Belief Revision with Bounded Treewidth*

Reinhard Pichler, Stefan Rümmele, and Stefan Woltran

Vienna University of Technology, Austria

Abstract. Problems arising from the revision of propositional knowledge bases have been intensively studied for two decades. Many different approaches to revision have thus been suggested, with the ones by Dalal or Satoh being two of the most fundamental ones. As is well known, most computational tasks in this area are intractable. Therefore, in practical applications, one requires sufficient conditions under which revision problems become efficiently solvable. In this paper, we identify such tractable fragments for the reasoning and the enumeration problem exploiting the notion of treewidth. More specifically, we present new algorithms based on dynamic programming for these problems in Dalal's setting and a tractability proof using Courcelle's Theorem for Satoh's approach.

1 Introduction

Since knowledge is continually evolving, there is a constant need to be able to revise a knowledge base as new information is received. In this paper, we restrict ourselves to *propositional knowledge bases*, i.e., they are given by propositional formulae. Problems arising from the revision of knowledge bases (referred to as *belief revision*) have been intensively studied for two decades. Formally, the problem of belief revision is usually specified as follows: Given a knowledge base (i.e., a formula) α and a formula β, find a revised knowledge base $\alpha \circ \beta$, such that β is true in all models of $\alpha \circ \beta$ and the change compared to the models of α is minimal. The following problems are of great interest:

- *Reasoning*. Given formulae α, β, and γ, decide if $\alpha \circ \beta \models \gamma$ holds.
- *Enumeration*. Given formulae α and β, compute the models of $\alpha \circ \beta$.

Several realizations for $\circ$ have been proposed in the literature and desired properties are formulated by the famous AGM-Postulates [1], or in terms of propositional logic and finite knowledge bases, by Katsuno and Mendelzon [2]. Two of the most fundamental approaches are due to Dalal [3] and Satoh [4]. Complexity results for the reasoning problem w.r.t. different $\circ$ operators are provided in [5] including $\Theta_2 P$- and $\Pi_2 P$-completeness for Dalal's, respectively Satoh's, operator.

An interesting approach to dealing with intractable problems is parameterized complexity theory. In fact, hard problems can become tractable if some problem parameter is bounded by a fixed constant. Such problems are called *fixed-parameter tractable*. One important parameter is treewidth, which measures the "tree-likeness" of a graph or, more generally, of a structure. By using a seminal result due to Courcelle [6], several fixed-parameter tractability (FPT) results in the area of AI and KR have been recently proven [7]. The goal of this work is to obtain tractability results also for belief revision.

* This work was supported by the Austrian Science Fund (FWF), project P20704-N18.

Courcelle's Theorem states that any property of finite structures, that is definable in monadic second-order logic (MSO), becomes tractable over structures with bounded treewidth. In this work, we show FPT for the reasoning problem with Satoh's operator $\circ_S$ by giving an MSO definition of this problem, i.e., defining the property "$\alpha \circ_S \beta \models \gamma$" of a structure representing formulae α, β, and γ. In order to prove an analogous result for Dalal's operator, we have to make use of an extension [8] of Courcelle's Theorem.

The proof of Courcelle's Theorem and its extension in [8] is "constructive": It works by transforming the MSO evaluation problem into a tree language recognition problem, which is then solved via a finite tree automaton (FTA). However, the "algorithms" resulting from such an MSO-to-FTA transformation are usually not practical due to excessively large constants. Consequently, Niedermeier states that MSO "is a very elegant and powerful tool for quickly deciding about FPT, but it is far from any efficient implementation" [9]. We therefore present a novel algorithm for the reasoning problem with Dalal's operator $\circ_D$. This algorithm is based on dynamic programming and builds upon an algorithm of [10] for the #SAT problem (i.e., the problem of counting all models of a given propositional formula). Moreover, we extend our reasoning algorithm to an algorithm for the enumeration problem of "$\circ_D$". As far as the complexity is concerned, our algorithms work in linear time for the reasoning problem and with linear delay (i.e., the time needed for computing the first model and for any further model of $\alpha \circ_D \beta$). Due to lack of space, we only present such dedicated algorithms for Dalal's approach (Satoh's approach can be handled by a similar dynamic programming algorithm, which will be included in the full version of this paper).

Results. Our main contributions are as follows.

- A novel algorithm based on dynamic programming for deciding $\alpha \circ_D \beta \models \gamma$ in *linear time* provided that the treewidth of the formulae α, β, and γ is bounded by a constant (for a formal definition of treewidth, see Section 2).
- An extension of the algorithm for the reasoning problem, such that also the set of all models of $\alpha \circ_D \beta$ is computed. In paticular, our algorithm works with linear delay if the formulae α and β have bounded treewidth.
- We show that the property "$\alpha \circ_S \beta \models \gamma$" is definable in MSO and that "$\alpha \circ_D \beta \models \gamma$" can be defined in the extension of MSO from [8]. In case of $\circ_S$ we thus establish FPT w.r.t. the treewidth of the reasoning problem. In case of $\circ_D$ we thus get an alternative proof of FPT, which follows of course also from our dedicated algorithm via dynamic programming.

2 Background

Throughout the paper, we assume a universe U of propositional atoms. A literal is either an atom a or a negated atom $\overline{a}$. For a set A of atoms, $\overline{A} = \{\overline{a} : a \in A\}$. Clauses are sets of literals. An interpretation (or assignment) I is a set of atoms and we define, for a clause c and $O \subseteq U$, $Mod_O(c) = \{I \subseteq O : (I \cup \overline{(O \setminus I)}) \cap c \neq \emptyset\}$. For a set C of clauses, $Mod_O(C) = \bigcap_{c \in C} Mod_O(c)$. For $O = U$, we write $Mod(C)$ instead of $Mod_O(C)$ for the set of classical models of C. By $At(C)$ we denote the set of atoms occurring in C. In what follows, we use the term *formula* to refer to a set of clauses. As usual, $\alpha \models \beta$ iff each model of formula α is also a model of formula β.

Revision Operators. The approaches of revision we deal with here rely on so-called model-based change operators. Such operators usually utilize a model distance $M \Delta M'$

which yields the set of atoms differently assigned in interpretations M and M', i.e. in our notation, $M \Delta M' = (M \setminus M') \cup (M' \setminus M)$. Assuming that α is consistent (we tacitly make this assumption throughout the paper), the operators due to Satoh [4] ("$\circ_S$"), and respectively, Dalal [3] ("$\circ_D$"), can be defined as follows:

$$Mod(\alpha \circ_S \beta) = \{J \in Mod(\beta) : \exists I \in Mod(\alpha) \text{ s.t. } I \Delta J \in \Delta^{min}(\alpha, \beta)\};$$
$$Mod(\alpha \circ_D \beta) = \{J \in Mod(\beta) : \exists I \in Mod(\alpha) \text{ s.t. } |I \Delta J| = |\Delta|^{min}(\alpha, \beta)\};$$

where we use $\Delta^{min}(\alpha, \beta) = min_{\subseteq}(\{I \Delta J : I \in Mod(\alpha), J \in Mod(\beta)\})$ with $min_{\subseteq}$ selecting elements which are minimal w.r.t. set inclusion, and $|\Delta|^{min}(\alpha, \beta) = min(\{|I \Delta J| : I \in Mod(\alpha), J \in Mod(\beta)\}$.

