, which we can express formally as the formula K_iK_jd. Hence the relevant formulas of the coordinated attack problem can be expressed in the language $\mathcal{L}_K(P, \mathcal{A})$ with $P = \{d, m_i, m_j\}$ and $\mathcal{A} = \{i, j\}$.

Definition 1. (Epistemic Models) *An* epistemic model *for the language $\mathcal{L}_K(P, \mathcal{A})$ is a tuple $\mathcal{M} = (W, \sim, L, W_d)$, where:*

- *W is a non-empty finite set of worlds.*
- *$W_d \subseteq W$ is a non-empty set of* designated worlds.
- *$\sim$ is a function assigning an equivalence relation $\sim_i$ on W to every agent $i \in \mathcal{A}$, called the* indistinguishability relation *for agent i.*
- *$L : W \rightarrow 2^P$ is a function assigning to each world w a* labelling*, which is the set of atoms true in w.*

When $W_d = \{w_d\}$ for some $w_d \in W$, $\mathcal{M}$ is called a single-pointed epistemic model*, and w_d is then often referred to as the* actual world*. By slight abuse of notation, we will then write $\mathcal{M} = (W, \sim, L, w_d)$. Otherwise, $\mathcal{M}$ is a* multi-pointed epistemic model.

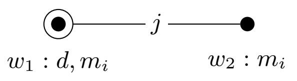

Fig. 1: An epistemic model $\mathcal{M}_0 = (W, \sim, L, w_1)$ for $\mathcal{L}_K(\{d, m_i, m_j\}, \{i, j\})$. In this model, $W = \{w_1, w_2\}$, $\sim_i = \{(w_1, w_1), (w_2, w_2)\}$, $\sim_j = \{(w_1, w_1), (w_2, w_2), (w_1, w_2), (w_2, w_1)\}$, $L(w_1) = \{d, m_i\}$ and $L(w_2) = \{m_i\}$. Even though they are not depicted for readability reasons, there are reflexive edges on each world, labelled by i and j. In general, we are going to omit reflexive edges in figures.

Definition 2. (Truth in an epistemic model) *Let $\mathcal{M} = (W, \sim, L, W_d)$ be an epistemic model. We define truth as follows, where $w \in W$, $p \in P$, $i \in \mathcal{A}$, and $\varphi, \psi \in \mathcal{L}_K(P, \mathcal{A})$:*

$(W, \sim, L, W_d) \models \varphi$	*iff*	*for all* $w \in W_d$, $(W, \sim, L, w) \models \varphi$
$(W, \sim, L, w) \models \top$		*always*
$(W, \sim, L, w) \models p$	*iff*	$p \in L(w)$
$(W, \sim, L, w) \models \neg\varphi$	*iff*	$(W, \sim, L, w) \not\models \varphi$
$(W, \sim, L, w) \models \varphi \wedge \psi$	*iff*	$(W, \sim, L, w) \models \varphi$ *and* $(W, \sim, L, w) \models \psi$
$(W, \sim, L, w) \models K_i\varphi$	*iff*	*for all* w' *s.t.* $w \sim_i w'$, $(W, \sim, L, w') \models \varphi$

Example 2. Figure 1 shows an epistemic model $\mathcal{M}_0$ representing the initial situation of the coordinated attack problem. Nodes depict the *worlds* of $\mathcal{M}_0$, and represent different possible states of affairs. Each node is labelled by its name, followed by the list of the atomic propositions that are true in the world. The world on the left, w_1, represented by a double circle, is the *actual world.* It means that *actually* general i has decided to attack at dawn (d), and *actually* the messenger is at the camp of general i (m_i), which is expressed by $\mathcal{M}_0 \models d \wedge m_i$. However, in the other world w_2, general i has not decided to attack at dawn, and d is not true. Since general j does not know whether general i has decided to attack at dawn or not, general j deems worlds w_1 and w_2 equally possible, which is illustrated by the *indistinguishability edge* labelled j between worlds w_1 and w_2. As such, $\mathcal{M}_0 \models \neg K_j d$.

Event models, defined next, represent changes to the situation resulting from the occurrence of events or execution of actions, which lead agents to update their knowledge.

Definition 3. (Event Models) *An* event model *for $\mathcal{L}_K(P, \mathcal{A})$ is a tuple $\mathcal{E} = (E, \sim, \mathsf{pre}, \mathsf{post}, E_d)$, such that:*

- *E is a non-empty finite set of* events*.*
- *$E_d \subseteq E$ is a non-empty set of* designated events*.*
- *$\sim$ is a function assigning an equivalence relation $\sim_i$ on W to every agent $i \in \mathcal{A}$, called the* indistinguishability relation *for agent i.*
- *$\mathsf{pre} : E \to \mathcal{L}_K(P, \mathcal{A})$ is a function assigning to each event a precondition.*
- *$\mathsf{post} : E \to \mathcal{L}_K(P, \mathcal{A})$ is a function assigning to each event a postcondition, which is a conjunction of literals (propositional variables and their negations, including $\top$) in which each atom appears at most once.*

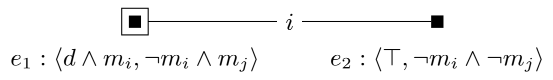
i
$e_1 : \langle d \wedge m_i, \neg m_i \wedge m_j \rangle$
$e_2 : \langle \top, \neg m_i \wedge \neg m_j \rangle$

Fig. 2: The event model $\mathcal{E}_{i \to j} = (E, \sim, \mathsf{pre}, \mathsf{post}, e_1)$. In this model, $E = \{e_1, e_2\}$, $\sim_i = \{(e_1, e_1), (e_2, e_2), (e_1, e_2), (e_2, e_1)\}$, $\sim_j = \{(e_1, e_1), (e_2, e_2)\}$, $\mathsf{pre}(e_1) = d \wedge m_i$, $\mathsf{post}(e_1) = \neg m_i \wedge m_j$, $\mathsf{pre}(e_2) = \top$, $\mathsf{post}(e_2) = \neg m_i \wedge \neg m_j$.

As with epistemic models, when $E_d = \{e_d\}$ for some $e_d \in E$, $\mathcal{E}$ is called a single-pointed event model *and e_d often the* actual event*. By slight abuse of notation, we will then write $\mathcal{E} = (E, \sim, \mathsf{pre}, \mathsf{post}, e_d)$. Otherwise, $\mathcal{E}$ is a* multi-pointed event model*.*

Example 3. Figure 2 depicts the event model $\mathcal{E}_{i \to j}$ representing the event/action where the messenger successfully delivers message d from i to j. Here the nodes are called *events*. They represent the possible outcomes of the action. Similarly to the epistemic model presented earlier, event e_1 on the left is represented by a double square, which indicates that it is the actual event. It is labelled by a pair of two formulas, which are respectively its precondition $\mathsf{pre}(e_1) = d \wedge m_i$ and its postcondition $\mathsf{post}(e_1) = \neg m_i \wedge m_j$. Intuitively, event e_1 models the outcome where the messenger safely arrives at general j's camp. The occurrence of d in the precondition of e_1 means that agent j gets to know that d is true (since agent j can distinguish e_1 from e_2). The postcondition $\neg m_i \wedge m_j$ of e_1 makes m_j true instead of m_i, that is, represents the fact that the messenger ends up being at general j instead of general i. The event e_2 represents the situation where the messenger gets caught (ends up being at neither of the two camps). As general i has no way of knowing wether the messenger will be successful in his quest, he does not know which of the two events will actually occur, and can hence not distinguish between e_1 and e_2. This is the reason we have an *indistinguishability edge* labelled i between the two events. In addition to event model $\mathcal{E}_{i \to j}$ depicted here, we will consider the additional event model $\mathcal{E}_{j \to i}$ that represents the symmetric situation where general j sends the messenger to general i's camp. It can be obtained from $\mathcal{E}_{i \to j}$ by swapping all occurrences of i and j.

Epistemic models can be updated with the application of event models through *product updates*, defined as follows.

Definition 4. (Product Update) *Let $\mathcal{M} = (W, \sim, L, W_d)$ be an epistemic model and $\mathcal{E} = (E, \sim, \mathsf{pre}, \mathsf{post}, E_d)$ an event model. Suppose that there exists at least one pair $(w, e) \in W_d \times E_d$ such that $(W, \sim, L, w) \models \mathsf{pre}(e)$. The* product update *of $\mathcal{M}$ with $\mathcal{E}$ is then the epistemic model $\mathcal{M} \otimes \mathcal{E} = \mathcal{M}'$ with $\mathcal{M}' = (W', \sim', L', W'_d)$ and*

- $W' = \{(w, e) \in W \times E \mid (W, \sim, L, w) \models \mathsf{pre}(e)\}$
- $\sim'_i = \{((w, e), (v, f)) \in W' \times W' \mid w \sim_i v$ *and* $e \sim_i f\}$
- $L'((w, e)) = (L(w) \cup \{p \mid \mathsf{post}(e) \models p\}) - \{p \mid \mathsf{post}(e) \models \neg p\}$

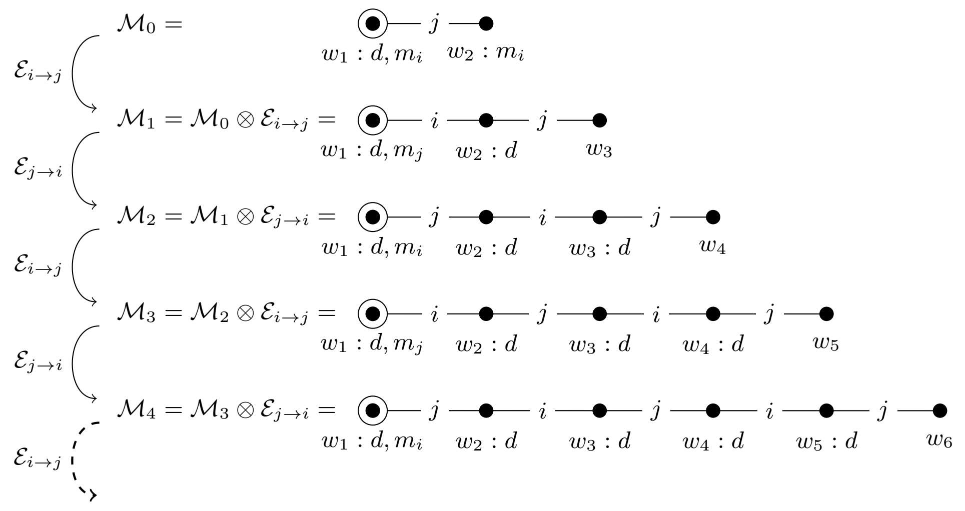

Fig. 3: The epistemic model $\mathcal{M}_0$ of Example 2 and its successive updates with event models $\mathcal{E}_{i\to j}$ and $\mathcal{E}_{j\to i}$.

– $W'_d = \{(w,e) \in W' \mid w \in W_d \textit{ and } e \in E_d\}$

Example 4. Figure 3 depicts the evolution of the coordinated attack problem when messages are sent back and forth. The initial situation, represented by the epistemic model $\mathcal{M}_0$ at the top of the figure, is the same as described in Example 2. Its update with the epistemic model $\mathcal{E}_{i\to j}$ is depicted immediately below, noted $\mathcal{M}_1$. Let us explain how $\mathcal{M}_1 = \mathcal{M}_0 \otimes \mathcal{E}_{i\to j}$ is built. In order to build the set of worlds of $\mathcal{M}_1$, one has to check, for each event of $\mathcal{E}_{i\to j}$, which worlds of $\mathcal{M}_0$ satisfy its precondition. The designated event e_1 has its precondition satisfied in the designated world w_1. The model $\mathcal{M}_1$ thus has a world (w_1, e_1), which is also designated and depicted as the leftmost world of the model (and renamed to w_1 for simplicity). As the postcondition of e_1 is $\neg m_i \wedge m_j$, the label of (w_1, e_1) is the same as w_1, except that m_i is now false and m_j true. Event e_2, that general i can not distinguish from event e_1, has its precondition satisfied in both worlds of $\mathcal{M}_0$. Event e_2 will hence create a complete copy of the original model $\mathcal{M}_0$, except the postcondition will be imposed on the copy. The postcondition of e_2 makes both m_i and m_j false. This explains why the two rightmost worlds of $\mathcal{M}_1$ is a copy of $\mathcal{M}_0$ with m_i removed.

Model $\mathcal{M}_n$ is the model resulting from n successful message passings. From the figure we can see that $\mathcal{M}_n$ for all n is a chain model in which the first world, w_1, is the actual world, and in which d holds in all worlds except the last, w_{n+2}. Furthermore, it is an alternating chain, where the edges alternate between i-edges and j-edges. From this it follows that we have $\mathcal{M}_n \models K_{k_1} K_{k_2} \cdots K_{k_n} d$ for any sequence of agents $k_1, \ldots, k_n$. E.g. in $\mathcal{M}_3$ we have both $\mathcal{M}_3 \models K_i K_j K_i d$ and

$\mathcal{M}_3 \models K_j K_i K_j d$. The conclusion is, as hinted at before, that after n successful message passings, the two generals will have shared knowledge to degree n that they will attack at dawn.

2.2 Generalisations of the DEL framework

One could also define the precondition of an event as a formula of the language $\mathcal{L}_{DK}(P, \mathcal{A})$ which is $\mathcal{L}_K(P, \mathcal{A})$ extended with modalities $[\mathcal{E}]$ for each event model $\mathcal{E}$. In this extended language, $[\mathcal{E}]\phi$ expresses that after the action $\mathcal{E}$ has occurred, ϕ is true. We here don't consider this extended language and the generalised preconditions it leads to. All our intractability results of course trivially generalise to this more general setting (since anything expressible in the more restricted setting is also expressible in the more general setting). The same holds for generalising to other frame conditions, that is, alternative conditions on $\sim$ that do not require the $\sim_i$ relations to be equivalence relations, but do not prevent them from being it either. This includes various versions of DEL where the underlying epistemic models represent beliefs rather than knowledge—or other propositional attitudes weaker than knowledge. Specifically, it includes DEL based on e.g. K or KD45 and also DEL based on plausibility models [2]. As we will later see, even our tractability results generalise from S5 to other frame conditions (in fact, arbitrary frame conditions), since the proof doesn't make use of any assumptions about frame conditions. In other words, all our results, both the tractability and intractability results, generalise to doxastic versions of DEL.

