Knowing-How Reasoning with Budgets Recasted: Universal Reachability Problem on VASS

Stéphane Demri[1], Laurent Doyen[1], and Raul Fervari[1,2] ✉

[1] Université Paris-Saclay, ENS Paris-Saclay, CNRS, Laboratoire Méthodes Formelles, 91190 Gif-sur-Yvette, France
[2] FAMAF, Universidad Nacional de Córdoba and CONICET, Argentina

Abstract. We investigate the decidability/complexity status of the model-checking problem for an ability-based logic expressing knowing how assertions and enriched with budget constraints. To do so, we introduce a new control-state reachability problem for complete vector addition systems with states where the transitions are labelled by letters from a finite alphabet. It is required that all runs labelled by a given word lead synchronously to the target control states. First, we show that the model-checking problem involving knowing how can be reduced to the new problem on VASS. Second, we establish Ack-completeness of both problems by properly adapting or using developments about well-structured transition systems, belief functions and length function theorems.

1 Introduction

Ubiquity of VASS for applications. Vector addition systems with states (VASS) [25,23] are ubiquitous models, closely related to Petri nets [40], that can be viewed as Minsky machines without zero-tests [37]. A wide range of decision problems have been considered for such counter machines, see e.g. [35,39,36,26], with new problems popping up regularly, for instance to recast problems from formal verification or database theory. For example, the satisfiability problem for FO2 over data words was shown to be equivalent to the reachability problem for VASS [5,14]. Decidability of the reachability problem was shown in the seminal papers [36,26]. Further developments on this problem are made in [40,27,29,30], including its Ack-completeness in [32,13,31]. The complexity class Ack [42] has been already successful to characterise the complexity of many problems involving counters, see e.g. [38,22,31]. More precisely, Ack is the class of Ackermannian problems closed under primitive-recursive reductions introduced in [42, Sec. 2.3], corresponding to the fast-growing complexity class $\mathcal{F}_\omega$ in the extended Grzegorczyk hierarchy (see details in [42]). In contrast, the covering and boundedness problems for VASS are "only" ExpSpace-complete [35,10,39,19], as well as variants [4,17]. To illustrate briefly the wide range of applications of VASS, let us mention that the covering problem can express the thread-state reachability problem for replicated finite-state programs [24], as well as decision problems for the parameterised verification of ad-hoc networks [16].

Knowing-how logics with budgets. This paper stems from investigations about knowing-how logics that have emerged recently as a novel formalism to express assertions related to the ability of an agent to achieve a certain goal [44,46,45]. This provides formal foundations to the epistemic concept of "knowing how", dedicated to strategic reasoning and automated planning. One of the appealing features of these logics [44,46,45] is the simplicity in the language, allowing to abstract the properties from the concrete strategies an agent may use. It has been shown that such logics enjoy interesting computational properties, e.g. the satisfiability problem for the logic from [44] is shown to be in $\mathsf{NP}^{\mathsf{NP}}$ [3], while its model-checking problem is in PSpace [18]. Interestingly, one of the logics considered therein is the extension of knowing how with numerical constraints, expressing not only that an agent is able to achieve a goal, but also that in doing so they are able to remain within a certain budget. For instance, one can express that a robot can complete an exploration mission without running out of fuel or in a certain amount of time. The need to express budget-like constraints about plans in ability-based logics is advocated in [33, Sec. 3.1] and [34]. Assuming that actions have costs, the execution of a sequence of actions requires that the agent stays always within the budget. Adding resource reasoning is a well-known paradigm in ATL-like logics [2,1,8], in energy games [6,11], and in multi-agent systems [9]. In [18], investigations can be found about the complexity of adding resource reasoning in the ability-based logic $\mathcal{L}_{kh}$ introduced in [44]; the new logic is called $\mathcal{L}_{kh}(\star)$. Though the decidability status of the model-checking problem for $\mathcal{L}_{kh}(\star)$ is therein left open, the subproblem in which the cost of actions is independent of the state, is shown to be ExpSpace-complete [18, Thm. 4].

Our contributions. In this work, we aim at determining whether model-checking for $\mathcal{L}_{kh}(\star)$ (Sec. 2.1) is decidable and, if so, to characterize its computational complexity. To do so, we introduce the universal control-state reachability problem for complete VASS with transitions labelled by letters from a finite alphabet. This is a variant of the (existential) control-state reachability problem asking for the existence of a finite sequence of letters such that all finite runs from an initial configuration following this sequence lead to a target control state (Sec. 2.2). Our first contribution is to show that the model-checking problem is reducible to the new problem for complete VASS (Sec. 3). Then, we show that model-checking for $\mathcal{L}_{kh}(\star)$ is Ack-complete (Thm. 1), in particular it requires non-primitive recursive time. For Ack-membership, we take advantage of the notion of belief functions [38], from the underlying well-structured transition system (WSTS) [21, Sec. 3] and the analysis about the length of bad sequences [38] and relying on [20] (Sec. 4). On the other hand, the hardness proof relies on the Ack-hardness of the control-state reachability problem for VASS with resets [43] (Sec. 5).