Subsequently, we refer to a *revision scenario* as either a pair of formulae α, β (in case of the enumeration problem) or a triple α, β, γ (in case of the reasoning problem). For revision scenarios, we assume unless stated otherwise, that the universe U is given by the set of atoms occurring in the involved formulae. In particular, we thus have that $Mod(\alpha \circ \beta)$ with $\circ \in \{\circ_S, \circ_D\}$ refers to a set of models over $At(\alpha \cup \beta)$ (resp. over $At(\alpha \cup \beta \cup \gamma)$ in the context of a reasoning problem).

As shown in [5], given formulae α, β, γ, deciding $\alpha \circ_S \beta \models \gamma$ is Π_2P-complete while deciding $\alpha \circ_D \beta \models \gamma$ is Θ_2P-complete. For both results, hardness holds even in case γ is a single atom.

Tree Decomposition and Treewidth. A *tree decomposition* of a graph $G = (V, E)$ is a pair (T, χ), where T is a tree and χ maps each node n of T (we use $n \in T$ as a shorthand below) to a *bag* $\chi(n) \subseteq V$, such that (1) for each $v \in V$, there is an $n \in T$, s.t. $v \in \chi(n)$; (2) for each $(v, w) \in E$, there is an $n \in T$, s.t. $v, w \in \chi(n)$; (3) for each $n_1, n_2, n_3 \in T$, s.t. n_2 lies on the path from n_1 to n_3, $\chi(n_1) \cap \chi(n_3) \subseteq \chi(n_2)$ holds.

A tree decomposition (T, χ) is *normalized* (or *nice*) [11], if (1) each $n \in T$ has ≤ 2 children; (2) for each $n \in T$ with two children n_1, n_2, $\chi(n) = \chi(n_1) = \chi(n_2)$; and (3) for each $n \in T$ with one child n', $\chi(n)$ and $\chi(n')$ differ in exactly one element.

The *width* of a tree decomposition is defined as the cardinality of its largest bag $\chi(n)$ minus one. It is known that every tree decomposition can be normalized in linear time without increasing the width [11]. The *treewidth* of a graph G, denoted as $tw(G)$, is the minimum width over all tree decompositions of G. For arbitrary but fixed $w \geq 1$, it is feasible in linear time to decide if a graph has treewidth $\leq w$ and, if so, to compute a tree decomposition of width w, see [12].

Tree Decompositions for Revision Scenarios. To build tree decompositions for revision scenarios (α, β, γ), we use incidence graphs[1] over $\Gamma = \alpha \cup \beta \cup \gamma$. Thus, for formulae α, β, γ, such a graph G is given by vertices $\Gamma \cup At(\Gamma)$ and has as edges all pairs (a, c) with an atom a appearing in a clause c of Γ. In case of normalized tree decompositions, we distinguish between six types of nodes: atom introduction (AI), clause introduction (CI), atom removal (AR), clause removal (CR), branch (B), and leaf (L) nodes. The first four types will be often augmented with the element e (either an atom or clauses) which is removed or added compared to the bag of the child node.

Example 1. Figure 1 shows the revision scenario $A \circ B$, which is used as a running example throughout the paper. Since the models of these formulae are $Mod(A) = \{\{x\}\}$ and $Mod(B) = \{\{\}, \{x, y, z\}\}$, it is easy to verify, that $|\Delta|^{min}(A, B) = 1$ and

[1] See [10] for justifications why incidence graphs are favorable over other types of graphs.

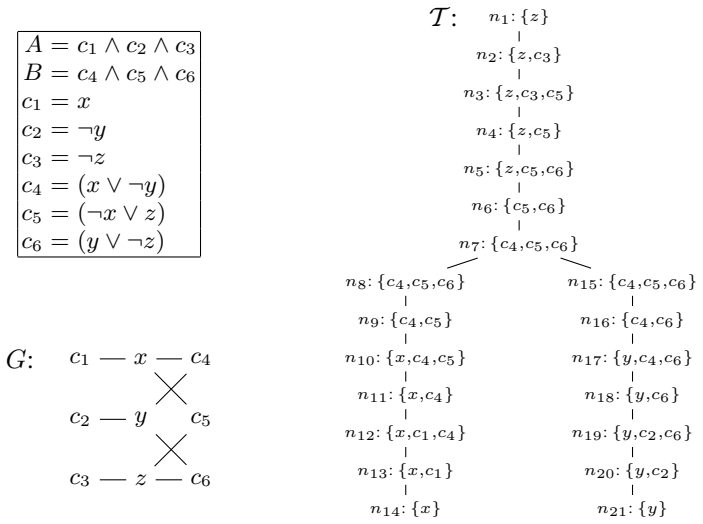

Fig. 1. Revision scenario $A \circ B$; incidence graph G; and normalized tree decomposition $\mathcal{T}$ of G.

$\Delta^{min}(A, B) = \{\{x\}, \{y, z\}\}$. Hence, $Mod(A \circ_D B) = \{\{\}\}$ and $Mod(A \circ_S B) = \{\{\}, \{x, y, z\}\}$. Figure 1 also shows the incidence graph G of this scenario, together with a normalized tree decomposition $\mathcal{T}$ of G having width 2. Actually, G cannot have a tree decomposition of width < 2, since only trees have treewidth = 1 and G contains cycles. Hence, the tree decomposition in Figure 1 is in fact optimal and we have $tw(G) = 2$. Examples for node types are n_{21} as (L) node, n_{20} as (c_2-CI) node, n_{16} as (y-AR) node, n_7 as (B) node, n_6 as (c_4-CR) node, and n_5 as (z-AI) node. ⊣

3 Applying Courcelle's Theorem

An important tool for establishing FPT of a decision problem is Courcelle's Theorem [6], stating that any property of finite structures definable in monadic second-order logic (MSO) becomes tractable (in fact, even linear), if the treewidth of the structures is bounded by a constant. This result was extended to counting problems as well as extremum problems [8]. We recall that MSO extends first-order logic by the use of *set variables* (denoted by upper case letters), which range over sets of domain elements.

In order to show the FPT of the aforementioned belief revision problems using Courcelle's Theorem, we first have to define how the problem instances can be modeled as finite structures. Let formulae α, β, γ be given by a structure $\mathcal{A}$ with signature $\sigma = \{atom(\cdot), clause_\alpha(\cdot), clause_\beta(\cdot), clause_\gamma(\cdot), pos(\cdot,\cdot), neg(\cdot,\cdot)\}$. $\mathcal{A}$ has domain $A = \Gamma \cup At(\Gamma)$, where $\Gamma = \alpha \cup \beta \cup \gamma$. Moreover, for each relation symbol in σ, a relation over A is contained in $\mathcal{A}$ with the following intended meaning: $atom$ designates the set of atoms, $clause_\alpha$, $clause_\beta$ and $clause_\gamma$ denote the set of clauses of α, β and γ respectively. Furthermore $pos(a, c)$ denotes that atom a occurs positively in clause c. Negative literals are described by $neg(a, c)$. The treewidth of a structure $\mathcal{A}$ is defined as the treewidth of the graph that we get by taking the set of domain elements (in our case, $A = \Gamma \cup At(\Gamma)$) as vertices and by considering two vertices (i.e., domain elements)

as adjacent if these domain elements jointly occur in some tuple of the structure, i.e., the edges of this graph are of the form (a, c) where either $pos(a, c)$ or $neg(a, c)$ is contained in the structure. Hence, the treewidth of $\mathcal{A}$ is precisely the treewidth defined via the *incidence graph* of Γ as described in the previous section.

