Logics of repeating values on data trees and branching counter systems

Sergio Abriola, Diego Figueira, Santiago Figueira

Logics of repeating values on data trees and branching counter systems*

Sergio Abriola[1], Diego Figueira[2], and Santiago Figueira[1]

[1] University of Buenos Aires, Argentina and ICC-CONICET, Argentina
[2] CNRS, LaBRI, France

Abstract. We study connections between the satisfiability problem for logics on data trees and Branching Vector Addition Systems (BVAS). We consider a natural temporal logic of "repeating values" (LRV) featuring an operator which tests whether a data value in the current node is repeated in some descendant node.
On the one hand, we show that the satisfiability of a restricted version of LRV on ranked data trees can be reduced to the coverability problem for Branching Vector Addition Systems. This immediately gives elementary upper bounds for its satisfiability problem, showing that restricted LRV behaves much better than downward-XPath, which has a non-primitive-recursive satisfiability problem.
On the other hand, satisfiability for LRV is shown to be reducible to the coverability for a novel branching model we introduce here, called Merging VASS (MVASS). MVASS is an extension of Branching Vector Addition Systems with States (BVASS) allowing richer merging operations of the vectors. We show that the control-state reachability for MVASS, as well as its bottom-up coverability, are in 3ExpTime.
This work can be seen as a natural continuation of the work initiated by Demri, D'Souza and Gascon for the case of data words, this time considering *branching* structures and counter systems, although, as we show, in the case of data trees more powerful models are needed to encode satisfiability.

1 Introduction

Logics for data trees. Finite data trees are ubiquitous structures that have attracted much attention lately. A *data tree* is a finite tree whose every position carries a *label* from a finite alphabet and a collection of *data values* from some infinite domain.[3] This structure has been considered in the realms of semistructured data as a simple abstraction of XML documents, timed automata, program verification, and generally in systems manipulating data values. Finding decidable

* We thank STIC AmSud, ANPCyT-PICT-2013-2011, UBACyT 20020150100002BA, and the Laboratoire International Associé "INFINIS".

[3] Other works have considered different simplifications of these structures, either having only one data value per node (*e.g.*, [2]) or ignoring the label (*e.g.*, [7]).

logics or automata models over data trees is a fundamental quest when reasoning on data-driven systems.

A wealth of specification formalisms on these structures (either for data trees or its 'word' version, data words) have been introduced, stemming from automata [25, 28], first-order logic [2, 19, 16, 4], XPath [20, 18, 15, 14, 13], or temporal logics [9, 24, 22, 11, 7, 21]. In full generality, most formalisms lead to undecidable reasoning problems and a well-known research trend consists of finding a good trade-off between expressiveness and decidability.

Interesting and surprising results have been exhibited about relationships between *logics* for data trees and *counter automata* [19, 18, 20]. This is why logics for data trees are not only interesting for their own sake but also for their deep relationships with counter systems.

This work. The aim of this work is to study the basic mechanism of "data repetition", common to many logics studied on data trees. For this, we study a basic logic that can navigate the structure of the tree through the use of CTL-like modalities, and on the other hand can make "data tests", by asking whether a data value is repeated in the subtree. More concretely, the data tests are formulas of the form $u \approx \mathrm{EF}v$, stating that the data value stored in attribute (also called *variable* here) u of the current node is equal to the data value stored in attribute v of some descendant. This *logic of repeating values*, or LRV, has been the center of a line of investigation studied in [6, 7] on *data words*, evidencing tight correspondences between reachability problems for Vector Addition Systems and the satisfiability problem. The current work pursues this question further, exhibiting connections between the satisfiability problem of LRV *over data trees* and the bottom-up coverability problem for branching counter systems. In order to obtain connections with branching Vector Addition Systems with States, or branching VASS [29], we also introduce a restriction where tests of the form $u \approx \mathrm{EF}v$ are only allowed when $u = v$. We denote this restriction by $\mathrm{LRV}^{\mathrm{D}}$. This symbiotic relation between counter systems and logics leads us to consider some natural extensions of both the logic and the branching counter systems. In particular, we introduce a new model of branching counter system of independent interest, with decidable coverability and control-state reachability problems, that captures LRV.

The extension of the logic LRV from words to trees is a very natural one. However, the techniques needed to encode the satisfiability of the logic into a counter system are not simple extensions from the ones provided on data words. The reason for this difficulty is manyfold: *a*) the fact that now the future is non-linear in addition to the possibility of having a data value repeating at several descendants in *different* variables, makes the techniques of [7] for propagating values of configurations impractical; *b*) further, this seems to be impossible for the case of data trees, and we could only show a reduction for the fragment $\mathrm{LRV}^{\mathrm{D}}$; *c)* in order to reduce the satisfiability problem for the full logic we will need to augment the power of branching VASS with the possibility to 'merge' counters in a more powerful way, somewhat akin to what has been done for encoding the satisfiability for FO^2 [19].

Contributions. The main contributions are the following:

- We show that the satisfiability for $\mathrm{LRV}^{\mathrm{D}}$ on k-ranked data trees is reducible, in exponential space, to the control-state reachability problem for VASS_k (*i.e.*, Branching VASS of rank k) in Section 5. Since the control-state reachability problem is decidable [29] in 2ExpTime [8], this reduction yields a decision procedure.
- We consider the addition of an operator $\mathrm{AG}_{\approx v}(\varphi)$ expressing "every descendant with the same v-attribute verifies φ", and we show that the logic resulting from adding positive instances of this operator is equivalent to the control-state reachability for Branching VASS, that is, there are reductions in both directions (Section 6).
- We introduce an extension of Branching VASS, called *Merging* VASS or MVASS in Section 4.2. This model allows for merging counters in branching rules in a form which is not necessarily component-wise, allowing for some weak form of counter transfers. We show that the bottom-up coverability (and control-state reachability) problem for MVASS is in 3ExpTime (Section 4.4). This is arguably a model of independent interest.
- We show that the satisfiability for LRV on k-ranked data trees can be reduced to the control-state reachability for MVASS_k in Section 7. As in the case of $\mathrm{LRV}^{\mathrm{D}}$, this yields a decision procedure.

Due to space limitations, technical details of the proofs are contained in the appendix.

Related work. The most closely related work is the one originated by Demri et al. in [5, 6] and pursued in [7]. These works study the satisfiability problem for temporal logics on *data words*, extended with the ability to test whether a data value is repeated in the past/future. Indeed, our current work is motivated by the deep correlations evidenced by these works, between counter systems and simple temporal logics on data words. The present manuscript expands this analysis to *branching* logics and counter systems.

There are several works showing links between reachability-like problems for counter systems and the satisfiability problem of logics on data trees. The first prominent example is that satisfiability for Existential MSO with two variables on data words ($\mathrm{EMSO}^2(+1, <, \sim)$) corresponds precisely to reachability of VASS [3], in the sense that there are reductions in both directions. On the other hand, EMSO^2 over (unranked) *data trees* was shown to have tight connections with the reachability problem for an *extension* of BVASS [19], called 'EBVASS'. This extension has features which are very close to the model we introduce here, MVASS, but it does not capture, nor is captured by, MVASS. One can draw a parallel between the situation of the satisfiablity for EMSO^2 and for LRV: while on data words both are inter-reducible to VASS, the extension to data-trees is non-trivial, and they no longer correspond to BVASS, but to *extensions* thereof.

In the course of the last decade, several logics for data trees have been proposed. Among those that feature navigation in terms of modalities such as temporal operators, one noticeable logic is that of XPath. Although the satisfiability

problem for XPath is undecidable, several fragments have been shown to be decidable through reduction to the reachability or coverability problems for counter systems [9, 18, 12]. In particular, the satisfiability problem for XPath with strict descendant (usually written $\downarrow_+$) on ranked data trees has already a non-primitive-recursive lower bound in complexity, as can be seen by adapting techniques shown for data words [17].

Modulo a simple coding, our logic LRV is captured by a fragment of regular-XPath, here called reg-XPath$_{\text{LRV}}$, on data trees where path expressions are allowed to use Kleene star on any expression (this what we denote by 'regular' XPath), and data tests are of the form $\langle \varepsilon \star \downarrow^* [\varphi] \rangle$ or $\langle \downarrow^n [\varphi] \star \downarrow^m [\psi] \rangle$ for some $n, m \in \mathbb{N}$ and $\star \in \{=, \neq\}$. There are, however, three provisos for this statement. First, in the aforementioned works on XPath the data model consists of data trees whose every position carries exactly *one* data value. In the present paper, we study 'multi-attributed' data trees where, essentially, each node carries a set of pairs 'attribute:value'. However, by means of a simple coding, such as putting every 'attribute:value' as a leaf of the corresponding node, one can easily translate LRV formulas to XPath formulas. Second, our LRV formulas are of the form $u \approx \mathrm{EF} v$ stating that the current data value under attribute u is repeated in a node x of the subtree under attribute v, but one cannot test that some property ψ further holds at the repeating node x. However, it was shown in [7] that one can extend the logic with this power, obtaining formulas of the form $x \approx \mathrm{EF} y[\psi]$, since this extended logic is PTime-reducible to the logic without these tests. Third, the LRV formulas cannot test for regular properties on the labeling of paths, and thus there is no precise characterization in terms of a natural fragment of regular-XPath, but one could add regular path tests to LRV to match the expressive power of reg-XPath$_{\text{LRV}}$ without changing any of our results.

In fact, the fragment reg-XPath$_{\text{LRV}}$ extends also the fragment DataGL considered in [1] and [13] containing only data tests of the form $\langle \varepsilon \star \downarrow^* [\varphi] \rangle$, which is known to be PSpace-complete on unranked data trees [13].

It is not hard to see that the satisfiability problem of LRV on unranked data trees is PSpace-complete following the techniques from [13]. On the other hand, on ranked data trees we know, by the discussion above, that if we would allow intermediate tests in a way to be able to encode the expressive power of XPath($\downarrow_+$) we would have a non-primitive recursive lower bound. It is therefore natural to limit the navigation *disallowing* intermediary tests. This natural fragment was already studied on data words [7], and we now study it on data trees.

2 Preliminaries

Let $\mathbb{N}^+ = \{1, 2, \dots\}$, $\mathbb{N} = \mathbb{N}^+ \cup \{0\}$, and $\underline{n} = \{1, \dots, n\}$ for every $n \in \mathbb{N}$. We use the bar notation $\bar{x}$ to denote a tuple of elements, where $\bar{x}[i]$, for $i > 0$, refers to the i-th element of the tuple. For any pair of vectors $\bar{x}, \bar{y} \in \mathbb{Z}^k$ we write $\bar{x} \leq \bar{y}$ if $\bar{x}[i] \leq \bar{y}[i]$ for all $1 \leq i \leq k$. The constant $\bar{\emptyset}$ refers to the (unique) vector of dimension 0, and the constant $\bar{e}_i$ refers to the vector (whose dimension will

always be clear from the context) so that $\bar{e}_i[i] = 1$ and $\bar{e}_i[j] = 0$ for all $j \neq i$. We write $\bar{0}$ for the tuple of all 0's (the dimension being implicit from the context).

A **linear set** of dimension k is a subset of $\mathbb{N}^k$ which is either empty or described as $\{\bar{v}_0 + \alpha_1\bar{v}_1 + \cdots + \alpha_n\bar{v}_n \mid \alpha_1, \ldots, \alpha_n \in \mathbb{N}\}$ for some $n \in \mathbb{N}$ and $\bar{v}_0, \ldots, \bar{v}_n \in \mathbb{N}^k$. Henceforward we assume that linear sets are represented by the **offset** $\bar{v}_0$ and the **generators** $\bar{v}_1, \ldots, \bar{v}_n$, where numbers are represented in binary. For ease of writing we will denote a linear set like the one above by "$\bar{v}_0 + \{\bar{v}_1, \ldots, \bar{v}_n\}^*$".

We fix once and for all an infinite domain of **data values** $\mathbb{D}$. A **data tree** of rank k over a finite set of labels $\mathbb{A}$ and a finite set of attributes $\mathbb{V}$, is a finite tree whose every node x contains a pair $(a, \mu) \in \mathbb{A} \times \mathbb{D}^{\mathbb{V}}$ and has no more than k children. In general, a will be called the **label** of x and $\mu(v)$ will be called the **data value** of attribute $v \in \mathbb{V}$ at x. The i**-ancestor** of a node x of a data tree T is the ancestor at distance i from x (*i.e.*, the 1-ancestor is the parent); while the i**-descendant**s of x are all the descendants of x at distance i.

3 Logic of repeating values on data trees

We will work with a temporal logic using CTL* modalities [26, 10] to navigate the tree —although this is not really essential to our results, in the sense that any other MSO definable data-blind operators could also be added to the logic obtaining similar results. The **Logic of Repeating Values** LRV contains the typical modalities from CTL*, such as EF, AF, EU, etc. as well as the possibility to test for the label of the current node, and *data tests*. Data tests are restricted to being very basic, as in [6], of the form "$u \approx \mathrm{EF}v$" stating "the data value of attribute u appears again at the attribute v of some descendant", or "$u \not\approx \mathrm{EF}v$" stating "there is a descendant node whose attribute v contains a different data value from the data value of the attribute u of the current node". Since LRV is closed under Boolean connectives, this means we can also express, for instance, that attribute u of all descendants have the same data value as the current node's: $\neg(u \not\approx \mathrm{EF}u)$.

Formally, formulas of LRV are defined by

$$\varphi ::= a \mid \varphi \wedge \varphi \mid \neg\varphi \mid \mathrm{EU}(\varphi, \psi) \mid u \approx \mathrm{EF}v \mid u \not\approx \mathrm{EF}v \mid u \approx \mathrm{EX}^i v \mid u \not\approx \mathrm{EX}^i v,$$

where a ranges over a finite set of labels $\mathbb{A}$, u, v range over a finite set of attribute variables $\mathbb{V}$ (also called just 'variables'), and $i \in \mathbb{N}^+$. Given a data tree T and a node x of T, the satisfaction relation $\models$ is defined in the usual way: $T, x \models a$ if a is the label of x; $T, x \models u \approx \mathrm{EF}v$ [resp. $T, x \models u \not\approx \mathrm{EF}v$] if there is a strict descendant y of x so that the u-attribute of x has the same [resp. different] data value as the v-attribute of y; $T, x \models u \approx \mathrm{EX}^i v$ [resp. $T, x \models u \not\approx \mathrm{EX}^i v$] if there exists an i-descendant of x whose v-attribute is equal [resp. distinct] to the u-attribute of x; and $T, x \models \mathrm{EU}(\varphi, \psi)$ if there is some strict descendant y of x so that $T, y \models \varphi$ and every other node z strictly between x and y verifies $T, z \models \psi$.

Note that the remaining CTL* modalities (EX, EG, EF, AX, AG, AF, AU) can be expressed using EU[4].

We call $\mathrm{LRV}^{\mathrm{D}}_n$ the logic using at most n attribute variables, whose only admissible data tests are of the form $u \approx \mathrm{EF}u$, $u \not\approx \mathrm{EF}u$, $u \approx \mathrm{EX}^i u$ or $u \not\approx \mathrm{EX}^i u$ (same variable in the left and right sides). Intuitively, this corresponds to the restriction where each attribute variable ranges over a *disjoint* set of data values (hence the letter 'D').

4 Models of branching counter systems

We present the models of counter systems we are going to work with. The first one is a well-known model, usually known as Branching Vector Addition System with States, or "BVASS", while the second one is a useful extension of the first one where the split/merge operation of the counters is controlled by the use of linear sets.

4.1 Branching VASS

A VASS of rank k and dimension n, or $n\mathrm{VASS}_k$, is a tuple $\mathcal{A} = \langle Q, U, B\rangle$, where Q is a finite set of states, $U \subseteq Q \times \mathbb{Z}^n \times Q$ is a set of unary rules, and $B \subseteq Q \times Q^{\leq k}$ is a finite set of branching rules. We notate $q \xrightarrow{\bar{v}} q'$ for a unary rule $(q, \bar{v}, q') \in U$, and $q \to (q_1, \ldots, q_i)$ for a branching rule $(q, q_1, \ldots, q_i) \in B$. A **configuration** is an element from $\mathit{Confs} := Q \times \mathbb{N}^n$. For a configuration $(q, \bar{n})$ we often use the term "counter i" instead of "$n[i]$" (in the case $n = 1$ we speak of *the* counter).

A **derivation tree** [resp. **incrementing derivation tree**] is a finite tree $\mathcal{D}$ whose every node x is either

- labeled with a pair $(p \xrightarrow{\bar{v}} p', (q, \bar{n})) \in U \times \mathit{Confs}$ so that $p \xrightarrow{\bar{v}} p'$ is a unary rule of U, $p = q$ and it has exactly one child, which is labeled $(r_1, (p_1, \bar{n}_1))$ so that $p' = p_1$ and

$$\bar{n} + \bar{v} = \bar{n}_1 \quad [\text{resp. } \bar{n} + \bar{v} \leq \bar{n}_1]; \tag{1}$$

- or labeled with a pair $((p, \bar{q}), (q, \bar{n})) \in B \times \mathit{Confs}$ so that $p \to \bar{q}$, with $\bar{q} \in Q^{k'}$ for some $k' \leq k$, is a branching rule of B, $p = q$ and it has exactly k' children, labeled $(r_1, (p_1, \bar{n}_1)), \ldots, (r_{k'}, (p_{k'}, \bar{n}_{k'}))$ so that $\bar{q} = (p_1, \ldots, p_{k'})$ and

$$\bar{n} = \sum_{i \leq k'} \bar{n}_i \quad [\text{resp. } \bar{n} \leq \sum_{i \leq k'} \bar{n}_i]. \tag{2}$$

Note that leaf nodes are necessarily labeled with rules of the form $q \to \bar{\emptyset} \in B$. Without loss of generality we will assume that the system contains rules $q \to \bar{\emptyset}$ for every state q.

[4] $\mathrm{EX}\varphi = \mathrm{EU}(\varphi, \bot)$, $\mathrm{EF}\varphi = \mathrm{EU}(\varphi, \top)$, $\mathrm{EG}\varphi = \mathrm{EU}(\varphi \wedge \neg\mathrm{EX}\top, \varphi)$, $\mathrm{AU}(\varphi, \psi) = \neg\mathrm{EU}(\neg\psi \wedge \neg\varphi, \neg\psi) \wedge \neg\mathrm{EG}(\neg\varphi)$, etc.

4.2 Merging VASS

We present an extension of the model above where the branching rules, now called *merging rules*, are more powerful: they allow us to reorganize the counters. Whereas in an (incrementing) derivation tree for VASS_k the component i of the configuration of a node depends only on the component i of its children and the rule applied, MVASS_k allows to have transfers *between components*. However, these transfers have some restrictions —otherwise the model would have non-elementary or undecidable coverability/reachability problems [23]. First, transfers between components are 'weak', in the sense that we cannot force a transfer of the whole value of a coordinate i to a distinct coordinate j of a child, we can only make sure that part of it will be transferred to component j and part of it will remain in component i. Second, these weak transfers can only be performed for any pair of coordinates i, j adhering to a partial order, where transfers occur from a bigger component to a smaller one.

A **Merging-VASS** of rank k and dimension n, or $n\mathrm{MVASS}_k$, is a tuple $\mathcal{A} = \langle Q, U, M, \preceq \rangle$, where $\preceq$ is partial order on $\underline{n}$, Q and U are as before, and M is a set of merging rules of the form $(q, S, \bar{q})$ where $q \in Q$, $\bar{q} \in Q^{k'}$ with $k' \leq k$, and $S \subseteq \mathbb{N}^{n\cdot(k'+1)}$ is a linear set of dimension $n \cdot (k'+1)$ of the form $\bar{0} + (B \cup S_0)^*$, where

1. all the elements of B are of the form $(\bar{e}_i, \bar{x}_1, \ldots, \bar{x}_{k'})$, where for each $1 \leq \ell \leq k'$, $\bar{x}_\ell \in \mathbb{N}^n$ is either $\bar{0}$ or $\bar{e}_j$ for some $j \prec i$; and
2. S_0 consists of the following $k' \cdot n$ vectors

$$S_0 = \bigcup_{1 \leq i \leq n} \{(\bar{e}_i, \bar{e}_i, \bar{0}, \bar{0}, \ldots, \bar{0}), (\bar{e}_i, \bar{0}, \bar{e}_i, \bar{0}, \ldots, \bar{0}), \ldots, (\bar{e}_i, \bar{0}, \ldots, \bar{0}, \bar{e}_i)\}. \quad (3)$$

The idea is that in point 1 we allow to transfer contents from component i to components of smaller order. For example, on dimension 3 and rank 2, a vector $\bar{v} = (1,0,0)(0,1,0)(0,0,1)$ in B would imply that during the merge one can transfer a quantity $m > 0$ from component 1 of the father into component 2 of the first child and component 3 of the second child, assuming $2, 3 \prec 1$). On the other hand, point 2 tells us that for every i we can always have some quantity of component i that is *not* transfered to other components, *i.e.*, that stays in component i. Continuing our example, the children configurations $(m, m'+s, t)$ and $(m, s, m'+t)$ can be merged into $(m+m', s, t)$ for every $m, m', s, t \geq 0$, using the vector $\bar{v}$ and S_0.