Finally, we can also consider to generalise the form of the postconditions. Often in DEL, the postconditions are expressed as mappings from propositions to epistemic formulas [10], so that the updated truth value of a proposition can potentially depend on the prior truth value of a complicated epistemic formula. Using such postconditions, many event models can be expressed more compactly. Given a postcondition of our type (a conjunction of literals) of size (length) n, we can always translate it into a mapping from propositions to epistemic formulas of size $\Theta(n)$ (at least when assuming, as is usually done, that we only encode how the mapping differs from the identity). Hence, we again get that any intractability result we might achieve for our type of postconditions transfers to the generalised postconditions.

2.3 Parameterized Complexity

In this section, we recall some notions of parameterized complexity. Parameterized complexity is a branch of complexity theory whose aim is to offer a finer-grained analysis of a computational problem, taking into account some characteristics of each instance. It studies *parameterized problems* [12], which resemble classical decision problems. Given an alphabet Σ, a parameterized problem L is a subset of $\Sigma^* \times \mathbb{N}$. Given an instance $\langle x, k\rangle$ of L, we call x the *main part* and k the *parameter*. The parameter k is a metric that gauges one dimension of x. For instance, if our problem is to model-check formulas of $\mathcal{L}_K(P, \mathcal{A})$, then x consists of a formula ϕ and a model $\mathcal{M}$, while k can e.g. be the modal depth of ϕ or the number of agents used in ϕ and $\mathcal{M}$.

In classical complexity theory, the class of tractable problems is P. The corresponding class in parameterized complexity theory is FPT, defined next.

Definition 5. (Fpt-algorithms and fixed-parameter tractability [14, Definition 1.4]) *Let L be a parameterized problem. An* fpt-algorithm *for problem L is an algorithm that solves L, for which there exists a computable function $f : \mathbb{N} \to \mathbb{N}$ and a polynomial $\mathcal{P}$, such that the running time of the algorithm on any instance $\langle x, k \rangle \in L$ is at most*

$$f(k) \cdot \mathcal{P}(|x|).$$

A parameterized problem is called fixed-parameter tractable *if there exists an fpt-algorithm solving it, and the class of all fixed-parameter tractable problems is denoted FPT.*

A well-known problem that is intractable in the traditional sense is SAT [9]. However, one of its parameterized variants p-SAT, where the number of propositional variables p of an instance is a parameter of the problem, is fixed-parameter tractable.

Indeed, let us try to check whether formula φ, built over p variables, is satisfiable. Checking whether a given assignment of the p variables satisfies the formula can be done in time $\mathcal{P}(|\varphi|)$, for some polynomial $\mathcal{P}$. As there are 2^{p} different assignments of the variables of φ, one could check them all against φ, which can be done in time $2^{\mathsf{p}} \cdot \mathcal{P}(|\varphi|)$. We have thus described an fpt-algorithm for the p-SAT problem. Intuitively, this means that if we consider the number of variables p of any instance of SAT to be a constant, SAT becomes tractable.

Proving that a parameterized problem is or is not fixed-parameter tractable can often be done through *fpt-reductions*, defined next. They can be seen as the parameterized complexity counterpart of classical polynomial-time reductions, and are useful for proving membership and hardness results for parameterized problems.

Definition 6. (Fpt-reductions [14, Definition 2.1]) *Let L and L' be two parameterized problems. An* fpt-reduction *from L to L' is a mapping $R : L \longrightarrow L'$ such that:*

- *$\langle x, k \rangle \in L$ iff $R(\langle x, k \rangle) \in L'$.*
- *R is computable by an fpt-algorithm,* i.e.*, there is a computable function f and a polynomial $\mathcal{P}$ such that $R(\langle x, k \rangle)$ can be computed in time $f(k) \cdot \mathcal{P}(|x|)$.*
- *There exists a computable function g such that if $\langle x, k \rangle \in L$ and $\langle x', k' \rangle = R(\langle x, k \rangle)$ then $k' \leq g(k)$.*

When there exists an fpt-reduction from L to L', we write $L \leq_{fpt} L'$.

It follows from the way fpt-reductions are defined that the problem class FPT is closed under fpt-reductions. More specifically, suppose $L \leq_{fpt} L'$. Then if L' belongs to FPT, so does L [14, Lemma 2.2]. Hence, to prove that a problem L'

Dynamic Belief Update (DBU)

Input: An initial epistemic model $\mathcal{M}$ on $\mathcal{L}_K(P, \mathcal{A})$;
A series of event models $\mathcal{E}_1, \ldots, \mathcal{E}_u$ on $\mathcal{L}_K(P, \mathcal{A})$;
A goal formula $\varphi_g \in \mathcal{L}_K(P, \mathcal{A})$.

Output: *Yes* if $\mathcal{M} \otimes \mathcal{E}_1 \otimes \cdots \otimes \mathcal{E}_u \models \varphi_g$
No otherwise

Fig. 4: The decision problem DBU considered in this paper

is not fixed-parameter tractable, it suffices to find an fpt-reduction to L' from a problem L known to be not fixed-parameter tractable (we call such problems *fixed-parameter intractable*). In this paper, we consider two complexity classes that are deemed fixed-parameter intractable, namely W[1] and para-NP [12]. W[1] is the class of problems that can be fpt-reduced to k-W2SAT, which is the problem where, given a 2CNF formula φ and a parameter k, one has to decide if there exists a valuation satisfying φ in which at most k variables are true. Para-NP is the class of parameterized problems that can be solved by a nondeterministic fpt-algorithm. Para-NP-hard problems are deemed fixed-parameter intractable, as W[1] $\subseteq$ para-NP [14].

In the remaining of this paper, we will allow problems to have multiple parameters. If a problem L has parameters $\mathsf{k}_1, \ldots, \mathsf{k}_n$, we denote it $\mathsf{k}_1 \cdots \mathsf{k}_n$-$L$. The instances of the problem are then of the form $\langle x, \mathsf{k}_1 + \cdots + \mathsf{k}_n \rangle$. Note that, for any permutation σ of $\mathsf{k}_1, \ldots, \mathsf{k}_n$, the notations $\mathsf{k}_1 \cdots \mathsf{k}_n$-$L$ and $\sigma(\mathsf{k}_1 \cdots \mathsf{k}_n)$-$L$ define the same problem. If the parameter sequence contains repetitions, e.g. $\mathsf{k}_1\mathsf{k}_2\mathsf{k}_1$-$L$, we still only count each distinct parameter once, so the instances of that problem would be of the form $\langle x, \mathsf{k}_1 + \mathsf{k}_2 \rangle$. When adding further parameters to a parameterized problem, we of course make it more constrained. That is, for any problem L and parameter sequences X and Y, the problem XY-L is at least as constrained as X-L. Hence the following is easily proved.

Proposition 1. *Let X and Y be sequences of parameters of a decision problem L. Then XY-$L \leq_{fpt} X$-L.*

2.4 Dynamic Belief Update

The decision problem considered in this paper is presented in Figure 4, following van de Pol *et al.* [20]. It is the problem of checking whether a certain epistemic formula is true after having updated an initial epistemic situation (epistemic model) with a sequence of epistemic actions (event models).[4] So it is about the

[4] The term "Dynamic Belief Update" might reasonably be considered a bit unfortunate. A potentially better name could be "Dynamic Knowledge Update" as we are here only considering models where the underlying accessibility relations are equivalence relations (*i.e.*, S5). However, the choice of naming of the problem was already

Param.	Description	Param.	Description
a	Number of agents	o	Goal formula's modal depth
c	Max. length of event preconditions	p	Number of prop. variables
e	Max. no. of events per event model	u	Number of event models
f	Length of goal formula		

Table 1: Parameters for DBU

Param. for DBU	Complexity
acfou	W[1]-hard
afopu	W[1]-hard
eu	FPT

Earlier known results [20]

Param. for DBU	Complexity
acefop	para-NP-hard
cfopu	W[1]-hard
acpu	W[1]-hard
acopu	FPT

New results of this paper

Table 2: Complexity results for the most general parameterized variants of DBU, from which all other results for our set of parameters can be immediately deduced. Results on the left table originate from [20], while results on the right table constitute the original contributions of this paper.

complexity of keeping track of "who knows what" when observing a sequence of actions taking place, where these actions can both change ontic facts and what the different agents know. A concrete example could be the coordinated attack problem presented above. The problem of whether n messages lead to nth-order shared knowledge can be stated as the following instance of DBU. We let $\mathcal{M}$ be the initial epistemic model $\mathcal{M}_0$ of the coordinated attack problem presented in Figure 1. For $m = 1, \ldots, n$, if m is odd we let $\mathcal{E}_m$ be the event model $\mathcal{E}_{i \to j}$ of Figure 2, and if m is even we let $\mathcal{E}_m$ be the symmetric event model $\mathcal{E}_{j \to i}$. Finally, we let the goal formula ϕ_g be the formula $\bigwedge_{k_1,\ldots,k_n} K_{k_1} K_{k_2} \cdots K_{k_n} d$. Now we have that the instance is positive if and only if the successful delivery of the first n messages lead to nth-order shared knowledge of d among agents i and j. Other problems that can be phrased as instances of DBU are the consecutive number puzzle, the muddy children puzzle, and problems related to epistemic games like Hanabi and Clue, as well as the false-belief tasks studied in cognitive psychology [7,5,3]. We can also think of DBU as the plan verification problem in epistemic planning [6]: Given an initial state (epistemic model), a sequence of actions (event models) and a goal formula, does the action sequence achieve the goal from the initial state?

DBU is PSPACE-complete, as proven by van de Pol *et al.* [20]. Their paper proposes various parameters as an attempt to identify the features that make DBU hard. Those parameters are given in Table 1, and any combination of the 7 parameters form a parameterized version of DBU. This leads us to the class of

established by van de Pol [20] from which our work departures, so it seems even more unfortunate if we decided to change the naming convention. Furthermore, all our results still hold when considering doxastic versions of DEL based on e.g. KD45 or plausibility models, as argued in Section 2.2.

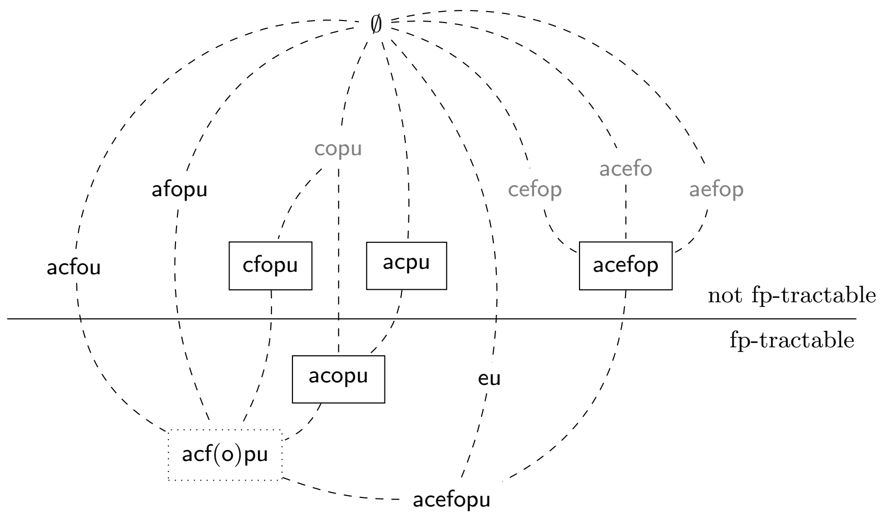

Fig. 5: Complexity results for the most general parameterized variants of DBU. Each parameter sequence X shown in the figure is short for the associated decision problem X-DBU. The problems above the vertical line are the ones that are not fixed-parameter tractable, and the ones below are those which are. The dashed edges show problem inclusion: the top end-point of the edge has a parameter sequence included in the sequence of the bottom end-point. This implies that the problem at the bottom reduces to the problem at the top, and hence that the problem at the bottom is easier than the problem at the top. The framed parameter sequences are associated to problems whose tractability is settled by one of the major results of this paper. The parameter sequence in dotted frame has its tractability settled as a corollary of one of the main results of this paper, and is part of the tractability frontier. Problems in gray formed the previously known border, but are now superseeded by one of our results. Notice that acfopu-DBU = acfpu-DBU.

problems of the form X-DBU, where X is a subsequence of the 7 parameters. For instance, acp-DBU is the dynamic belief update problem where the parameters are the number of agents, the length of the preconditions and the number of propositional variables. There are $2^7 = 128$ problems of this form. Prior to our work, the (fixed-parameter) tractability or intractability of 114 of them was already known [20]. Of the remaining 14 problems, we show intractability of 11 of them, and tractability for 3. We also provide alternative proofs for some problems that were previously settled. Table 2 summarizes the known results, including the new ones of this paper, and Figure 5 shows the same results alternatively, by providing an outline of the tractability frontier. Both only mention the strongest results, as all other results can be immediately deduced from them through Proposition 1, and the observation that, for any sequence of parameters X of DBU, Xf-DBU $\leq_{fpt}$ Xfo-DBU (if we constrain the length of the goal formula, we are also constraining its modal depth).

It can be hard to keep track of 128 different versions of the same problem. However, many are obviously interdependent in the sense that the (in)tractability of one immediately implies the (in)tractability of the other, e.g. through Proposition 1. To keep track of dependency and which problems are still open, we developed a small script, which can be found at `https://github.com/arnaudlequen/dbuproblemfinder`. The script allowed us to find the open problems that would solve most other open problems, and keeping track of the remaining open problems as we gradually settled more cases.

2.5 Single-pointed vs multi-pointed DBU

We have defined the DBU problem on multi-pointed epistemic models and event models. In classical DEL, one usually only considers single-pointed models with a single designated world (actual world) of each epistemic model and a single designated event (actual event) of each event model. In our intractability proofs, we are however going to use multi-pointed models extensively, and they are also essential to represent non-deterministic actions in epistemic planning [6]. We here show that the question of fixed-parameter tractability is independent of whether we consider single- or multi-pointed models.