Models of a formula φ can then be stated by the MSO property (see also [7]):

$$mod_\varphi(I) \equiv \forall x[x \in I \rightarrow atom(x)] \wedge \\ \forall c[clause_\varphi(c) \rightarrow \exists a((pos(a, c) \wedge a \in I) \vee (neg(a, c) \wedge a \notin I))].$$

Towards an MSO-encoding for $\alpha \circ_S \beta$ we define three more helper formulae. The first one yields all models J of β together with the possible differences to models of α. The other two characterize valid triples $I \Delta J = K$, respectively proper subsets $X \subset Y$.

$$modD_{\alpha,\beta}(J, K) \equiv mod_\beta(J) \wedge \exists I[mod_\alpha(I) \wedge \mathit{diff}(I, J, K)],$$
$$\mathit{diff}(I, J, K) \equiv \forall a[a \in K \leftrightarrow ((a \in I \wedge a \notin J) \vee (a \notin I \wedge a \in J))],$$
$$sub(X, Y) \equiv \forall a(a \in X \rightarrow a \in Y) \wedge \exists b(b \in Y \wedge b \notin X).$$

We put things together to characterize the models of $\alpha \circ_S \beta$:

$$rev^S_{\alpha,\beta}(J) \equiv \exists K[modD_{\alpha,\beta}(J, K) \wedge \forall J' \forall K'(sub(K', K) \rightarrow \neg modD_{\alpha,\beta}(J', K'))].$$

It is now easy to see that the MSO formula $\forall J(rev^S_{\alpha,\beta}(J) \rightarrow Mod_\gamma(J))$ characterizes the reasoning problem $\alpha \circ_S \beta \models \gamma$ for Satoh's revision operator. We thus obtain via Courcelle's Theorem the following result.

Theorem 1. *The reasoning problem $\alpha \circ_S \beta \models \gamma$ is fixed-parameter linear w.r.t. the treewidth, i.e., it is solvable in time $\mathcal{O}(f(w) \cdot \|\alpha \cup \beta \cup \gamma\|)$, where f is a function depending only on the treewidth w of the revision scenario (α, β, γ).*

For Dalal's operator, we require additional machinery. Following the notation of [8], a model of $Mod(\alpha \circ_D \beta)$ can be described by a *linear extended monadic second-order extremum problem* $\min_{modD_{\alpha,\beta}(J,K)} |K|$, where $modD_{\alpha,\beta}(J, K)$ is the MSO-formula given above in the MSO-characterization of Satoh's revision operator. By using the extension of Courcelle's Theorem of [8], we thus get the following FPT result.

Theorem 2. *Assuming unit cost for arithmetic operations, the reasoning problem $\alpha \circ_D \beta \models \gamma$ is fixed-parameter linear w.r.t. the treewidth, i.e., it is solvable in time $\mathcal{O}(f(w) \cdot \|\alpha \cup \beta \cup \gamma\|)$, where f is a function depending only on the treewidth w of (α, β, γ).*

Proof. The key observation is that $\alpha \circ_D \beta \models \gamma$ holds if and only if $|\Delta|^{min}(\alpha, \beta \wedge \gamma) < |\Delta|^{min}(\alpha, \beta \wedge \neg\gamma)$. Moreover, both expressions $|\Delta|^{min}(\alpha, \beta \wedge \gamma)$ and $|\Delta|^{min}(\alpha, \beta \wedge \neg\gamma)$ can be characterized by linear extended MSO extremum problems. By Theorem 5.6 of [8], those can be evaluated in linear time if we assume unit cost for arithmetic operations and if the treewidth of $\alpha \cup \beta \cup \gamma$ is bounded by a fixed constant. □

4 The Dynamic Programming Approach for $\circ_D$

In this section, we show how the theoretical results from Section 3 can be put to practice by dynamic programming. Due to the space restrictions we discuss here only the realization for the Dalal-revision operator $\circ_D$ in detail.

We start with an algorithm to decide $\alpha \circ_D \beta \models \gamma$. The very idea of such an algorithm is to associate certain objects (so-called bag assignments) to each node n of a tree decomposition for this problem, such that certain information about the subproblem represented by the subtree rooted at n remains available. Consequently, results for the entire problem can be read off the root of the tree decomposition.

We then make use of our algorithm also for the enumeration problem. Hereby, we traverse the tree decomposition a second time, but starting from the root, where we already have identified certain objects which will allow us to compute the models of the revised knowledge base. However, to guarantee that the enumeration does not provide duplicate models, some additional adjustments in the data structure will be necessary.

4.1 Reasoning Problem

For the problem $\alpha \circ_D \beta \models \gamma$, we restrict ourselves here to scenarios where γ is a single atom occurring in $At(\alpha \cup \beta)$ in order to keep the presentation simple. In what follows, we fix $\mathcal{T} = (T, \chi)$ to be a normalized tree decomposition of the incidence graph for $\alpha \cup \beta$. We refer to the root node of T as n_{root}, and we require that the bags of n_{root} and of all leaf nodes of T do not contain any clauses. Such a tree decomposition is easily obtained from a normalized one by suitably adding (CI)- and (CR)-nodes. Additionally, we require that γ appears in $\chi(n_{root})$. Finally, we assume $\alpha \cap \beta = \emptyset$ holds, thus for any clause $c \in \alpha \cup \beta$, its origin $o(c)$ is either α or β. Also recall that we fix $U = At(\alpha \cup \beta)$.

For a node $n \in T$, we denote by T_n the subtree of T rooted at n. For a set S of elements (either atoms or clauses), $n|_S$ is a shorthand for $\chi(n) \cap S$; moreover, $n{\downarrow_S}$ is defined as $\bigcup_{m \in T_n} m|_S$, and $n{\Downarrow_S}$ abbreviates $n{\downarrow_S} \setminus n|_S$.

Definition 1. *A tuple $\vartheta = (n, M_\alpha, M_\beta, C)$, where $n \in T$, $M_\alpha, M_\beta \subseteq n|_U$, and $C \subseteq n|_{\alpha\cup\beta}$ is called a* bag assignment (for node n).

Bag assignments for a node n implicitly talk about interpretations over $n{\downarrow_U}$. The following definition makes this more precise.

Definition 2. *For a bag assignment $\vartheta = (n, M_\alpha, M_\beta, C)$ and $\varphi \in \{\alpha, \beta\}$, define*

$$E_\varphi(\vartheta) = \big\{K \subseteq n{\downarrow_U} :\ K \setminus (n{\Downarrow_U}) = M_\varphi;\\ (C \cap \varphi) \cup (n{\Downarrow_\varphi}) = \{c \in n{\downarrow_\varphi} :\ K \in \mathit{Mod}_{n{\downarrow_U}}(c)\}\big\}.$$

In other words, we associate with a bag assignment $(n, M_\alpha, M_\beta, C)$ all interpretations K that extend M_α in such a way, that all clauses from α appearing in C and in bags below node n are satisfied. The same is done for β. Bag assignments for which such extended interpretations exist for both α and β are of particular interest for us.

Definition 3. *A bag assignments ϑ is called* bag model *iff $E_\alpha(\vartheta) \neq \emptyset \neq E_\beta(\vartheta)$.*

We next rephrase the main features of the definition of $\circ_D$ in terms of bag models and then show that bag models for the root node capture $\circ_D$ as expected.