A derivation tree [resp. incrementing derivation tree] is defined just as before, with the sole difference being that condition (2) is replaced with

$$\begin{aligned} &(\bar{n}, \bar{n}_1, \ldots, \bar{n}_{k'}) \in S \\ &[\text{resp. } (\bar{n}, \bar{n}'_1, \ldots, \bar{n}'_{k'}) \in S \text{ for } (\bar{n}'_1, \ldots, \bar{n}'_{k'}) \leq (\bar{n}_1, \ldots, \bar{n}_{k'})]. \end{aligned} \quad (4)$$

Notice that this is a generalization of VASS_k. Indeed, VASS_k corresponds to the restriction where all the k'-ary merging rules have $S = \bar{0} + S_0^*$ for S_0 as defined in (3). Note that an (incrementing) derivation tree for $n\mathrm{VASS}_k$ is, in particular,

an (incrementing) derivation tree for $n\text{MVASS}_k$. As before, we assume that there are always rules $(q, \emptyset, \bar{\emptyset})$ for every state q.

Jacquemard et al. [19] study an extension of BVASS, 'EBVASS', in relation to the satisfiability of $\text{FO}^2(<, +1, \sim)$ over *unranked* data trees. EBVASS has some features for merging counters. While MVASS and EBVASS are incomparable in computational power, it can be seen that without the restriction $j \prec i$ in condition 1, MVASS would capture EBVASS. In fact, this condition is necessary for the (elementary) decidability of the coverability problem for MVASS, while the status of the coverability problem for EBVASS is unknown.

4.3 Decision problems

Given a counter system $\mathcal{A}$, a set of states $\widehat{Q}$, and a configuration $(q, \bar{n})$ of $\mathcal{A}$, we write $(q, \bar{n}) \rightsquigarrow_{\mathcal{A}} \widehat{Q}$ [resp. $(q, \bar{n}) \rightsquigarrow^+_{\mathcal{A}} \widehat{Q}$] if there exists a derivation tree [resp. incrementing derivation tree] for $\mathcal{A}$ with root configuration $(q, \bar{n})$, so that all the leaves have configurations from $\widehat{Q} \times \{\bar{0}\}$. The reachability and incrementing reachability problems are defined as follows.

Problem:	VASS_k **reachability problem** [resp. VASS_k **incrementing reachability problem**]
Input:	an $n\text{VASS}_k$ $\mathcal{A}$ with states Q, a set of states $\widehat{Q} \subseteq Q$, and a configuration $(q, \bar{n})$ of $\mathcal{A}$.
Output:	'Yes' iff $(q, \bar{n}) \rightsquigarrow_{\mathcal{A}} \widehat{Q}$ [resp. $(q, \bar{n}) \rightsquigarrow^+_{\mathcal{A}} \widehat{Q}$].

Observe that when $k = 1$ this problem is equivalent to the reachability and coverability problems for Vector Addition Systems with States.

The $\textbf{MVASS}_k$ **reachability problem** and $\textbf{MVASS}_k$ **incrementing reachability problem** are defined just as before but considering $\mathcal{A}$ to be an $n\text{MVASS}_k$ instead of a $n\text{VASS}_k$. We will often refer to these problems as Reach( ) and Reach^+( ). We also remark that the incrementing reachability problem is simply a restatement of the *coverability problem*. In particular, it is monotone: if $(q, \bar{n}) \rightsquigarrow^+_{\mathcal{A}} \widehat{Q}$ and $\bar{n}' \geq \bar{n}$ then $(q, \bar{n}') \rightsquigarrow^+_{\mathcal{A}} \widehat{Q}$. We define the **control-state reachability problem** CSReach as the problem of, given $\mathcal{A}, q, \widehat{Q}$, whether $(q, \bar{n}) \rightsquigarrow_{\mathcal{A}} \widehat{Q}$ for some $\bar{n}$. It is easy to see that this problem is equivalent to the problem of whether $(q, \bar{0}) \rightsquigarrow^+_{\mathcal{A}} \widehat{Q}$.

In [8] the coverability problem (or equivalently, the incrementing reachability problem) for a single-state formulation, called BVAS, is studied. A BVAS consists of a tuple $\langle n, R_1, R_2 \rangle$, where R_1 is a set of unary rules, R_2 is a set of binary rules (both rules included in $\mathbb{Z}^n$ which add up a vector). The *size* of a given BVAS is defined as $n\ell$, where ℓ represents the maximum binary size of an entry in $R_1 \cup R_2$.

Proposition 1. *[8] Coverability for* BVAS *is 2ExpTime-complete. If the dimension n is fixed, the problem is in ExpTime.*

4.4 Decidability of Reach$^+$(MVASS)

The arguments used in [8] to prove the previous proposition can be adapted to show a similar result for MVASS: the REACH$^+$ and CSREACH problems are in 3ExpTime.

Theorem 2. REACH$^+$(MVASS$_k$) *and* CSREACH(MVASS$_k$) *are in 3ExpTime for every* $k \geq 1$. *If the dimension* n *is fixed, the problem is in 2ExpTime.*

Proof (idea). Using Lemma 9, we show that if there is an incrementing derivation $\mathcal{D}$ for $(q, \bar{n}) \rightsquigarrow^+_{\mathcal{A}} \widehat{Q}$, where $\mathcal{A} = (Q, U, M, \preceq)$ is an nMVASS$_k$ counter system, then there is a contraction $\mathcal{D}'$ of $\mathcal{D}$ with height bounded doubly-exponentially in the dimension. We show that if a nMVASS$_k$ $\mathcal{A} = (Q, U, M)$ has an incrementing derivation $\mathcal{D}$ for $(q, \bar{n}') \rightsquigarrow^+_{\mathcal{A}} \widehat{Q}$, $\bar{n}' \geq \bar{n}$, then there a contraction $\mathcal{D}'$ of $\mathcal{D}$ which is also an incrementing derivation for $(q, \bar{n}'') \rightsquigarrow^+_{\mathcal{A}} \widehat{Q}$, $\bar{n}'' \geq n$, whose height is bounded by

$$(\max(U^+) + \max \bar{n} + |Q|)^{2^{p(k)}} \tag{5}$$

for a polynomial function $p : \mathbb{N} \to \mathbb{N}$.

The next argument basically follows the schema (i)–(iii) of [8, p.7]. Let $\mathcal{D}$ be an incrementing derivation for $(q, \bar{n}') \rightsquigarrow^+_{\mathcal{A}} \widehat{Q}$, and let π be a root-to-leaf path of $\mathcal{D}$ which is larger than the bound. Let $\mathcal{A}'$ be a kMVASS$_n$ whose set of rules consists of:

- The unary rules $(q, \bar{v}, q')$ contained in the unary rules of π.
- Suppose we have a node x of π with configuration $((q, \bar{n}), (q, S, \bar{q}))$ and with children labeled $((q_1, \bar{n}_1), r_1), \ldots, ((q_s, \bar{n}_s), r_s)$, so that the next element after x in π is the j-th child of x. Further, suppose that this merging rule is preceded by a unary rule; that is, the parent x' of x is labeled with $((p, \bar{n}'), (p, \bar{w}, q))$ —it is not hard to see that without any loss of generality we can always assume that a merging rule is preceded by a unary rule. Let $B = \{\bar{b}_1, \ldots, \bar{b}_m\}$ be the basis of S, that is, $B^* = S$. Let $B' = \{\bar{b}_i \mid \bar{b}_i[h] = 0 \text{ for all } n \cdot (1 + j) + 1 \leq h \leq n \cdot (2 + j)\}$, and $B'' = B \setminus B'$. Note that B' is the set of bases that do not touch the j-th component. We then have

  $$(\bar{n}, \bar{n}_1, \ldots, \bar{n}_s) = \alpha_1 \bar{b}'_1 + \cdots + \alpha_{m'} \bar{b}'_{m'} + \beta_1 \bar{b}''_1 + \cdots + \beta_{m''} \bar{b}''_{m''}$$

  for $B' = \{\bar{b}'_1, \ldots, \bar{b}'_{m'}\}$ and $B'' = \{\bar{b}''_1, \ldots, \bar{b}''_{m''}\}$. Let $\bar{v}' = -(\alpha_1 \bar{b}'_1 + \cdots + \alpha_{m'} \bar{b}'_{m'}) \in \mathbb{Z}^{n \cdot (s+1)}$ and $\bar{v} \in \mathbb{Z}^n$ the restriction of $\bar{v}'$ to the first n components. Note that $\bar{v}$ contains non-positive entries only. Then, produce the unary rule $(p, \bar{w} + \bar{v}, q)$ and a merging rule (q, S', q') so that $S' \subseteq \mathbb{N}^{2n}$ is the restriction of S to the components corresponding to the j-th child.

Note that if we relabel accordingly π we obtain an incrementing derivation for $(q, \bar{n}') \rightsquigarrow^+_{\mathbb{A}'} \widehat{Q}$. Then, by Lemma 9, there is a contraction of π which is still a

correct incrementing derivation for $(q, \bar{n}'') \rightsquigarrow^+_{\mathbb{A}'} \widehat{Q}$ for some $\bar{n}'' \geq \bar{n}$ and so that its length is at most

$$(\max(U'^+) + \max(\bar{n}) + 1)^{2^{p(k)}}$$

where U' is the set of unary rules of $\mathcal{A}'$. Note that $\max(U'^+) \leq \max(U^+)$ because we have only added unary rules with smaller positive entries.

We can then unfold back the subtrees hanging from nodes of π to obtain an MVASS incrementing derivation for $(q, \bar{n}'') \rightsquigarrow^+_{\mathbb{A}} \widehat{Q}$ whose number of leaves at height greater than the bound (10) has decreased in at least 1. Repeating the same argument a finite number of times we obtain an incrementing derivation for $(q, \bar{n})$ of height bounded by (10).

Thus, to decide the incrementing reachability problem, it suffices to search for a derivation of doubly-exponential height, whose vectors may contain triply-exponential entries in principle. As a consequence of this, the verification of the existence of such a derivation can be performed in alternating double exponential space, as it is shown in [8, Theorem 8], and thus the incrementing reachability for MVASS is in 3ExpTime. See Appendix A for more details.

If n is fixed, the height of the witnessing derivation becomes singly exponential and thus the problem is in 2ExpTime (as explained in [8, Theorem 8]). □

5 Satisfiability of LRV$^{\mathrm{D}}$ on data trees

We call SAT_k the satisfiability problem on finite k-ranked data-trees. The main result of this section is the following.

Theorem 3. *SAT_k-$\mathrm{LRV}^{\mathrm{D}}_n$ is ExpSpace-reducible to* $\textsc{CSReach}(n\mathrm{VASS}_k)$.

In the proof of the theorem, the number of attribute variables of the formula will become the dimension of the VASS_k. Since the CSReach problem for VASS_k is decidable in 2ExpTime, this yields a decidable procedure for SAT_k-$\mathrm{LRV}^{\mathrm{D}}$ for every k. For the case $k = 1$, *i.e.*, on *data words*, it has been shown [7] that there is a reduction from SAT_1-LRV_n to $\textsc{CSReach}(2^n\text{-}\mathrm{VASS}_1)$, where the dimension of the VASS_1 is exponential in the number of variables. However, it is easy to see that the proof of [7] also yields a reduction from SAT_1-$\mathrm{LRV}^{\mathrm{D}}_n$ to $\textsc{CSReach}(n\mathrm{VASS}_1)$. Thus, this theorem has been shown for $k = 1$, and here we generalize it to $k > 1$. However, there are a number of problems that appear if one tries to "extend" the proof of [7] to the branching setup. In particular, the non-linearity of the future in addition to the possibility of having a data value repeating at several descendants in different variables, calls for a non-standard way of propagating the values of configurations, which is not contemplated in VASS_k. This is why we are only able to show the reduction for the 'disjoint' fragment $\mathrm{LRV}^{\mathrm{D}}$, and which leads us to consider the extended model MVASS_k in Section 7. This propagation problem does not appear when one only considers that the classes of different values are disjoint, that is, that all formulas of the type $v \star \mathrm{EF} w$ with $\star \in \{\approx, \not\approx\}$ have $v = w$, motivating the study of SAT_k-$\mathrm{LRV}^{\mathrm{D}}_n$.

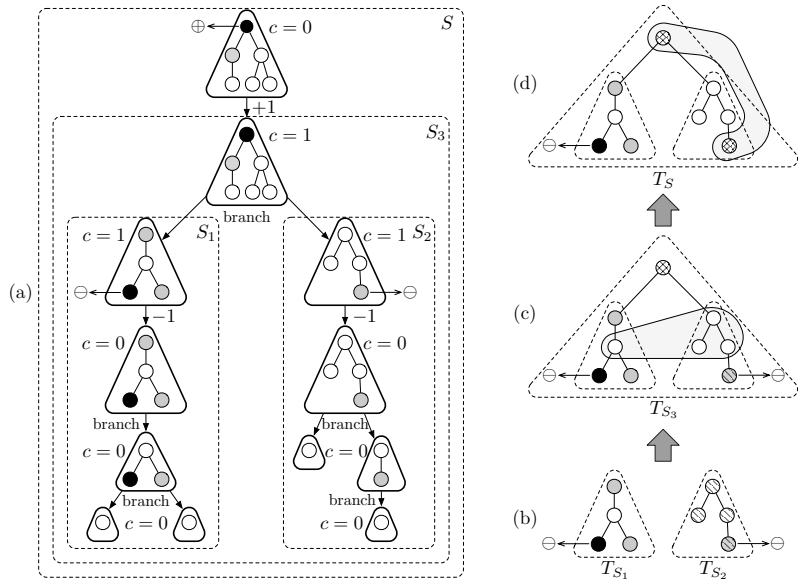
Proof idea. We start by analyzing a restricted case, which serves as building block: the logic $\mathrm{LRV}_1^{\mathrm{D}-}$ whose only formulas are conjuncts of terms of the form $v \star \mathrm{EX}^i v$, $v \star \mathrm{EF} v$, or their negation, where $\star \in \{\approx, \not\approx\}$. We show that for any formula φ of $\mathrm{LRV}_1^{\mathrm{D}-}$, there is a $1\mathrm{VASS}_k$ $\mathcal{A}_\varphi^k = \langle Q, U, B \rangle$, a set of initial states $Q_0 \subseteq Q$, and a set of final states $\widehat{Q} \subseteq Q$ such that $\mathrm{SAT}_k(\varphi)$ iff there is a derivation tree with a starting node in $q_0 \in Q_0$ that is a solution to $\textsc{CSReach}(\mathcal{A}_\varphi^k, q_0, \widehat{Q})$—it is easy to see that this problem is equivalent to CSReach as stated in Section 4.3. We then extend this construction to the automaton $\mathcal{B}_\varphi^k$, enabling a reduction from the full logic $\mathrm{LRV}_1^{\mathrm{D}}$, but still restricted to only one variable. Finally, because of the disjointness of the variables, it is easy to extend these constructions to the full logic $\mathrm{LRV}_n^{\mathrm{D}}$. See Appendix B for the full proof.

Here we only give a brief explanation of the construction of $\mathcal{A}_\varphi^k = \langle Q, U, B \rangle$ for the logic $\mathrm{LRV}_1^{\mathrm{D}-}$. For the sake of simplicity, we assume our logic has no labels; their addition to the construction is straightforward. Since LRV can only deal with data (in)equality and since in this case we consider $n = 1$, we will interchangeably speak of an equivalence relation between the nodes of the tree or of the particular data values.

We define the EX-length of a formula ψ as the maximum i such that ψ contains a subformula of the form $v \star \mathrm{EX}^i v$. Let d be the EX-length of φ. The set Q consists of all valid (d, k)-frames, where a (d, k)-frame is a tree of depth d and rank k, equipped with an equivalence relation, and with some extra attributes (node-labeling functions) to include some special marks and semantic information of future requirements of the form $(\neg)v \approx \mathrm{EF} v$ and $(\neg)v \not\approx \mathrm{EF} v$. The initial states Q_0 are those frames F satisfying the *local* part of φ (that is, subformulas of φ the form $v \star \mathrm{EX}^i v$). *Future requirements* (that is, subformulas of φ of the form $v \star \mathrm{EF} v$) may not be satisfied locally in F. The set $\widehat{Q}$ is the singleton with a frame consisting in a single node. The basic idea is that the counter of $\mathcal{A}_\varphi^k$ keeps track of how many future requirements are not yet satisfied. Some nodes of the frames may have extra information in the form of labels $\oplus$ or $\ominus$. States F whose root is labeled with $\oplus$ are *points of increment*: $\mathcal{A}^k$ will have unary rule in U that increments the (sole) counter in 1. A point of increment denotes that some subformula $v \approx \mathrm{EF} v$ of φ should hold, but it is not satisfied *locally*, that is, inside F. Leaves with $\ominus$ are those not related to ancestors in the frame with the same data value, they can thus be "joined" into the same equivalence class to a future requirement originated at some distant ancestor. States with leaves $\ominus$-labeled are *points of decrement*: $\mathcal{A}^k$ will have a unary rule in U to decrement the counter depending on the number of equivalence classes of leaves labeled with $\ominus$. The branching rules B of $\mathcal{A}^k$ are of the form $F \to (F_1, \ldots, F_i)$, where F_j overlaps with an adequate part of F.

Example 4. The following figure illustrates a scheme of an incrementing derivation S of the $1\mathrm{VASS}_2$ $\mathcal{A}_\varphi^2$ (a) and some steps (b, c and d) in the bottom up construction of the data tree T_S satisfying φ, for $\varphi = \neg v \approx \mathrm{EX} v \wedge \neg v \approx \mathrm{EX}^2 v \wedge v \approx \mathrm{EF} v$. Triangles represent $(2, 2)$-frames. Shades of gray represent the equivalence classes, which only make sense inside any frame. The counter is notated with c, and

arrows represent the (unary/branching) transitions of the derivation. Notice that the top branching is 'incremental', and that the local requirements of φ (namely, $\neg v \approx \mathrm{EX}v$ and $\neg v \approx \mathrm{EX}^2 v$) are satisfied in the root of the top frame.



The construction of T_S is bottom-up, and we show three steps: (a), (b) and (c). Notice that in (b) each of T_{S_1} and T_{S_2} has its own partition (no intersection). In (c) we process the root of S_3 by tying together T_{S_1} and T_{S_2} with a common parent, who lives in a single class of the partition. Notice that the partitions of T_{S_1} and T_{S_2} are properly joined (grey area), according to the information in the root of S_3. Finally in (d) we construct T_S. The root of S is a point of increment, so we match $\oplus$ with some $\ominus$ in T_{S_3}. In this case, we match it with the right-hand $\ominus$, and so we join them by putting them in the same partition (grey area). We have satisfied the future requirement $v \approx \mathrm{EF}v$ of φ.

On the one hand, any incrementing derivation S that is a solution to $\textsc{CSReach}(\mathcal{A}^k_\varphi, q_0, \widehat{Q})$ for some $q_0 \in Q_0$, can be translated into a data tree T_S whose root satisfies φ. In fact, any semantic information contained in the labels of nodes in frames of S will be satisfied in the corresponding nodes of T_S. The difficult part is to show that $\ominus$-leaves will have the necessary conditions to be joined with the equivalence class of an $\oplus$-ancestor, making true the formula $v \approx \mathrm{EF}v$. This will be a consequence of the fact that the incrementing derivation satisfies CSReach.

On the other hand, if φ is satisfiable in some k-ranked data tree T then we can build an incrementing derivation tree S_T that is a solution to $\textsc{CSReach}(\mathcal{A}^k_\varphi, q_0, \widehat{Q})$ for some $q_0 \in Q_0$. Following the ideas of the previous part, we proceed from the root toward the leaves using the structure and equivalence classes of T to determine in each step the corresponding states (including the semantic labels

and the labels $\ominus$, $\oplus$) and rules of S_T. From the construction, and using the incrementing nature of the derivation, it will follow that S_T is a solution to the control-state reachability problem.

For the construction of $\mathcal{B}^k_\varphi$, the information given by the (d,k)-frames will be supplemented by the addition of sets of formulas containing information about the EU operator and the Boolean connectives. The way to do this is standard (see *e.g.*, [6]).

Complexity. Let $\mathrm{LRV}^{\mathrm{D}}_{n,d}$ be the fragment of $\mathrm{LRV}^{\mathrm{D}}_{n}$ where each formula has EX-length at most d. By inspecting the above reduction, we can bound the number of states of the constructed $n\mathrm{VASS}_k$ by $O(p(n)^{k^{d+1}} \cdot (k^{d+1})^{k^{d+1}} \cdot 2^{p(|\varphi|)})$ for some polynomial p, and we can bound the maximum value among the entries in unary rules by k^d. Furthermore, we can reduce our branching VASS to an equivalent (single-state) BVAS with an addition of a constant number of new dimensions. This transformation increases the binary size of the maximum entry of the unary rules at most logarithmically over the number of states of our original $n\mathrm{VASS}_k$ (see Appendix B.4 for more details). Now, using Proposition 1, Theorem 3, and the above complexity analysis, we obtain:

Proposition 5. *SAT_k-$\mathrm{LRV}^{\mathrm{D}}_{n,d}$ is in ExpTime for fixed k, n, d; it is in 2ExpTime for fixed k, n or fixed d, k; and it is in 3ExpTime for fixed k.*

6 Obtaining equivalence with VASS_k

In the previous section we have seen a reduction into the control-state reachability problem for VASS_k. A natural question is whether there exists a reduction in the other direction: can $\textsc{CSReach}(\mathrm{VASS}_k)$ be reduced into the k-satisfiability for $\mathrm{LRV}^{\mathrm{D}}$? For the case $k = 1$, this has been shown to be the case [7]: there exists a polynomial-space reduction from $\textsc{CSReach}(\mathrm{VASS}_1)$ to $\mathrm{SAT}_1(\mathrm{LRV})$.