Proposition 2. *Let X denote a subsequence of the 7 parameters of Table 1, and let X-DBU$_s$ denote the X-DBU problem restricted to single-pointed epistemic models and event models. Then X-DBU $\leq_{fpt}$ X-DBU$_s$ and X-DBU$_s$ $\leq_{fpt}$ X-DBU.*

Proof. The reduction X-DBU$_s$ $\leq_{fpt}$ X-DBU is trivial, as any problem in X-DBU$_s$ is also a problem in X-DBU. We then prove the other direction, X-DBU $\leq_{fpt}$ X-DBU$_s$. To prove this, we first define a translation $\cdot_s$ from multi-pointed models over the language $\mathcal{L}_K(P, \mathcal{A})$ into single-pointed models. To define the translation, we first pick a fresh agent j and fresh propositional symbol p_j. Let $\mathcal{A}' = \mathcal{A} \cup \{j\}$ and $\mathcal{P}' = \mathcal{P} \cup \{p_j\}$. The translation will then take any multi-pointed epistemic (or event) model over $\mathcal{L}_K(P, \mathcal{A})$ and turn it into a single-pointed epistemic (or event) model over $\mathcal{L}_K(P', \mathcal{A}')$. For any epistemic model $\mathcal{M} = (W, \sim, L, W_d)$, the translated model $\mathcal{M}_s = (W', \sim', L', w_d)$ is defined by:

- $W' = W \cup \{w_d\}$, where $w_d \notin W$.
- $L'(w) = L(w)$ for all $w \in W$, and $L'(w_d) = \{p_j\}$.
- $\sim'_i = \sim_i$ for all $i \in \mathcal{A}$, and $\sim'_j = ((W_d \cup \{w_d\}) \times (W_d \cup \{w_d\}))^r$ (where R^r denotes the reflexive closure of relation R).

Note that by this construction, for any formula ϕ in $\mathcal{L}_K(P, \mathcal{A})$, we will have $\mathcal{M} \models \phi$ iff $\mathcal{M}_s \models K_j(\neg p_j \to \phi)$. Similarly, for any event model $\mathcal{E} = (E, \sim, \mathsf{pre}, \mathsf{post}, E_d)$, we define its translation $\mathcal{E}_s = (E', \sim', \mathsf{pre}', \mathsf{post}', e_d)$ by:

- $E' = E \cup \{e_d\}$, where $e_d \notin E$.
- $\sim'_i = \sim_i$ for all $i \in \mathcal{A}$, and $\sim'_j = ((E_d \cup \{e_d\}) \times (E_d \cup \{e_d\}))^r$.
- $\mathsf{pre}'(e) = \neg p_j \wedge \mathsf{pre}(e)$ for all $e \in E$, and $\mathsf{pre}'(e_d) = p_j$.
- $\mathsf{post}'(e) = \mathsf{post}(e)$ for all $e \in E$, and $\mathsf{post}'(e_d) = \top$.

Claim. For any epistemic model $\mathcal{M}$ and event model $\mathcal{E}$ over $\mathcal{L}_K(P, \mathcal{A})$, the models $(\mathcal{M} \otimes \mathcal{E})_s$ and $\mathcal{M}_s \otimes \mathcal{E}_s$ are isomorphic. Proof of claim: We create a bijection

f between the worlds of the two models as follows. The model $(\mathcal{M} \otimes \mathcal{E})_s$ has a single designated world w_d. The model $\mathcal{M}_s \otimes \mathcal{E}_s$ also has a single designated world (w_d, e_d). So we let $f(w_d) = (w_d, e_d)$. Now take any other world w of $(\mathcal{M} \otimes \mathcal{E})_s$. This will be a world (w, e) of $\mathcal{M} \otimes \mathcal{E}$. By definition, we then have that w is a world of $\mathcal{M}$, e is an event of $\mathcal{E}$, and $\mathsf{pre}(e)$ is true in w. Since w is a world of $\mathcal{M}$, it is also a world of $\mathcal{M}_s$. Since e is an event of $\mathcal{E}$, it is also an event of $\mathcal{E}_s$. Since $\mathsf{pre}(e)$ is true in w, (w, e) must then be a world of $\mathcal{M}_s \otimes \mathcal{E}_s$. We can hence let $f((w, e)) = (w, e)$. We then first need to prove that this is a bijection. As f is clearly injective, it suffices to prove that it is surjective. So let (w, e) be any world of $\mathcal{M}_s \otimes \mathcal{E}_s$. Either w is a world of $\mathcal{M}$ or $w = w_d$. If $w = w_d$ then $e = e_d$, since e_d is the only event of $\mathcal{E}_s$ that has its precondition satisfied in w_d. But then $(w, e) = (w_d, e_d)$, and we know that this is in the image of f. Now suppose $w \neq w_d$. Then also $e \neq e_d$, since the precondition of e_d is p_j that is only satisfied in w_d. This implies that w belongs to $\mathcal{M}$ and e to $\mathcal{E}$. By definition of the product update, we also have that $\mathsf{pre}(e)$ is satisfied in w. This implies that (w, e) also belongs to $(\mathcal{M} \otimes \mathcal{E})_s$, and hence that (w, e) must be in the image of f, as required. We now know that f is a bijection, and need to check that it satisfies the conditions for being an isomorphism. First we check that for any world w of $(\mathcal{M} \otimes \mathcal{E})_s$, w and $f(w)$ satisfy the same propositional variables. This is trivial for all world except w_d of $(\mathcal{M} \otimes \mathcal{E})_s$. The world w_d of $(\mathcal{M} \otimes \mathcal{E})_s$ satisfies only p_j, so we need to show that $f(w_d) = (w_d, e_d)$ also only satisfies p_j. But since w_d only satisfies p_j and the event e_d has the trivial postcondition $\top$, it immediately follows. Now we only need to check that the two models also have the same indistinguishability relations between worlds related by f. Suppose $w \sim_i v$ for a pair of distinct worlds w, v of $(\mathcal{M} \otimes \mathcal{E})_s$ and an agent i. Since the indistinguishability relations of both models are trivially reflexive, we can assume $w \neq v$. Suppose first $i = j$. Suppose further that $w = w_d$. Then $v = (w', e')$ for some $w' \in W_d$ and $e' \in E_d$. Since $w' \in W_d$, in $\mathcal{M}_s$ we have $w_d \sim_j w'$. Since $e' \in E_d$, in $\mathcal{E}_s$ we have $e_d \sim_j e'$. Hence $(w_d, e_d) \sim_j (w', e')$ in $\mathcal{M}_s \otimes \mathcal{E}_s$. Since $(w_d, e_d) = f(w_d)$ and $(w', e') = f((w', e'))$, we then get $f(w) \sim_i f(v)$, as required. Suppose that instead $w \neq w_d$, still assuming $i = j$. We can assume that also $v \neq w_d$, as otherwise the case is symmetric to the previous one. In this case, we must have that $w = (w', e')$ and $v = (v', f')$ where $w', v' \in W_d$ and $e', f' \in E_d$. It follows that $w' \sim_j v'$ in $\mathcal{M}_s$ and $e' \sim_j f'$ in $\mathcal{E}_s$, and hence $(w', e') \sim_j (v', f')$ in $\mathcal{M}_s \otimes \mathcal{E}_s$. This again shows the required. This covers the case where $i = j$. Now assume $i \neq j$. Then we must have that $w = (w', e')$ and $v = (v', f')$, where $w' \sim_i v'$ in $\mathcal{M}$ and $e' \sim_i f'$ in $\mathcal{E}$. Then also $w' \sim_i v'$ in $\mathcal{M}_s$ and $e' \sim_i f'$ in $\mathcal{E}_s$. Hence $(w', e') \sim_i (v', f')$ in $\mathcal{M}_s \otimes \mathcal{E}_s$, as required. This completes the proof of the claim.

To prove X-$\mathsf{DBU} \leq_{fpt} X$-$\mathsf{DBU}_s$, we define a mapping R from X-DBU to X-DBU_s as follows. For each X-DBU instance $\langle x, k \rangle$, we let $R(\langle x, k \rangle) = \langle x', k' \rangle$ be the X-DBU_s instance where

- If x consists of the elements $\mathcal{M}$, $\mathcal{E}_1, \ldots, \mathcal{E}_u$ and ϕ_g, then x' consists of the elements $\mathcal{M}_s$, $(\mathcal{E}_1)_s, \ldots, (\mathcal{E}_u)_s$ and $K_j(\neg p_j \rightarrow \phi_g)$.
- $k' = g(k) = k + 13$.

R and g are clearly computable. We hence only need to prove that $\langle x, k\rangle$ is a positive instance of X-DBU iff $\langle x', k'\rangle$ is a positive instance of X-DBU_s. As earlier concluded, we have $\mathcal{M}\otimes\mathcal{E}_1\otimes\cdots\otimes\mathcal{E}_u \models \phi_g$ iff $(\mathcal{M}\otimes\mathcal{E}_1\otimes\cdots\otimes\mathcal{E}_u)_s \models K_j(\neg p_j \rightarrow \phi)$. By the claim above, we get that $(\mathcal{M}\otimes\mathcal{E}_1\otimes\cdots\otimes\mathcal{E}_u)_s \models K_j(\neg p_j \rightarrow \phi)$ is equivalent to $\mathcal{M}_s \otimes (\mathcal{E}_1)_s \otimes \cdots \otimes (\mathcal{E}_u)_s \models K_j(\neg p_j \rightarrow \phi)$. Hence x is a positive instance iff x' is. The only thing left to check is the size of the parameter $k' = g(k)$. We need to ensure that k' is less than or equal to the sum of the parameters in X for the problem x'. Compared to x, the input x' uses 1 additional agent, the maximal length of preconditions is at most 3 higher, the maximal number of events per event model is at most 1 higher, the length of the goal formula is 6 higher, it's modal depth is 1 higher, and the number of propositional variables is 1 higher. Hence the sum of the parameters in X is at most $1 + 3 + 1 + 6 + 1 + 1 = 13$ higher for x' than x. Hence, we can bound k' by $k + 13$.

3 New intractability results

In this section, we present proofs for the intractability of three parameterized variants of DBU, namely acefop-DBU, cfopu-DBU, and acpu-DBU. As the sequences of parameters considered consist of four parameters or more, the intractability of numerous other problems is also settled by our proofs. However, we will only mention explicitly, as corollaries, the problems whose intractability was previously unknown.

Theorem 1. *acefop-DBU is fixed-parameter intractable (more precisely, para-NP-hard). In other words, the Dynamic Belief Update problem is intractable even when restricting the number of propositional variables and agents (p,a), the maximum number of events in event models (e), the maximum length of event preconditions (c), and the length and modal depth of the goal formula (f,o).*

Proof. The proof is by an fpt-reduction from an NP-hard problem to an instance of acefop-DBU with fixed values of a, c, e, f, o and p, thus proving para-NP-hardness of the latter (since the NP-hard problem doesn't have any parameter, the reduction is also a regular polynomial reduction). The construction used in the proof is an adaptation of the proof of Theorem 19 of Bolander *et al.* [6]. The general idea is to simulate, through an instance of DBU, the execution of a fixed nondeterministic Turing machine M that solves a given NP-hard problem (any NP-hard problem will do). We begin by encoding the initial configuration of the machine (*i.e.*, its tape, the position of its head and its internal state) into the initial epistemic model. Then, we build a series of event model updates, such that the epistemic model after n product updates contains the representation of every configuration of M that can be reached in exactly n transitions (computation steps). Finally, we build a goal formula that checks whether an accepting configuration was encountered in the process or not. Thus, the DBU instance is positive if and only if M accepts the word in the input.

Let $M = (\mathcal{S}, \Gamma, q_0, \delta, q_f)$ be any nondeterministic Turing machine that solves an NP-hard problem in polynomial time, with states $\mathcal{S} = \{q_0, q_1, \ldots, q_f\}$, where

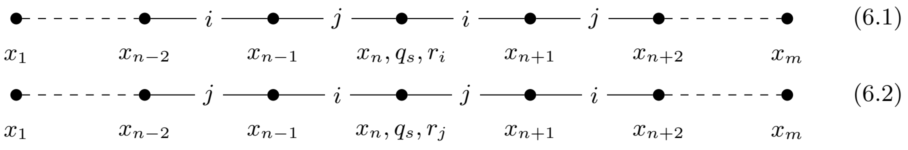

Fig. 6: Two information cells for agent k, both representing the ID $x_1 \cdots x_{n-1} q_s x_n \cdots x_m$ of the Turing machine $M = (\mathcal{S}, \Gamma, q_0, \delta, q_f)$, where $x_i \in \Gamma$ and $q_s \in \mathcal{S}$. This ID represents the configuration of M where the word on the tape is $x_1 \cdots x_m$, where M is in state q_s, and the head is at the nth symbol x_n of the word on the tape. Recall that $\sim_k = \sim_i \cup \sim_j$ and $\sim_g = W \times W$ are implicitly assumed, where W is the set of all worlds. Note the similarity to the chain models obtained when formalising the coordinated attack problem, cf. Figure 3. In both cases, the models have the structure of an alternating i, j-chain of worlds.

q_0 is the only initial state, q_f is the only accepting state, Γ is the set of tape symbols including the blank symbol # and δ is the transition function [17]. The DBU instance we build has agents $\mathcal{A} = \{i, j, k, g\}$ and propositional variables $P = \Gamma \cup \mathcal{S} \cup \{r_i, r_j, t\}$. Information cells for agent k (*i.e.*, sets $W_k \subseteq W$ of maximum size that are closed under $\sim_k$) are used to encode configurations of M, and agents i and j are used to distinguish the right and the left of each cell of the tape that we encode. We will in all epistemic models enforce $\sim_k = \sim_i \cup \sim_j$ by having $\sim_k = \sim_i \cup \sim_j$ in the initial model, and $\sim_k = \sim_i \cup \sim_j$ in all event models. We will similarly enforce $\sim_g$ to be the universal relation—*i.e.*, make any two worlds indistinguishable—by making all pairs of worlds in the initial model indistinguishable, and by making all pairs of events of all event models indistinguishable. For simplicity, the $\sim_k$ and $\sim_g$ indistinguishability relations will not be explicitly drawn. Furthermore, the reflexive and transitive closure of all indistinguishability relations drawn is implicitly assumed.