2 Preliminaries

In this section, we present the model-checking problem for an ability-based logic with budgets, which is the starting point of this work. Its second part is dedicated

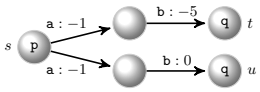


Fig. 1: A model for $\mathcal{L}_{kh}(\star)$ ($r = 1$).

to a new decision problem for complete VASS, called the *universal control-state reachability problem*, in which we ask if there exists a word σ such that all finite runs labelled by σ lead to target control states, while maintaining non-negative counters.

2.1 Knowing-how logics with budgets: the model-checking problem

We present below the knowing-how logic with budgets, first introduced in [18] and called herein $\mathcal{L}_{kh}(\star)$. We introduce standard notions like models, plans and strong executability (see e.g. [44,46,45]) and budget-related notions such as weight functions, computations and budget compatibility.

Linear plans. Let Act be a countable set of action symbols, and Act^* be the set of finite sequences over Act. Elements of Act^* are called *plans*, with ε being the *empty plan*. Given $\sigma \in \mathsf{Act}^*$, let $|\sigma|$ be the length of σ (with $|\varepsilon| \stackrel{\text{def}}{=} 0$). For $0 \leq k \leq |\sigma|$, the plan σ_k is σ's initial segment up to (and including) the kth position ($\sigma_0 \stackrel{\text{def}}{=} \varepsilon$). For $0 < k \leq |\sigma|$, the action $\sigma[k]$ is the one in σ's kth position.

Let $(\mathrm{R}_{\mathsf{a}})_{\mathsf{a}\in\mathsf{Act}}$ be a family of binary relations $\mathrm{R}_{\mathsf{a}} \subseteq \mathrm{S}\times\mathrm{S}$, for some set S. Define $\mathrm{R}_{\varepsilon} \stackrel{\text{def}}{=} \{(s,s) \mid s \in \mathrm{S}\}$ and, for all $\sigma \neq \varepsilon \in \mathsf{Act}^*$ and $\mathsf{a} \in \mathsf{Act}$, let $\mathrm{R}_{\sigma\mathsf{a}} \stackrel{\text{def}}{=} \{(s,t) \in \mathrm{S}\times\mathrm{S} \mid \exists t' \in \mathrm{S} \text{ s.t. } (s,t') \in \mathrm{R}_{\sigma} \text{ and } (t',t) \in \mathrm{R}_{\mathsf{a}}\}$. Consider a plan $\sigma \in \mathsf{Act}^*$: for all $s \in \mathrm{S}$, we define $\mathrm{R}_{\sigma}(s) \stackrel{\text{def}}{=} \{t \in \mathrm{S} \mid (s,t) \in \mathrm{R}_{\sigma}\}$. For all $X \subseteq \mathrm{S}$, $\mathrm{R}_{\sigma}(X) \stackrel{\text{def}}{=} \bigcup_{s\in X} \mathrm{R}_{\sigma}(s)$.

Models, weight functions and computations. Models of $\mathcal{L}_{kh}(\star)$ are of the form $\mathcal{S} = (\mathrm{S}, (\mathrm{R}_{\mathsf{a}})_{\mathsf{a}\in\mathsf{Act}}, wf, \mathrm{L})$ where: (1) S is a non-empty set of states; (2) $(\mathrm{R}_{\mathsf{a}})_{\mathsf{a}\in\mathsf{Act}}$ is a collection of binary relations on S ($s \xrightarrow{\mathsf{a}} t$ denotes that $(s,t) \in \mathrm{R}_{\mathsf{a}}$); (3) $\mathrm{L} : \mathrm{S} \to 2^{\mathsf{Prop}}$ is a labelling, for some set Prop of atomic propositions; and (4) $wf : \mathrm{S}\times\mathsf{Act}\times\mathrm{S} \to \mathbb{Z}^r$ is a so-called *weight function* of dimension r. The value $wf(s,\mathsf{a},t)$ is read as the cost of executing the action a from the state s to t (or equivalently to fire the transition $s \xrightarrow{\mathsf{a}} t$).

A *computation* is a sequence $\lambda = s_0 \xrightarrow{\mathsf{a}_1} s_1 \cdots \xrightarrow{\mathsf{a}_K} s_K$, and its (accumulated) *weight* is $wf(\lambda) \stackrel{\text{def}}{=} \sum_{k=1}^{K} wf(s_{k-1}, \mathsf{a}_k, s_k)$ (empty computations have weight $\mathbf{0}$).