The existence of a reduction would show, intuitively, that one can express in the logic that there is a tree that verifies all the conditions for being a derivation. Without the use of data tests, one can easily encode trees that verify all the conditions except perhaps (1) and (2) regarding the vectors. For this, let us assume without loss of generality that all unary rules contain a vector $\bar{e}_i$ or $-\bar{e}_i$. The data values are used to ensure the next two conditions:

- Along any branch, every node containing a rule of the form $q \xrightarrow{\bar{e}_i} q'$ has a unique data value. In other words, we cannot find two nodes encoding an increment of component i with the same data value so that one is the ancestor of the other.
- For every node with a unary rule $q \xrightarrow{\bar{e}_i} q'$ there exists a descendant with a rule $p \xrightarrow{-\bar{e}_i} p'$ and the same data value.

These two conditions imply that after incrementing component i there must be *at least* one corresponding decrement of component i. Note that there could be *more* decrements than increments, which is not a problem since we work under the 'incrementing' semantics.

Interestingly, these two conditions can be expressed in LRV, but we do not know how to encode it in $\mathrm{LRV}^{\mathrm{D}}$ (we conjecture that they are not expressible).

Adding the operator $\mathbf{AG}_{\approx v}(\varphi)$. We add a new operator $\mathrm{AG}_{\approx v}(\varphi)$ to $\mathrm{LRV}^{\mathrm{D}}$, where $T, x \models \mathrm{AG}_{\approx v}(\varphi)$ if every descendant of x with the same v-attribute verifies φ. The fragment of $\mathrm{LRV}^{\mathrm{D}}_n$ extended with positive occurrences of $\mathrm{AG}_{\approx v}(\varphi)$ (that is, where $\mathrm{AG}_{\approx}$ occurs always under an even number of negations) is called $\mathrm{LRV}^{\mathrm{D}}_n(\mathrm{AG}^{+}_{\approx})$.

Now, in $\mathrm{LRV}^{\mathrm{D}}_n(\mathrm{AG}^{+}_{\approx})$ one can express: *for every node x containing a rule $q \xrightarrow{\bar{e}_i} q'$, we have that all descendants of x with the same v_i attribute contain a rule of the form $p \xrightarrow{-\bar{e}_i} p'$*. This, added to the property that every increment for component i must verify $v_i \approx \mathrm{EF} v_i$, ensures that the tree indeed encodes a derivation tree.

Theorem 6. $\textsc{CSReach}(n\mathrm{VASS}_k)$ *is PTime-reducible to* SAT_k-$\mathrm{LRV}^{\mathrm{D}}_1(\mathrm{AG}^{+}_{\approx})$.

Proof (idea). We show the idea for $n = 1$, as this case generalizes to any n straightforwardly, and without changing the number of variables in the logic. For every $1\mathrm{VASS}_k$ $\mathcal{C}^k = \langle Q, U, B\rangle$, $q_0 \in Q$ and $\widehat{Q} \subseteq Q$ we define $\varphi \in \mathrm{LRV}^{\mathrm{D}}_1(\mathrm{AG}^{+}_{\approx})$, such that $\mathrm{SAT}_k(\varphi)$ iff $\textsc{CSReach}(\mathcal{C}^k_\varphi, q_0, \widehat{Q})$. We want this φ to force various properties in all its models, so that every model corresponds to a derivation tree of $\textsc{CSReach}(\mathcal{C}^k_\varphi, q_0, \widehat{Q})$. In particular, we want:

- Each node is labeled with either a rule of $U \cup B$ or an extra label $*$ for dummy nodes that will be ignored (this is to force exact k-branching for all non-leaves). We can assume without loss of generality that all unary rules in U of the form $q \xrightarrow{c} q'$ have either $c = 1$ or $c = -1$. In particular, formulas φ_{inc} and φ_{dec} express that the label is an increment or a decrement rule, respectively.
- If a node is labeled with an empty rule $q \to \bar{\emptyset}$, then it is a leaf.
- The root is labeled with a rule of the form $(q_0, \dots) \in U \cup B$.
- Each node labeled with an increment rule has a descendant in the same equivalence class (*i.e.* with same value for the only attribute v), and all its descendants in the same equivalence class are labeled with a decrement rule: this can be expressed by $\varphi_{\mathrm{inc}} \to \big(v \approx \mathrm{EF} v \wedge \mathrm{AG}_{\approx v}(\varphi_{\mathrm{dec}})\big)$.

All the above properties, except the last one, can be expressed in $\mathrm{LRV}^{\mathrm{D}}_1$; for the last one we use (positively) $\mathrm{AG}_{\approx}$. The final formula φ consists of a conjunction of all these properties, among others (so that the occurrence of $\mathrm{AG}_{\approx}$ remains positive). Then one verifies that a solution to the control-state reachability problem of $\mathcal{C}^k, q_0, \widehat{Q}$ can be used to construct a model for φ; and that a data tree satisfying φ can be used to construct an incrementing derivation tree for $\textsc{CSReach}(\mathcal{C}^k, q_0, \widehat{Q})$. See Appendix C for more details. □

The satisfiability for this extension still has a reduction to the control-state reachability for VASS_k:

Theorem 7. SAT_k-$\mathrm{LRV}^{\mathrm{D}}(\mathrm{AG}^{+}_{\approx})$ *is ExpSpace-reducible to* $\textsc{CSReach}(\mathrm{VASS}_k)$.

7 From LRV to MVASS$_k$

The reduction from LRV$^{\mathrm{D}}$ to VASS$_k$ from Section 5 cannot be extended to treat LRV. The main problem is that the branching nature of the counters in a CSReach(VASS$_k$) will be insufficient to represent some classes of data trees (which can be needed to model some formulas). When we have tests of the form $u_1 \approx \mathrm{EF} u_2$ with $u_1 \neq u_2$ distinct variables, we can no longer reason in terms of "one coordinate i for each variable u_i", where the i-th component in the configuration of the VASS$_k$ codes, intuitively, how many distinct data values must be seen on variable u_i in the subtree as shown in Section 5. In fact, when working with LRV, a data value may appear in *several* variables, as a result of allowing formulas like $u_1 \approx \mathrm{EF} u_2 \wedge u_1 \approx \mathrm{EF} u_3$. This means that we need to reason in terms of *sets* of variables, where each component i is associated with a non-empty subset U_i of the variables appearing in the input formula φ; this time, component i counts how many data values must appear in the subtree under all the variables of U_i. This, in principle, poses no problem for the non-branching case: in fact, this kind of coding (indexing one coordinate of the configuration for each subset of variables) was used in [6] to show a reduction from LRV to VASS on data words. However, on data trees, this coding breaks with the semantics of the branching rules of VASS$_k$.

As an example, suppose we work with two variables u, v and we thus have dimension 3 —the first component is associated with $\{u\}$, the second with $\{v\}$ and the third with $\{u, v\}$. Suppose that there are n ancestor nodes that have to satisfy both $u \approx \mathrm{EF} u$ and $u \approx \mathrm{EF} v$, which at the current configuration of the VASS$_k$ is witnessed by the vector $(0, 0, n)$. Intuitively, this means that there are n data values that must appear in the subtree under a variable u and also under v (though not necessarily at the same node) in the data tree the automaton is trying to find. Hence, as part of the "branching" instruction of this configurations into the configuration of the left and right children, one must contemplate the possibility of obtaining, for instance, $(n, 0, 0)\,(0, n, 0)$, saying that the left subtree contains n distinct data values for u, and the right child contains n data values for v. But it could be $(n - t, 0, t)\,(0, n - t, 0)$, or $(0, 0, n - t)\,(0, 0, t)$, etc. In other words, components need to be mixed in a more complex way that is not allowed in VASS$_k$ branching rules. In particular, some sort of transfers between coordinates must be necessary. This is precisely the behavior that we can encode into MVASS.

Theorem 8. *SAT_k-LRV$_n$ is reducible to* CSReach(2^n-MVASS$_k$).

Proof (idea).

Using the merging rules as described in Section 7, the reduction from LRV$^{\mathrm{D}}$ to VASS$_k$ of Section 5 and Appendix B can be modified to obtain a reduction from LRV to MVASS$_k$. Frames and its notion of validity are extended to treat set of variables. In particular, now the points of increment and decrement are always relative to a set of variables. This follows, very roughly, the idea of coding from [7] in the setup built in Section 5, but now some special care must be considered because of the non-linearity of a tree. One must decide in advance to which leaf

of the frame the satisfaction of data demands will be delegated. The resulting MVASS_k now has dimension *exponential* in the number of variables of the input formula. Concretely, in order to encode this logic we need to make the following changes to the set of frames $\mathcal{F}_{d,k}$ we work with.

First of all, the labelling function ℓ_1 now labels pairs of *sets* of formulas. These formulas labelled by ℓ_1 are of the form

- in the first component $u \approx \mathrm{EF}v$ or $\neg(u \approx \mathrm{EF}v)$
- in the second component $u \not\approx \mathrm{EF}v$ or $\neg(u \not\approx \mathrm{EF}v)$

for any pair of variables u, v used in the input formula. For simplicity, we write $\psi \in \ell_1(x)$ (or, alternatively, that x is ℓ_1-labelled with ψ) to denote that ψ is either in the first or second component of $\ell_1(x)$.

Further, instead of having one equivalence relation $\equiv$ over the set of nodes, we have an equivalence relation $\equiv$ over pairs (x, u) where x is a node of hte frame and u an attribute variable of the input formula φ. This is to account for the possibility that different attributes can have the same data value.

In light of this, the formulas labeled by ℓ_1 must 'respect' $\equiv$. That is, if $u \approx \mathrm{EF}v \in \ell_1(x)$ [resp. $u \not\approx \mathrm{EF}v \in \ell_1(x)$] and $(x, u) \equiv (x, u')$ then $u' \approx \mathrm{EF}v \in \ell_1(x)$ [resp. $u' \not\approx \mathrm{EF}v \in \ell_1(x)$].

More importantly, the labelling ℓ_2 must be changed to reflect the fact that

(1) there may be several demands for the same attribute, as a result of formulas like $u \approx \mathrm{EF}v \wedge u' \approx \mathrm{EF}v$ (as we will see next, this is the reason for the first parameter of $\oplus$),
(2) there may be several attributes in a demand for equality, as a result of formulas like $u \approx \mathrm{EF}v \wedge u \approx \mathrm{EF}v'$,
(3) a point of decrement needs to be a point that has some attributes U which are not connected by equality to any 'local' ancestor and they are connected possibly to some other attributes V in the descendants.

Formally, the mapping ℓ_2 now labels nodes with $\oplus(U, V)$ and/or $\ominus(U, V)$, where U, V are sets of attribute variables. Each node x can receive more than one $\oplus$ or $\ominus$ label, that is, ℓ_2 is a function from nodes to subsets of $\{\oplus(U, V) \mid U, V \subseteq \mathbb{V}\} \cup \{\ominus(U, V) \mid U, V \subseteq \mathbb{V}\}$, assuming $\mathbb{V}$ is the set of variables used in the input formula.[5] The idea is that $\oplus(U, V)$ holding at x means that there must be a data value appearing in the subtree at x under all the variables of V (possibly at different nodes), which is equal to the u-attribute of the k-ancestor of x, for every $u \in U$. On the other hand, $\ominus(U, V)$ holding at x means that the data value of the U-attributes of x (which are all the same) do no not appear in any i-ancestor of x ($i \leq k$), and they will appear in the future with attributes V.

We add the following conditions.

- For any two labels $\oplus(U, V)$ and $\oplus(U', V')$ at the same node, U and U' are disjoint. For any two labels $\ominus(U, V)$ and $\ominus(U', V')$ at the same node, U and U' are disjoint.

[5] It is worth remarking that $\ell_2(x)$ is always a set of size linear in $|\mathbb{V}|$ due to the next conditions.

- For every leaf x which is ℓ_2-labeled with $\oplus(U,V)$ we have that U is an equivalence class of $\{(u,v) \mid (r,u) \equiv (r,v)\}$, where r is the root node.
- For every leaf x which is ℓ_2-labeled with $\ominus(U,V)$ we have that for some $v \in \mathbb{V}$ we have

$$U = \{u \mid (x,u) \equiv (x,v)\},$$
$$V = \{u \mid [v \approx \mathrm{EF}u] \in \ell_1(x)\},$$

  and that there is no ancestor y of x so that $(x,u) \equiv (y,v')$ for some $u \in U$, $v' \in \mathbb{V}$.
- There exists an ℓ_1-labelling $u \approx \mathrm{EF}v$ holding at the root r if, and only if, there exists a node x at some depth i so that either
  - $(r,u) \equiv (x,v)$, or
  - $(r,u) \equiv (x,v')$ for some v' and $v' \approx \mathrm{EF}v$ in $\ell_1(x)$, or
  - $i = k$ and x is ℓ_2-labeled with $\oplus(U,V)$ with $U = \{u' \mid (r,u) \equiv (r,u')\}$ and $v \in V$.
- There exists an ℓ_1-labelling $u \not\approx \mathrm{EF}v$ holding at the root r if, and only if, there exists a node x at some depth i so that either
  - $(r,u) \not\equiv (x,v)$, or
  - $(r,u) \equiv (x,v)$ and $v \not\approx \mathrm{EF}v$ in $\ell_1(x)$.

1-step consistency is preserved as before. Now a *point of increment for* V is a frame whose root is labeled with $\oplus(U,V)$ for some U; whereas a *point of decrement for* W is a frame whose root is labeled with $\ominus(U,V)$ for $U \cup V = W$.

Finally, the automaton $\mathcal{A}^k_\varphi$ is built as for the other reduction, with the exception that now its dimension is exponential. As in the previous reductions, the frames are the states of the automaton, where the initial frames are those that do not contain $\oplus/\ominus$ tags at nodes at distance $< k$ from the root, and whose root labeling is consistent with the satisfaction of the formula. In the automaton, we have one coordinate associated with every non-empty subset $V \subseteq \mathbb{V}$ of attribute variables (remember that we use $\bar{e}_V$ to denote $\bar{e}_i$ for the coordinate i associated with V and $\bar{e}_\emptyset$ to denote $\bar{0}$). Unary rules now follow a logic of first decrementing all the $\ominus$ at the root, and then incrementing all the $\oplus$ at the root. (The $\oplus/\ominus$ tags of other nodes are differed to the moment when they will be at the root of a frame.) That is, unary rules $(F_1, -\bar{e}_{U\cup V}, F_2)$ whenever F_1 has $\ominus(U,V)$ at the root, F_2 is just like F_1 but without the $\ominus(U,V)$ at the root. We have unary rules $(F_1, \bar{e}_V, F_2)$ whenever F_1 has no $\ominus$-labels at the root, it has a $\oplus(U,V)$ label at the root, and F_2 is the result of removing $\oplus(U,V)$ at the root. Merging rules are built as it was explained before. That is, we have $(F_1, \bar{0} + B^*, (F'_1, \ldots, F'_{k'}))$ whenever F_1 is 1-consistent with $(F'_1, \ldots, F'_{k'})$, and B consists of all vectors $(\bar{e}_V \bar{e}_{U_1} \cdots \bar{e}_{U_k})$, so that $V \neq \emptyset$ and $V = \bigcup_i U_i$. The partial order $\preceq$ will then be the subset ordering on the components: $i \preceq j$ if the set associated to i is contained in that associated to j. It is not hard to check that if $\mathcal{A}^k_\varphi$ has a solution for CSReach if and only if φ is satisfiable on k-ranked data trees using precisely the same ideas as in the proof of Proposition 13. (See Appendix E.) □

As a corollary, due to Theorem 2, we have that SAT_k-LRV is decidable. We remark that, similarly as done in [7], one can add formulas of the form $u \star \mathrm{EF}[\varphi]v$ stating that there is a descendant witnessing $u \star \mathrm{EF}v$ and verifying φ, while preserving this reduction.

8 Discussion

We have shown connections between counter systems and data logics on ranked data trees. In particular, this has yielded decision procedures for data logics and a new model of branching computation of VASS.

While in the present work the focus has been put on *ranked* data trees, we envisage working also on unranked trees in the future. In particular, we remark that these logics can be naturally extended to the unranked case, but that there are no well-known models of branching counter systems with *unbounded* branching. This may lead to new natural models featuring some sort of unbounded parallel computations with good computational properties.

We are also interested in considering other modalities in our logics, with branching tests such as $\mathrm{EX}^i v \star \mathrm{EF}u$ and $\mathrm{EF}u \approx \mathrm{EF}v$, or tests including past such as $u \approx \mathrm{EF}^{-1}v$ and $\mathrm{EF}^{-1}u \approx \mathrm{EF}v$.

We were unable to show the precise complexity of $\mathrm{CSReach}(\mathrm{MVASS}_k)$, which lies between 2ExpTime and 3ExpTime. We leave this for future work. We believe that SAT_k-$\mathrm{LRV}(\mathrm{AG}^{+}_{\approx})$ is equivalent to the control-state reachability problem for MVASS_k, in the sense of existence of computable reductions from and to.

References

1. David Baelde, Simon Lunel, and Sylvain Schmitz. A sequent calculus for a modal logic on finite data trees. In *25th EACSL Annual Conference on Computer Science Logic, CSL 2016, August 29 - September 1, 2016, Marseille, France*, pages 32:1–32:16, 2016.
2. Mikoaj Bojańczyk, Anca Muscholl, Thomas Schwentick, and Luc Segoufin. Two-variable logic on data trees and XML reasoning. *JACM*, 56(3):13, 2009.
3. Mikołaj Bojańczyk, Claire David, Anca Muscholl, Thomas Schwentick, and Luc Segoufin. Two-variable logic on data words. *ACM Trans. Comput. Log.*, 2010.
4. Benedikt Bollig, Aiswarya Cyriac, Paul Gastin, and K Narayan Kumar. Model checking languages of data words. In *FoSSaCS*, pages 391–405. Springer, 2012.
5. Stéphane Demri, Deepak D'Souza, and Régis Gascon. Decidable temporal logic with repeating values. In *LFCS*, volume 4514 of *LNCS*, pages 180–194. Springer, 2007.
6. Stéphane Demri, Deepak D'Souza, and Régis Gascon. Temporal logics of repeating values. *J. Log. Comput.*, 22(5):1059–1096, 2012.
7. Stéphane Demri, Diego Figueira, and M. Praveen. Reasoning about data repetitions with counter systems. In *LICS*, pages 33–42. IEEE Press, 2013.
8. Stéphane Demri, Marcin Jurdziński, Oded Lachish, and Ranko Lazić. The covering and boundedness problems for Branching Vector Addition Systems. *J. Comput. Syst. Sci.*, 79(1):23–38, 2013.

9. Stéphane Demri and Ranko Lazić. LTL with the freeze quantifier and register automata. *ACM Trans. Comput. Log.*, 10(3), 2009.
10. E Allen Emerson and Joseph Y Halpern. sometimes and not never revisited: on branching versus linear time temporal logic. *JACM*, 33(1):151–178, 1986.
11. Diego Figueira. Forward-XPath and extended register automata on data-trees. In *ICDT*. ACM, 2010.
12. Diego Figueira. Alternating register automata on finite data words and trees. *Log. Methods Comput. Sci.*, 8(1), 2012.
13. Diego Figueira. Decidability of downward XPath. *ACM Trans. Comput. Log.*, 13(4), 2012.
14. Diego Figueira. On XPath with transitive axes and data tests. In *PODS*, pages 249–260. ACM, 2013.
15. Diego Figueira, Santiago Figueira, and Carlos Areces. Basic model theory of XPath on data trees. In *ICDT*, pages 50–60. ACM, 2014.
16. Diego Figueira and Leonid Libkin. Pattern logics and auxiliary relations. In *CSL-LICS*, pages 40:1–40:10, 2014.
17. Diego Figueira and Luc Segoufin. Future-looking logics on data words and trees. In *International Symposium on Mathematical Foundations of Computer Science*, pages 331–343. Springer, 2009.
18. Diego Figueira and Luc Segoufin. Bottom-up automata on data trees and vertical XPath. In *STACS*, volume 9 of *LIPIcs*, pages 93–104. LZI, 2011.
19. Florent Jacquemard, Luc Segoufin, and Jerémie Dimino. FO2$(<, +1, \sim)$ on data trees, data tree automata and branching vector addition systems. *Logical Methods in Computer Science*, 12(2), 2016.
20. Marcin Jurdziński and Ranko Lazić. Alternating automata on data trees and XPath satisfiability. *ACM Trans. Comput. Log.*, 12(3):19, 2011.
21. Ahmet Kara, Thomas Schwentick, and Thomas Zeume. Temporal logics on words with multiple data values. In *FST&TCS*, 2010.
22. O. Kupferman and M. Vardi. Memoryful Branching-Time Logic. In *LICS*, pages 265–274. IEEE Press, 2006.
23. Ranko Lazić and Sylvain Schmitz. Nonelementary complexities for branching VASS, MELL, and extensions. *ACM Trans. Comput. Log.*, 16(3):20, 2015.
24. A. Lisitsa and I. Potapov. Temporal logic with predicate λ-abstraction. In *TIME*, pages 147–155. IEEE Press, 2005.
25. Frank Neven, Thomas Schwentick, and Victor Vianu. Finite state machines for strings over infinite alphabets. *ACM Trans. Comput. Log.*, 5(3):403–435, 2004.
26. Amir Pnueli. The temporal logic of programs. In *FOCS*, pages 46–57. IEEE Press, 1977.
27. Charles Rackoff. The covering and boundedness problems for Vector Addition Systems. *Theoret. Comput. Sci.*, 6(2):223–231, 1978.
28. Luc Segoufin. Automata and logics for words and trees over an infinite alphabet. In *CSL*, pages 41–57. Springer, 2006.
29. Kumar Neeraj Verma and Jean Goubault-Larrecq. Karp-Miller trees for a branching extension of VASS. *Discrete Math. Theor. Comput. Sci.*, 7(1):217–230, 2005.