A configuration of the machine can be represented by an Instantaneous Description (ID) [17]. Following Bolander *et al.* [6], we represent IDs by epistemic models as illustrated in Figure 6. This pair of information cells for agent k offers two unique representations of an ID [6], and we call *represented ID* an information cell for k that has the form of either (6.1) or (6.2). Each world represents one cell of the tape of the machine, and is marked with a propositional variable representing the symbol in the cell. One world is marked with two additional propositions: one for the current state of the machine (q_s), as well as either r_i or r_j. This world represents the current position of the head and is called the *current world* (not to be confused with the *actual world* of an epistemic model). The propositions r_i and r_j are used to distinguish between the right and the left of the current cell. If r_i (resp. r_j) is true, then the cell at the right of the current one is reachable through an i-edge (resp. j-edge).

We proceed to show how to build the initial epistemic model and event models. Suppose that in its initial configuration, M is in state q_0 and with the word $\omega = x_1 \cdots x_m$ on its tape. Then the initial epistemic model $\mathcal{M}_0$ is the represented ID of the initial configuration of M, as shown in Figure 7. In addition to that, we add a designated world w_t only labeled by the propositional variable t. Its purpose is to make sure the model doesn't end up being empty,

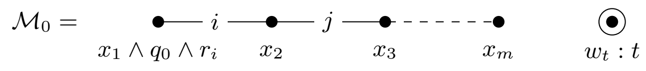

Fig. 7: The initial epistemic model $\mathcal{M}_0$ for the Turing machine M with input word $\omega = x_1 \cdots x_m$. It consists of the represented ID of the initial configuration of M plus an additional designated world w_t only accessible from the other worlds by the $\sim_g$ relation (recall that $\sim_k = \sim_i \cup \sim_j$ and $\sim_g = W \times W$ are implicitly assumed).

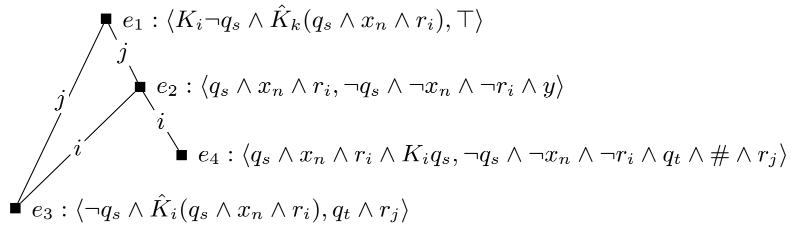
$e_1 : \langle K_i \neg q_s \wedge \hat{K}_k(q_s \wedge x_n \wedge r_i), \top \rangle$
j
$e_2 : \langle q_s \wedge x_n \wedge r_i, \neg q_s \wedge \neg x_n \wedge \neg r_i \wedge y \rangle$
j
i
i
$e_4 : \langle q_s \wedge x_n \wedge r_i \wedge K_i q_s, \neg q_s \wedge \neg x_n \wedge \neg r_i \wedge q_t \wedge \# \wedge r_j \rangle$
$e_3 : \langle \neg q_s \wedge \hat{K}_i(q_s \wedge x_n \wedge r_i), q_t \wedge r_j \rangle$

Fig. 8: The transition component τ_l^i, for a transition l of the form $\delta(q_s, x_n) = (q_t, y, R)$, where $x_n \neq y$.

which could otherwise happen if at some point no transition can be applied to any ID.

The next step consists in building the series of event models, which are all copies of a single model $\mathcal{E}_{trans}$. The aim of $\mathcal{E}_{trans}$ is to simulate one step of the Turing machine M, by applying all applicable transitions to each represented ID of the previous epistemic model. The event model mainly consists in a disjoint union of several sub-event models, that we call *transition components*, whose purpose is to attempt to apply a transition of the Turing machine M to a represented ID. For each transition l, *i.e.*, each element of the transition function δ, we construct an i-transition component τ_l^i and a j-transition component τ_l^j. We construct these transition components such that given an ID s and valid transition l for s, applying τ_l^i (resp. τ_l^j) to the represented ID of s, of the form (6.1) (resp. (6.2)), will result in the represented ID of the successor of s after l was applied. Applying to an ID s a transition component whose form does not match the represented ID of s, or whose transition is not applicable to s, will yield no worlds.

Figure 8 shows an example of an i-transition component. The j-transition component can be obtained by swapping i and j everywhere. Other transitions, such as $\delta(q_s, x_n) = (q_t, y, L)$ or transitions satisfying $x_n = y$, can be handled similarly. Let us try to explain the intuition behind this construction. It is very similar to the construction of Bolander *et al.* [6]. Event e_1 makes sure that, after the update, worlds that represent cells of the tape that are unaffected by the transition are left unchanged. It copies into the updated model every world of the represented ID, except the world representing the current head position and the one at its right. Event e_2 copies the current world, noted w, but removes the propositional variables that mark the head of the machine. It also updates the tape symbol. If the cell to the right of the current position of the head is not

$e_f : \langle q_f, \top \rangle$
$e_t : \langle t, \top \rangle$

Fig. 9: Event model σ. The purpose of e_f is to make sure to preserve any world marked with q_f, as the existence of such a world would mean that an accepting configuration has been reached (in one of the represented executions). Event e_t copies the world w_t as the only designated event, making sure always to preserve a single designated world (actual world) after each update. Recall that we have implicitly assumed $e_f \sim_g e_t$.

blank, then there exists a world w' to the right of the current world w, *i.e.*, such that $w \sim_i w'$. Event e_3 adds to w' the propositional variables that make it into the current world of the updated model. It updates as well the current state of the machine, from q_s to q_t. If the cell to the right of the current position of the head is blank, then no world is to the right of the current world. Event e_4 then creates it with a blank symbol, and sets it to be the current world of the updated model. Applying the i-transition component of Figure 8 to a represented ID s of the form (6.2) results in no world. Indeed, in s, the current world is instead labeled by r_j, and thus, no world verifies r_i. Therefore, no event has its precondition satisfied, as each of the four events $e_1, \dots, e_4$ has a precondition requiring r_i to hold in at least one world. Similarly, if the transition is not applicable to the ID represented by s, then the current world of s is labeled by $q'_s \neq q_s$ and/or $x'_n \neq x_n$, and thus does not satisfy $q_s \wedge x_n$. And as before, each of the four events $e_1, \dots, e_4$ has a precondition requiring $q_s \wedge x_n$ to hold in at least one world.

In order to build $\mathcal{E}_{trans}$, we need to introduce another component σ, which consists of two events, e_f and e_t. Those events, depicted in Figure 9, carry to the updated model the information that will eventually allow the goal formula to check whether the instance is positive or not. Building $\mathcal{E}_{trans}$ is then straightforward. In addition to σ, it consists in the disjoint union of the i- and j-transition components τ_l^i and τ_l^j associated to every transition l of M. Recall again that we implicitly assume to also add a g-edge between any pair of events. Applying $\mathcal{E}_{trans}$ to an epistemic model that contains the representations of all IDs reachable in n transitions results in a model containing the representations of all IDs reachable in $n+1$ transitions. If a model contains a world where q_f is true, then in the updated model, there is also a world where q_f if true.

By assumption, there exists a polynomial $\mathcal{P}$ such that, for any word ω', M accepts ω' iff M accepts it in at most $\mathcal{P}(|\omega'|)$ steps. Then, for our given input ω, we only need to simulate $\mathcal{P}(|\omega|)$ steps of M, and thus create a series of $\mathcal{P}(|\omega|)$ product updates of $\mathcal{M}_0$ with the event model $\mathcal{E}_{trans}$. In the final model, the only designated world is w_t, which is linked by a g-edge to every other remaining world. The goal formula $\hat{K}_g q_f$ must thus be true in the final model iff a world verifying q_f has been reach after some initial sequence of product updates, *i.e.*, if M can reach an accepting state in at most $\mathcal{P}(|\omega|)$ steps. Thus, M accepts input ω iff the instance of DBU with initial state $\mathcal{M}_0$, with $\mathcal{P}(|\omega|)$ copies of the event model $\mathcal{E}_{trans}$ and with goal formula $\hat{K}_g q_f$ is positive. Note that we indeed here have fixed values for all of the parameters a, c, e, f, o and p. We initially fixed 4 agents, so $\mathsf{a} = 4$. We defined $P = \Gamma \cup \mathcal{S} \cup \{r_i, r_j, t\}$, where Γ and $\mathcal{S}$ are finite sets of the fixed Turing machine M, so also p has a fixed value. The event models we have constructed are completely independent of the problem instance we consider (the input word), and even independent of which Turing machine M we start out with. Thus also c and e have fixed values. The goal formula is $\phi_g = \hat{K}_g q_f$, which clearly also has a fixed length and a fixed model depth independent of the input word, so also f and o have fixed values. We have hence successfully fpt-reduced the problem "Does M accept input ω?", where M is fixed and ω is the input, to the problem acefop-DBU. We comply with the conditions of Definition 6: we respectively satisfy the second and third conditions as the reduction is polynomial, and all parameters of acefop-DBU are constants, by construction. Finally, as M solves an NP-hard problem, acefop-DBU is para-NP-hard.

Corollary 1. *acp-DBU, acep-DBU and acfp-DBU are all fixed-parameter intractable.*

The corollary is by the application of Proposition 1 to Theorem 1. In addition to settling those four open problems, Theorem 1 shows a stronger result, which is that all parameterized versions of DBU that do not have u as a parameter are fixed-parameter intractable. This settles in itself the fixed-parameter intractability of 64 out of the total 128 problems. It also constitutes an alternative proof of the intractability of three different problems shown separately by van de Pol *et al.* [20], which are acefo-DBU, cefop-DBU and aefop-DBU.

We now prove fixed-parameter intractability of two further problems that were left open by van de Pol *et al.* [20]: cfopu-DBU and acpu-DBU. We here show that both are fixed-parameter intractable, which implies the fixed-parameter intractability of cfpu-DBU and acp-DBU. Our proofs of both theorems are adaptations of the fixed-parameter intractability proof of copu-DBU by van de Pol *et al.* [20]. In addition to strengthening their construction to be able to generalize their intractability results, we also simplify their construction in a few places. The general point is to show W[1]-hardness by a reduction from the earlier mentioned W[1]-complete problem k-W2SAT: Given a 2CNF input formula φ and a parameter k, decide whether there exists a valuation satisfying φ in which at most k variables are true.

In the following we assume the variables of φ are named $x_1, \ldots, x_m$. The general trick in constructing an fpt-reduction from k-W2SAT to a parameterized DBU problem is as follows. First we define epistemic (sub)models that can be used to encode propositional valuations over $\{x_1, \ldots, x_m\}$. We call these *valuation gadgets* and use $\mathcal{M}_v$ to denote the valuation gadget encoding the valuation v. The initial model of the DBU instance is then the model $\mathcal{M}_0$ where 0 denotes the valuation with $0(x_i) = 0$ for all i (the valuation that sets every variable false). We then construct an event model that can take any set of valuation gadgets and for each gadget $\mathcal{M}_v$ it constructs m new gadgets $\mathcal{M}_{v[x_1 \mapsto 1]}, \ldots, \mathcal{M}_{v[x_m \mapsto 1]}$ (where $v[x \mapsto 1]$ is the mapping that is as v except $v(x) = 1$). After updating k times with this event model, we are guaranteed to have gadgets representing all valuations where at most k variables are true. If we have no bound on f, we can now directly use the goal formula of the DBU instance to check that there

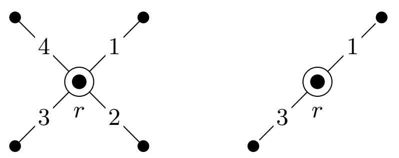

Fig. 10: Left: A valuation gadget for $m = 4$ representing the valuation 0 in which all x_i, $i = 1, \dots, m$, are false. Right: The gadget for the valuation where x_2 and x_4 are true (since the outgoing 2- and 4-edges have been deleted).

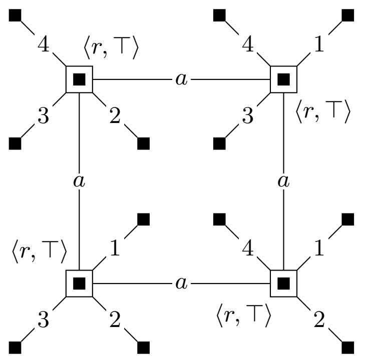
4
$\langle r, \top \rangle$
4
1
a
3
2
3
$\langle r, \top \rangle$
a
a
$\langle r, \top \rangle$
1
4
1
a
3
2
$\langle r, \top \rangle$
2

Fig. 11: The pointed event model $\mathcal{E}$ for $m = 4$. The unlabelled events are implicitly labelled $\langle \top, \top \rangle$.

exists a gadget making φ true. This is what we do for the intractability proof of acpu-DBU. If we have a bound on f, as in the intractability proof of cfopu-DBU, we need to perform product updates with additional event models that mark the gadgets making φ true.

Theorem 2. *cfopu-DBU is fixed-parameter intractable (W[1]-hard). In other words, the Dynamic Belief Update problem is intractable even when restricting the number of propositional variables (p), the number of event models (u), the maximum length of event preconditions (c), and the length and modal depth of the goal formula (f,o).*

Proof. The main contribution of this proof over the proof of the fixed-parameter intractability of copu-DBU by van de Pol *et al.* [20] is the construction of an additional event model ($\mathcal{E}_\varphi$) that allows us to only consider a goal formula of fixed length (while still preserving the fixed bound on the event preconditions). Let φ and k be given (an instance of k-W2SAT), where φ has variables $var(\phi) = \{x_1, \dots, x_m\}$. We will now create an instance of DBU that can decide the k-W2SAT instance, *i.e.*, whether there exists a valuation satisfying ϕ and setting at most k variables true. The DBU instance will be using agents $\mathcal{A} = \{1, \dots, m, a, b\}$. For each valuation v over $var(\varphi)$, we define the gadget $\mathcal{M}_v$ as the star-shaped model with a single root world satisfying proposition r, and for each x_i with $v(x_i) = 0$ it has an outgoing i-edge to a unique world satisfying no propositions. The construction is illustrated for $m = 4$ in Figure 10. Now consider the event model $\mathcal{E}$ illustrated for $m = 4$ in Figure 11. The events with no label are implicitly labelled $\langle \top, \top \rangle$, *i.e.*, they are events that preserve any world to which they are applied. The events labelled $\langle r, \top \rangle$ only apply to the roots of gadgets. When $\mathcal{E}$ is applied to a gadget $\mathcal{M}_v$, it creates m copies of the gadget, where in the first gadget x_1 is made true (by removing the outgoing 1-edge), in the second x_2 is made true (by removing the outgoing 2-edge), etc. These gadgets are furthermore connected by a-edges via their root worlds. When this event model is applied k times to the initial gadget model $\mathcal{M}_0$, we achieve a model with m^{k} gadgets connected by a-edges via their root worlds. Each gadget is obtained by starting with the initial gadget representing the valuation 0, and then making at most k variables true by consecutively removing k edges from the gadget model. Since we might attempt to remove the same edge multiple times, this construction gives us a representation of all valuations where at most k variables are true (except the valuation 0 that can be checked separately). Hence the final model $\mathcal{M}_0 \otimes \mathcal{E}^{\mathsf{k}}$ contains a gadget for each valuation with at most k variables set true (except the valuation 0).