Definition 4. *For any bag model $\vartheta = (n, M_\alpha, M_\beta, C)$, define*

$$\delta(\vartheta) = \min\big\{\, |I_\alpha \Delta I_\beta| \,:\, I_\alpha \in E_\alpha(\vartheta), I_\beta \in E_\beta(\vartheta)\big\}; \quad \textit{and}$$
$$\mathcal{E}(\vartheta) = \big\{I_\beta \in E_\beta(\vartheta) \,:\, \exists I_\alpha \in E_\alpha(\vartheta), |I_\alpha \Delta I_\beta| = \delta(\vartheta)\big\}.$$

Theorem 3. *Let Θ be the set of all bag models ϑ for n_{root}, such that no bag model ϑ' for n_{root} with $\delta(\vartheta') < \delta(\vartheta)$ exists. Then, $Mod(\alpha \circ_D \beta) = \bigcup_{\vartheta \in \Theta} \mathcal{E}(\vartheta)$.*

Proof. ($\subseteq$): Let $J \in Mod(\alpha \circ_D \beta)$. Hence, $J \in Mod(\beta)$ and there exists an $I \in Mod(\alpha)$, such that $|I \Delta J| = |\Delta|^{min}(\alpha, \beta) = k$. Consider $\vartheta = (n_{root}, M_\alpha, M_\beta, \emptyset)$ where $M_\alpha = n_{root}|_I$ and $M_\beta = n_{root}|_J$. Since we assumed that no clauses are stored in $\chi(n_{root})$ and $n_{root} \downarrow_U = U$, $J \in Mod(\beta)$ yields that $J \in Mod_{n\downarrow_U}(c)$ holds for each $c \in n_{root} \downarrow_\beta = n_{root} \Downarrow_\beta$. The same argumentation applies to I and α. Hence, $I \in E_\alpha(\vartheta)$, $J \in E_\beta(\vartheta)$, and thus ϑ is a bag-model. To show $\vartheta \in \Theta$, it remains to show that no other $\vartheta' \in \Theta$ exists with $\delta(\vartheta') < \delta(\vartheta) \leq k$. Towards a contradiction, suppose such a $\vartheta' = (n_{root}, M'_\alpha, M'_\beta, C')$ exists. By definition, then there exists an $I'_\alpha \in E_\alpha(\vartheta')$ and an $I'_\beta \in E_\beta(\vartheta')$ with $|I'_\alpha \Delta I'_\beta| = \delta(\vartheta')$. Let $\varphi \in \{\alpha, \beta\}$. By definition of $E_\varphi(\cdot)$, we obtain $I'_\varphi \in Mod_{n_{root}\downarrow_U}((C' \cap \varphi) \cup (n_{root} \Downarrow_\varphi))$. Again $C' = \emptyset$ by our assumption for n_{root}, and thus $n_{root} \Downarrow_\varphi = \varphi$. We also know $U = n_{root}\downarrow_U$. $I'_\varphi \in Mod(\varphi)$ follows. Hence, we have found models I'_α, I'_β for α, and resp. β, such that $|I'_\alpha \Delta I'_\beta| < k$. A contradiction to our assumption that $|\Delta|^{min}(\alpha, \beta) = k$. The other direction holds by essentially the same arguments. □

We now put our concept of bag models to work also below the root node. Our goal is to characterize bag models ϑ without an explicit computation of $E_\varphi(\vartheta)$. To this end, first note that bag models for leaf nodes n are easily built from all pairs of interpretations over the atoms in the bag $\chi(n)$; also recall that we assumed that no clause is in $\chi(n)$. Thus, formally, the set of all bag models for a leaf node n is given by $\{(n, M, N, \emptyset) : M, N \subseteq n|_U\}$. For each such bag model $\vartheta = (n, M, N, \emptyset)$, $\delta(\vartheta) = |M \Delta N|$ is clear. Next, we define a relation $\prec_\mathcal{T}$ between bag assignments, such that all bag models of a node are accordingly linked to bag models of the child(ren) node(s). We thus can propagate, starting from the leaves, bag models upwards the tree decomposition. Afterwards, we will show how $\delta(\vartheta)$ can be treated accordingly.

Definition 5. *For bag assignments $\vartheta = (n, M_\alpha, M_\beta, C)$ and $\vartheta' = (n', M'_\alpha, M'_\beta, C')$, we have $\vartheta' \prec_\mathcal{T} \vartheta$ iff n has a single child n', and the following properties are satisfied, depending on the node type of n:*

1. *(c-CR): $M_\alpha = M'_\alpha$, $M_\beta = M'_\beta$, $C = C' \setminus \{c\}$, $c \in C'$;*
2. *(c-CI): $M_\alpha = M'_\alpha$, $M_\beta = M'_\beta$, and $C = C' \cup \{c\}$ if $M_{o(c)} \in Mod_{n|_U}(c)$; and $C = C'$ otherwise;*
3. *(a-AR): $M_\alpha = M'_\alpha \setminus \{a\}$, $M_\beta = M'_\beta \setminus \{a\}$, $C = C'$;*
4. *(a-AI): one of the following cases applies*
   - $M_\alpha = M'_\alpha \cup \{a\}$, $N = M'_\beta \cup \{a\}$, $C = C' \cup \{c \in n|_{\alpha\cup\beta} : a \in c\}$;
   - $M_\alpha = M'_\alpha \cup \{a\}$, $M_\beta = M'_\beta$, $C = C' \cup \{c \in n|_\alpha : a \in c\} \cup \{d \in n|_\beta : \overline{a} \in d\}$;
   - $M_\alpha = M'_\alpha$, $M_\beta = M'_\beta \cup \{a\}$, $C = C' \cup \{c \in n|_\alpha : \overline{a} \in c\} \cup \{d \in n|_\beta : a \in d\}$;
   - $M_\alpha = M'_\alpha$, $M_\beta = M'_\beta$, $C = C' \cup \{c \in n|_{\alpha\cup\beta} : \overline{a} \in c\}$.

For branch nodes, we extend (with slight abuse of notation) $\prec_\mathcal{T}$ to a ternary relation.

Definition 6. *For bag assignments $\vartheta = (n, M_\alpha, M_\beta, C)$, $\vartheta' = (n', M'_\alpha, M'_\beta, C')$ and $\vartheta'' = (n'', M''_\alpha, M''_\beta, C'')$, we have $(\vartheta', \vartheta'') \prec_\mathcal{T} \vartheta$ iff n has two children n', n'', $M_\alpha = M'_\alpha = M''_\alpha$, $M_\beta = M'_\beta = M''_\beta$, and $C = C' \cup C''$.*

Lemma 1. *Let $\vartheta, \vartheta', \vartheta''$ be bag assignments, such that $\vartheta' \prec_{\mathcal{T}} \vartheta$ (resp. $(\vartheta', \vartheta'') \prec_{\mathcal{T}} \vartheta$). Then, ϑ is a bag model iff ϑ' is a bag model (resp. both ϑ' and ϑ'' are bag models).*

Proof. For the proof, one has to distinguish between the node types. Here, we only show the case where ϑ is a bag assignment for a (c-CI) node n with child m. In this case, $\vartheta' \prec_{\mathcal{T}} \vartheta$ holds exactly for assignments of the form $\vartheta = (n, M_\alpha, M_\beta, C)$ and $\vartheta' = (m, M_\alpha, M_\beta, C')$, where $C = C' \cup \{c\}$ if c appears in $\varphi \in \{\alpha, \beta\}$ and M_φ is a partial model of c (i.e., $M_\varphi \in Mod_{n|_U}(c)$); and $C = C'$ otherwise. Consider the case $c \in \alpha$ (the other case is symmetric). We show $E_\alpha(\vartheta) = E_\alpha(\vartheta')$ and $E_\beta(\vartheta) = E_\beta(\vartheta')$. The assertion then follows. There is only room to sketch a proof for $E_\alpha(\vartheta) = E_\alpha(\vartheta')$.