A Proof of Theorem 2

Preliminaries In this section we work with a slightly different version of the incrementing reachability problem which can easily be shown to be equivalent. This will simplify some of our arguments.

Problem: MVASS_k incrementing reachability problem
Input: a $n\text{VASS}_k$ $\mathcal{A}$ with states Q, a set of states $\widehat{Q} \subseteq Q$, and a configuration $(q, \bar{n})$ of $\mathcal{A}$.
Output: 'Yes' iff $(q, \bar{n}') \rightsquigarrow^+_{\mathcal{A}} \widehat{Q}$ for some $\bar{n}' \geq \bar{n}$.

The difference is that we look for an incrementing derivation of a 'bigger' configuration $(q, \bar{n}')$ than the one $(q, \bar{n})$ received as input. It is straightforward to see that this is essentially the same problem.

We say that a tree $\mathcal{D}$ is an incrementing derivation tree for $(q, \bar{n}) \rightsquigarrow^+_{\mathcal{A}} \widehat{Q}$ if $\mathcal{D}$ is an incrementing derivation that is a witness for $(q, \bar{n}) \rightsquigarrow^+_{\mathcal{A}} \widehat{Q}$. Throughout this section we write 'derivation' as short for 'incrementing derivation tree'. Given a derivation $\mathcal{D}$ and a node x thereof, we write $\rho_{\mathcal{D}}(x)$ to denote the vector of the configuration at x and $\sigma_{\mathcal{D}}(x)$ to denote its state. We will usually write ϵ to denote the node at the root of $\mathcal{D}$. We adapt the main concepts of the 2ExpTime proof for $\text{Reach}^+(\text{VASS}_k)$ of [8] to our setup. A **contraction** of a derivation $\mathcal{D}$ is the result of applying a finite number of times the following operation. Let x be a node of $\mathcal{D}$ with configuration $(q, \bar{n})$ and x' a descendant of x with configuration $(q', \bar{n}')$ so that $q = q'$. Consider the result of

- replacing the subtree at x with the subtree at x' (*i.e.*, removing all descendants of x which are not descendants of x' and identifying x' with x);
- for every ancestor y of x with configuration $(p, \bar{m})$, replacing the configuration with $(p, \bar{m} + (\bar{n}' - \bar{n}))$.

We denote this substitution with $\mathcal{D}[x \leftarrow x']$. We say that a configuration $(q, \bar{n})$ is **bigger** than a configuration $(p, \bar{m})$ if $p = q$ and $\bar{n} \geq \bar{m}$. Note that if $\mathcal{D}$ is a derivation for $(q, \bar{n}) \rightsquigarrow^+_{\mathcal{A}} \widehat{Q}$ and the configuration at node x' is bigger than the configuration at an ancestor x of x', then $\mathcal{D}[x \leftarrow x']$ is a derivation for $(q, \bar{n}') \rightsquigarrow^+_{\mathcal{A}} \widehat{Q}$ for some $\bar{n}' \geq \bar{n}$. Thus, if $\mathcal{D}$ is a witness for the incrementing reachability problem instance $(\mathcal{A}, \widehat{Q}, (a, \bar{n}'))$, so is $\mathcal{D}[x \leftarrow x']$. (Indeed, this is true both when $\mathcal{A}$ is a VASS or a MVASS.) For a set of unary rules $U \subseteq Q \times \mathbb{Z}^k \times Q$, let $\max(U^+) \in \mathbb{N}$ be the maximum positive value contained in unary rules; that is,

$$\max\big(U^+\big) \stackrel{def}{=} \max(\{0\} \cup \{\bar{v}[i] \mid (q, \bar{v}, q') \in U, i \in \underline{k}, \bar{v}[i] > 0\}\big).$$

For a derivation of $\mathcal{D}$ of a $k\text{MVASS}_n$ $\mathcal{A}$, and a set $I \subseteq \underline{k}$, we define the **restriction to** I of $\mathcal{D}$, and we note it $\mathcal{D}|_I$, as the result of

- replacing each configuration $(q, \bar{n})$ of a node with $(q, \bar{n}[I])$, where $\bar{n}[I] \in \mathbb{N}^{|I|}$ is the restriction of $\bar{n}$ to the component indices of I;
- replacing every unary rule $(q, \bar{v}, q')$ in a node with $(q, \bar{v}[I], q')$; and

– replacing every merging rule $(q, S, \bar{q})$ in a node with $(q, S[I], \bar{q})$, where $S[I] = \{\bar{s}[I] \mid \bar{s} \in S\}$.

In a similar way, we consider the restriction $\mathcal{A}|_I$ of the automaton $\mathcal{A}$ as the $|I|$-MVASS$_n$ resulting from replacing the rules as described above. Note that if I is $\preceq$-downward closed (*i.e.*, if $i \preceq j$ and $j \in I$, then $i \in I$) then $\mathcal{D}|_I$ is actually a derivation of $\mathcal{A}|_I$.

Lemma 9. *Let $\mathcal{A} = \langle Q, U, M, \leq \rangle$ be a MVASS$_1$ of dimension k, and let $\mathcal{D}$ be a derivation for $(q, \bar{n}) \rightsquigarrow^+_{\mathcal{A}} \widehat{Q}$. Then, there is a contraction of $\mathcal{D}$ which is a derivation for $(q, \bar{n}') \rightsquigarrow^+_{\mathcal{A}} \widehat{Q}$ for some $\bar{n}' \geq \bar{n}$ and whose length is bounded by $(\max(U^+) + \max(\bar{n}))^{2^{p(k)}}$ for a polynomial $p(\ )$.*

Proof. This proof follows arguments similar to those from Rackoff [27, Section 3] as described in [8, Lemma 4]. Let $m(\mathcal{D}, \bar{n}, \widehat{Q}, \mathcal{A})$ be the smallest height of a contraction of $\mathcal{D}$ that is a derivation for $(q, \bar{n}') \rightsquigarrow^+_{\mathcal{A}} \widehat{Q}$ for some $\bar{n}' \geq \bar{n}$. For $L, k \in \mathbb{N}$ we let:

$$M_L(k) = \sup \{m(\mathcal{D}', \bar{n}, \widehat{Q}, \mathcal{A}) : \mathcal{D}' \text{ is a derivation for } (q, \bar{n}') \rightsquigarrow^+_{\mathcal{A}} \widehat{Q}, \bar{n}' \geq \bar{n} \text{ and } |Q| \cdot (\max(U^+) + \max(\bar{n}) + 1) \leq L\}.$$

We show that the number $M_L(k)$ is well-defined in the next lemma.

In the context of the partially ordered set $(\underline{k}, \preceq)$, let $\downarrow i = \{j \in \underline{k} \mid j \prec i\}$ for every $i \in \underline{k}$.

Lemma 10. *For all $L \in \mathbb{N}$, the following inequalities hold:*

$$M_L(k) \leq \begin{cases} L & \text{if } k = 0, \\ M_L(k-1) \cdot \prod_{i \in \underline{k}} B_i & \text{if } k \geq 1, \end{cases} \tag{6}$$

for $B_i = M_L(|\downarrow i|) \cdot \left(\prod_{j \succ i} B_j\right)^2 + L$, and $\prod \emptyset = 1$ by convention.

Proof. We proceed by induction on k. The case $k = 0$ is trivial, as there are no counters, and thus the height of minimal contractions is bounded by $|Q|$ by a pumping argument. For every $k \geq 1$, it is sufficient to prove that for every derivation $\mathcal{D}$ for $(q, \bar{n}') \rightsquigarrow^+_{\mathcal{A}} \widehat{Q}$ where $\bar{n}' \geq \bar{n}$ and $|Q| \cdot (\max(U^+) + \max(\bar{n}) + 1) \leq L$, the following inequality holds:

$$m(\mathcal{D}, \bar{n}, \widehat{Q}, \mathcal{A}) \leq M_L(k-1) \cdot \prod_{i \in \underline{k}} B_i. \tag{7}$$

For a set of components $I \subseteq \underline{k}$, we say that $\mathcal{D}$ is **I-bounded** if for every $i \in I$ and for every configuration $(q, \bar{v})$ of $\mathcal{D}$ we have $\bar{v}[i] < B_i$. We consider the following two cases: (a) $\mathcal{D}$ is $\underline{k}$-bounded, and (b) $\mathcal{D}$ is not $\underline{k}$-bounded. Assume that $\mathcal{D}$ has minimal height. We define $\rho_{\mathcal{D}}(x)$ [resp. $\sigma_{\mathcal{D}}(x)$] for any node x of $\mathcal{D}$ as the vector $\bar{v}$ [resp. state p] contained in the configuration $(p, \bar{v})$ of $\mathcal{D}$ at x.

(a) Assume that $\mathcal{D}$ is $\underline{k}$-bounded. Note that if $\rho_{\mathcal{D}}(x) = \rho_{\mathcal{D}}(x')$, $\sigma_{\mathcal{D}}(x) = \sigma_{\mathcal{D}}(x')$ and x is an ancestor of x' then the derivation $\mathcal{D}[x \leftarrow x']$ obtained by the contracting substitution is also a derivation for $(q, \bar{n}') \rightsquigarrow^{+}_{\mathcal{A}} \widehat{Q}$. By performing such substitutions repeatedly, we will eventually obtain a contraction of $\mathcal{D}$ that is a $\underline{k}$-bounded derivation for $(q, \bar{n}') \rightsquigarrow^{+}_{\mathcal{A}} \widehat{Q}$ with height bounded by

$$|Q| \cdot \prod_i B_i \leq L \cdot \prod_i B_i \leq M_L(k). \tag{8}$$

(b) Suppose now that $\mathcal{D}$ is not $\underline{k}$-bounded and suppose that it is minimal in the sense of contractions. Let $i_0 \in \underline{k}$ be a $\preceq$-minimal index so that $\mathcal{D}$ is $\uparrow i_0$-bounded for $\uparrow i_0 = \{j \mid j \succ i\}$. Note that

- for some node x of $\mathcal{D}$ we have $\rho_{\mathcal{D}}(x)[i_0] > B_{i_0}$; and
- for all $j \succ i_0$ and for every node y of $\mathcal{D}$ we have $\rho_{\mathcal{D}}(y)[j] \leq B_j$.

Let x_0 be the lowest node (*i.e.*, closest to the leaf) so that $\rho_{\mathcal{D}}(x_0)[i_0] > B_{i_0}$. We first bound the distance between x_0 and the root, and then we bound the distance between x_0 and the leaf.

(I) Consider the subderivation $\mathcal{D}_1$ from the root to x_0. We show that the height of $\mathcal{D}_1$ is bounded by

$$\mathit{contr} = M_L(|{\downarrow} i_0|) \cdot \prod_{j \succ i_0} B_j.$$

For the sake of contradiction, suppose that its height is larger than *contr*. Note that at each step of $\mathcal{D}_1$ we can increase component i_0 in at most

$$\mathit{incstep} = \sum_{j \succ i_0} B_j + \max(U^+). \tag{9}$$

If we restrict $\mathcal{D}_1$ to the components ${\downarrow} i_0$ we obtain a derivation for $\mathcal{A}|_{{\downarrow} i_0}$ with smaller dimension. In fact it is a derivation thanks to the inequality of (4) —all the increments coming from transfers from the components $j \succeq i_0$ can considered in the inequality.

By inductive hypothesis on $\mathcal{D}|_{{\downarrow} i_0}, \mathcal{A}|_{{\downarrow} i_0}$, there are two nodes y, y' (y ancestor of y') at distance $\leq$ *contr* from the root that can be contracted (*i.e.*, so that $\mathcal{D}|_{{\downarrow} i_0}[y \leftarrow y']$ is a derivation for $(q, \bar{n}') \rightsquigarrow^{+}_{\mathcal{A}|_{{\downarrow} i_0}} \widehat{Q}$ for $\bar{n}' \geq \bar{n}$), and whose values for all the components $j \succ i_0$ coincide (*i.e.*, $\rho_{\mathcal{D}}(y)[j] = \rho_{\mathcal{D}}(y')[j]$ for all $j \succ i_0$). The contraction of y, y' on $\mathcal{D}_1|_{{\downarrow} i_0}$ is a derivation of $\mathcal{A}|_{{\downarrow} i_0}$ by inductive hypothesis. Further, since the values of components $j \succ i_0$ coincide, the contraction of y, y' on $\mathcal{D}_1|_{\{j \mid j \neq i_0\}}$ is also a derivation. Finally, by (9), the increase from y to y' on

component i_0 cannot be greater than $\mathit{contr} \cdot \mathit{incstep}$; and since

$$\begin{aligned}
\mathit{contr} \cdot \mathit{incstep} + \max(\bar{n}) &= M_L(|{\downarrow} i_0|) \cdot \prod_{j \succ i_0} B_j \cdot \sum_{j \succ i_0} B_j + \max(\bar{n}) \\
&\leq M_L(|{\downarrow} i_0|) \cdot \prod_{j \succ i_0} B_j \cdot \sum_{j \succ i_0} B_j + L \\
&\leq M_L(|{\downarrow} i_0|) \cdot \big(\prod_{j \succ i_0} B_j \big)^2 + L \\
&= B_{i_0}
\end{aligned}$$

we thus have that the contraction $\mathcal{D}_1'$ of y, y' on $\mathcal{D}_1$ is a derivation as well, for all the components. Further, the root of $\mathcal{D}_1'$ has a configuration greater than $(q, \bar{n})$ and the leaf remains unchanged, that is, it contains the configuration $(\rho_{\mathcal{D}_1}(x_0), \sigma_{\mathcal{D}_1}(x_0))$. Therefore, by minimality of $\mathcal{D}$ we have that $\mathcal{D}_1$ must have height bounded by contr.

(II) On the other hand, let $\mathcal{D}_2$ be the subderivation between x_0 and the leaf of $\mathcal{D}$. Since $\mathcal{D}_2$ is $({\uparrow} i_0 \cup \{i_0\})$-bounded (except for the root), we have that $\mathcal{D}_2$ cannot have height larger than

$$\mathit{contr}' = M_L(|{\downarrow} i_0|) \cdot \prod_{j \succeq i_0} B_j$$

using similar arguments as before.

Thus, by (I) cum (II), we have that the height of $\mathcal{D} = \mathcal{D}_1 \mathcal{D}_2$ is bounded by

$$\begin{aligned}
\mathit{contr} + \mathit{contr}' &= M_L(|{\downarrow} i_0|) \cdot \prod_{j \succ i_0} B_j \;+\; M_L(|{\downarrow} i_0|) \cdot \prod_{j \succeq i_0} B_j \\
&\leq M_L(k-1) \cdot \prod_i B_i = M_L(k)
\end{aligned}$$

Note that by definition of B_i, for any $\preceq$-minimal i, $B_i = M_L(0) + L = 2L$. For any other i, the number of recursive calls needed to compute B_i is bounded by $2^k \cdot k$. This is because at each recursive call for B_i we produce two instances of B_j for every $j \succ i$, and thus in the product each B_t with maximal t will have an exponent $2^{|\{j | i \prec j \preceq t\}|}$; repeating this for each such t (not more than k times) we obtain the bound. We then have, for every i, that $B_i \leq (M_L(|{\downarrow} i|))^{2^k \cdot k} + 2^{k+1} \cdot k \cdot L$, and thus

$$M_L(k) \leq M_L(k-1)^{2^k \cdot k^2} \cdot (2L)^{2^k \cdot k^2} + 2^{2^k} \cdot k^2 \cdot L.$$

Therefore, $M_L(k)$ is bounded by a function $L^{2^{p(k)}}$ for some polynomial $p(\,)$.

We have now all the necessary elements to prove Theorem 2.

Proof (Complete proof of Theorem 2). Using Lemma 9, we show that if there is an incrementing derivation $\mathcal{D}$ for $(q, \bar{n}) \rightsquigarrow_{\mathcal{A}}^{+} \widehat{Q}$, where $\mathcal{A} = (Q, U, M, \preceq)$ is

an nMVASS$_k$ counter system, then there is a contraction $\mathcal{D}'$ of $\mathcal{D}$ with height bounded doubly-exponentially in the dimension. We show that if a nMVASS$_k$ $\mathcal{A} = (Q, U, M)$ has an incrementing derivation $\mathcal{D}$ for $(q, \bar{n}') \rightsquigarrow^+_{\mathcal{A}} \widehat{Q}$, $\bar{n}' \geq \bar{n}$, then there a contraction $\mathcal{D}'$ of $\mathcal{D}$ which is also an incrementing derivation for $(q, \bar{n}'') \rightsquigarrow^+_{\mathcal{A}} \widehat{Q}$, $\bar{n}'' \geq n$, whose height is bounded by

$$(\max(U^+) + \max \bar{n} + |Q|)^{2^{p(k)}} \tag{10}$$

for a polynomial function $p : \mathbb{N} \to \mathbb{N}$.

The next argument basically follows the schema (i)–(iii) of [8, p.7]. Let $\mathcal{D}$ be an incrementing derivation for $(q, \bar{n}') \rightsquigarrow^+_{\mathcal{A}} \widehat{Q}$, and let π be a root-to-leaf path of $\mathcal{D}$ which is larger than the bound. Let $\mathcal{A}'$ be a kMVASS$_n$ whose set of rules consists of:

- The unary rules $(q, \bar{v}, q')$ contained in the unary rules of π.
- Suppose we have a node x of π with configuration $((q, \bar{n}), (q, S, \bar{q}))$ and with children labeled $((q_1, \bar{n}_1), r_1), \ldots, ((q_s, \bar{n}_s), r_s)$, so that the next element after x in π is the j-th child of x. Further, suppose that this merging rule is preceded by a unary rule; that is, the parent x' of x is labeled with $((p, \bar{n}'), (p, \bar{w}, q))$ —it is not hard to see that without any loss of generality we can always assume that a merging rule is preceded by a unary rule. Let $B = \{\bar{b}_1, \ldots, \bar{b}_m\}$ be the basis of S, that is, $B^* = S$. Let $B' = \{\bar{b}_i \mid \bar{b}_i[h] = 0 \text{ for all } n \cdot (1 + j) + 1 \leq h \leq n \cdot (2 + j)\}$, and $B'' = B \setminus B'$. Note that B' is the set of bases that do not touch the j-th component. We then have

  $$(\bar{n}, \bar{n}_1, \ldots, \bar{n}_s) = \alpha_1 \bar{b}'_1 + \cdots + \alpha_{m'} \bar{b}'_{m'} + \beta_1 \bar{b}''_1 + \cdots + \beta_{m''} \bar{b}''_{m''}$$

  for $B' = \{\bar{b}'_1, \ldots, \bar{b}'_{m'}\}$ and $B'' = \{\bar{b}''_1, \ldots, \bar{b}''_{m''}\}$. Let $\bar{v}' = -(\alpha_1 \bar{b}'_1 + \cdots + \alpha_{m'} \bar{b}'_{m'}) \in \mathbb{Z}^{n \cdot (s+1)}$ and $\bar{v} \in \mathbb{Z}^n$ the restriction of $\bar{v}'$ to the first n components. Note that $\bar{v}$ contains non-positive entries only. Then, produce the unary rule $(p, \bar{w} + \bar{v}, q)$ and a merging rule (q, S', q') so that $S' \subseteq \mathbb{N}^{2n}$ is the restriction of S to the components corresponding to the j-th child.

Note that if we relabel accordingly π we obtain an incrementing derivation for $(q, \bar{n}') \rightsquigarrow^+_{\mathbb{A}'} \widehat{Q}$. Then, by Lemma 9, there is a contraction of π which is still a correct incrementing derivation for $(q, \bar{n}'') \rightsquigarrow^+_{\mathbb{A}'} \widehat{Q}$ for some $\bar{n}'' \geq \bar{n}$ and so that its length is at most

$$(\max(U'^+) + \max(\bar{n}) + 1)^{2^{p(k)}}$$

where U' is the set of unary rules of $\mathcal{A}'$. Note that $\max(U'^+) \leq \max(U^+)$ because we have only added unary rules with smaller positive entries.

We can then unfold back the subtrees hanging from nodes of π to obtain an MVASS incrementing derivation for $(q, \bar{n}'') \rightsquigarrow^+_{\mathbb{A}} \widehat{Q}$ whose number of leaves at height greater than the bound (10) has decreased in at least 1. Repeating the same argument a finite number of times we obtain an incrementing derivation for $(q, \bar{n})$ of height bounded by (10).