Note that a clause $(\neg)x_i \vee (\neg)x_j$ is true in a valuation v iff the formula $(\neg)K_i r \vee (\neg)K_j r$ is true at the root of the gadget $\mathcal{M}_v$. We now construct an additional event model $\mathcal{E}_\varphi$ as follows. It has a single designated event labelled $\langle r, \top\rangle$. For each clause $(\neg)x_i \vee (\neg)x_j$ of φ, it has an additional event labelled $\langle r \wedge \neg((\neg)K_i r \vee (\neg)K_j r), f\rangle$, where f is a new propositional variable denoting "failure". All events of $\mathcal{E}_\varphi$ are connected by b-edges. Each event with postcondition f checks whether a particular clause of φ is false in the gadget to which it is applied. If it is, a b-accessible world satisfying f is created. When $\mathcal{E}_\varphi$ is applied to a valuation gadget, it will hence preserve the root (due to the event $\langle r, \top\rangle$), and additionally it will add a b-accessible f-world for each unsatisfied clause. If there are no unsatisfied clauses, it will only preserve the root. Hence, if we apply $\mathcal{E}_\varphi$ to the model $\mathcal{M}_0 \otimes \mathcal{E}^{\mathsf{k}}$ containing gadgets for all the relevant valuations, the resulting model $\mathcal{M}_0 \otimes \mathcal{E}^{\mathsf{k}} \otimes \mathcal{E}_\varphi$ will contain an r-world with no b-accessible f-worlds iff φ is true in one of the valuations. Hence, we can check whether φ is true in one of the relevant valuations by checking the goal formula $\varphi_g := \hat{K}_a(r \wedge K_b \neg f)$ in the model $\mathcal{M}_0 \otimes \mathcal{E}^{\mathsf{k}} \otimes \mathcal{E}_\varphi$.

To sum up, given a k-$\mathsf{W2SAT}$ instance φ with parameter k, we reduce it to the DBU instance with initial model $\mathcal{M}_0$, with k copies of the event model $\mathcal{E}$ followed by the event model $\mathcal{E}_\varphi$ and with goal formula φ_g. We now only have to verify that the reduction is an fpt-reduction from k-$\mathsf{W2SAT}$ to copu-DBU. Building the epistemic model $\mathcal{M}_0$ and the k copies of the event model $\mathcal{E}$ is clearly polynomial in m and k and hence in the input size of the k-$\mathsf{W2SAT}$ instance. Building $\mathcal{E}_\varphi$ is polynomial in the formula φ and hence also in the input size of the k-$\mathsf{W2SAT}$ instance. Finally, the goal formula has a fixed length. This shows that the reduction is computable by an fpt-algorithm. We then only need to show that the parameters of the translated cfopu-DBU instance can be bound by a computable function in k. The parameters $\mathsf{c}, \mathsf{f}, \mathsf{o}, \mathsf{p}$ all have a fixed value independent of the k-$\mathsf{W2SAT}$ instance, and u is $\mathsf{k}+1$. So the parameters are clearly bound by a computable function in k, and the proof is complete.

Theorem 3. *acpu-DBU is fixed-parameter intractable ($W[1]$-hard).*

Proof. The main contribution of this proof, compared to the proof of the fixed-parameter intractability of copu-DBU by van de Pol *et al.* [20], is that we show how to create gadgets that encode the truth value of the different variables via worlds at different depths of the model rather than via different agents. This is necessary since we have a as a parameter, so we need to put a bound on the

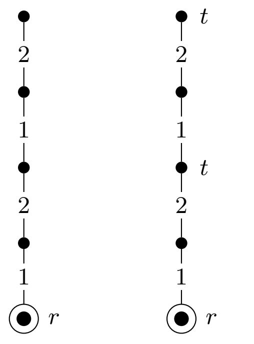

Fig. 12: Left: A valuation gadget for $m = 4$ representing the valuation 0 in which all x_i, $i = 1, \dots, m$, are false. Right: The gadget for the valuation where x_2 and x_4 are true (since the worlds in distance 2 and 4 from the root have label t).

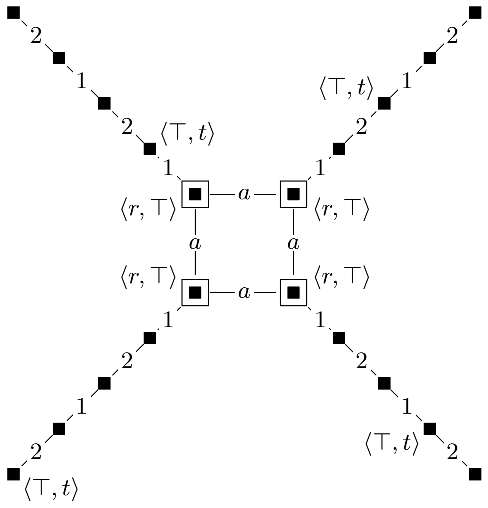

Fig. 13: The pointed event model $\mathcal{E}$ for $m = 4$. The unlabelled events are implicitly labelled $\langle \top, \top \rangle$.

number of agents. However, this is done at the expense of the modal depth of the goal formula, which is not restricted in our proof, and depends on the size of the k-W2SAT instance being reduced. When referring to worlds at different depths of a model, and with no bound on the depth of a model, we usually also need preconditions of unbounded length. But our construction shows that it is possible to still do with only preconditions of bounded length. In our proof, in order to encode a valuation, we use chains of worlds linked by alternating agents (as in the formalisation of the coordinated attack problem, and as in the encoding of the Turing machine tape in Theorem 1). This trick, central to our proof, resembles a trick used by de Haan and van de Pol [15]. The main difference is that they encode the truth-value of a single variable as a chain, and create as many chains as there are true propositional variables in the encoded valuation, whereas we encode an entire valuation in a single chain.

Essentially, the structure of this proof is as the previous, except we need a different type of gadgets. Let φ and k be given with $var(\varphi) = \{x_1, \dots, x_m\}$. Let $\mathcal{A} = \{1, 2, a\}$. For each valuation v, we define the gadget $\mathcal{M}_v$ as an alternating $1, 2$-chain of worlds with a root world satisfying r, and where the world at distance i from the root makes the propositional variable t true iff $v(x_i) = 1$. The construction is illustrated for $m = 4$ in Figure 12. Now consider the event model $\mathcal{E}$ illustrated for $m = 4$ in Figure 13. As in the previous proof, when this event model is applied to a gadget $\mathcal{M}_v$, it creates m copies of the gadget, where in the first gadget x_1 is made true (by adding t to the world at distance 1 from the root), in the second x_2 is made true (by adding t to the world at distance

2 from the root), etc. As before, these gadgets will be connected by a-edges via their root worlds. Also as before, when this event model is applied k times to the initial gadget model $\mathcal{M}_0$, we achieve a model with m^{k} gadgets containing at least one gadget for each valuation making at most k variables true (again except the valuation 0 that can be treated separately). The only essential difference is that instead of making use of agents to encode the truth value of the different variables, we use the depth of the event model. This means we can use a as a parameter in our reduction (the number of agents is fixed independently of the input).

Let $\psi_1 := \hat{K}_1 t$, $\psi_2 := \hat{K}_1 \hat{K}_2 t$, $\psi_3 := \hat{K}_1 \hat{K}_2 \hat{K}_1 t$, etc. Then note that φ is true in the valuation v iff the formula $\varphi[\psi_i/x_i]$ is true in the root of the gadget $\mathcal{M}_v$. Hence, to check whether φ is true in a valuation making at most k variables true, we can check whether the formula $\varphi_g := \hat{K}_a \varphi[\psi_i/x_i]$ is true in $\mathcal{M}_0 \otimes \mathcal{E}^{\mathsf{k}}$. To sum up, given a k-$\mathsf{W2SAT}$ instance φ with parameter k, we reduce it to the DBU instance with initial model $\mathcal{M}_0$, with k copies of the event model $\mathcal{E}$ and with goal formula φ_g. Building $\mathcal{M}_0$ and the k copies of $\mathcal{E}$ is polynomial in m and k, and building φ_g is polynomial in m and the length of φ. Hence the DBU instance can be computed in polynomial time in the size of the k-$\mathsf{W2SAT}$ instance, and is hence computable by an fpt-algorithm. We then only need to show that the parameters of the translated acpu-DBU instance can be bound by a computable function in k. This trivially holds, as the parameters a, c, p all have fixed value independent of the k-$\mathsf{W2SAT}$ instance, and u is k.

4 New tractability results

In this section we prove tractability of acopu-DBU (and hence also acfpu-DBU and acfopu-DBU). By Proposition 2, we can freely choose to consider single- or multi-pointed models. We here only consider single-pointed models, as it becomes technically simpler and allows us to rely on many existing results that only exist for single-pointed models. The overall idea of our proof is to reduce the size of models via stratified bisimulation contractions (k-bisimulation contractions). In the first subsection below, we define stratified bisimulations (k-bisimulations) and some well-known results concerning them, followed by a new result concerning the relation between k-bisimulations and product updates. In the next subsection we then consider partition refinements as a way to compute k-bisimulation contractions, and study the computational complexity of partition refinement algorithms. In the final subsection, all the previous results will be combined to prove our tractability result.

4.1 Stratified bisimulations and bisimulation contractions

We use the variant of the definition of stratified bisimulations provided by Yu et al. [23], but otherwise refer to Blackburn et al. [4] for a thorough introduction to bisimulations in modal logic (the latter contains a slightly different, but equivalent, definition of stratified bisimulations).

Definition 7. [23, Definition 4.9] *Let $\mathcal{M}$ be an epistemic model. Two worlds w and w' are called* 0-bisimilar*, written $w \leftrightarroweq_0 w'$, if $L(w) = L(w')$. For $k > 0$, they are* k-bisimilar*, written $w \leftrightarroweq_k w'$, if they are 0-bisimilar and for every agent $i \in \mathcal{A}$:*

- [forth$_k$] *If $v \sim_i w$ then there is a $v' \sim_i w'$ such that $v \leftrightarroweq_{k-1} v'$.*
- [back$_k$] *If $v' \sim_i w'$ then there is a $v \sim_i w$ such that $v \leftrightarroweq_{k-1} v'$.*

Two epistemic models $\mathcal{M} = (W, \sim, L, w)$ and $\mathcal{M}' = (W', \sim', L', w')$ are called k-bisimilar*, written $\mathcal{M} \leftrightarroweq_k \mathcal{M}'$, if w and w' are k-bisimilar in the disjoint union of the two models.*[5]

Note that k-bisimilarity implies n-bisimiliarity for all $n < k$. A crucial property of k-bisimulations is that they preserve modal equivalence up to depth k. More precisely, we have the following.

Proposition 3. [4, Proposition 2.31] *Two (single-pointed) epistemic models are k-bisimilar iff they agree on all formulas of modal depth at most k.*

Another crucial property of k-bisimulations is that we can reduce the size of a model by taking its k-bisimulation contraction, and doing so will give us a model that is k-bisimilar to the original model. The k-bisimulation contraction is achieved by identifying all k-bisimilar worlds of the model. Hence, for instance, the 0-bisimulation contraction of a model is the one in which we identify any pair of worlds that has the same label. More precisely, we define as follows.

Definition 8. [23, Definition 4.11] *The* k-bisimulation contraction *of an epistemic model $\mathcal{M}$, denoted $\lfloor \mathcal{M} \rfloor_k$, is the quotient structure of $\mathcal{M}$ with respect to the $\leftrightarroweq_k$-relation on its worlds.*

The following notational convention will later come in handy.

Definition 9. *Let $\mathcal{M}$ be an epistemic model and $\mathcal{E}$ an event model. We introduce $\mathcal{M} \otimes_k \mathcal{E}$ as shorthand for $\lfloor \mathcal{M} \otimes \mathcal{E} \rfloor_k$.*

Proposition 4. [23, Proposition 4.12] *For any model $\mathcal{M}$ and any $k \geq 0$, we have $\lfloor \mathcal{M} \rfloor_k \leftrightarroweq_k \mathcal{M}$.*

Combining Propositions 3 and 4, we can conclude that if we need to check the truth of a formula ϕ of modal depth k in an epistemic model $\mathcal{M}$, we can first compute $\lfloor \mathcal{M} \rfloor_k$, and then check whether ϕ holds there. Computing the k-bisimulation contraction of course takes time, so it is not obvious that this would be a better approach to the model checking of ϕ in $\mathcal{M}$, even if the contracted model is smaller. The real advantage appears if we can perform a k-bisimulation

[5] The disjoint union of the two models simply takes the union of the worlds of the two models (these world sets are supposed to be disjoint), the union of the indistinguishable relations, and preserves the labels of all worlds. To preserve single-pointedness, we need to choose a single designated world of the resulting model, but this can be chosen arbitrarily, as the construction of the disjoint union is only there to simplify the definition of the k-bisimulation between distinct models.

contraction after each of a series of product updates, as we can then prove that the models stay within a certain size (as opposed to e.g. the coordinated attack problem, where each product update produces a model that is one world larger than the previous). For instance, 0-bisimulation contraction always gives us a model with at most $2^{|P|}$ worlds, as any two worlds with the same label are identified. The main complication of this idea is that we need to figure out how to pick the right value of k. The following result by Yu et al. [23] shows that when the event models only have propositional preconditions (preconditions of modal depth 0), it is simple.