We have $V = n{\downarrow_U} = m{\downarrow_U}$, $W = n \Downarrow_U = m \Downarrow_U$. It is sufficient to show $(C \cap \alpha) \cup n \Downarrow_\alpha = \{d \in n{\downarrow_\alpha} : K \in Mod_V(d)\}$ iff $(C' \cap \alpha) \cup m \Downarrow_\alpha = \{d \in m{\downarrow_\alpha} : K \in Mod_V(d)\}$ for each $K \subseteq V$ s.t. $K \setminus W = M_\alpha$. Fix such a K and note that $n \Downarrow_\alpha = m \Downarrow_\alpha$. One can show $(C \cap \alpha) = \{d \in n|_\alpha : K \in Mod_V(d)\}$ iff $(C' \cap \alpha) = \{d \in m|_\alpha : K \in Mod_V(d)\}$ by observing that $K \in Mod_V(c)$ iff $M_\alpha \in Mod_{n|_U}(c)$. □

Next, we define recursively a number assigned to bag models n and show that this number in fact matches the minimal distance $\delta(\vartheta)$ defined above.

Definition 7. *Let $\vartheta = (n, M_\alpha, M_\beta, C)$ be a bag model. We define*

$$\rho(\vartheta) = \begin{cases} |M_\alpha \Delta M_\beta| & \text{if } n \text{ is a leaf node} \\ \rho(\vartheta') & \text{if } n \text{ is type (CI) or (CR); and } \vartheta' \prec_{\mathcal{T}} \vartheta \\ \min\{\rho(\vartheta') : \vartheta' \prec_{\mathcal{T}} \vartheta\} & \text{if } n \text{ is type (AR)} \\ \min\{\rho(\vartheta') : \vartheta' \prec_{\mathcal{T}} \vartheta\} & \text{if } n \text{ is type (a-AI) and } a \in M_\alpha \text{ iff } a \in M_\beta \\ \min\{\rho(\vartheta') : \vartheta' \prec_{\mathcal{T}} \vartheta\} + 1 & \text{if } n \text{ is type (a-AI) and } a \notin M_\alpha \text{ iff } a \in M_\beta \\ \min\{\vartheta' \times \vartheta'' : (\vartheta', \vartheta'') \prec_{\mathcal{T}} \vartheta\} & \text{if } n \text{ is type (B)} \end{cases}$$

where $\vartheta' \times \vartheta''$ stands for $\rho(\vartheta') + \rho(\vartheta'') - |M_\alpha \Delta M_\beta|$.

Lemma 2. *For any bag model ϑ, $\delta(\vartheta) = \rho(\vartheta)$.*

Example 2. In Fig. 2, we list all bag models ϑ of the tree decomposition from Example 1 together with values $\rho(\vartheta)$. For instance, leaf node n_{21} has bag models for all pairs of interpretations over $\{y\}$. If we go upwards the tree, we observe that bag models for n_{19} additionally contain clauses from $\{c_2, c_6\}$ satisfied by the respective assignments. In the next node n_{18} only those bag models survive where c_2 was contained, since n_{18} is a (c_2-CR) node. Due to space restrictions, we cannot discuss all steps in detail. Note that for the root, the bag model $\vartheta = (n_1, \emptyset, \emptyset, \emptyset)$ is the one with minimal $\rho(\vartheta)$. It can be checked that $\mathcal{E}(\vartheta) = \{\{\}\}$ as expected (recall that $Mod(\alpha \circ_D \beta) = \{\{\}\}$). ⊣

Theorem 4. *Assuming unit cost for arithmetic operations, $\alpha \circ_D \beta \models \gamma$ can be decided in time $O(f(w) \cdot \|\alpha \cup \beta\|)$, where f is a function depending only on the treewidth w of (α, β).*

Proof. Lemma 1 suggests the following algorithm: first, we establish the bag models ϑ for leaf nodes together with their value for $\delta(\vartheta) = \rho(\vartheta)$; then we compute all remaining bag models via $\prec_{\mathcal{T}}$ in a bottom-up manner, and keep track of $\delta(\cdot)$ using the definition of ρ, which is indeed feasible thanks to Lemma 2. As soon as we have the bag models for the root node together with their δ-values we know that bag models in

$(n_1, \emptyset, z, \emptyset)$: 2	(n_5, z, z, c_5c_6): 1	$(n_8, \emptyset, \emptyset, c_4)$: 0	$(n_{15}, \emptyset, \emptyset, c_6)$: 1
$(n_1, \emptyset, \emptyset, \emptyset)$: 1	(n_5, z, z, c_5): 0	$(n_8, \emptyset, \emptyset, c_5)$: 1	$(n_{15}, \emptyset, \emptyset, c_4)$: 0
$(n_2, z, z, \emptyset)$: 1	$(n_5, z, \emptyset, c_6)$: 1	$(n_9, \emptyset, \emptyset, c_4)$: 0	$(n_{16}, \emptyset, \emptyset, c_6)$: 1
$(n_2, z, \emptyset, \emptyset)$: 2	$(n_5, z, \emptyset, c_5c_6)$: 2	$(n_9, \emptyset, \emptyset, c_5)$: 1	$(n_{16}, \emptyset, \emptyset, c_4)$: 0
$(n_2, \emptyset, z, c_3)$: 2	$(n_5, \emptyset, z, c_5c_6)$: 2	(n_{10}, x, x, c_4): 0	$(n_{17}, \emptyset, y, c_6)$: 1
$(n_2, \emptyset, \emptyset, c_3)$: 1	$(n_5, \emptyset, z, c_5)$: 1	$(n_{10}, x, \emptyset, c_5)$: 1	$(n_{17}, \emptyset, \emptyset, c_4)$: 0
(n_3, z, z, c_5): 1	$(n_5, \emptyset, \emptyset, c_6)$: 0	(n_{11}, x, x, c_4): 0	$(n_{18}, \emptyset, y, c_6)$: 1
$(n_3, z, \emptyset, \emptyset)$: 1	$(n_5, \emptyset, \emptyset, c_5c_6)$: 1	$(n_{11}, x, \emptyset, \emptyset)$: 1	$(n_{18}, \emptyset, \emptyset, \emptyset)$: 0
$(n_3, z, \emptyset, c_5)$: 2	$(n_6, \emptyset, \emptyset, c_6)$: 1	(n_{12}, x, x, c_1c_4): 0	(n_{19}, y, y, c_6): 0
$(n_3, \emptyset, z, c_5c_3)$: 2	$(n_6, \emptyset, \emptyset, \emptyset)$: 0	$(n_{12}, x, \emptyset, c_1)$: 1	$(n_{19}, y, \emptyset, \emptyset)$: 1
$(n_3, \emptyset, \emptyset, c_3)$: 0	$(n_6, \emptyset, \emptyset, c_5)$: 1	$(n_{12}, \emptyset, x, c_4)$: 1	$(n_{19}, \emptyset, y, c_2c_6)$: 1
$(n_3, \emptyset, \emptyset, c_5c_3)$: 1	$(n_7, \emptyset, \emptyset, c_4c_6)$: 1	$(n_{12}, \emptyset, \emptyset, \emptyset)$: 0	$(n_{19}, \emptyset, \emptyset, c_2)$: 0
(n_4, z, z, c_5): 1	$(n_7, \emptyset, \emptyset, c_5c_6)$: 2	(n_{13}, x, x, c_1): 0	$(n_{20}, y, y, \emptyset)$: 0
$(n_4, z, \emptyset, \emptyset)$: 1	$(n_7, \emptyset, \emptyset, c_4)$: 0	$(n_{13}, x, \emptyset, c_1)$: 1	$(n_{20}, y, \emptyset, \emptyset)$: 1
$(n_4, z, \emptyset, c_5)$: 2	$(n_7, \emptyset, \emptyset, c_4c_5)$: 1	$(n_{13}, \emptyset, x, \emptyset)$: 1	$(n_{20}, \emptyset, y, c_2)$: 1
$(n_4, \emptyset, z, c_5)$: 2		$(n_{13}, \emptyset, \emptyset, \emptyset)$: 0	$(n_{20}, \emptyset, \emptyset, c_2)$: 0
$(n_4, \emptyset, \emptyset, \emptyset)$: 0		$(n_{14}, x, x, \emptyset)$ 0	$(n_{21}, y, y, \emptyset)$: 0
$(n_4, \emptyset, \emptyset, c_5)$: 1		$(n_{14}, x, \emptyset, \emptyset)$ 1	$(n_{21}, y, \emptyset, \emptyset)$: 1
		$(n_{14}, \emptyset, x, \emptyset)$ 1	$(n_{21}, \emptyset, y, \emptyset)$: 1
		$(n_{14}, \emptyset, \emptyset, \emptyset)$ 0	$(n_{21}, \emptyset, \emptyset, \emptyset)$: 0

Fig. 2. All bag models for the tree decomposition from Example 1.