Thus, to decide the incrementing reachability problem, it suffices to search for a derivation of doubly-exponential height, whose vectors may contain triply-exponential entries in principle. As a consequence of this, the verification of the existence of such a derivation can be performed in alternating double exponential space, as it is shown in [8, Theorem 8], and thus the incrementing reachability for MVASS is in 3ExpTime. See Appendix A for more details.

If n is fixed, the height of the witnessing derivation becomes singly exponential and thus the problem is in 2ExpTime (as explained in [8, Theorem 8]). □

B Proof of Theorem 3

In this section we show that the problem of satisfiability of $\mathrm{LRV}_n^{\mathrm{D}}$ over k-ranked data trees can be reduced to the problem of control-state reachability for $n\mathrm{VASS}_k$.

We start with a simple logic and we gradually increase the complexity to finally obtain the desired result. We begin with a simple logic, notated $\mathrm{LRV}_1^{\mathrm{D}-}$, which consists only of formulas $\varphi \in \mathrm{LRV}_1^{\mathrm{D}}$ that are conjuncts of terms of the form $v \star \mathrm{EX}^i v$, $v \star \mathrm{EF} v$, or their negation, where $\star \in \{\approx, \not\approx\}$. We first show, in §B.1, that the problem of satisfiability of $\mathrm{LRV}_1^{\mathrm{D}-}$ over k-ranked data trees can be reduced to the problem of control-state reachability for$1\mathrm{VASS}_k$. These initial results are of no great interest by themselves, but the constructions we make for their proofs will be useful for §B.2, when we extend the expressive power of $\mathrm{LRV}_1^{\mathrm{D}-}$ with the missing operators (negation, conjunction, disjunction and 'until') to obtain $\mathrm{LRV}_1^{\mathrm{D}}$, where we prove an analogous reduction. In §B.3 we easily generalize to arbitrary n to obtain the proof of Theorem 3.

B.1 A simple logic: $\mathbf{LRV_1^{D-}}$

In this section we show the following:

Proposition 11. *There is a computable reduction from SAT_k-$\mathrm{LRV}_1^{\mathrm{D}-}$ to* $\mathrm{CSReach}(1\mathrm{VASS}_k)$.

That is, we show that for any formula φ of $\mathrm{LRV}_1^{\mathrm{D}-}$, there is a $1\mathrm{VASS}_k$ $\mathcal{A}_\varphi^k$ such that φ is satisfiable in a k-ranked data tree iff $\mathcal{A}_\varphi^k$ has a solution to the control-state reachability problem.

We begin with some definitions used for the construction of $\mathcal{A}_\varphi^k$, and then proceed to the proof of the reduction.

Valid (d,k)-frames A **valid** (d,k)**-frame** (or often just (d,k)**-frame**) is a tuple $F = \langle N, E, \ell_1, \ell_2, \equiv \rangle$ such that

- $\langle N, E\rangle$ is a k-ranked ordered tree of height at most d, whose non-empty set of nodes is N and whose set of edges is E.
- $\ell_1 : N \to \{([v \approx \mathrm{EF}v], [v \not\approx \mathrm{EF}v]), ([v \approx \mathrm{EF}v], [\neg v \not\approx \mathrm{EF}v]), ([\neg v \approx \mathrm{EF}v], [v \not\approx \mathrm{EF}v]), ([\neg v \approx \mathrm{EF}v], [\neg v \not\approx \mathrm{EF}v])\}$ is a node-labeling function
- $\ell_2 : N \to \{\varepsilon, \oplus, \ominus\}$ is another node-labeling function
- $\equiv$ is an equivalence relation over N

where ℓ_1 satisfies the following validity conditions:

1. If $x \in N$ is such that $\ell_1(x) = ([\neg v \approx \mathrm{EF}v], [\neg v \not\approx \mathrm{EF}v])$ then x has no children.
2. If $x \in N$ is a leaf, and x is at distance $< d$ from the root of F, then $\ell_1(x) = ([\neg v \approx \mathrm{EF}v], [\neg v \not\approx \mathrm{EF}v])$.
3. Let y be a descendant of x, with $x \equiv y$, then $\pi_1(\ell_1(x)) = [v \approx \mathrm{EF}v]$, and if $\pi_2(\ell_1(y)) = [v \not\approx \mathrm{EF}v]$, then $\pi_2(\ell_1(x)) = [v \not\approx \mathrm{EF}v]$. If $x \not\equiv y$, then $\pi_2(\ell_1(x)) = [v \not\approx \mathrm{EF}v]$.
4. If y is a descendant of x and $\pi_2(\ell_1(x)) = [\neg v \not\approx \mathrm{EF}v]$ then $x \equiv y$ and $\pi_2(\ell_1(y)) = [\neg v \not\approx \mathrm{EF}v]$.
5. If y is a descendant of x and $\pi_1(\ell_1(x)) = [\neg v \approx \mathrm{EF}v]$, then $x \not\equiv y$.
6. If $\pi_2(\ell_1(x)) = [v \not\approx \mathrm{EF}v]$ and all children y of x satisfy $x \equiv y$, then there is some child z of x such that $\pi_2(\ell_1(z)) = [v \not\approx \mathrm{EF}v]$

and ℓ_2 satisfies the following conditions:

a. $\ell_2(r) = \oplus$ iff r is the root of F, $\pi_1(\ell_1(r)) = [v \approx \mathrm{EF}v]$, and there is no descendant x of r with $x \equiv r$
b. If $\ell_2(x) = \ominus$, then x is a leaf of F at maximum distance from the root (i.e. at distance d), and there is no node y at distance $< k$ from the root with $x \equiv y$.
c. If x, y are leaves with $x \equiv y$, and $\ell_2(x) = \ominus$, then $\ell_2(y) = \ominus$.

Let $\mathcal{F}_{d,k}$ be the set of all valid (d,k)-frames. We will work with many (d,k)-frames so we need a notation to distinguish the component of each of them. Unless otherwise stated, a (d,k)-frame F is a tuple $F = \langle N^F, E^F, \ell_1^F, \ell_2^F, \equiv^F \rangle$.

We say that a (d,k)-frame F' is an **extension** of a (d,k)-frame F, or that F is a **(d,k)-subframe** of F', if $N^F \subseteq N^{F'}$ and E^F, ℓ_1^F, ℓ_2^F, and $\equiv^F$ are the restrictions of $E^{F'}$, $\ell_1^{F'}$, $\ell_2^{F'}$, and $\equiv^{F'}$, respectively, to N^F.

If T is any tree-shaped structure (in particular, a data tree, but it could also have, *e.g.* other node labeling functions), we notate $T(x)$ as the subtree of T generated by x and all its descendants (hence the root of $T(x)$ is x). Let F' be a (d,k)-frame, and let $x \in N^{F'}$. We name $F(x)$ as the (d,k)-subframe of F' induced by x.

Let F be a (d,k)-frame with root r, such that $\ell_2^F(x) = \epsilon$ for all x in N^F, and let $x_1, \ldots, x_i$ be the children of r, ordered from left to right. Let G_i be the (d,k)-frames of F induced by x_i. We say that F is **1-consistent** with the (d,k)-frames $F_1, \ldots F_i$, and G_i is a (d,k)-subframe of F_i for all i. Further, we say that F is a **point of decrement** if there is a leaf $x \in N^F$ such that $\ell_2(x) = \ominus$. More precisely, we say that it is a point of decrement **of value** p if it is a point of decrement with a maximum of p $\equiv$-equivalence classes of leaves y with $\ell_2^F(y) = \ominus$. We say that F is a **point of increment** if $\ell_2^F(r) = \oplus$ and F is not a point of decrement.

The automaton $\mathcal{A}_\varphi^k$ We recall from Section 5 that the EX-length of φ is the maximum i such that the a term of the form $v \star \mathrm{EX}^i v$ is a subformula of φ. Let d be the EX-length of φ. We define the 1VASS$_k$ $\mathcal{A}_\varphi^k$ as follows:

- The set of states of $\mathcal{A}^k_\varphi$ consists of $\mathcal{F}_{d,k}$, the set of all valid (d,k)-frames.
- Unary rules. Let F_1 and F_2 be (d,k)-frames.
  - $F_1 \xrightarrow{-n} F_2$ if F_1 is a point of decrement of value n, and F_2 is equal to F_1, except that $\ell_2^{F_2}$ is defined as follows:

$$\ell_2^{F_2}(x) = \begin{cases} \epsilon & x \text{ is a leaf of } F_1; \\ \ell_2^{F_1}(x) & \text{otherwise.} \end{cases}$$

  - $F_1 \xrightarrow{+1} F_2$ if F_1 is a point of increment, and F_2 is equal to F_1, except that $\ell_2^{F_2}$ is defined as follows:

$$\ell_2^{F_2}(x) = \begin{cases} \epsilon & x \text{ is the root of } F_1; \\ \ell_2^{F_1}(x) & \text{otherwise.} \end{cases}$$

- Branching rules: $F \to (F_1, \ldots, F_i)$ (with $i \leq k$), if F is 1-consistent with $F_1, \ldots, F_i$.

We define the following sets of states of Q, to be used as inputs of the control-state reachability problem.

- Q_0 is the set of *initial* (d,k)-frames. We say that F, of root r, is initial iff the following conditions hold:
  - F, r satisfies all terms of φ of the form $v \star \mathrm{EX}^i v$ or $\neg v \star \mathrm{EX}^i v$;
  - if $v \approx \mathrm{EF}v$ (resp. $v \not\approx \mathrm{EF}v$) is a positive conjunct of φ, then the root $r \in N^F$ satisfies $\pi_1(\ell_1^F(r)) = [v \approx \mathrm{EF}v]$ (resp. $\pi_2(\ell_1^F(r)) = [v \not\approx \mathrm{EF}v]$);
  - if $v \approx \mathrm{EF}v$ (resp. $v \not\approx \mathrm{EF}v$) is a negative conjunct of φ, then for all descendants y of the root $r \in N^F$, it holds that $\pi_1(\ell_1^F(y)) = [\neg v \approx \mathrm{EF}v]$ (resp. $\pi_2(\ell_1^F(y)) = [\neg v \not\approx \mathrm{EF}v]$).
- $\widehat{Q}$ is the singleton containing the (d,k)-frame in $\mathcal{F}_{d,k}$ that consists solely of one node.

We define the **control-state reachability problem for initial sets** CSReach as the problem of, given $\mathcal{A}, Q_0, \widehat{Q}$, whether $(q, \bar{n}) \rightsquigarrow_{\mathcal{A}} \widehat{Q}$ for some $\bar{n}$ and some $q \in Q_0$. It is easy to see that this problem is equivalent to the problem of whether $(q, \bar{0}) \rightsquigarrow^+_{\mathcal{A}} \widehat{Q}$ for some $q \in Q_0$.

Observation 12 *The problem of control-state reachability problem for initial sets is equivalent to the problem of control-state reachability, as defined in §4.3.*

Consequently, it is enough to prove:

Proposition 13. $\mathrm{CSReach}(\mathcal{A}^k_\varphi, Q_0, \widehat{Q})$ *implies* $SAT_k(\varphi)$.

Proof. Suppose S is a solution tree for the control-state reachability problem of $\mathcal{A}^k_\varphi$. We first construct a (node-decorated) k-ranked data tree $T = \langle N, E, \ell_2, \equiv \rangle$ (with the labeling function such that $\ell_2(x) \in \{\epsilon, \ominus\}$) and then we define our solution data tree by removing the node decoration: $T' = \langle N, E, \equiv \rangle$. This last

tree will satisfy φ at its root. The tree T will be constructed by induction: for every subtree R of S we construct a tree T_R such that, as we will formally verify in the second part of the proof, its structure of nodes and edges derives from the structure of the (d,k)-frames (states of $\mathcal{A}^k_\varphi$) of R, and where the semantics of the labels ℓ_1 in R are satisfied in T_R. The tree T will finally be T_S. We then verify that φ is true at the root of T_S.

Construction of the data tree Given a subderivation R, we construct a (labeled) data tree $T_R = \langle N^R, E^R, \ell_2^R, \equiv^R \rangle$. We also identify each node x in a (d,k)-frame of R with a corresponding node $\mathrm{id}_R(x)$ in T_R. This mapping id_R will be surjective, and, whenever F is a (d,k)-frame in R, $\mathrm{id}_R \restriction N^F$ is injective.

We proceed by induction in the complexity of R.

Leaf. For the base case, let F be a leaf of S. Observe that by construction of $\mathcal{A}^k_\varphi$, $N^F = \{x\}$. Then the corresponding tree is $T_F = \langle N^F, E^F, \ell_2^F, \equiv^F \rangle$ and we define $\mathrm{id}_F(x)$ to be the same node in T_F, i.e. $\mathrm{id}_F(x) = x$.

Branching rule. Let S_0 be an incrementing derivation subtree of S such that its root, a (d,k)-frame F_0, branches into (d,k)-frames $F_1, \ldots, F_i$. We define T_{S_0} as follows. Let r be the root of F_0, and let $a_1, \ldots, a_i$ be the children of r, ordered from left to right. Let $S_j = S_0(F_j)$.

By inductive hypothesis, all S_j correspond with trees T_{S_j}, with equivalence relations $\equiv^{T_{S_j}}$ and labeling function $\ell_2^{T_{S_j}}$. Then we define T_{S_0} as follows: the root of T_{S_0} is some node $\tilde{r}$, and the subtrees hanging from its i children are T_{S_j}, for $1 \leq j \leq i$. See Figure 1 for an example in the case $k = 2$.

For each node x in some (d,k)-frame of S_0, we define $\mathrm{id}_{S_0}(x)$ as follows: if x is in a (d,k)-frame of some S_j, we keep the identification $\tilde{x}$ it had in T_{S_j}, i.e. $\mathrm{id}_{S_0}(x) = \mathrm{id}_{S_j}(x)$; if x is the root of F_0, namely $x = r$, then $\mathrm{id}_{S_0}(x) = \tilde{r}$; and for each node $x \neq r$ of F_0, let (by 1-consistency) x' be the corresponding copy of x in F_j and define $\mathrm{id}_{S_0}(x) = \mathrm{id}_{S_j}(x')$. It remains to define the labeling ℓ_2 and the equivalence relation $\equiv$ of T_{S_0}. We define ℓ_2 as follows

$$\ell_2(x) = \begin{cases} \epsilon & \text{if } x = \tilde{r}; \\ \ell_2^{T_{S_j}}(x) & \text{if } x \text{ is in a } (d,k)\text{-frame of } S_j. \end{cases}$$

We define $\equiv$ as the smallest equivalence relation such that:

- $\equiv \restriction_{T_{S_j}} = \equiv^{T_{S_j}}$ for each $1 \leq j \leq i$
- if $x \equiv^{F_0} y$ then $\mathrm{id}_{S_0}(x) \equiv \mathrm{id}_{S_0}(y)$.

Unary rule. Let S_0 be a incrementing derivation subtree of S, with root (d,k)-frame F_0 and an only child, the (d,k)-frame F_1, as the outcome of a unary rule. Let $S_1 = S_0(F_1)$. Let $T_{S_1} = \langle N, E, \ell_2, \equiv \rangle$ be the tree that is constructed from S_1 by inductive hypothesis.

If the rule for the transition from F_0 to F_1 was a $-n$ decrement, then n is the maximal such that there are leaves $x_1, \ldots, x_n$ of F_0 with $x_i \not\equiv^1 x_j$ $(i \neq j)$, and

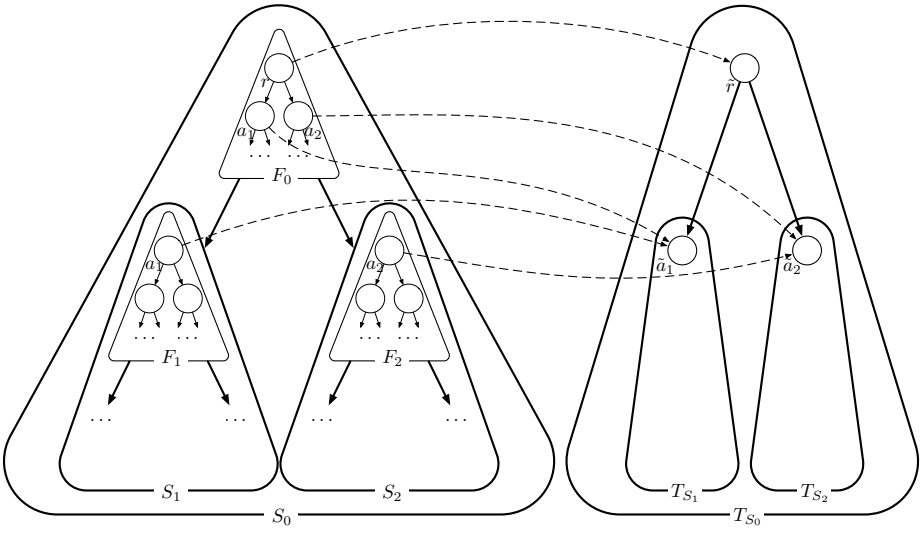


Fig. 1. The incrementing derivation tree S_0, starting with a branching rule, and the constructed T_{S_0}. The dotted lines represent some pairs of the mapping id_{S_0}.

$\ell_2^{F_0}(x_i) = \ominus$. We define $T_{S_0} = \langle N, E, \tilde{\ell_2}, \equiv \rangle$ (i.e. the same tree structure as T_{S_1}, and the same equivalence relation over it, but a different labeling function), where $\tilde{\ell}_2$ is defined below, and the identification mapping id_{S_0} is defined as follows: $\mathrm{id}_{S_0} \restriction S_1 = \mathrm{id}_{S_1}$, and for any $x \in N^{F_0}$, if x' is the corresponding copy of x in F_1 (recall that the underlying trees of F_0 and F_1 are equal), we let $\mathrm{id}_{S_0}(x) = \mathrm{id}_{S_1}(x')$. The labeling $\tilde{\ell}_2$ is defined as follows:

$$\tilde{\ell}_2(x) = \begin{cases} \ominus & \text{if } (\exists y \in N^{F_0})\ \ell_2^{F_0}(y) = \ominus \text{ and } \mathrm{id}_{S_0}(y) = x; \\ \ell_2(x) & \text{otherwise.} \end{cases}$$

In other words, $\tilde{\ell}_2$ adds $\ominus$ labels to the nodes that correspond with $\ominus$ leaves in F_0, and for all other nodes keeps the labeling of ℓ_2. See Figure 2 for an illustration of this process.

If the rule was an increment, as we will see in Lemma 14, we can assume that there is a node y in some (d,k)-frame F of S_1 such that $\ell_2^F(y) = \ominus$ and also $\ell_2(\mathrm{id}_{S_1}(y)) = \ominus$. The idea now will be to join the equivalence classes of the root of T_{S_0} with that of $\mathrm{id}_{S_0}(y) = \mathrm{id}_{S_1}(y)$, and to remove the $\ominus$ labels in the nodes of these equivalence classes. Let r be the root of F_0, and define $T_{S_0} = \langle N, E, \tilde{\ell}_2, \tilde{\equiv} \rangle$ (i.e. the same tree structure as T_{S_1}, but different equivalence relation and labeling function) where $\tilde{\ell}_2$ and $\tilde{\equiv}$ are defined below, and the identification mapping id_{S_0} is defined as in the case of a $-n$ decrement.

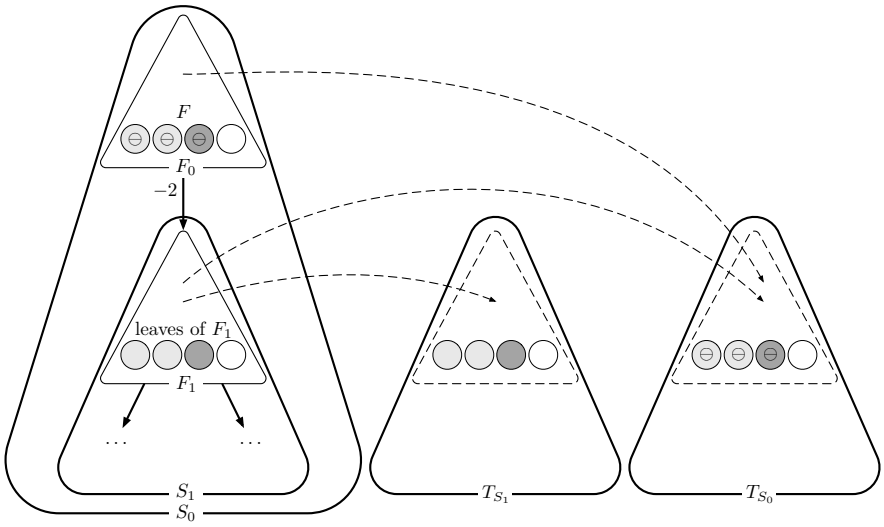


Fig. 2. The incrementing derivation tree S_0, starting with a decrement -2 rule, and the constructed T_{S_0}. The colours represent the equivalence classes. The dotted lines represent the mappings id_{S_0} and id_{S_1}.