Proposition 5. [23, Proposition 4.14] *Let $\mathcal{M}$ and $\mathcal{M}'$ be two (single-pointed) epistemic models with $\mathcal{M} \leftrightarroweq_k \mathcal{M}'$. Let $\mathcal{E} = (E, \sim, \mathsf{pre}, \mathsf{post}, e)$ be a (single-pointed) event model with propositional preconditions (preconditions of modal depth 0), and suppose that $\mathcal{M} \models \mathsf{pre}(e)$ and $\mathcal{M}' \models \mathsf{pre}(e)$. Then $\mathcal{M} \otimes \mathcal{E} \leftrightarroweq_k \mathcal{M}' \otimes \mathcal{E}$.*

Suppose we are trying to solve a DBU instance with initial epistemic model $\mathcal{M}$, event models $\mathcal{E}_1, \ldots, \mathcal{E}_u$ that only have propositional preconditions, and goal formula ϕ_g. Let o denote the modal depth of ϕ_g. By Propositions 4, we have $\mathcal{M} \otimes \mathcal{E}_1 \leftrightarroweq_{\mathsf{o}} \mathcal{M} \otimes_{\mathsf{o}} \mathcal{E}_1$. Applying Proposition 5, we then get $(\mathcal{M} \otimes \mathcal{E}_1) \otimes \mathcal{E}_2 \leftrightarroweq_{\mathsf{o}} (\mathcal{M} \otimes_{\mathsf{o}} \mathcal{E}_1) \otimes \mathcal{E}_2$. Applying again Proposition 4, we get $(\mathcal{M} \otimes_{\mathsf{o}} \mathcal{E}_1) \otimes \mathcal{E}_2 \leftrightarroweq_{\mathsf{o}} (\mathcal{M} \otimes_{\mathsf{o}} \mathcal{E}_1) \otimes_{\mathsf{o}} \mathcal{E}_2$. Combining these, we get

$$\mathcal{M} \otimes \mathcal{E}_1 \otimes \mathcal{E}_2 \leftrightarroweq_{\mathsf{o}} \mathcal{M} \otimes_{\mathsf{o}} \mathcal{E}_1 \otimes_{\mathsf{o}} \mathcal{E}_2.$$

More generally, by repeated applications of Propositions 4 and 5, we get that

$$\mathcal{M} \otimes \mathcal{E}_1 \otimes \cdots \otimes \mathcal{E}_u \leftrightarroweq_{\mathsf{o}} \mathcal{M} \otimes_{\mathsf{o}} \mathcal{E}_1 \otimes_{\mathsf{o}} \cdots \otimes_{\mathsf{o}} \mathcal{E}_u$$

In other words, we can replace the original sequence of product updates (left) by a sequence where after each update, we take the o-bisimulation contraction (right). This gives us a resulting model that is o-bisimilar to the model that would have resulted from performing the original product update sequence. Hence the two resulting models assign the same truth-value to the goal formula ϕ_g (by Proposition 3). This gives us an alternative way to decide DBU instances: after each product update, we take the o-bisimulation contraction. This can potentially be a computational advantage, as the o-bisimulation contractions can prevent models from growing arbitrarily large when undergoing a sequence of updates.

Example 5. Consider again the coordinated attack problem. Both of the event models $\mathcal{E}_{i \to j}$ and $\mathcal{E}_{j \to i}$ only have propositional preconditions (more specifically, the precondition formulas are $d \wedge m_i$, $d \wedge m_j$ and $\top$). Hence, given any goal formula ϕ_g of modal depth o, we can check whether ϕ_g holds after the successful delivery of u messages by checking whether the following holds:

$$\mathcal{M} \otimes_{\mathsf{o}} \underbrace{\mathcal{E}_{i \to j} \otimes_{\mathsf{o}} \mathcal{E}_{j \to i} \otimes_{\mathsf{o}} \cdots}_{\mathsf{u} \text{ event models}} \models \phi_g.$$

In the coordinated attack problem, the size of an o-bisimulation contracted model is actually linear in o, as we will now show. Consider the $\mathcal{M}_n$ models of Figure 3. Consider any two worlds w_k and w_l in $\mathcal{M}_n$ where $\mathsf{o}+1 < k, l < n+2-\mathsf{o}$. Then from w_k and w_l we can by any path of length $\leq \mathsf{o}$ only reach worlds where d holds, so w_k and w_l will satisfy the same formulas of depth at most o, and hence by Proposition 3, $w_k \leftrightarroweq_{\mathsf{o}} w_l$. In other words, w_k and w_l will be identified in the o-bisimulation contraction. So any pair of worlds between the leftmost o worlds and the rightmost o of $\mathcal{M}_n$ will be identified, and the contracted model $\lfloor \mathcal{M}_n \rfloor_{\mathsf{o}}$ can hence at most contain $2\mathsf{o}+1$ worlds. This implies that when we need to check whether ϕ_g holds after a sequence of updates (message passings), we only need to consider models of a size that is linear in the model depth o of the goal formula, not linear in the number u of updates.

In order for the idea presented above to work in general, we still need a bit more. First of all, in DBU we can generally not assume that events have propositional preconditions. So we can not rely on Proposition 5 in its current form. We hence generalise it in the following way.

Proposition 6. *Suppose $\mathcal{M} \leftrightarroweq_k \mathcal{M}'$. Let $\mathcal{E} = (E, \sim, \mathsf{pre}, \mathsf{post}, e)$ be an event model with preconditions of modal depth at most n, where $n \leq k$, and suppose that $\mathcal{M} \models \mathsf{pre}(e)$ and $\mathcal{M}' \models \mathsf{pre}(e)$. Then $\mathcal{M} \otimes \mathcal{E} \leftrightarroweq_{k-n} \mathcal{M}' \otimes \mathcal{E}$.*

Proof. The proof is by induction on the value of $k-n$. The base case is $k-n=0$. So suppose $\mathcal{M} = (W, \sim, L, w)$, $\mathcal{M}' = (W', \sim', L', w')$ and $\mathcal{E} = (E, \sim, \mathsf{pre}, \mathsf{post}, e)$ are given, satisfying the assumptions of the proposition, where $k-n=0$. We need to prove that $\mathcal{M} \otimes \mathcal{E}$ and $\mathcal{M}' \otimes \mathcal{E}$ are 0-bisimilar. As we have assumed $\mathcal{M} \leftrightarroweq_k \mathcal{M}'$, we immediately get that $\mathcal{M} \leftrightarroweq_0 \mathcal{M}'$. This means $w \leftrightarroweq_0 w'$, in other words that $L(w) = L(w')$. Hence, clearly, we must also have $L((w,e)) = L((w',e))$, as (w,e) and (w',e) are two worlds with identical labels that have been updated with the same event. Hence $(w,e) \leftrightarroweq_0 (w',e)$, i.e., $\mathcal{M} \otimes \mathcal{E} \leftrightarroweq_0 \mathcal{M}' \otimes \mathcal{E}$, as required. This covers the base case.

For the induction step, suppose the conclusion holds for $k-n=m$ and consider the case of $k-n=m+1$. Since $m+1>0$, we must have $k-n>0$ and hence $k-n-1 \geq 0$. Suppose $\mathcal{M} = (W, \sim, L, w)$, $\mathcal{M}' = (W', \sim', L', w')$ and $\mathcal{E} = (E, \sim, \mathsf{pre}, \mathsf{post}, e)$ are given, satisfying the assumptions of the proposition, where $k-n=m+1$. Suppose further that $\mathcal{M} \otimes \mathcal{E} = (\bar{W}, \bar{\sim}, \bar{L}, (w,e))$ and $\mathcal{M}' \otimes \mathcal{E} = (\bar{W}', \bar{\sim}', \bar{L}', (w',e))$. We need to prove $\mathcal{M} \otimes \mathcal{E} \leftrightarroweq_{k-n} \mathcal{M}' \otimes \mathcal{E}$, i.e., $(w,e) \leftrightarroweq_{k-n} (w',e)$. That $(w,e) \leftrightarroweq_0 (w',e)$ holds is proved exactly as in the base case. So we only need to prove [forth$_{k-n}$] and [back$_{k-n}$] of Definition 7 for the pair of worlds (w,e) and (w',e). [back$_{k-n}$] and [forth$_{k-n}$] are symmetric, so it suffices to prove [forth$_{k-n}$]. To prove [forth$_{k-n}$], suppose $(v,f) \mathrel{\bar{\sim}_i} (w,e)$. We then need to find a $(v',f') \mathrel{\bar{\sim}'_i} (w',e)$ such that $(v,f) \leftrightarroweq_{k-n-1} (v',f')$. Since $(v,f) \mathrel{\bar{\sim}_i} (w,e)$, we get $v \sim_i w$ and $f \sim_i e$, by the definition of product update. Since we have assumed $\mathcal{M} \leftrightarroweq_k \mathcal{M}'$, we must have $w \leftrightarroweq_k w'$. Hence, by [forth$_k$] of the bisimilarity between w and w', there exists $v' \sim'_i w'$ such that $v \leftrightarroweq_{k-1} v'$. Let $\mathcal{N} = (W, \sim, L, v)$, $\mathcal{N}' = (W', \sim', L', v')$ and $\mathcal{F} = (E, \sim, \mathsf{pre}, \mathsf{post}, f)$. Then $\mathcal{N} \leftrightarroweq_{k-1} \mathcal{N}'$. Since $v \leftrightarroweq_{k-1} v'$, the two worlds v and v' satisfy the same formulas up to modal depth $k-1$, by Proposition 3. Since $k-n-1 \geq 0$, we have $k-1 \geq n$. The two worlds v and v' must hence also satisfy the same formulas up to modal depth n, in particular we get $\mathcal{N} \models \mathsf{pre}(f)$ iff $\mathcal{N}' \models \mathsf{pre}(f)$. Since $(v, f) \bar{\sim}_i (w, e)$, necessarily we must have $\mathcal{N} \models \mathsf{pre}(f)$. Hence also $\mathcal{N}' \models \mathsf{pre}(f)$. We therefore have $\mathcal{N}' \otimes \mathcal{F} = (\bar{W}', \bar{\sim}', \bar{L}', (v', f))$. We now have that $\mathcal{N} \leftrightarroweq_{k-1} \mathcal{N}'$, that $\mathcal{F}$ has preconditions of modal depth at most n, and that $\mathcal{N} \models \mathsf{pre}(f)$ and $\mathcal{N}' \models \mathsf{pre}(f)$. Since $k-1-n = (k-n)-1 = (m+1)-1 = m$, we can apply the induction hypothesis to conclude that $\mathcal{N} \otimes \mathcal{F} \leftrightarroweq_{k-1-n} \mathcal{N}' \otimes \mathcal{F}$. Hence $(v, f) \leftrightarroweq_{k-1-n} (v', f)$. Since $v' \sim_i' w'$, $f \sim_i e$, $\mathcal{N}' \models \mathsf{pre}(f)$ and $\mathcal{N}' \models \mathsf{pre}(e)$, we get $(v', f) \bar{\sim}_i' (w', e)$. Letting $f' = f$, we have then found the required $(v', f') \bar{\sim}_i' (w', e)$ such that $(v, f) \leftrightarroweq_{k-n-1} (v', f')$. This completes the proof.

This result above is new. It is perhaps not a surprising result, but it might still be of independent interest outside of the study of DBU. In particular, the original version, Proposition 5, was the main ingredient in proving decidability of the plan existence problem in epistemic planning with propositional preconditions [23]. Our generalised result could potentially be applied to the study of the complexity of the corresponding bounded plan existence problem.

The result above can be applied as follows. Consider a DBU instance where o as usual denotes the modal depth of the goal formula, c denotes the maximal length of its event preconditions and u denotes the number of event models. Note that c is also an upper bound on the modal depth of the event preconditions. We can now reason as follows, recalling that k-bisimilarity implies n-bisimilarity for all $n < k$:

$$\begin{array}{ll}
\quad \mathcal{M} \otimes \mathcal{E}_1 \ \leftrightarroweq_{\mathsf{o}+(\mathsf{u}-1)\mathsf{c}} \ \mathcal{M} \otimes_{\mathsf{o}+\mathsf{uc}} \mathcal{E}_1 & \text{by Proposition 4} \\
\Downarrow & \\
\quad \mathcal{M} \otimes \mathcal{E}_1 \otimes \mathcal{E}_2 \ \leftrightarroweq_{\mathsf{o}+(\mathsf{u}-2)\mathsf{c}} \ \mathcal{M} \otimes_{\mathsf{o}+\mathsf{uc}} \mathcal{E}_1 \otimes \mathcal{E}_2 & \text{by Proposition 6} \\
\Downarrow & \\
\quad \mathcal{M} \otimes \mathcal{E}_1 \otimes \mathcal{E}_2 \ \leftrightarroweq_{\mathsf{o}+(\mathsf{u}-2)\mathsf{c}} \ \mathcal{M} \otimes_{\mathsf{o}+\mathsf{uc}} \mathcal{E}_1 \otimes_{\mathsf{o}+\mathsf{uc}} \mathcal{E}_2 & \text{by Proposition 4} \\
\Downarrow & \\
\quad \mathcal{M} \otimes \mathcal{E}_1 \otimes \mathcal{E}_2 \otimes \mathcal{E}_3 \ \leftrightarroweq_{\mathsf{o}+(\mathsf{u}-3)\mathsf{c}} \ \mathcal{M} \otimes_{\mathsf{o}+\mathsf{uc}} \mathcal{E}_1 \otimes_{\mathsf{o}+\mathsf{uc}} \mathcal{E}_2 \otimes \mathcal{E}_3 & \text{by Proposition 6} \\
\Downarrow & \\
\quad \dots & \\
\Downarrow & \\
\quad \mathcal{M} \otimes \mathcal{E}_1 \otimes \cdots \otimes \mathcal{E}_u \ \leftrightarroweq_{\mathsf{o}} \ \mathcal{M} \otimes_{\mathsf{o}+\mathsf{uc}} \mathcal{E}_1 \otimes_{\mathsf{o}+\mathsf{uc}} \cdots \otimes_{\mathsf{o}+\mathsf{uc}} \mathcal{E}_3 &
\end{array}$$

Hence, since we are checking a goal formula of modal depth o, we can replace the original sequence of updates (left) by a sequence where after each update, we take the $(\mathsf{o} + \mathsf{uc})$-bisimulation contraction (right). What is left is only to prove that taking the $(\mathsf{o} + \mathsf{uc})$-bisimulation contraction after each update is sufficient for tractability when the parameters include a, c, o, p and u.