Θ as defined in Theorem 3 characterize the models of $\alpha \circ_D \beta$. Due to our assumption that γ is just a single atom occurring in $\chi(n_{root})$, it remains to check whether for each $(n_{root}, M_\alpha, M_\beta, C) \in \Theta$, $\gamma \in M_\beta$ holds.

The effort needed for processing a leaf node as well as the transition from child to parent nodes only depends on the treewidth but not on $\|\alpha \cup \beta\|$. The size of $\mathcal{T}$ is linear bounded by the size of $\alpha \cup \beta$, thus the desired time bound for our algorithm follows. □

4.2 Enumeration Problem

Our reasoning algorithm from Section 4.1 gathers the following information along the bottom-up traversal of $\mathcal{T}$: (1) all bag models $\vartheta = (n, M_\alpha, M_\beta, C)$ for all nodes n in $\mathcal{T}$, (2) the minimal distance $\delta(\vartheta)$ between the models $I_\alpha \in E_\alpha(\vartheta)$ and $I_\beta \in E_\beta(\vartheta)$, and (3) the relation $\prec_\mathcal{T}$ indicating which bag model(s) ϑ' at the child node n' (resp. at the two child nodes n' and n'') give rise to which bag model ϑ at a node n in $\mathcal{T}$. In principle, this is all the information needed to enumerate the models in $\alpha \circ_D \beta$ by starting with the bag models ϑ in Θ from Theorem 3 (i.e., the bag models ϑ at the root node n_{root}, s.t. no other bag model ϑ' at n_{root} with smaller value of $\delta(\cdot)$ exists) and determining $\mathcal{E}(\vartheta)$ for every $\vartheta \in \Theta$ by traversing $\mathcal{T}$ in top-down direction following the $\prec_\mathcal{T}$ relation in reversed direction. However, such an enumeration algorithm faces two problems:

(1) In Definition 7 (with $\rho(\vartheta) = \delta(\vartheta)$, by Lemma 2) we computed the minimum value attainable by $\rho(\vartheta)$ over all possible bag models ϑ' (resp. pairs $(\vartheta', \vartheta'')$) with $\vartheta' \prec_\mathcal{T} \vartheta$ (resp. $(\vartheta', \vartheta'') \prec_\mathcal{T} \vartheta$). Hence, when we now follow the $\prec_\mathcal{T}$ relation in the reversed direction, we have to make sure that, from any ϑ, we only continue with bag models ϑ' (resp. with pairs $(\vartheta', \vartheta'')$) that actually lead to the minimal value of $\rho(\vartheta)$.

(2) For distinct bag models ϑ, ϑ' for any node n, $\mathcal{E}(\vartheta) \cap \mathcal{E}(\vartheta') = \emptyset$ is *not* guaranteed. More precisely, suppose that two bag models $\vartheta = (n, M_\alpha, M_\beta, C)$ and $\vartheta' =$

$(n, M'_\alpha, M'_\beta, C')$ fulfill the condition $M_\beta = M'_\beta$. Then it may well happen that some model I_β is contained both in $E_\beta(\vartheta)$ and $E_\beta(\vartheta')$, s.t. I_β has minimal distance $\delta(\vartheta)$ from some $I_\alpha \in E_\alpha(\vartheta)$ and also minimal distance $\delta(\vartheta')$ from some $I'_\alpha \in E_\alpha(\vartheta')$. However, for our enumeration algorithm we want to avoid the computation of duplicates since this would, in general, destroy the linear time upper bound on the delay.

The first problem is dealt with below by restricting the relation $\prec_\mathcal{T}$ to a subset $\ll_\mathcal{T}$ of $\prec_\mathcal{T}$. For the second problem, we shall extend the relation $\ll_\mathcal{T}$ on bag models to a relation on sets of bag models. We start with the definition $\ll_\mathcal{T}$ on bag models. Let us introduce some additional notation first: We identify the components of a bag assignment $\vartheta = (n, M_\alpha, M_\beta, C)$ as $\vartheta_{node} = n$; $\vartheta_\alpha = M_\alpha$; $\vartheta_\beta = M_\beta$; and $\vartheta_{clause} = C$. If Θ is a set of bag assignments, s.t. ϑ_β is identical for all $\vartheta \in \Theta$, then we write Θ_β to denote ϑ_β for any $\vartheta \in \Theta$. Finally, we write $\mathcal{E}(\Theta)$ as a short-hand for $\bigcup_{\vartheta \in \Theta} \mathcal{E}(\vartheta)$.

Definition 8. *Let ϑ, ϑ', and optionally, ϑ'' be bag models, s.t. ϑ'_{node} (and, optionally, also ϑ''_{node}) is a child of $n = \vartheta_{node}$. We define $\vartheta' \ll_\mathcal{T} \vartheta$ (resp. $(\vartheta', \vartheta'') \ll_\mathcal{T} \vartheta$) iff $\vartheta' \prec_\mathcal{T} \vartheta$ (resp. $(\vartheta', \vartheta'') \prec_\mathcal{T} \vartheta$) and one of the following conditions is fulfilled:*

(i) n is of type (CI) or (CR);
(ii) n is of type (AR) and $\delta(\vartheta) = \delta(\vartheta')$;
(iii) n is of type (a-AI), $a \in \vartheta_\alpha$ iff $a \in \vartheta_\beta$, and $\delta(\vartheta) = \delta(\vartheta')$;
(iv) n is of type (a-AI), $a \notin \vartheta_\alpha$ iff $a \in \vartheta_\beta$, and $\delta(\vartheta) = \delta(\vartheta') + 1$; or
(v) n is of type (B) and $\delta(\vartheta) = \delta(\vartheta') + \delta(\vartheta'') - |\vartheta_\alpha \Delta \vartheta_\beta|$.

We now extend $\ll_\mathcal{T}$ from a relation on bag models to a relation on sets Θ of bag models ϑ (with identical component ϑ_β). By slight abuse, we reuse the same symbol $\ll_\mathcal{T}$.

Definition 9. *Let $\Theta \neq \emptyset$ a set of bag models for $n \in \mathcal{T}$, s.t. for all $\vartheta, \vartheta' \in \Theta$, $\vartheta_\beta = \vartheta'_\beta$.*

(i) Suppose that n is either of type (CI), (CR) or of type (a-AI) with $a \notin \Theta_\beta$. Then we define $\Theta' \ll_\mathcal{T} \Theta$ for $\Theta' = \{\vartheta' : \vartheta'_\beta = \Theta_\beta$ and $\vartheta' \ll_\mathcal{T} \vartheta$ for some $\vartheta \in \Theta\}$.