The relation $\widetilde{\equiv}$ joins the equivalence classes of $\mathrm{id}_{S_0}(r)$ and $\mathrm{id}_{S_0}(y)$, i.e.

$$z \widetilde{\equiv} w \text{ iff } z \equiv w \text{ or } (z \equiv \mathrm{id}_{S_0}(r) \text{ and } w \equiv \mathrm{id}_{S_0}(y))$$

and $\tilde{\ell}_2$ is defined as follows:

$$\tilde{\ell}_2(x) = \begin{cases} \epsilon & \text{if } x = \mathrm{id}_{S_1}(z) \text{ for some } z \equiv^F y; \\ \ell_2(x) & \text{otherwise.} \end{cases}$$

See Figure 3 for an illustration of this process.

Observe that for the construction of we are ignoring the non-deterministic increments in the derivation (which is a solution to the control-state reachability problem), as well as the exact distribution of the counters.

Lemma 14. *Let S_0 be an incrementing derivation subtree of S (a solution to the control-state reachability problem) with root (d,k)-frame F_0, such that F_0 has an only child, the (d,k)-frame F_1, which is the product of an increment rule. Let $S_1 = S_0(F_1)$. Let $T_{S_1} = \langle N, E, \ell_2, \equiv \rangle$ and $T_{S_0} = \langle N, E, \tilde{\ell}_2, \widetilde{\equiv} \rangle$ be given as in the construction. Then there is a node $p \in N$ such that $\ell_2(p) = \ominus$. Furthermore, if for any p with $\ell_2(p) = \ominus$ we define $P \subseteq N$ as*

$$P = \{x \in N \mid \ell_2(x) = \ominus \wedge x \equiv p\}$$

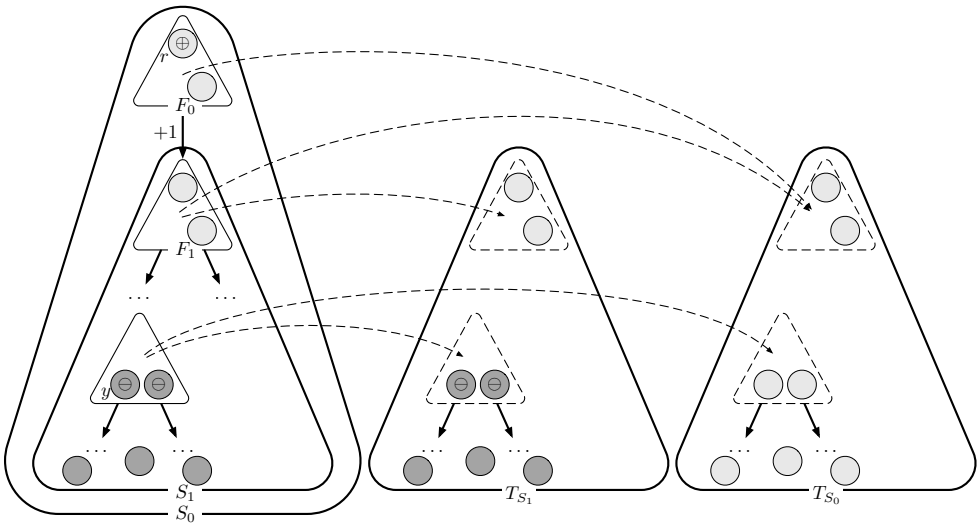


Fig. 3. The incrementing derivation tree S_0, starting with an increment rule, and the constructed T_{S_0}. The colours represent the equivalence classes. The dotted lines represent the mappings id_{S_0} and id_{S_1}.

then if both $x, y \in P$ for some $x, y \in N$, and we have $F_x, F_y \in S_1$ such that $\tilde{x} \in F_x, \tilde{y} \in F_y$ and $\mathrm{id}_{S_1}(\tilde{x}) = x, \mathrm{id}_{S_1}(\tilde{y}) = y$ (and thus with $\ell_2^{F_x}(\tilde{x}) = \ominus$ and $\ell_2^{F_y}(\tilde{y}) = \ominus$), then we have that $F_x = F_y$ and $\tilde{x} \equiv_{F_x} \tilde{y}$.

That is, the $\ominus$-labeled nodes of T_{S_1} that in T_{S_0} are ϵ-labeled correspond to the equivalence class of $\ominus$-labeled leaves of a single (d, k)-frame in S_0. In other words, when the join of classes is done in the construction of T_{S_0} because of an increment operation, the nodes y in T_{S_0} with $\ell_2^{T_{S_1}}(y) = \ominus$ that have $\ell_2^{T_{S_0}}(y) = \epsilon$ cannot correspond via $\mathrm{id}_{S_0}^{-1}$ to $\ominus$-labeled leaves in more than one (d, k)-frame; they correspond with exactly one decrement of the counter in the incrementing derivation tree.

Proof (Proof of Lemma 14). We will prove the lemma by induction on S_1.

Base case. If S_1 is such that there is no increment operation in S_1 then, as S_1 is an incrementing derivation subtree of a solution of the control-state reachability problem, S_1 must contain a point of decrement F with a node $\tilde{y}$ with $\ell_2^F(\tilde{y}) = \ominus$ and, since there is no increment operation in S_1, we have that $\ell_2(\mathrm{id}_{S_1})(\tilde{y}) = \ominus$ (that is, $p = \mathrm{id}_{S_1}(\tilde{y})$).

For the second claim, assume that $p \in N$ is any node with $\ell_2(p) = \ominus$, and P is defined accordingly. Suppose by way of contradiction that $x, y \in N$ are such that $x, y \in P$, but there is no single $F \in S_1$ such that there are $\tilde{x}, \tilde{y} \in F$ with $\mathrm{id}_{S_1}(\tilde{x}) = x, \mathrm{id}_{S_1}(\tilde{y}) = y$, $\ell_2^F(\tilde{x}) = \ominus, \ell_2^F(\tilde{y}) = \ominus$, and $\tilde{x} \equiv_F \tilde{y}$. We will prove

this leads to $x \not\equiv y$, a contradiction with our assumption that $x, y \in P$, and consequently we will conclude that there is such a (d,k)-frame F.

First, observe that there cannot be a (d,k)-frame F_x and $\tilde{x} \in F_x$ with $\mathrm{id}_{S_1}(\tilde{x}) = x$ and $\ell_2^{F_x}(\tilde{x}) = \ominus$ such that $\tilde{x}$ has an ancestor $\tilde{a} \in F_x$ with $\tilde{a} \equiv_{F_x} \tilde{x}$, since leaves of (d,k)-frames are not labeled with $\ominus$ if they have an in-frame ancestor. Now, since increments have not been used in the construction of T_{S_1}, the equivalence classes of (d,k)-frames of S_1 coincide with their identification via id_{S_1} in T_{S_1}, and therefore there cannot be an ancestor z of x at distance d or less from x with $z \equiv x$. The analogous result holds for y. Now, $x \equiv y$, and $\ell_2(x) = \ell_2(y) = \ominus$ implies that there are $F_x, F_y \in S_1$, $\tilde{x} \in F_x, \tilde{y} \in F_y$ with $\mathrm{id}_{S_1}(\tilde{x}) = x, \mathrm{id}_{S_1}(\tilde{y}) = y$ and with $\ell_2^{F_x}(\tilde{x}) = \ominus, \ell_2^{F_y}(\tilde{y}) = \ominus$. Observe from the definition of the decrement rule in $\mathcal{A}_\varphi^k$ and the conditions on $\ominus$, that there cannot be $\tilde{w} \in F_w$ with $\mathrm{id}_{S_1}(\tilde{w}) = x$ and $F_w \neq F_x$. Analogously for y. So, as we are assuming $F_x \neq F_y$, from 1-consistency and the increment-free construction of T_{S_1}, $x \equiv y$ implies that there is a least common ancestor z in N with $x \equiv z \equiv y$, and a chain of nodes $z = z_0, \ldots, z_n = x$ in N such that z_{i+1} is a descendant of z_i at distance at most d, and $z_i \equiv z_{i+1}$ (and the same for a chain towards y). But this contradicts the observation that there cannot be an ancestor z of x at distance d or less with $x \equiv z$.

Induction. Observe that, since S is a solution to the control-state reachability problem, for every increment in S_0 there must be at least one decrement in S_1. Suppose there are m increments and $n \geq m$ decrements in S_0. Then there are $m-1$ increments and $n > m-1$ decrements in S_1. From the inductive hypothesis, there must remain at least one node $p \in N$ with $\ell_2(p) = \ominus$.

For the second claim, let $p \in N$ with $\ell_2(z) = \ominus$, $P = \{x \in N \mid \ell_2(x) = \ominus \wedge x \equiv p\}$, and $x, y \in P$. Let $x, y \in P$, and let $F_x, F_y \in S_1$, $\tilde{x} \in F_x, \tilde{y} \in F_y$ be such that $\mathrm{id}_{S_1}(\tilde{x}) = x, \mathrm{id}_{S_1}(\tilde{y}) = y$ We want to prove that $F_x = F_y$ and $\tilde{x} \equiv_{F_x} \tilde{y}$.

As $\ell_2(x) = \ell_2(y) = \ominus$, x and y cannot have been joined with the equivalence class of an ancestor node of T_{S_1} as a result of an increment operation. Therefore, we are in a similar case as in the base step: there must a common ancestor z of x, y, such that $x \equiv z \equiv y$ and a chain of nodes $z = z_0, \ldots, z_n = x$ in N such that z_{i+1} is a descendant of z_i at distance at most d, and $z_i \equiv z_{i+1}$, but this is a contradiction with the fact that $\ell_2(x) = \ominus$.

Verification By ignoring the labeling function ℓ_2 of $T_S = \langle N, E, \equiv, \ell_2 \rangle$ we obtain the desired data tree. We show that the data tree $T = \langle N, E, \equiv \rangle$ satisfies φ at the root. To ease the notation we write id for id_S.

From the definition of Q_0 and the construction of T, it is clear that any conjunct of φ of the form $v \star \mathrm{EX}^i v$ or $\neg v \star \mathrm{EX}^i v$, with $\star \in \{\approx, \not\approx\}$, is satisfied in the root of T.

Next, we show that conjuncts of φ of the type $v \star \mathrm{EF} v$ or $\neg v \star \mathrm{EF} v$ are also satisfied in the root of T. Recall from the construction of T that all nodes of N correspond via id^{-1} to nodes $x_1, \ldots, x_n$ in (d,k)-frames $F_1, \ldots, F_n$ of S such that $\ell_1^{F_i}(x_i) = \ell_1^{F_1}(x_1)$ for all $i = 1 \ldots n$. Therefore, it is enough to verify that if x is a node in some (d,k)-frame F of S, then $\mathrm{id}(x)$ satisfies the semantics of

$\pi_1(\ell_1^F(x))$ and of $\pi_2(\ell_1^F(x))$. Furthermore, it is enough to verify the above when x is the root of F. In what follows, we write ℓ_j instead of ℓ_j^F $(j = 1, 2)$.

We will use the following facts:

Fact 15 *In the construction of T, nodes that are at distance of at most d and non-equivalent in some frame are never made equivalent in T.*

Fact 16 *In the construction of T, nodes that are equivalent for some tree $T_{S'}$ with $S' \subseteq S$ are kept equivalent for T_S.*

Fact 17 *If $x \not\equiv^F z$ for all $x \neq z \in F$, and $\ell_2(x) \neq \oplus$ (recall that x is the root of F), then in the construction of T the equivalence class of* $\mathrm{id}(x)$ *is never joined with the equivalence class of a descendant.*

We consider the four different cases, two for every projection π_1 or π_2 of $\ell_1(x)$:

- In the case $\pi_1(\ell_1(x)) = [v \approx \mathrm{EF}v]$, we show the formula $v \approx \mathrm{EF}v$ is satisfied in $\mathrm{id}(x)$:
  - *Local satisfaction.* If there is some descendant $y \in F$ of x such that $x \equiv^F y$, then, from Fact 16, $\mathrm{id}(x) \equiv^T \mathrm{id}(y)$ and thus $T, \mathrm{id}(x) \models v \approx \mathrm{EF}v$.
  - *Non-local satisfaction.* If there is no such frame as in the previous sub-item, then (since $\pi_1(\ell_1(x)) = [v \approx \mathrm{EF}v]$) there is a frame F with $\ell_2^F(x) = \oplus$. Thus, from the construction of T and Fact 16, there is a descendant of $\mathrm{id}(x)$ in its equivalence class.
- In the case $\pi_1(\ell_1(x)) = [\neg v \approx \mathrm{EF}v]$, we consider two subcases to consider. If $\pi_2(\ell_1(x)) = [\neg v \not\approx \mathrm{EF}v]$, from the validity condition 1 the nodes $\mathrm{id}^{-1}(\mathrm{id}(x))$ have no descendants (and thus neither does $\mathrm{id}(x)$), and therefore $\neg v \approx \mathrm{EF}v$ is trivially satisfied at $\mathrm{id}(x)$. Otherwise, if $\pi_2(\ell_1(x)) = [v \not\approx \mathrm{EF}v]$, we show that for all descendants $\mathrm{id}(y)$ of $\mathrm{id}(x)$ at distance k of $\mathrm{id}(x)$ in T we have $\mathrm{id}(x) \not\equiv \mathrm{id}(y)$:
  - *Local satisfaction.* If $0 < k \leq d$ then taking $\tilde{y}$ in the frame F such that $\mathrm{id}(\tilde{y}) = \mathrm{id}(y)$, from condition 5, $x \not\equiv \tilde{y}$ and then from Fact 15 we have $\mathrm{id}(x) \not\equiv \mathrm{id}(y)$.
  - *Non-local satisfaction.* If $k > d$, since $\pi_1(\ell_1(x)) = [\neg v \approx \mathrm{EF}v]$ and x is the root of F, then $\ell_2(x) \neq \oplus$. As we have seen that $x \not\equiv^F z$ for all $x \neq z \in F$, Fact 17 indicates that for any $\tilde{y}$ descendant of $\mathrm{id}(x)$ we have $\mathrm{id}(x) \not\equiv \tilde{y}$.
- In the case $\pi_2(\ell_1(x)) = [v \not\approx \mathrm{EF}v]$, we distinguish two cases:
  - *Local satisfaction.* If there is a descendant $y \in F$ of x (and thus at distance at most d from x) such that $x \not\equiv^F y$, then, from construction of T, there is also a descendant $\mathrm{id}(y)$ in N such that $\mathrm{id}(x) \not\equiv^{T_{S_F}} \mathrm{id}(y)$. Thus, from Fact 15, $\mathrm{id}(x)$ satisfies $v \not\approx \mathrm{EF}v$.
  - *Non-local satisfaction.* If there is no descendant $y \in F$ with $x \not\equiv^F y$, then, from condition 6 there is chain of descendants of x, $y_1 \in F_1, \ldots, y_n \in F_n$ such that $\forall i < n\, y_i \equiv y_{i+1}$, and $\forall i\, \pi_2(\ell_1(y_i)) = [v \not\approx \mathrm{EF}v]$, and such that y_n has no child z with $z \equiv y_n$. As $\pi_2(\ell_1(y_n)) = [v \not\approx \mathrm{EF}v]$, condition 2

implies that there must be a child z of y_n, and then necessarily $z \not\equiv y_n$. From Fact 15 and Fact 16, therefore $\mathrm{id}(x) \not\equiv \mathrm{id}(z)$ and then $v \not\approx \mathrm{EF}v$ is satisfied in $\mathrm{id}(x)$.
- Finally, in the case $\pi_2(\ell_1(x)) = [\neg v \not\approx \mathrm{EF}v]$, the formula $\neg v \not\approx \mathrm{EF}v$ is satisfied in $\mathrm{id}(x)$ since, by validity condition 4 and construction of T, for all of the descendants $\mathrm{id}(y)$ of $\mathrm{id}(x)$ we have that $\mathrm{id}(x) \equiv \mathrm{id}(y)$ and $\ell_1(y) = [\neg v \not\approx \mathrm{EF}v]$.

This concludes the proof of Proposition 13.

Proposition 18. *$SAT_k(\varphi)$ implies* $\textsc{CSReach}(\mathcal{A}^k_\varphi, Q_0, \widehat{Q})$.

Proof. Let T be a k-ranked finite data tree whose root satisfies φ. We want to see that there is an incrementing derivation tree S of $\mathcal{A}^k_\varphi$ that starts at a node of Q_0 with the counter at 0 and ends with all leaves in $\widehat{Q}$ with the counter in 0.

We construct the incremental derivation tree from the root to the leaves. The idea is simply to identify which states ((d,k)-frames) of the automaton $\mathcal{A}^k_\varphi$ correspond to portions of T, adding first the appropriate values of ℓ_1 and ℓ_2, and then performing unary and binary operations in the expected way.

From a node in T to a (d,k)-frame For any node x of T we associate a (d,k)-frame F_x, defined as follows:

- N^{F_x} is the maximal subtree of T height d that hangs from x
- $\equiv^{F_x} = \equiv\restriction_{N^{F_x}}$
- $\ell_1^{F_x}(y)$ is consistent with T (*i.e.* $\ell_1^{F_x}(y) = ([v \approx \mathrm{EF}v], [v \not\approx \mathrm{EF}v])$ if y has a descendant in F_x with the same data value and other with different data value, etc.)
- $\ell_2^{F_x}$ defined as follows (*cf.* items a and b of the conditions for the (d,k)-frames):

$$\ell_2(y) = \begin{cases} \oplus & \text{if } y = x,\ \pi_1(\ell_1(y)) = [v \approx \mathrm{EF}v], \text{ and} \\ & \text{there is no descendant } z \text{ of } y \text{ with } z \equiv y; \\ \ominus & \text{if } y \text{ is a descendant of } x \text{ at distance } d \text{ from it, } x \not\equiv y, \text{ and} \\ & \text{for all descendant } z \text{ of } x \text{ at distance } < d \text{ we have } y \not\equiv z; \\ \epsilon & \text{otherwise.} \end{cases}$$

Let r be the root of T. We select as the initial state of S the (d,k)-frame F_r.

Construction of the incrementing derivation tree For this step, to make clearer whether we are referring to a node in T or in some (d,k)-frame, we will use a function id from the roots of the (d,k)-frames into T, as in Proposition 13. If $\tilde{x}$ is the root of F, and $F = F_x$, then we set $\mathrm{id}(\tilde{x}) = x$.

We now decide which rules are invoked on each (d,k)-frame F_x we construct. Let $\tilde{x} \in \mathrm{id}^{-1}(x)$ be the root of F.

If F_x is neither a point of decrement nor a point of increment, and if x has descendants, then we use a binary rule to branch F_x into the two (d,k)-frames F_{a_1} and F_{a_2}, where a_1, a_2 are the two children of x in T. The particular way in

which the counter of $\mathcal{A}^k_\varphi$ has been divided between these two (d,k)-frames will be explained afterwards (see **Verification**).

If F_x is a point of decrement of value n then from F_x we transition, via an n-decrement, to the (d,k)-frame $F^- = \langle N^{F_x}, E^{F_x}, \ell_1^{F_x}, \ell_2^-, \equiv^{F_x}\rangle$ (*i.e.* F^- is equal to F_x except for the ℓ_2 labeling function), where

$$\ell_2^-(\tilde{y}) = \begin{cases} \epsilon & \text{if } \ell_2(\tilde{y}) = \ominus; \\ \ell_2^{F_x}(\tilde{y}) & \text{otherwise.} \end{cases}$$

If $\tilde{x}^-$ is the root of F^-, we keep $\mathrm{id}(\tilde{x}^-) = \mathrm{id}(\tilde{x})$.

If F_x is a point of increment, we transition to the (d,k)-frame $F^+ = \langle N^{F_x}, E^{F_x}, \ell_1^{F_x}, \ell_2^+, \equiv^{F_x}\rangle$ (*i.e.* F^- is equal to F_x except for the ℓ_2 labeling function), where

$$\ell_2^+(\tilde{y}) = \begin{cases} \epsilon & \text{if } \ell_2(\tilde{y}) = \oplus; \\ \ell_2(\tilde{y}) & \text{otherwise.} \end{cases}$$

If $\tilde{x}^+$ is the root of F^+, we keep $\mathrm{id}(\tilde{x}^+) = \mathrm{id}(\tilde{x})$.

Verification We show that the constructed incrementing derivation tree S is a solution to the control-state reachability problem. Observe that labels $\oplus$ of ℓ_2 are assigned to the roots of (d,k)-frames when the corresponding node of T has a descendant at distance greater than d with the same equivalence class (but it has none at distance d or less). Also, $\ominus$ labels are assigned to leaves when they do not have an ancestor of the same class at distance less than d from the root of the frame. Thus, for every frame in S whose root r has $\ell_2(r) = \oplus$, there is at least one descendant frame in S with a leaf y with $\ell_2(y) = \ominus$.

Therefore, in the incrementing derivation tree S the total number of increments of the counter has been some number m and the total value of the decrements some $n \geq m$. If $n > m$, we modify S to add an spontaneous increase of the counter in $n - m$ at the first frame, which can be done since we are working on the control-state reachability problem. So we can assume that $m = n$. It remains to assign, for each instance of the branching rule, a way in which the counter has been divided between the branches. We do that as follows: if at a node F of S the counter is at c and the next operation of S is a branching $F \to F_1 \mid \cdots \mid F_i$, then for each i we assign c_i to F_i, where c_i is the number of decrements in the derivation tree of $S(F_i)$. In this way, all the operations of S are valid (there are no n-decrements when the counter is less than n), and the counter in all leaves ends up at 0. Therefore S is a solution for the control-state reachability problem.