4.2 Partition refinements

Bisimulation contractions can be computed using a partition refinement algorithm [1]. We will here slightly generalise this to computing k-bisimulation contractions. A *partition* of a set W is a set $P = \{W_1, \dots, W_n\}$ with $W_1 \cup \cdots \cup W_n =$

W and all W_i being non-empty and pairwise disjoint. The elements W_i of the partition are called *blocks*. To *split a block* W_i in a partition $P = \{W_1, \dots, W_n\}$ means to replace the block W_i by a set of sub-blocks $W_i^1, \dots, W_i^m$ such that $\{W_i^1, \dots, W_i^m\}$ forms a partition of W_i. The result of the split is hence the new partition $P' = \{W_1, \dots, W_{i-1}, W_i^1, \dots, W_i^m, W_{i+1}, \dots, W_n\}$. A refinement step consists in splitting one or more blocks of a partition.

Our partitions and refinements will be on the set of worlds W of epistemic models $\mathcal{M} = (W, \sim, L, w)$. For any subset $W' \subseteq W$ and agent $a \in \mathcal{A}$, we are going to use the notation $w \sim_a W'$ to denote the existence of a $v \in W'$ with $w \sim_a v$. When W' is a block of a partition, $w \sim_a W'$ simply means that agent a has an (indistinguishability) edge to the block W'.

Partition refinement algorithms are algorithms that start with an initial partition that is then iteratively refined through a series of refinement steps (usually until a fixed-point is reached). We here sketch one of the standard partition refinement algorithms, but refer to Paige and Tarjan [19] or Aceto et al. [1] for a more thorough introduction to partition refinements. For all $k > 0$, we build partition P_k from P_{k-1} by the following *refinement step*:

> *Refinement step.* Each block W_i of P_{k-1} is split into sub-blocks $W_i^1, \dots, W_i^m$ defined as follows. Any pair of worlds $w, v \in W_i$ belong to the same sub-block W_i^j iff for every agent $a \in \mathcal{A}$ and every block W_b of P_{k-1}, $w \sim_a W_b$ iff $v \sim_a W_b$.

Example 6. Let us illustrate partition refinement on the models $\mathcal{M}_n$ of the coordinated attack problem (Figure 3). Consider the particular case of $\mathcal{M}_4$. There are three different labels in $\mathcal{M}_4$, the label $\{d, m_j\}$ of w_1, the label $\{d\}$ of the worlds w_2–w_5, and the empty label of w_6. Hence, the initial partition P_0 of $\mathcal{M}_3$ will contain three blocks $W_1 = \{w_1\}$, $W_2 = \{w_2, w_3, w_4, w_5\}$ and $W_3 = \{w_6\}$. Now consider what happens when we apply the refinement step to $P_0 = \{W_1, W_2, W_3\}$. Nothing can happen to W_1 and W_3 as these are already singletons. So consider W_2. We need to split worlds of W_2 into separate sub-blocks if they differ on which agents have edges to which blocks of P_0. Since w_2 has an i-edge to block W_1 and w_3 doesn't, w_2 and w_3 will end in distinct sub-blocks. A similar argument gives that w_4 and w_5 will end in distinct sub-blocks, since $w_5 \sim_j W_3$, but $w_4 \not\sim_j W_3$. We hence only need to check whether w_3 and w_4 will also end in separate sub-blocks. For $k = i, j$, we have $w_3 \sim_k W_l$ iff $l = 2$. The same holds with w_3 replaced by w_4. Hence w_3 and w_4 will still end in the same sub-block. Hence the refinement step will result in the partition $P_1 = \{W_1, W_1', W_2', W_3', W_3\}$ where $W_1' = \{w_2\}$, $W_2' = \{w_3, w_4\}$ and $W_3' = \{w_5\}$.

The following result is according to our knowledge also new, however it is probably at least implicitly known by experts on bisimulation and partition refinement, even if not explicitly spelled out in the existing literature.

Proposition 7. *Let $\mathcal{M}$ be an epistemic model. After performing k refinement steps on the initial partition of $\mathcal{M}$, two worlds are in the same block iff they are k-bisimilar.*

Proof. Suppose $\mathcal{M} = (W, \sim, L, v)$. For each k, we let P_k denote the kth partition of W, that is, the result of performing k refinement steps on the initial partition P_0. We need to prove that two worlds are in the same block of P_k iff they are k-bisimilar. The proof is by induction on k. The base case is $k = 0$. By definition, we have that two worlds are in the same block of P_0 iff they are 0-bisimilar. This covers the base case.

For the induction step, suppose the claim holds for k and consider $k + 1$. Suppose $P_{k+1} = \{W_1, \dots, W_n\}$. Let $w, w' \in W$ be chosen arbitrarily. We need to prove that $w \leftrightarroweq_{k+1} w'$ iff $w, w' \in W_i$ for some $i \in \{1, \dots, n\}$. First we prove the "if" direction. So suppose $w, w' \in W_i$. Since the initial partition only puts worlds in the same block if they have the same label, and since each refinement step only splits existing blocks into sub-blocks, also w and w' will necessarily have the same label. This proves $w \leftrightarroweq_0 w'$. We then only have to prove [forth$_{k+1}$] and [back$_{k+1}$]. We only prove [forth$_{k+1}$], the other one being symmetric. So suppose $v \sim_a w$ for some agent $a \in \mathcal{A}$. We then need to find a $v' \sim_a w'$ such that $v \leftrightarroweq_k v'$. Since P_{k+1} is the result of applying a refinement step to P_k, by definition of the refinement step we have that for all blocks W_b of P_k, $w \sim_a W_b$ iff $w' \sim_a W_b$. Let W_j denote the block of P_k containing v. Then $w \sim_a W_j$, and hence $w' \sim_a W_j$. From $w' \sim_a W_j$ we get the existence of a $v' \in W_j$ satisfying $w' \sim_a v'$. We now only need to prove $v \leftrightarroweq_k v'$. However, this trivially follows from the induction hypothesis, as P_k is the partition after k refinement steps, and v and v' both belong to the same block W_j of that partition.

We now prove the "only if" direction of the induction step. So suppose $w \leftrightarroweq_{k+1} w'$. We need to show that w and w' belong to the same block of P_{k+1}. Since $w \leftrightarroweq_{k+1} w'$ implies $w \leftrightarroweq_k w'$, the induction hypothesis immediately implies that they belong to the same block of P_k. So we only need to show that w and w' are not split into separate blocks when performing the refinement step on P_k. This amounts to proving that for every agent $a \in \mathcal{A}$ and every block W_b of P_k, we have $w \sim_a W_b$ iff $w' \sim_a W_b$. So let a and b be chosen, we then need to prove $w \sim_a W_b$ iff $w' \sim_a W_b$. We only prove the "only if" direction, the other being symmetric. So suppose $w \sim_a W_b$. Then there exists a $v \in W_b$ such that $w \sim_a v$. Since $w \leftrightarroweq_{k+1} w'$, by [forth$_{k+1}$] we get the existence of a $v' \sim_a w'$ such that $v \leftrightarroweq_k v'$. By induction hypothesis again, we get that v and v' belong to the same block of P_k, in other words, both v and v' belong to W_b. Since $w' \sim_a v'$, we get $w' \sim_a W_b$, as required.

It follows from Proposition 7 that the quotient model made from the blocks after having performed k partition refinement steps is identical to the k-bisimulation contraction of the model. The following two results, of which the first is from the existing literature, address the complexity of k-bisimulation contractions.

Proposition 8. [1, Theorem 0.2.6] *The k-bisimulation contraction of $\mathcal{M}$ can be computed in quadratic time in the size of $\mathcal{M}$.*

Proposition 9. *The k-bisimulation contraction of $\mathcal{M}$ has a size bounded by a computable function $h(k, \mathsf{a}, \mathsf{p})$, where a is the number of agents in $\mathcal{M}$ and p is the number of propositional variables.*

Proof. In the following, we will refer to the number of blocks in a partition as the *size* of the partition (not to be confused with its size in terms of the memory required to store it). For all $k \geq 0$, we let P_k denote the kth partition of $\mathcal{M}$, that is, P_0 is the initial partition of $\mathcal{M}$, and for all $k > 0$, P_k is the result of performing a single refinement step on P_{k-1}.

If we can show that the number of worlds of $\lfloor \mathcal{M} \rfloor_k$ is bounded by a function in k, a and p, so will the size of the model itself: the number of edges is bounded by $n^2\mathsf{a}$, where n is the number of worlds of the model; and the length of the world labels is bounded by p. The number of worlds of $\lfloor \mathcal{M} \rfloor_k$ is by Proposition 7 equal to the size of P_k. It hence suffices to find a function $g(k, \mathsf{a}, \mathsf{p})$ that provides an upper bound on the size of P_k. We define g by:

$$\begin{aligned} g(0, \mathsf{a}, \mathsf{p}) &= 2^{\mathsf{p}} \\ g(k+1, \mathsf{a}, \mathsf{p}) &= 2^{\mathsf{a}g(k,\mathsf{a},\mathsf{p})} g(k, \mathsf{a}, \mathsf{p}) \end{aligned}$$

We then only need to prove that P_k has a size bounded by $g(k, \mathsf{a}, \mathsf{p})$. The proof is by induction on k.

The base case is $k = 0$. We here need to show that P_0 has size at most $g(0, \mathsf{a}, \mathsf{p}) = 2^{\mathsf{p}}$. However, this is trivial, as the initial partition consists of a single block for each label of $\mathcal{M}$, and the number of labels is bound by 2^{p}. For the induction step, suppose the claim holds for k and consider $k+1$. By definition, P_{k+1} is the result of performing a refinement step on $P_k = \{W_1, \ldots, W_n\}$. By induction hypothesis, we have $n \leq g(k, \mathsf{a}, \mathsf{p})$. Let the *signature* of a world w (wrt. P_k) be the set $\{(a, j) \in \mathcal{A} \times \mathbb{N} \mid w \sim_a W_j\}$. Now note that two worlds in the same block of P_k are split into distinct sub-blocks by the refinement step iff they have distinct signatures. Hence, the number of sub-blocks created by a split of a block is bounded by the number of signatures, which in turn is bounded by $2^{\mathsf{a}n}$. Since we are starting out with n blocks, and each block is split into at most $2^{\mathsf{a}n}$ sub-blocks, the size of P_{k+1} is bounded by $2^{\mathsf{a}n} n \leq 2^{\mathsf{a}g(k,\mathsf{a},\mathsf{p})} g(k, \mathsf{a}, \mathsf{p}) = g(k+1, \mathsf{a}, \mathsf{p})$, as required.

4.3 The tractability result

With all the above results on k-bisimulations and partition refinements in place, we are finally ready to prove our main positive result, fixed-parameter tractability of acopu-DBU.

Theorem 4. *acopu-DBU is fixed-parameter tractable. In other words, the Dynamic Belief Update problem becomes tractable when restricting the number of propositional variables and agents (p,a), the number of event models (u), the maximum length of event preconditions (c), and the modal depth of the goal formula (o).*

Proof. To prove that acopu-DBU is fixed-parameter tractable, we need to find an fpt-algorithm that solves it. The algorithm is the one we have already sketched above: After each product update, we take the $(\mathsf{o} + \mathsf{uc})$-bisimulation contraction of the resulting model. As already shown, the goal formula will hold in this contracted update sequence iff it holds in the original update sequence. What is left is then only to show that this algorithm is fpt. To show this, we need to find a computable function $g : \mathbb{N}^5 \to \mathbb{N}$ and a polynomial $\mathcal{P}$ such that the running time of the algorithm for an input where the main part has length n is at most $g(\mathsf{a}, \mathsf{c}, \mathsf{o}, \mathsf{p}, \mathsf{u}) \cdot \mathcal{P}(n)$ (we choose to keep the different parameters separate, to make the role of each independent parameter more clear).

Let us compute an upper bound on the time it takes to compute the product update of an epistemic model with at most m worlds with one of the event models in the input. The number of events in the event models is limited by n (the input size). So to compute the product update, we need to create at most mn world-event pairs, and for each of these we need to check a precondition of size at most c. Model checking is polynomial in the sum of the size of the model and the formula to be checked [21], so checking each precondition is polynomial in $m + \mathsf{c}$. For each world-event pair that survives (where the precondition of the event is satisfied in the world), we need to enforce the postcondition in the resulting world. This can be done in time p. So handling each world-event pair takes time at most $\mathsf{p}(m + \mathsf{c})^{O(1)}$, and since there are at most mn such world-event pairs, the total time to compute the worlds of the updated model (and their labels) is at most $mn\mathsf{p}(m + \mathsf{c})^{O(1)}$. In addition to computing the worlds of the updated model, we need to compute the indistinguishability relations. We can do this by iterating through each of the at most $\mathsf{a}m^2$ indistinguishability edges of the epistemic model and the at most $\mathsf{a}n^2$ indistinguishability edges of the event model. For each such choice of two indistinguishability edges $w \sim_i v$ and $e \sim_j f$, we add an indistinguishability edge $(w, e) \sim_i (v, f)$ to the updated model if $i = j$ and both (w, e) and (v, f) exist in the model. This can hence be done in time at most $\mathsf{a}^2 n^2 m^2$. So the total time it takes to compute the product update is at most $mn\mathsf{p}(m + \mathsf{c})^{O(1)} + \mathsf{a}^2 n^2 m^2$. The conclusion is that computing the product update is polynomial in m, n, c, a and p, and the size of the resulting model is polynomial in the same parameters.