(ii) Suppose that n is of type (a-AI) with $a \in \Theta_\beta$. Then we define $\Theta' \ll_\mathcal{T} \Theta$ for $\Theta' = \{\vartheta' : \vartheta'_\beta = \Theta_\beta \setminus \{a\}$ and $\vartheta' \ll_\mathcal{T} \vartheta$ for some $\vartheta \in \Theta\}$.

(iii) Suppose that n is of type (a-AR). Then we define $\Theta'_1 \ll_\mathcal{T} \Theta$ and $\Theta'_2 \ll_\mathcal{T} \Theta$ for $\Theta'_1 = \{\vartheta' : \vartheta'_\beta = \Theta_\beta$ and $\vartheta' \ll_\mathcal{T} \vartheta$ for some $\vartheta \in \Theta\}$ and $\Theta'_2 = \{\vartheta' : \vartheta'_\beta = \Theta_\beta \cup \{a\}$ and $\vartheta' \ll_\mathcal{T} \vartheta$ for some $\vartheta \in \Theta\}$.

(iv) Suppose that n is of type (B). Then we define $\Theta' \ll_\mathcal{T} \Theta$ for $\Theta' = \{\vartheta' : \exists \vartheta''$ with $(\vartheta', \vartheta'') \ll_\mathcal{T} \vartheta$ for some $\vartheta \in \Theta\}$.

Moreover, for every $\hat{\Theta} \subseteq \Theta'$ with $\Theta' \ll_\mathcal{T} \Theta$, we define $(\hat{\Theta}, \Theta'') \ll_\mathcal{T} \Theta$, where $\Theta'' = \{\vartheta'' : \exists \vartheta \in \Theta$ and $\exists \vartheta' \in \hat{\Theta}$ with $(\vartheta', \vartheta'') \ll_\mathcal{T} \vartheta\}$.

The following lemma states that every model in $\mathcal{E}(\Theta)$ at node n can be computed via $\mathcal{E}(\Theta')$ (and optionally $\mathcal{E}(\Theta'')$) at the child node(s) of n and, conversely, that every element in $\mathcal{E}(\Theta')$ (and optionally $\mathcal{E}(\Theta'')$) can indeed be extended to a model of $\mathcal{E}(\Theta)$.

Lemma 3. *Let $n \in \mathcal{T}$ and Θ be a non-empty set of bag models for n, s.t. for all $\vartheta, \vartheta' \in \Theta$, $\vartheta_\beta = \vartheta'_\beta$. Then the following properties hold:*

(i) Suppose n is of type (CI), (CR), (AR), or of type (a-AI), s.t. $a \notin \Theta_\beta$. Then $I \in \mathcal{E}(\Theta)$ iff $I \in \mathcal{E}(\Theta')$, s.t. $\Theta' \ll_\mathcal{T} \Theta$.

(ii) Suppose n is of type (a-AI), s.t. $a \notin \Theta_\beta$. Then $I \in \mathcal{E}(\Theta)$ iff $(I \setminus \{a\}) \in \mathcal{E}(\Theta')$, s.t. $\Theta' \ll_\mathcal{T} \Theta$.

(iii) Suppose n is of type (B). Then $I \in \mathcal{E}(\Theta)$ iff $I = I' \cup I''$ for some $I' \in \mathcal{E}(\hat{\Theta})$ and $I'' \in \mathcal{E}(\Theta'')$, where $\hat{\Theta} \subseteq \Theta'$, $\Theta' \ll_{\mathcal{T}} \Theta$, and $(\hat{\Theta}, \Theta'') \ll_{\mathcal{T}} \Theta$.

For our enumeration algorithm, we start at the root node of $\mathcal{T}$ and first partition the relevant bag models ϑ according to ϑ_β. Formally, let Θ be the set of all bag models ϑ for n_{root}, such that no bag model ϑ' for n_{root} with $\delta(\vartheta') < \delta(\vartheta)$ exists. Then we partition Θ into $\Theta_1, \ldots, \Theta_n$, such that for each $\vartheta, \vartheta' \in \Theta_i$, $\vartheta_\beta = \vartheta'_\beta$, and for each $\vartheta \in \Theta_i$, $\vartheta' \in \Theta_j$ with $i \neq j$, $\vartheta_\beta \neq \vartheta'_\beta$. Clearly, the sets $\mathcal{E}(\Theta_i)$ are pairwise disjoint. Hence, no duplicates will be computed when we compute $\mathcal{E}(\Theta_1), \ldots, \mathcal{E}(\Theta_n)$ separately.

Given a set Θ of bag models, we compute $\mathcal{E}(\Theta)$ by implementing an appropriate *iterator* for every node n in $\mathcal{T}$. The iterator provides functions open, get_current, and get_next. In addition, other functions like close (to deallocate state information) are needed which we do not discuss here.

The open *function.* The open function serves to initialize the state information at each node of a given subtree of $\mathcal{T}$. For instance, it is convenient (in particular, for branch nodes) to store in a Boolean flag first whether get_next() has not yet been called since the initialization with the call of function open. Moreover, for an (AR) node, there can exist two sets Θ_1 and Θ_2 with $\Theta_i \ll_{\mathcal{T}} \Theta$. We have to store in the state of the (AR) node, which one of these two sets is currently being processed at the child node.

The open function takes as input a set Θ of bag models ϑ with identical ϑ_β and recursively calls open(Θ') with $\Theta' \ll_{\mathcal{T}} \Theta$. If the current node is of type (CI), (CR), or (AI), then Θ' is unique. Likewise, Θ' is unique for the first child of a branch node. In case of an (AR) node, Θ' corresponds to Θ_1 from Definition 9, case (iii), provided that it is non-empty. Otherwise, $\Theta' = \Theta_2$ is chosen. The children of a branch node are treated asymmetrically by the $\ll_{\mathcal{T}}$-relation and, hence, also by the open function. In the first place, we only compute Θ' with $\Theta' \ll_{\mathcal{T}} \Theta$ for the *first child* of every branch node. As we shall explain below, the function get_next() computes the set of assignments $\mathcal{E}(\Theta)$, returning one such assignment per call. For branch nodes, we thus compute for the first child the set $\mathcal{E}(\Theta')$ with $\Theta' \ll_{\mathcal{T}} \Theta$. For each assignment I' thus returned, get_next() also yields $\Gamma' = \{\hat{\vartheta} : I' \in \mathcal{E}(\hat{\vartheta})\} \subseteq \Theta'$. Then the subtree rooted at the second child node is processed with Θ'', s.t. $(\Gamma', \Theta'') \ll_{\mathcal{T}} \Theta$. Hence, for every assignment I', we have to compute Θ'' (which is uniquely determined by Γ' and Θ) and call open(Θ''), before we can retrieve the assignments I'' in $\mathcal{E}(\Theta'')$ with get_next().

The get_next *and* get_current *function.* Suppose that a node n in $\mathcal{T}$ has been initialized by a call of open(Θ) with $\vartheta_{node} = n$ for every $\vartheta \in \Theta$. Then we can call get_next() for this node in order to retrieve the first resp. the next assignment I in $\mathcal{E}(\Theta)$. In addition to the assignment I, the get_next function also provides a set $\Gamma \subseteq \Theta$ as output, s.t. $\Gamma = \{\hat{\vartheta} : I \in \mathcal{E}(\hat{\vartheta})\}$. As we have already seen, this set Γ of bag assignments is needed when we encounter a branch node on our way back to the root, in order to determine the set Θ'' for the second child. The get_current function is called (for the first child of a branch node) to retrieve once again the result from the previous call to get_next. If no next assignment exists, then get_next returns the value "Done".