This concludes the proof of Proposition 18.

From Propositions 13 and 18 we obtain Proposition 11

Observation 19 *There exists a similar reduction when considering the logic that allows node-labeling and formulas of the type $\psi := a$. In this case, the construction of $\mathcal{A}^k_\varphi$ can be easily extended so as to prove the analog of Proposition 13 in the presence of these type of formulas.*

B.2 Adding Boolean and *Until* operators: $\mathbf{LRV_1^D}$

We extend $\mathrm{LRV}_1^{\mathrm{D}-}$ with $\wedge$, $\vee$, $\neg$ and EU. We will combine the idea of [5, Section 3.2] with our previous approach of §B.1 in order to deal with this more expressive logic, to obtain the following generalization of Proposition 11:

Proposition 20. *There is a computable reduction from SAT_k-$\mathrm{LRV}_1^{\mathrm{D}}$ to* $\mathrm{CSReach}(\mathrm{1VASS}_k)$.

Let φ be a formula. We define $cl(\varphi)$ to be the standard *closure* of φ: the smallest F set of formulas that contains φ, is closed under subformulas, and satisfies the following conditions:

- If $\psi \in F$ and ψ is not of the form $\neg\psi_1$ for some ψ_1, then $\neg\psi \in F$.
- If $\mathrm{EU}(\psi_2, \psi_1) \in F$ then $\mathrm{EX}(\mathrm{EU}(\psi_2, \psi_1)) \in F$.

An atom of φ is a subset A of $cl(\varphi)$ which is maximally consistent in that it satisfies the following conditions:

- For every $\neg\psi \in cl(\varphi)$, we have $\neg\psi \in A$ iff $\psi \notin A$.
- For every $\psi_1 \wedge \psi_2 \in cl(\varphi)$, we have $\psi_1 \wedge \psi_2 \in A$ iff ψ_1 and ψ_2 are in A.
- For every $\psi_1 \vee \psi_2 \in cl(\varphi)$, we have $\psi_1 \vee \psi_2 \in A$ iff ψ_1 or ψ_2 is in A.
- For every $\mathrm{EU}(\psi_2, \psi_1) \in cl(\varphi)$, we have $\mathrm{EU}(\psi_2, \psi_1) \in A$ iff either $\mathrm{EX}(\psi_2) \in A$ or both $X(\psi_1) \in A$ and $\mathrm{EX}(\mathrm{EU}(\psi_2, \psi_1)) \in A$.

We denote by $Atom(\varphi)$ the set of atoms of φ. Let $A, A_1, \ldots, A_i \in Atom(\varphi)$. We say that A is **1-consistent** with $A_1, \ldots, A_i$ if for every $\mathrm{EX}(\psi) \in cl(\varphi)$, we have $\mathrm{EX}(\psi) \in A$ iff $\psi \in \bigcup_{1 \leq j \leq i} A_j$.

We recall the definition of EX-length given in §5.

As in §B.1, we show that for any formula φ of $\mathrm{LRV}_1^{\mathrm{D}}$, there is a $\mathrm{1VASS}_k$ $\mathcal{B}_\varphi^k$ and sets $Q_0, \widehat{Q}$ such that $\mathrm{SAT}_k(\varphi)$ iff $\mathrm{CSReach}(\mathcal{B}_\varphi^k, Q_0, \widehat{Q})$.

The idea behind the definition of $\mathcal{B}_\varphi^k$ is similar to the one of $\mathcal{A}_\varphi^k$ defined in §B.1. However, $\mathcal{B}_\varphi^k$ will encode more information. For the definition of $\mathcal{B}_\varphi^k$, recall the definition of (d,k)-frame given in §B.1. We say that an atom A is **locally consistent** with a (d,k)-frame F if, for r the root of F, and for all $\gamma \in A$ of the form $v \star \mathrm{EF}v$ or $\neg v \star \mathrm{EF}v$ (for $\star \in \{\approx, \not\approx\}$), r is labeled appropriately in ℓ_1.

The automaton $\mathcal{B}_\varphi^k$. We define the $\mathrm{1VASS}_k$ $\mathcal{B}_\varphi^k$ as follows:

- The set of states Q of $\mathcal{B}_\varphi$ consists of the set
$$\{(F, A) \in \mathcal{F}_{d,k} \times Atom(\varphi) \mid A \text{ is locally consistent with } F\}.$$
- Unary rules. Let F_1 and F_2 be (d,k)-frames, and let A be an atom.
  - $(F_1, A) \xrightarrow{-n} (F_2, A)$ if F_1 and F_2 are as in the rule $F_1 \xrightarrow{-n} F_2$ of $\mathcal{A}_\varphi^k$.
  - $(F_1, A) \xrightarrow{+1} (F_2, A)$ if F_1 and F_2 are as in the rule $F_1 \xrightarrow{+1} F_2$ of $\mathcal{A}_\varphi^k$.
- Branching rules: $(F, A) \to ((F_1, A_1), \ldots, (F_i, A_i))$ (with $i \leq k$), if F is 1-consistent with $F_1, \ldots, F_i$, and A is 1-consistent with $A_1, \ldots, A_i$.

We now define the two sets Q_0 and $\widehat{Q}$, to be used as inputs of the control-state reachability problem.

- Q_0 is the set of all $(F, A) \in \mathcal{F}_{d,k} \times \mathit{Atom}(\varphi)$ which are *initial*. We say that (F, A) is initial iff F satisfies the same conditions of an initial state of $\mathcal{A}^k_\varphi$ and $\varphi \in A$.
- $\widehat{Q}$ is the set of all the states of the form (F, A), where F is the only (d, k)-frame that consists solely of one node, and where A is such that if $\mathrm{EU}(\rho, \psi) \in cl(\varphi)$ then $\mathrm{EU}(\rho, \psi) \notin A$.

B.3 The general case: $\mathbf{LRV}^{\mathbf{D}}_{\boldsymbol{n}}$

To prove Theorem 3 it is enough to introduce a small modification to the construction of $\mathcal{F}_{d,k}$ as seen in Proposition 11, changing the dimension (polynomially in n) of the codomains of ℓ_1 and of ℓ_2 in order to maintain information for all the variables. For the case of n disjoint variables and $\varphi \in \mathrm{LRV}^{\mathrm{D}}_n$ we make the following changes to the definition of the (d, k)-frames that will constitute the states of the automaton $\mathcal{B}^k_\varphi$:

- Let $V_i = \{([v_i \approx \mathrm{EF}v_i], [v_i \not\approx \mathrm{EF}v_i]), ([\neg v_i \approx \mathrm{EF}v_i], [v_i \not\approx \mathrm{EF}v_i]), ([v_i \approx \mathrm{EF}v_i], [\neg v_i \not\approx \mathrm{EF}v_i]), ([\neg v_i \approx \mathrm{EF}v_i], [\neg v_i \not\approx \mathrm{EF}v_i])\}$. Now we have $\ell_1 : N \to \prod_{1 \le i \le n} V_i$.
- $\ell_2 : N \to \prod_{1 \le i \le n} \{\epsilon_{v_i}, \oplus_{v_i}, \ominus_{v_i}\}$.
- $\equiv_1, \ldots, \equiv_n$ are equivalence relations over the nodes of the frames.

The notions of validity and 1-consistency between frames are then adjusted accordingly. There are now n counters in the automaton, and n instances of our previous unary rules for $\mathcal{B}^k_\varphi$, one for every disjoint variable.

B.4 Complexity

We will make a short analysis on the complexity of the reduction corresponding to Theorem 3. For this part, we will assume we are working without labels, but their addition to the logic does not change the complexity classes in our results.

First, we rapidly note that the maximum size of an entry in a unary rule of the n-counter automaton $\mathcal{B}^k_\varphi$ is bounded by k^d, corresponding with a (d, k)-frame of maximum depth where all leaves are labeled via ℓ_2 with $\ominus$ and all leaves belong to different equivalence classes; such (d, k)-frame is a point of decrement of value k^d, and there cannot be points of decrement of higher value.

By making an analysis on the size of $\mathcal{F}_{d,k}$ (the set of valid (d, k)-frames) and of $\mathit{Atom}(\varphi)$ (which is used for the temporal portion), we can obtain an upper bound on the size of the automaton $\mathcal{B}^k_\varphi$ of our reduction in Subsection B.2, generalized to $\mathrm{LRV}^{\mathrm{D}}_n$. Each state of $\mathcal{B}^k_\varphi$ is basically a k-ranked tree of depth at most d (equipped with a node-labeling function and an equivalence class) and a set of (roughly) subformulas of φ. There are $O(2^{p_1(|\varphi|)})$ such sets, for some polynomial p_1, where $|\varphi|$ denotes the number of subformulas of φ. On the other hand, there are $O(k^{d+1})$

many k-ranked tree of depth at most d, and so there are $O((k^{d+1})^{k^{d+1}})$ many such trees with a binary relation defined on its nodes. Further, each leaf can be labeled with $p_2(n)$-many labels, where $p_2(n)$ is the product of the size of the codomains of the functions ℓ_1 and ℓ_2 for (d,k)-frames in the n-dimensional case, as in Subsection B.3. So there are $O((k^{d+1})^{k^{d+1}} \cdot p_2(n)^{k^{d+1}})$ many k-ranked trees of depth at most d equipped with node-labeling functions and an equivalence class.

Let $\mathrm{LRV}^{\mathrm{D}}_{n,d}$ be the fragment of $\mathrm{LRV}^{\mathrm{D}}_{n}$ where each formula has EX-length at most d. We have obtained:

Proposition 21. *Given $\varphi \in \mathrm{LRV}^{\mathrm{D}}_{n,d}$, the number of states of $\mathcal{B}^k_\varphi$ is*

$$O(p(n)^{k^{d+1}} \cdot (k^{d+1})^{k^{d+1}} \cdot 2^{p(|\varphi|)})$$

for some polynomial p.

Observe also that the reduction of Theorem 3 can be done using only exponential space, as we can codify in exponential space the lists of states and the rules between states, while checking if each state represents a valid (d,k)-frame and if each rule corresponds to one of our unary or branching rules.

Finally, in order to use the Prop. 1 from [8], we need to translate our $n\mathrm{VASS}_k$ $\mathcal{B}^k_\varphi$ into a BVAS $B = \langle n + \tilde{n}, R_1, R_2 \rangle$ Where $R_1, R_2 \subseteq \mathbb{Z}^{n+\tilde{n}}$ are unary and binary rules, respectively, and we need to measure the maximum binary size of entries in the rules R_1, R_2. We also need to specify a set of axioms, which adds a linear size to the input. We will show how we can build a branching VASS $\mathcal{C} = \langle C, U_C, B_C \rangle$ with a constant number of states; a fixed increase $\tilde{n}$ in the dimension; and a new bound (that dominates the bound of $\log_2(k^d)$ for $\mathcal{B}^k_\varphi$) to the binary size of the maximum entry of the rules, a bound that is logarithmic on $|\mathcal{B}^k_\varphi|$. Afterwards, we can translate $\mathcal{C}$ to a BVAS in a standard way, which does not increase our complexities, and only needs a single axiom.

Let $q_0, \ldots, q_N$ the states of $\mathcal{B}^k_\varphi$. $\mathcal{C}$ will have three states: $C = q_a, q_b, q_c$. For each unary rule on $\mathcal{B}^k_\varphi$ of the form $q_i \xrightarrow{\bar{v}} q_j$, U_c contains the rules:

- $q_a \xrightarrow{(\bar{v},\bar{w})} q_b$, where $\bar{w} = (-i, -(N-i), j, N-j, 0, 0)$.
- $q_b \xrightarrow{(\bar{v},\bar{w})} q_a$, where $\bar{w} = (j, N-j, -i, -(N-i), 0, 0)$.
- $q_a \xrightarrow{(\bar{v},\bar{w})} q_c$, where $\bar{w} = (-i, -(N-i), 0, 0, j, N-j)$.
- ...And so on for all combinations $q_i \to q_j$ with $i, j \in \{a, b, c\}$.

For branching rules on $\mathcal{B}^k_\varphi$ of the form $q_k \to (q_i, q_j)$, B_C contains the rules[6]:

- $q_a, q_b \xrightarrow{\bar{w}} q_c$, where $\bar{w} = (-i, -(N-i), j, -(N-j), k, N-k)$.
- And so on for all combinations with the three states.

For branching rules of higher branching, the idea is similar, but we have to introduce new counters in order to simulate k-branching with only binary branching.

[6] The branching rules B_C of $\mathcal{C}$ will allow the addition of a vector, in order to facilitate a later transition to a BVAS; this feature is unessential, but useful for the purpose of clarity.

C Proof of Theorem 6

We will start by proving the result for $n = 1$, as the general case follows from small modifications to the proofs for this restricted case. Let $\mathcal{C}^k = \langle Q, U, B\rangle$ be a 1VASS_k, let $q_0 \in Q$, and let $\widehat{Q} \subseteq Q$. We can assume without loss of generality that all unary rules in U of the form $q \xrightarrow{c} q'$ have either $c = 1$ or $c = -1$. We will define[7] a formula $\varphi^{\mathcal{C}^k}$ of $\text{LRV}_1^{\text{D}}(\text{AG}^+_{\approx})$, over an adequate set of labels, such that $\text{CSReach}(\mathcal{C}^k, q_0, \widehat{Q})$ iff $\text{SAT}_k(\varphi^{\mathcal{C}^k})$.

For the signature of the logic, we will consider a set of labels

$$L = (U \cup B \cup \{*\}) \times \underline{k},$$

where $*$ is a symbol to represent that the node is a 'dummy node' that will be ignored in the translation to an incrementing derivation tree; also, dummy nodes will always be to the right of nodes labeled with any (t, i), for $t \neq *$. We notate

$$\varphi_{\text{inc}} = \bigvee_{1\leq j\leq k} \bigvee_{t=q\xrightarrow{+1}q'} (t, j) \qquad \text{and} \qquad \varphi_{\text{dec}} = \bigvee_{1\leq j\leq k} \bigvee_{t=q\xrightarrow{-1}q'} (t, j).$$

For $t = q \xrightarrow{v} \dots$ a branching or unary rule, we notate $\pi_h(t) = q$. We want to construct a formula $\varphi^{\mathcal{C}^k}$ whose satisfiability over k-ranked data trees is equivalent to $\text{CSReach}(\mathcal{C}^k, q_0, \widehat{Q})$.

The truth of $\varphi^{\mathcal{C}^k}$ in a data tree T, r (where r is the root of T) expresses:

- $\pi_h(\pi_1(\ell(r))) = q_0$
- For each node x of T, we have:
  - i. If $\pi_1(\ell(x)) = q \xrightarrow{c} q'$ then $T, x \models \text{EX}(a, 1) \wedge \text{EX}(*, 2) \wedge \dots \wedge \text{EX}(*, k)$, where $\pi_h(a) = q'$.
  - ii. If $\pi_1(\ell(x)) = q \to (q_1, \dots q_j)$, with $j > 0$, then $T, x \models \text{EX}(a_1, 1) \wedge \dots \wedge \text{EX}(a_j, j) \wedge \text{EX}(*, j + 1) \wedge \dots \wedge \text{EX}(*, k)$, where $\pi_h(a_i) = q_i$.
  - iii. If $\pi_1(\ell(x)) = *$ then $T, x \models \neg\text{EX}(\top)$.
  - iv. If $\pi_1(\ell(x)) = q \to \bar{\emptyset}$, then $T, x \models \neg\text{EX}(\top)$.
  - v. If $T, x \models \neg\text{EX}(\top)$ and $\pi_1(\ell(x)) = q \to \bar{\emptyset}$, then $q \in \widehat{Q}$.
  - vi. If $\pi_1(\ell(x)) = q \xrightarrow{+1} q'$ then $T, x \models v \approx \text{EF}v \wedge \text{AG}_{\approx v}(\varphi_{\text{dec}})$.

In the framework of k-ranked data trees, item i and item ii give nodes tagged with a non-empty rule, exactly k-children, where the first ones are associated with the rule itself, and the rest are dummy nodes. Item iii ensures that dummy nodes are leaves, and item iv ensures that nodes corresponding to the empty rule are leaves. Item vi says that nodes tagged with an increment rule have a descendant in its same class, and that all descendants in the same class are tagged with a decrement rule. Item v assures that leaves of the tree that are tagged with an empty rule correspond with states in $\widehat{Q}$. Observe that the conditions i, ii, iii, iv,

[7] Abuse of notation: $\varphi^{\mathcal{C}^k}$ also depends on q_0 and $\widehat{Q}$.

and v taken together imply that $T, x \models \neg \mathrm{EX}\top$ iff either $\pi_1(\ell(x)) = q \to \bar{\emptyset}$ or $\pi_1(\ell(x)) = *$.

We now write the formulas of the logic that correspond to all these conditions.

$$\varphi_0^{\mathcal{C}^k} = \bigvee_{\substack{1 \le z \le k \\ t = q_0 \to \cdots \in U \cup B}} (t, z)$$

$$\varphi_1^{\mathcal{C}^k} = \bigwedge_{\substack{1 \le z \le k \\ t = q \xrightarrow{c} q' \in U}} \left((t, z) \to \left(\bigvee_{\pi_h(a) = q'} \mathrm{EX}(a, 1) \wedge \bigwedge_{2 \le j \le k} \mathrm{EX}(*, j) \right) \right) \tag{i}$$

$$\varphi_2^{\mathcal{C}^k} = \bigwedge_{\substack{1 \le z \le k \\ t = q \to (q_1, \ldots q_j) \in B}} \left((t, z) \to \bigvee_{\substack{a_1, \ldots, a_j \in U \cup B \\ \text{s.t.} (\forall i) \pi_h(a_i) = q_i}} \left(\bigwedge_{1 \le i \le j} \mathrm{EX}(a_i, i) \wedge \bigwedge_{j < i \le k} \mathrm{EX}(*, i) \right) \right) \tag{ii}$$

$$\varphi_3^{\mathcal{C}^k} = \bigwedge_{1 \le z \le k} (*, z) \to \neg \mathrm{EX}\top \tag{iii}$$

$$\varphi_4^{\mathcal{C}^k} = \bigwedge_{\substack{1 \le z \le k \\ t = q \to \bar{\emptyset}}} (t, z) \to \neg \mathrm{EX}\top \tag{iv}$$

$$\varphi_5^{\mathcal{C}^k} = \left(\neg \mathrm{EX}\top \wedge \bigvee_{\substack{1 \le z \le k \\ t = q \to \bar{\emptyset}}} (t, z) \right) \to \bigvee_{\substack{1 \le z \le k \\ t = q \to \bar{\emptyset}, q \in \widehat{Q}}} (t, z) \tag{v}$$

$$\varphi_6^{\mathcal{C}^k} = \varphi_{\mathrm{inc}} \to \left(v \approx \mathrm{EF} v \wedge \mathrm{AG}_{\approx v}(\varphi_{\mathrm{dec}}) \right) \tag{vi}$$

For $i = 1 \ldots 6$, let $\psi_i^{\mathcal{C}^k} = \varphi_i^{\mathcal{C}^k} \wedge \mathrm{AG}(\varphi_i^{\mathcal{C}^k})$ and finally let

$$\varphi^{\mathcal{C}^k} = \varphi_0^{\mathcal{C}^k} \wedge \bigwedge_{1 \le i \le 6} \psi_i^{\mathcal{C}^k}.$$

Observe that the size of φ is polynomial on the size of $\mathcal{C}^k$. Note also that $\mathrm{AG}_{\approx}$ appears only positively in $\varphi^{\mathcal{C}^k}$, and hence $\varphi^{\mathcal{C}^k} \in \mathrm{LRV}_1^{\mathrm{D}}(\mathrm{AG}_{\approx}^+)$..

Proposition 22. *If there is a solution to the problem of the* $\textsc{CSReach}(\mathcal{C}^k, q_0, \widehat{Q})$ *then there is a solution to the SAT$_k$ problem of* $\varphi^{\mathcal{C}^k}$ *over the logic* $\mathrm{LRV}_1^{\mathrm{D}}(\mathrm{AG}_{\approx}^+)$.

Proof (Sketch of the proof). Let S be an incrementing derivation tree that is a solution of $\textsc{CSReach}(\mathcal{C}^k, q_0, \widehat{Q})$. We want to prove that there is a k-ranked tree T_S where the root r satisfies $\varphi^{\mathcal{C}^k}$. Nodes of T_S will correspond to the nodes of S, and their labels will be determined by the rule that is invoked on them in S and by their position as children of some other node (and we choose to assign $\pi_2(\ell(r)) = 1$). When a node has at least one child but strictly less than k, we add dummy nodes (with labels of the form $(*, i)$) in the proper order so as to arrive to

exactly k children. With all this, $\psi_1^{\mathcal{C}^k} \wedge \psi_2^{\mathcal{C}^k} \wedge \psi_3^{\mathcal{C}^k} \wedge \psi_4^{\mathcal{C}^k}$ is satisfied at r. Since S is a solution for $\text{CSREACH}(\mathcal{C}^k, q_0, \widehat{Q})$, $\varphi_0^{\mathcal{C}^k}$ holds at the root of T_S, and all leaves z of T_S with $\pi_1(\ell(z))$ of the form $q \to \bar{\emptyset}$ have $\pi_h(\pi_1(\ell(z))) \in \widehat{Q}$, and thus $\psi_5^{\mathcal{C}^k}$ is satisfied at r. For the equivalence relation, we first put all nodes in different equivalence classes. Then, since S is a solution for the control-state reachability problem, whenever there is a $(+1)$ increment rule there is a (-1) decrement rule further down, and we can make this assignation injectively, yielding corresponding joining of the equivalence classes. This is enough to satisfy $\psi_6^{\mathcal{C}^k}$ at r.