After each product update, we need to compute the $(\mathsf{o} + \mathsf{uc})$-bisimulation contraction of the updated model. According to Proposition 8, this can be done in quadratic time in the size of the model. Hence, the time it takes to compute the product update and the following contraction is still polynomial in m, n, c, a and p. We now take a closer look at the value of m. In the initial product update, m is bounded by n. So computing the first contracted update, $\lfloor \mathcal{M} \otimes \mathcal{E}_1 \rfloor_{\mathsf{o}+\mathsf{uc}}$, is polynomial in n, c, a and p. In all the following product updates, m is bounded by the size of the $(\mathsf{o} + \mathsf{uc})$-bisimulation contracted model. According to Proposition 9, the $(\mathsf{o} + \mathsf{uc})$-bisimulation contraction of a model has a size bounded by $h(\mathsf{o} + \mathsf{uc}, \mathsf{a}, \mathsf{p})$ for some computable function h.

We need to perform u contracted updates in total. As each of these are polynomial in $h(\mathsf{o} + \mathsf{uc}, \mathsf{a}, \mathsf{p})$, n, c, a and p, the total sequence of contracted updates can be done in polynomial time in $h(\mathsf{o} + \mathsf{uc}, \mathsf{a}, \mathsf{p})$, n, c, a, p and u. The final step is model checking of a formula of size at most n in a model of size at most $h(\mathsf{o} + \mathsf{uc}, \mathsf{a}, \mathsf{p})$. This is polynomial in n and $h(\mathsf{o} + \mathsf{uc}, \mathsf{a}, \mathsf{p})$. Hence, the entire algorithm runs in polynomial time in $h(\mathsf{o} + \mathsf{uc}, \mathsf{a}, \mathsf{p})$, n, and the parameters a, c,

p, and u. In other words, it runs in time

$$h(\mathsf{o} + \mathsf{uc}, \mathsf{a}, \mathsf{p})^{O(1)} \mathsf{a}^{O(1)} \mathsf{c}^{O(1)} \mathsf{p}^{O(1)} \mathsf{u}^{O(1)} n^{O(1)}.$$

It immediately follows that we can find a computable function $g : \mathbb{N}^5 \to \mathbb{N}$ and a polynomial $\mathcal{P}$ such that the running time is at most $g(\mathsf{a}, \mathsf{c}, \mathsf{o}, \mathsf{p}, \mathsf{u}) \cdot \mathcal{P}(n)$. This completes the proof.

Note the role played by the different parameters in the proof above. All 5 parameters are included in the expression $h(\mathsf{o} + \mathsf{uc}, \mathsf{a}, \mathsf{p})$ providing a bound on the size of the contracted updates. The parameter p is required to make sure that the initial partition doesn't become too big, and a is required to make sure that a block can't be split into too many sub-blocks. The parameters o, u and c of the first argument to h are required to give an upper bound on the k for which we need to do k-bisimulation contractions. It is crucial that this upper bound doesn't depend on n, as h is not polynomial in its first argument. So this account for the role of all 5 parameters. And, indeed, we already formally proved that all 5 of them are needed to ensure tractability, as can be seen from Figure 5.

The tractability result above is proven for a restricted version of DEL where we consider all agent relations to be equivalence relations. However, note that we never use the assumption that the relations are equivalence relations, so the theorem above immediately generalises to arbitrary frame classes.

5 Discussion and future work

We managed to solve all of the remaining 14 open tractability problems for the dynamic belief update problem. In most cases, our results were negative, *i.e.*, we proved fixed-parameter intractability. Out of the original $2^7 = 128$ problems, $128 - 14 = 114$ were already solved, but we actually ended up providing new proofs of 84 of them, if we count all our results and corollaries.

The *short Turing machine acceptance problem* (STMA) is the acceptance problem of single-tape nondeterministic Turing machines with bound k on the number of computation steps. It is a parameterized problem with parameter k known to be W[1]-complete, *i.e.*, fixed-parameter intractable [8]. The proof of Theorem 1 gives us a construction allowing us to encode an instance of STMA as a DBU instance. Since the parameter k is the number of computation steps, which translates into the parameter u in the DBU instance, we can do an fpt-reduction from STMA to acfou-DBU, *i.e.*, we can replace e, p by u in the fixed-parameter intractability result of Theorem 1. We have to drop the parameters e and p as their sizes depend on the alphabet of the Turing machine. This reduction then immediately gives W[1]-hardness of acfou-DBU. This result was already established by van de Pol *et al.* [20], but with our Turing machine construction in Theorem 1, we get this additional result essentially for free.

Our tractability result of Section 4 ties back to the opening discussion of this paper. We wrote that "a conjecture is that higher-order thinking is the most prominent source of intractability, and situations requiring modest depth of reasoning are the easiest to untangle." It is often not clear what depth of reasoning is required in a given situation, e.g. what depth of reasoning is required to solve a given DBU instance. However, our results of Section 4 shed some light on it. We showed that the depth of reasoning required is never higher than $\mathsf{o}+\mathsf{uc}$, since we can always iteratively replace each model by its $(\mathsf{o}+\mathsf{uc})$-contraction and still arrive at the same result (and the $(\mathsf{o}+\mathsf{uc})$-contraction is only guaranteed to preserve modal equivalence up to modal depth $\mathsf{o}+\mathsf{uc}$). So if we use all of o, u and c as fixed parameters, we are guaranteed to have fixed bound on the required depth of reasoning. It then turns out that this in itself is not sufficient to guarantee tractability, as we can still exploit the complexity coming from a great amount of agents (as in the proof of Theorem 2) or many propositions (as in the fixed-parameter intractability proof for acfou-DBU by van de Pol *et al.* [20]). However, if we bound all of these parameters, we get tractability.

We could also turn things around and define a depth-limited version of DBU where we put a bound on the depth of reasoning that the algorithm is allowed to make use of. Let us use $\mathsf{DBU}(d)$ to denote the depth-limited version of DBU with depth parameter d. There are multiple possible ways to formally define $\mathsf{DBU}(d)$, but one would be to preserve the same input as for DBU and then require the output to be yes iff $\mathcal{M} \otimes_d \mathcal{E}_1 \otimes_d \cdots \otimes_d \mathcal{E}_u \models \phi_g$. The complication is of course that DBU and $\mathsf{DBU}(d)$ are not in general guaranteed to give the same answer for a given input, except if d satisfies certain restrictions, like $d \geq \mathsf{o}+\mathsf{uc}$ that according to the results of Section 4 will suffice. It might seem to be a bug if $\mathsf{DBU}(d)$ gives a different answer than DBU, since the real problem we are trying to solve is clearly DBU. However, it shouldn't necessarily be considered a bug. For instance, an agent with no Theory of Mind will normally answer a first-order false-belief task incorrectly, since it requires a first-order Theory of Mind to do the required reasoning (it requires depth 1 reasoning). Similarly, an agent with a first-order Theory of Mind can not be expected to respond correctly to a second-order false-belief task (it requires depth 2 reasoning). In other words, if we put a limit on the depth of reasoning of a given agent or algorithm, then of course if a certain problem requires reasoning of higher orders than that, the agent or algorithm might come up with a wrong answer to the problem.

Let us consider the complexity of $\mathsf{DBU}(d)$. First of all, it is *not* tractable. Had it been tractable, then also cou-DBU would be tractable, as when $d \geq \mathsf{o}+\mathsf{uc}$, the two problems DBU and $\mathsf{DBU}(d)$ coincide. The question is then, which additional parameters do we need to make $\mathsf{DBU}(d)$ tractable? Consider the proof of Theorem 4. If we replace the expression $\mathsf{o}+\mathsf{uc}$ by d, it actually provides us with a proof that we can check whether $\mathcal{M} \otimes_d \mathcal{E}_1 \otimes_d \cdots \otimes_d \mathcal{E}_u \models \phi_g$ holds in time $h(d, \mathsf{a}, \mathsf{p})^{O(1)} \mathsf{a}^{O(1)} \mathsf{c}^{O(1)} \mathsf{p}^{O(1)} \mathsf{u}^{O(1)} n^{O(1)}$, where n is the size of the main part of the input. In other words, this is the time it takes to check an instance of $\mathsf{DBU}(d)$. It hence follows that ap-$\mathsf{DBU}(d)$ is fixed-parameter tractable. If we are looking for the source of complexity in the ability of humans to attribute mental states to ourselves and others, it might be reasonable to assume a fixed upper bound on the number of agent (a) and number of propositions (p), since we are usually reasoning about a fixed environment (the world of agents and facts around us). And then, indeed, if we fix those, the depth-limited dynamic belief update problem becomes tractable. In this sense, there is in fact some evidence that the actual source of complexity in epistemic reasoning is the depth of the required reasoning, and that as long as we put a limit on that depth, epistemic reasoning is tractable. This informal discussion of course needs to be turned into a more thorough and formal investigation, including a proper investigation of the most correct way to define the depth-limited DBU problem. We leave this for future work.

Other ideas for future studies include considering the plan *synthesis* problem instead of the plan *verification* problem, and for instance then to include variants of the additional parameters introduced for the classical planning problem by Kronegger et al. [18].

Acknowledgments

We would like to thank the anonymous reviewers of both this paper and the earlier version presented at DaLí 2020 for many helpful comments that allowed us to improve the paper.

References

1. Aceto, L., Ingólfsdóttir, A., Srba, J.: The algorithmics of bisimilarity. In: Advanced Topics in Bisimulation and Coinduction, Cambridge tracts in theoretical computer science, vol. 52, pp. 100–172. Cambridge University Press, Cambridge, England (2012). https://doi.org/10.1017/CBO9780511792588.004
2. Baltag, A., Smets, S.: A qualitative theory of dynamic interactive belief revision. In: Bonanno, G., van der Hoek, W., Wooldridge, M. (eds.) Logic and the Foundations of Game and Decision Theory (LOFT7). Texts in Logic and Games, vol. 3, pp. 13–60. Amsterdam University Press (2008)
3. Baral, C., Bolander, T., van Ditmarsch, H., McIlrath, S.: Epistemic planning (dagstuhl seminar 17231). In: Dagstuhl Reports. vol. 7. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik (2017)
4. Blackburn, P., de Rijke, M., Venema, Y.: Modal Logic, Cambridge Tracts in Theoretical Computer Science, vol. 53. Cambridge University Press, Cambridge, UK (2001). https://doi.org/10.1017/CBO9781107050884
5. Bolander, T.: Seeing is believing: Formalising false-belief tasks in dynamic epistemic logic. In: Jaakko Hintikka on Knowledge and Game-Theoretical Semantics, pp. 207–236. Springer (2018)
6. Bolander, T., Andersen, M.: Epistemic planning for single- and multi-agent systems. Journal of Applied Non-classical Logics - JANCL **21**, 9–34 (01 2011). https://doi.org/10.3166/jancl.21.9-34
7. Bolander, T., Charrier, T., Pinchinat, S., Schwarzentruber, F.: DEL-based epistemic planning: Decidability and complexity. Artificial Intelligence (2020, to appear). https://doi.org/https://doi.org/10.1016/j.artint.2020.103304, `http://www.sciencedirect.com/science/article/pii/S0004370219301146`
8. Cesati, M.: The turing way to parameterized complexity. J. Comput. Syst. Sci. **67**, 654–685 (12 2003). https://doi.org/10.1016/S0022-0000(03)00073-4

9. Cook, S.A.: The complexity of theorem-proving procedures. In: Proceedings of the Third Annual ACM Symposium on Theory of Computing. p. 151–158. STOC '71, Association for Computing Machinery, New York, NY, USA (1971). https://doi.org/10.1145/800157.805047, `https://doi.org/10.1145/800157.805047`
10. van Ditmarsch, H., Kooi, B.: Semantic results for ontic and epistemic change. In: Bonanno, G., van der Hoek, W., Wooldridge, M. (eds.) Logic and the Foundation of Game and Decision Theory (LOFT 7). pp. 87–117. Texts in Logic and Games 3, Amsterdam University Press (2008)
11. Ditmarsch, H.v., van der Hoek, W., Kooi, B.: Dynamic Epistemic Logic. Springer Publishing Company, Incorporated, 1st edn. (2007)
12. Downey, R.G., Fellows, M.R.: Fundamentals of Parameterized Complexity. Springer Publishing Company, Incorporated (2013)
13. Fagin, R., Moses, Y., Halpern, J.Y., Vardi, M.Y.: Reasoning about knowledge. MIT press (2003)
14. Flum, J., Grohe, M.: Parameterized Complexity Theory (Texts in Theoretical Computer Science. An EATCS Series). Springer-Verlag, Berlin, Heidelberg (2006)
15. de Haan, R., van de Pol, I.: On the computational complexity of model checking for dynamic epistemic logic with S5 models. Journal of Applied Logics **8**(3), 621–658 (April 2021)
16. Haselager, W.: Cognitive science and folk psychology: The right frame of mind. Sage (1997)
17. Hopcroft, J.E., Motwani, R., Ullman, J.D.: Introduction to Automata Theory, Languages, and Computation (3rd Edition). Addison-Wesley Longman Publishing Co., Inc., USA (2006)
18. Kronegger, M., Pfandler, A., Pichler, R.: Parameterized complexity of optimal planning: A detailed map. In: Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence. pp. 954–961. IJCAI '13, AAAI Press (2013)
19. Paige, R., Tarjan, R.E.: Three partition refinement algorithms. SIAM Journal on Computing **16**(6), 973–989 (1987). https://doi.org/10.1137/0216062
20. van de Pol, I., van Rooij, I., Szymanik, J.: Parameterized complexity of theory of mind reasoning in dynamic epistemic logic. J of Log Lang and Inf **27**, 255—-294 (2018). https://doi.org/https://doi.org/10.1007/s10849-018-9268-4
21. Vardi, M.Y.: On the complexity of bounded-variable queries. In: Proceedings of the Fourteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems. pp. 266–276 (1995)
22. Verbrugge, R.: Logic and social cognition: The facts matter, and so do computational models. Journal of Philosophical Logic **38**(6), 649–680 (12 2009). https://doi.org/10.1007/s10992-009-9115-9
23. Yu, Q., Wen, X., Liu, Y.: Multi-agent epistemic explanatory diagnosis via reasoning about actions. In: Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence (IJCAI). pp. 27–33 (2013)
24. Zawidzki, T.W.: Mindshaping: A new framework for understanding human social cognition. MIT Press (2013)