In order to compute the first resp. next assignment I, we traverse $\mathcal{T}$ downwards by recursive calls of get_next() until the leaves are reached. In the leaves, we start with the assignment $I = \Theta_\beta$ and also set $\Gamma = \Theta$. This assignment I and the set Γ are now updated on the way back to the root. The only modifications to I are in fact done when we are at an (a-AI) node or at a (B) node. For (AI) nodes, we add a in case a is added

Function get_next for a branch node n with child nodes n', n''
  Let Θ be the input parameter of the previous call of function open
  if first *then*
    first = False;
    $(I', \Gamma') = n'$.get_next();
    Let Θ'' s.t. $(\Gamma', \Theta'') \ll_{\mathcal{T}} \Theta$;
    n''.open(Θ'')
    $(I'', \Gamma'') = n''$.get_next()
  else
    $(I', \Gamma') = n'$.get_current()
    $(I'', \Gamma'') = n''$.get_next()
    if $(I'', \Gamma'') =$ undefined (i.e., the call of n''.get_next() returned "Done") *then*
      $(I', \Gamma') = n'$.get_next();
      if $(I', \Gamma') =$ undefined (i.e., the call of n'.get_next() returned "Done") *then*
        return "Done"
      endif
      Let Θ'' s.t. $(\Gamma', \Theta'') \ll_{\mathcal{T}} \Theta$;
      n''.open(Θ'')
      $(I'', \Gamma'') = n''$.get_next()
    endif
  endif
  return $(I' \cup I'', \{\vartheta \in \Theta : \exists \gamma' \in \Gamma'$ and $\exists \gamma'' \in \Gamma''$, s.t. $(\gamma', \gamma'') \ll_{\mathcal{T}} \vartheta\})$

Fig. 3. Function get_next() for a branch node.

to the respective Θ_β. For (B) nodes, we set $I = I' \cup I''$, where I' (resp. I'') is the assignment returned by the call of get_next() for the first (resp. second) child node. In Figure 3 we give the pseudo-code of the get_next function in case of a branch node. For the remaining node types, the implementation of get_next is even simpler.

Theorem 5. *Given formulae α and β, the models in $Mod(\alpha \circ_D \beta)$ can be computed with delay $O(f(w) \cdot |\alpha \cup \beta|)$, where f is a function depending only on the treewidth w of (α, β).*

Proof. By our definition of $\ll_{\mathcal{T}}$ on sets and by Lemma 3, we can be sure that (1) every assignment in $Mod(\alpha \circ_D \beta)$ is eventually computed by our iterator-based implementation via recursive calls of get_next and (2) no assignment is computed twice. Indeed, our set-based definition of the $\ll_{\mathcal{T}}$-relation groups together bag models ϑ with identical ϑ_β and, for any bag models ϑ' with $\vartheta_\beta \neq \vartheta'_\beta$, we trivially have $\mathcal{E}(\vartheta) \cap \mathcal{E}(\vartheta') = \emptyset$.

As far as the complexity is concerned, note that the recursive calls of the open function come down to a top-down traversal of $\mathcal{T}$. (In fact, by the asymmetric treatment of the children of a branch node, open is only called for the nodes along the left-most path in $\mathcal{T}$.) Similarly, each call of get_next and get_current leads to a single traversal of the subtree below the current node n. The work actually carried out inside each call is independent of the size of $\mathcal{T}$. Hence, in total, we end up with a time bound that is linear in the size of $\mathcal{T}$ and, hence, in the size of α and β. □

5 Conclusion

The quest for (fixed-parameter) tractability has been pursued in many areas of KR and AI. However, in the context of belief revision, very few activities have been undertaken

in this direction – apart from the work of Darwiche [13], which relies on compilation techniques. To the best of our knowledge, neither Courcelle's Theorem [6] (or one of its extensions as [8]) nor dynamic programming approaches (along the lines of tree decompositions) have been applied to belief revision problems, so far. For other KR formalisms though such approaches already proved to be successful (see, e.g., [7, 14]).

In this work, we have identified new tractable classes of revision problems with respect to two of the most fundamental approaches [3, 4]. Moreover, we provided novel dynamic programming algorithms for Dalal's revision operator [3] (i.e. for the problem of deciding $\alpha \circ_D \beta \models \gamma$, and enumerating the models of $\alpha \circ_D \beta$) which run in linear time (resp. with linear delay) if the treewidth of the revision scenario is bounded.

Future work includes to apply the methods used in this paper also to update operators. The approach due to Winslett [15], for instance, can be shown tractable by "plain" MSO, while the approach due to Forbus [16] is less accessible to such techniques since the concept of cardinality is "hidden" in the characterization (see, e.g. [17] for further discussions on this problem). Another direction of research is to apply our methods to approaches for iterated belief revision. This however calls for the additional requirement that the outcome of a single revision step has to be of bounded treewidth as well. It is an interesting research question of its own how to ensure such a property.

References

1. Alchourrón, C., Gärdenfors, P., Makinson, D.: On the logic of theory change: Partial meet functions for contraction and revision. Journal of Symbolic Logic **50** (1985) 510–530
2. Katsuno, H., Mendelzon, A.: Propositional knowledge base revision and minimal change. Artificial Intelligence **52** (1991) 263–294
3. Dalal, M.: Investigations into a theory of knowledge base revision. In: Proc. AAAI'88, AAAI Press / The MIT Press (1988) 449–479
4. Satoh, K.: Nonmonotonic reasoning by minimal belief revision. In: Proceedings of the International Conference on Fifth Generation Computer Systems. (1988) 455–462
5. Eiter, T., Gottlob, G.: On the complexity of propositional knowledge base revision, updates, and counterfactuals. Artificial Intelligence **57** (1992) 227–270
6. Courcelle, B.: Graph rewriting: An algebraic and logic approach. In: Handbook of Theoretical Computer Science, Vol. B. Elsevier Science Publishers (1990) 193–242
7. Gottlob, G., Pichler, R., Wei, F.: Bounded treewidth as a key to tractability of knowledge representation and reasoning. In: Proc. AAAI'06, AAAI Press (2006) 250–256
8. Arnborg, S., Lagergren, J., Seese, D.: Easy problems for tree-decomposable graphs. Journal of Algorithms **12** (1991) 308–340
9. Niedermeier, R.: Invitation to Fixed-Parameter Algorithms. Oxford University Press (2006)
10. Samer, M., Szeider, S.: Algorithms for propositional model counting. In: Proc. LPAR'07. Vol. 4790 of LNCS. Springer (2007) 484–498
11. Kloks, T.: Treewidth, Computations and Approximations. Springer (1994)
12. Bodlaender, H.L.: A linear-time algorithm for finding tree-decompositions of small treewidth. SIAM J. Comput. **25** (1996) 1305–1317
13. Darwiche, A.: On the tractable counting of theory models and its application to truth maintenance and belief revision. Journal of Applied Non-Classical Logics **11** (2001) 11–34
14. Jakl, M., Pichler, R., Woltran, S.: Answer-set programming with bounded treewidth. To appear in Proc. IJCAI (2009)
15. Winslett, M.: Reasoning about action using a possible models approach. In: Proc. (AAAI'88), AAAI Press / The MIT Press (1988) 89–93
16. Forbus, K.: Introducing actions into qualitative simulation. Proc. IJCAI (1989) 1273–1278
17. Szeider, S.: Monadic second order logic on graphs with local cardinality constraints. In: Proc. MFCS'08. Vol. 5162 of LNCS. Springer (2008) 601–612