Proposition 23. *If there is a solution to the SAT_k problem of $\varphi^{\mathcal{C}^k}$ over the logic $\text{LRV}_1^{\text{D}}(\text{AG}_{\approx}^{+})$ then there is a solution to $\text{CSREACH}(\mathcal{C}^k, q_0, \widehat{Q})$.*

Proof (Sketch of the proof). Let T be a k-ranked tree whose root r satisfies $\varphi^{\mathcal{C}^k}$. We want to construct S_T, an incrementing derivation tree of $\mathcal{C}^k$ that is a solution to the control-state reachability problem. The idea is that the conjuncts $\psi_1^{\mathcal{C}^k}$, $\psi_2^{\mathcal{C}^k}$, and $\psi_4^{\mathcal{C}^k}$ provide a natural translation from the nodes and labels of each node of T to the structure and invoked rules of S_T. Because of $\psi_3^{\mathcal{C}^k}$, we can ignore those nodes $x \in T$ with $\pi_h(\ell(x)) = *$. $\varphi_0^{\mathcal{C}^k}$ ensures that the derivation tree starts from q_0, while $\psi_5^{\mathcal{C}^k}$ implies that all leaves of S_T are in $\widehat{Q}$. To check that all leaves of S_T have the counter set at 0, observe that $\psi_6^{\mathcal{C}^k}$ ensures that whenever an increment rule ($+1$ to the counter) is invoked, there is further down in the derivation an application of a decrement rule (-1 to the counter), furthermore, there is at least one decrement for each increment, since $\psi_6^{\mathcal{C}^k}$ also ensures that two nodes labeled with some increment rule cannot be in the same equivalence class if one is ancestor of the another.

For the general case of arbitrary n, we can assume without loss of generality that all unary rules in U of the form $q \xrightarrow{v} q'$ have either $v = e_i$ or $v = -e_i$ for some $1 \leq i \leq n$. Now, adding for each counter in $\mathcal{C}^k$ adequate new versions of φ_{inc}, φ_{dec}, and $\varphi_6^{\mathcal{C}^k}$, we can extend the previous arguments, yielding a proof of Theorem 6. Observe that this does not necessitate an increase in the number of variables of the logic.

D Proof of Theorem 7

To show SAT_k-$\text{LRV}_n^{\text{D}}(\text{AG}_{\approx}^{+})$ to $\text{CSREACH}(n\text{VASS}_k)$ we will proceed incrementally, as it was done in Section B. We begin with a simple logic, notated $\text{LRV}_1^{\text{D}-}(\text{AG}_{\approx}^{+})$, which consists only of formulas $\varphi \in \text{LRV}_1^{\text{D}}(\text{AG}_{\approx}^{+})$ that are conjuncts of terms of the form $v \star \text{EX}^i v$, $v \star \text{EF} v$, or their negation, where $\star \in \{\approx, \not\approx\}$, and, all occurrences of $\text{AG}_{\approx v}(\varphi)$ are of form $\text{AG}_{\approx v}(a)$, where a is a label, and whose labels rage over a finite fixed set L.

Let φ be a formula $\varphi \in \text{LRV}_1^{\text{D}-}(\text{AG}_{\approx}^{+})$, let

$$H = \{a \in L \mid \text{AG}_{\approx v}(a) \text{ is a subformula of } \varphi\}$$

and let $h = \#H$. We construct a $(h+1)$VASS$_k$ $\mathcal{C}^k_\varphi$ (via a procedure similar to that of Section B, using Observation 19 to take labels into account), with two distinguished set of states Q_0 and $\widehat{Q}$ such that $\mathrm{SAT}_k(\varphi)$ iff $\textsc{CSReach}(\mathcal{C}^k_\varphi, Q_0, \widehat{Q})$ (recall Observation 12).

Valid (d,k)-frames We adjust our notion of valid (d,k)-frames to this framework. A (d,k)-frame is now a tuple $F = \langle N, E, \ell, \ell_1, \ell_2, \equiv\rangle$ that satisfies similar conditions as those of Section B, and $\ell : N \to L$. We state next the differences with the (d,k)-frames seen in Section B.1.

- For $x \in N$, $\ell_1(x)$ is a tuple $([\star_1 v \approx \mathrm{EF}v], [\star_2 v \not\approx \mathrm{EF}v], S)$, where $\star_1, \star_2$ can be either the empty string or $\neg$, and where S is a potentially empty set containing elements of the form $\mathrm{AG}_{\approx v}(a)$ with $a \in L$.
- *Validity:* We extend the validity conditions of Section B with the following rules:
  - If $x \in N$ satisfies $\pi_1(\ell_1(x)) = [v \approx \mathrm{EF}v]$ and $\mathrm{AG}_{\approx v}(a) \in \pi_3(\ell_1(x))$, and if $y \in N$ is a descendant of x with $x \equiv y$, then $\ell(y) = a$.
  - If $x \in N$ satisfies $\mathrm{AG}_{\approx v}(a), \mathrm{AG}_{\approx v}(b) \in \pi_3(\ell_1(x))$ for $a \neq b$, then $\pi_1(\ell_1(x)) = [\neg v \approx \mathrm{EF}v]$.
- We change the labeling function ℓ_2, such that its codomain is

$$\{\epsilon, \ominus_{a_1}, \ldots, \ominus_{a_n}, \ominus, \oplus, \oplus_{a_1}, \ldots, \oplus_{a_n}\}.$$

- For a root $r \in N$, $\ell_2(r) = \oplus_{a_i}$ iff $\pi_1(\ell_1(x)) = [v \approx \mathrm{EF}v]$ and $\mathrm{AG}_{\approx v}(a) \in \pi_3(\ell_1(r))$ and there is no descendant in the same frame in the same equivalence class.
- For a root $r \in N$, $\ell_2(r) = \oplus$ iff $\pi_1(\ell_1(x)) = [v \approx \mathrm{EF}v]$ and $\pi_3(\ell_1(r)) = \emptyset$ and there is no descendant in the same frame in the same equivalence class.
- For leaves $z \in N$, $\ell_2(z) = \ominus_{a_i}$ implies that $\ell(z) = a_i$, $\mathrm{AG}_{\approx v}(a_i) \in \pi_3(\ell_1(z))$ and there is no node in the frame at distance $< d$ from the root with the same equivalence class as z.
- For leaves $z \in N$, $\ell_2(z) = \ominus$ implies that $\pi_3(\ell_1(z)) = \emptyset$ and that there is no node in the frame at distance $< d$ from the root with the same equivalence class as z.

A (d,k)-frame is an **a-point of decrement of value** p iff it has a maximum of p leaves $z_1, \ldots, z_p$ in different equivalence classes and with $\ell_2(z_j) = \ominus_a$ for all j. It is a **neutral point of decrement of value** p iff it has a maximum of p leaves $z_1, \ldots, z_p$ in different equivalence classes and with $\ell_2(z_j) = \ominus$ for all j. A (d,k)-frame is an **a-point of increment** if its root r has $\ell_2(r) = \oplus_a$ and it is not an a-point of decrement. It is a **neutral point of increment** if $\ell_2(r) = \oplus$ and it is not an a-point of decrement for any $a \in L$.

The automaton $\mathcal{C}^k_\varphi$ We define the $(h+1)$VASS$_k$ $\mathcal{C}^k_\varphi$ and the sets $Q_0, \widehat{Q}$ similarly as in the case of $\mathcal{A}^k_\varphi$ in Section B, using Observation 19 to take labels into account with the notion of 1-consistency between the (d,k)-frames just defined.

We define $\mathcal{C}^k_\varphi$ as follows:

– We fix the dimension to be $h+1$. Let $H = \{a_1, \ldots, a_h\}$. For $i \leq h$, the i-th coordinate will correspond with the label a_i. Intuitively, i-th coordinate will we used to count instances of $v \approx \mathrm{EF}v \wedge \mathrm{AG}_{\approx v}(a)$, yet unsatisfied.
– The set of states of $\mathcal{C}^k_\varphi$ is $\mathcal{F}_{d,k}$, the set consisting of all valid (d,k)-frames as defined above.
– Unary rules. Let F_1 and F_2 be (d,k)-frames. We have the following rules: [8]
  - $F_1 \xrightarrow{-ne_i} F_2$ if F_1 is an a_i point of decrement of value n, and F_2 is equal to F_1, except that $\ell_2^{F_2}$ is defined as follows:

$$\ell_2^{F_2}(x) = \begin{cases} \epsilon & x \text{ is a leaf of } F_1 \text{ with } \ell_2^{F_1}(x) = \ominus_{a_i}; \\ \ell_2^{F_1}(x) & \text{otherwise.} \end{cases}$$

  - $F_1 \xrightarrow{-me_{h+1}} F_2$ if either:
    * F_1 is a neutral point of decrement of value m and F_2 is equal to F_1, except that $\ell_2^{F_2}$ is defined as follows:

$$\ell_2^{F_2}(x) = \begin{cases} \epsilon & x \text{ is a leaf of } F_1 \text{ with } \ell_2^{F_1}(x) = \ominus; \\ \ell_2^{F_1}(x) & \text{otherwise.} \end{cases}$$

    or
    * F_1 is a a_i-point of decrement of value $n \geq m$ for some a_i, and F_2 is an a_i point of decrement of value $n-m$ such that F_2 has m fewer instances of distinct equivalence classes of nodes with $\ominus_{a_i}$, that is: F_2 is equal to F_1, except for $\ell_2^{F_2}$: for any node x, $\ell_2^{F_1}(x) = \ominus_{a_i}$ implies either $\ell_2^{F_2}(x) = \ominus_{a_i}$ or $\ell_2^{F_2}(x) = \ominus_{a_i}$, and if $\ell_2^{F_1}(x) \neq \ominus_{a_i}$ then $\ell_2^{F_1}(x) = \ell_2^{F_2}(x)$.
  - $F_1 \xrightarrow{e_i} F_2$ if F_1 is an a_i point of increment, and F_2 is equal to F_1, except that $\ell_2^{F_2}$ is defined as follows:

$$\ell_2^{F_2}(x) = \begin{cases} \epsilon & x \text{ is the root of } F_1; \\ \ell_2^{F_1}(x) & \text{otherwise.} \end{cases}$$

  - $F_1 \xrightarrow{e_{h+1}} F_2$ if F_1 is a neutral point of increment, and F_2 is equal to F_1, except that $\ell_2^{F_2}$ is defined as follows:

$$\ell_2^{F_2}(x) = \begin{cases} \epsilon & x \text{ is the root of } F_1; \\ \ell_2^{F_1}(x) & \text{otherwise.} \end{cases}$$

– Branching rules: $F \to (F_1, \ldots, F_i)$, if F is 1-consistent with $F_1, \ldots, F_i$.

We define the sets $Q_0, \widehat{Q}$:

– Q_0 consists of initial frames. A (d,k)-frame F of root r is initial iff the following conditions hold:

[8] There will we a non-deterministic choice of the order of decrements if there are leaves with different $\ominus_{a_j}$, but the order of these operations is not relevant.

- F, r satisfies all terms of φ of the form $v \star \mathrm{EX}^i v$ or $\neg v \star \mathrm{EX}^i v$;
- if $v \approx \mathrm{EF}v$ (resp. $v \not\approx \mathrm{EF}v$) is a positive conjunct of φ, then the root $r \in N^F$ satisfies $\pi_1(\ell_1^F(r)) = [v \approx \mathrm{EF}v]$ (resp. $\pi_2(\ell_1^F(r)) = [v \not\approx \mathrm{EF}v]$);
- if $v \approx \mathrm{EF}v$ (resp. $v \not\approx \mathrm{EF}v$) is a negative conjunct of φ, then for all descendants y of the root $r \in N^F$, it holds that $\pi_1(\ell_1^F(y)) = [\neg v \approx \mathrm{EF}v]$ (resp. $\pi_2(\ell_1^F(y)) = [\neg v \not\approx \mathrm{EF}v]$).
- If $b \in L$ and b is a conjunct of φ, then $\ell(r) = b$, for r the root of F.
- If $\mathrm{AG}_{\approx v}(a)$ is a conjunct of φ, then $\mathrm{AG}_{\approx v}(a) \in \pi_3(\ell_1(r))$, for r the root of F.

– $\widehat{Q}$ is the set of (d,k)-frames in $\mathcal{F}_{d,k}$ that consists solely of one node.

The analogous results to Propositions 13 and 18 hold for $\varphi \in \mathrm{LRV}_1^{\mathrm{D}-}(\mathrm{AG}_{\approx}^{+})$ and the corresponding $(h+1)\mathrm{VASS}_k$ $\mathcal{C}_\varphi^k$ and the states $Q_0, \widehat{Q}$. Hence we arrive at

Proposition 24. *$SAT_k(\varphi)$ iff $\textsc{CSReach}(\mathcal{C}_\varphi^k, Q_0, \widehat{Q})$.*

As in Proposition 20, we can extend these results to the full logic, and then extend for the general case where we allow any formula $\mathrm{AG}_{\approx v}(\eta)$ to appear in φ.

E Proof of Theorem 8

Proof (Complete proof of Theorem 8).

Using the merging rules as described in Section 7, the reduction from $\mathrm{LRV}^{\mathrm{D}}$ to VASS_k of Section 5 and Appendix B can be modified to obtain a reduction from LRV to MVASS_k. Frames and its notion of validity are extended to treat set of variables. In particular, now the points of increment and decrement are always relative to a set of variables. This follows, very roughly, the idea of coding from [7] in the setup built in Section 5, but now some special care must be considered because of the non-linearity of a tree. One must decide in advance to which leaf of the frame the satisfaction of data demands will be delegated. The resulting MVASS_k now has dimension *exponential* in the number of variables of the input formula. Concretely, in order to encode this logic we need to make the following changes to the set of frames $\mathcal{F}_{d,k}$ we work with.

First of all, the labelling function ℓ_1 now labels pairs of *sets* of formulas. These formulas labelled by ℓ_1 are of the form

– in the first component $u \approx \mathrm{EF}v$ or $\neg(u \approx \mathrm{EF}v)$
– in the second component $u \not\approx \mathrm{EF}v$ or $\neg(u \not\approx \mathrm{EF}v)$

for any pair of variables u, v used in the input formula. For simplicity, we write $\psi \in \ell_1(x)$ (or, alternatively, that x is ℓ_1-labelled with ψ) to denote that ψ is either in the first or second component of $\ell_1(x)$.

Further, instead of having one equivalence relation $\equiv$ over the set of nodes, we have an equivalence relation $\equiv$ over pairs (x, u) where x is a node of hte frame and u an attribute variable of the input formula φ. This is to account for the possibility that different attributes can have the same data value.

In light of this, the formulas labeled by ℓ_1 must 'respect' $\equiv$. That is, if $u \approx \mathrm{EF}v \in \ell_1(x)$ [resp. $u \not\approx \mathrm{EF}v \in \ell_1(x)$] and $(x,u) \equiv (x,u')$ then $u' \approx \mathrm{EF}v \in \ell_1(x)$ [resp. $u' \not\approx \mathrm{EF}v \in \ell_1(x)$].

More importantly, the labelling ℓ_2 must be changed to reflect the fact that

(1) there may be several demands for the same attribute, as a result of formulas like $u \approx \mathrm{EF}v \wedge u' \approx \mathrm{EF}v$ (as we will see next, this is the reason for the first parameter of $\oplus$),
(2) there may be several attributes in a demand for equality, as a result of formulas like $u \approx \mathrm{EF}v \wedge u \approx \mathrm{EF}v'$,
(3) a point of decrement needs to be a point that has some attributes U which are not connected by equality to any 'local' ancestor and they are connected possibly to some other attributes V in the descendants.

Formally, the mapping ℓ_2 now labels nodes with $\oplus(U,V)$ and/or $\ominus(U,V)$, where U, V are sets of attribute variables. Each node x can receive more than one $\oplus$ or $\ominus$ label, that is, ℓ_2 is a function from nodes to subsets of $\{\oplus(U,V) \mid U,V \subseteq \mathbb{V}\} \cup \{\ominus(U,V) \mid U,V \subseteq \mathbb{V}\}$, assuming $\mathbb{V}$ is the set of variables used in the input formula.[9] The idea is that $\oplus(U,V)$ holding at x means that there must be a data value appearing in the subtree at x under all the variables of V (possibly at different nodes), which is equal to the u-attribute of the k-ancestor of x, for every $u \in U$. On the other hand, $\ominus(U,V)$ holding at x means that the data value of the U-attributes of x (which are all the same) do no not appear in any i-ancestor of x ($i \leq k$), and they will appear in the future with attributes V.

We add the following conditions.

- For any two labels $\oplus(U,V)$ and $\oplus(U',V')$ at the same node, U and U' are disjoint. For any two labels $\ominus(U,V)$ and $\ominus(U',V')$ at the same node, U and U' are disjoint.
- For every leaf x which is ℓ_2-labeled with $\oplus(U,V)$ we have that U is an equivalence class of $\{(u,v) \mid (r,u) \equiv (r,v)\}$, where r is the root node.
- For every leaf x which is ℓ_2-labeled with $\ominus(U,V)$ we have that for some $v \in \mathbb{V}$ we have

$$U = \{u \mid (x,u) \equiv (x,v)\},$$
$$V = \{u \mid [v \approx \mathrm{EF}u] \in \ell_1(x)\},$$

  and that there is no ancestor y of x so that $(x,u) \equiv (y,v')$ for some $u \in U$, $v' \in \mathbb{V}$.
- There exists an ℓ_1-labelling $u \approx \mathrm{EF}v$ holding at the root r if, and only if, there exists a node x at some depth i so that either
  - $(r,u) \equiv (x,v)$, or
  - $(r,u) \equiv (x,v')$ for some v' and $v' \approx \mathrm{EF}v$ in $\ell_1(x)$, or
  - $i = k$ and x is ℓ_2-labeled with $\oplus(U,V)$ with $U = \{u' \mid (r,u) \equiv (r,u')\}$ and $v \in V$.

[9] It is worth remarking that $\ell_2(x)$ is always a set of size linear in $|\mathbb{V}|$ due to the next conditions.

- There exists an ℓ_1-labelling $u \not\approx \mathrm{EF}v$ holding at the root r if, and only if, there exists a node x at some depth i so that either
  - $(r, u) \not\equiv (x, v)$, or
  - $(r, u) \equiv (x, v)$ and $v \not\approx \mathrm{EF}v$ in $\ell_1(x)$.

1-step consistency is preserved as before. Now a *point of increment for V* is a frame whose root is labeled with $\oplus(U, V)$ for some U; whereas a *point of decrement for W* is a frame whose root is labeled with $\ominus(U, V)$ for $U \cup V = W$.

Finally, the automaton $\mathcal{A}^k_\varphi$ is built as for the other reduction, with the exception that now its dimension is exponential. As in the previous reductions, the frames are the states of the automaton, where the initial frames are those that do not contain $\oplus/\ominus$ tags at nodes at distance $< k$ from the root, and whose root labeling is consistent with the satisfaction of the formula. In the automaton, we have one coordinate associated with every non-empty subset $V \subseteq \mathbb{V}$ of attribute variables (remember that we use $\bar{e}_V$ to denote $\bar{e}_i$ for the coordinate i associated with V and $\bar{e}_\emptyset$ to denote $\bar{0}$). Unary rules now follow a logic of first decrementing all the $\ominus$ at the root, and then incrementing all the $\oplus$ at the root. (The $\oplus/\ominus$ tags of other nodes are differed to the moment when they will be at the root of a frame.) That is, unary rules $(F_1, -\bar{e}_{U\cup V}, F_2)$ whenever F_1 has $\ominus(U, V)$ at the root, F_2 is just like F_1 but without the $\ominus(U, V)$ at the root. We have unary rules $(F_1, \bar{e}_V, F_2)$ whenever F_1 has no $\ominus$-labels at the root, it has a $\oplus(U, V)$ label at the root, and F_2 is the result of removing $\oplus(U, V)$ at the root. Merging rules are built as it was explained before. That is, we have $(F_1, \bar{0} + B^*, (F'_1, \ldots, F'_{k'}))$ whenever F_1 is 1-consistent with $(F'_1, \ldots, F'_{k'})$, and B consists of all vectors $(\bar{e}_V \bar{e}_{U_1} \cdots \bar{e}_{U_k})$, so that $V \neq \emptyset$ and $V = \bigcup_i U_i$. The partial order $\preceq$ will then be the subset ordering on the components: $i \preceq j$ if the set associated to i is contained in that associated to j. It is not hard to check that if $\mathcal{A}^k_\varphi$ has a solution for CSReach if and only if φ is satisfiable on k-ranked data trees using precisely the same ideas as in the proof of Proposition 13. (See Appendix E.) □