Fig. 1 shows an example of a model with budgets: state labels (propositions p and q) are shown inside states, and some state names are shown outside states.

Constraints on plans. Intuitively, the works [44,46,45] establish that an agent knows how to achieve condition ψ given condition φ, when she has an appropriate plan that allows her to go from any state in which φ holds, only to states in which ψ holds. In order to characterise what 'appropriate' means, one can impose restrictions on what qualifies as an adequate plan.

Let $(\mathrm{R}_{\mathsf{a}})_{\mathsf{a}\in\mathsf{Act}}$ be a collection of binary relations. A plan $\sigma \in \mathsf{Act}^*$ is *strongly executable* (SE) at $s \in \mathrm{S}$ iff for all $k \in [0, |\sigma| - 1]$ and $t \in \mathrm{R}_{\sigma_k}(s)$, we have $\mathrm{R}_{\sigma[k+1]}(t) \neq \emptyset$, that is every computation on a prefix of σ can be prolonged to a computation of σ. We define the set $\mathrm{SE}(\sigma) \stackrel{\text{def}}{=} \{s \in \mathrm{S} \mid \sigma \text{ is SE at } s\}$. Strong executability is a notion inherited from conformant planning [12].

A plan $\sigma = \mathsf{a}_1 \cdots \mathsf{a}_K$ is $\boldsymbol{b}$-*compatible* at s ($\boldsymbol{b} \in \mathbb{N}^r$) iff for every computation $\lambda = s_0 \xrightarrow{\mathsf{a}_1} s_1 \cdots \xrightarrow{\mathsf{a}_K} s_K$ with $s_0 = s$, we have for all $L \in [1, K]$, $\boldsymbol{b} + \mathit{wf}(\lambda_{\leq L}) \geq \mathbf{0}$ (with $\lambda_{\leq L} \stackrel{\text{def}}{=} s_0 \xrightarrow{\mathsf{a}_1} s_1 \cdots \xrightarrow{\mathsf{a}_L} s_L$). The plan σ is $\boldsymbol{b}$-*compatible* at a set $X \subseteq \mathrm{S}$ iff it is $\boldsymbol{b}$-compatible at all $s \in X$. The tuple $\boldsymbol{b}$ is understood as the *initial budget*.

The logic $\mathcal{L}_{kh}(\star)$. Let us present below a natural way to extend the logic from [44]. We write $\mathcal{L}_{kh}(r)$ to indicate our ability-based logic with $r \geq 0$ resource types. If $r = 0$, then $\mathcal{L}_{kh}(r)$ corresponds to the logic from [44], written $\mathcal{L}_{kh}$ here. The logic $\mathcal{L}_{kh}(\star)$ denotes the version in which the number of resource types is arbitrary. For $r \geq 0$, the formulae of $\mathcal{L}_{kh}(r)$ have the following syntax:

$$\varphi ::= \mathsf{p} \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathsf{Kh}^{\boldsymbol{b}}(\varphi, \varphi),$$

where $\mathsf{p} \in \mathsf{Prop}$, and $\boldsymbol{b} \in \mathbb{N}^r$. Formulae of the form $\mathsf{Kh}^{\boldsymbol{b}}(\varphi, \psi)$ are read as *"when φ holds, the agent knows how to make ψ true with budget $\boldsymbol{b}$"*.

Semantics. The satisfaction relation $\Vdash$ is defined as follows:

$$\begin{array}{ll}
\mathcal{S}, s \Vdash \mathsf{p} & \stackrel{\text{def}}{\Leftrightarrow} \mathsf{p} \in \mathrm{L}(s), \\
\mathcal{S}, s \Vdash \neg\varphi & \stackrel{\text{def}}{\Leftrightarrow} \mathcal{S}, s \nVdash \varphi, \\
\mathcal{S}, s \Vdash \varphi \vee \psi & \stackrel{\text{def}}{\Leftrightarrow} \mathcal{S}, s \Vdash \varphi \text{ or } \mathcal{S}, s \Vdash \psi, \\
\mathcal{S}, s \Vdash \mathsf{Kh}^{\boldsymbol{b}}(\varphi, \psi) & \stackrel{\text{def}}{\Leftrightarrow} \text{there is a plan } \sigma \in \mathsf{Act}^* \text{ such that} \\
& \quad (1)\ [\![\varphi]\!]^{\mathcal{S}} \subseteq \mathrm{SE}(\sigma), \quad (2)\ \mathrm{R}_\sigma([\![\varphi]\!]^{\mathcal{S}}) \subseteq [\![\psi]\!]^{\mathcal{S}}, \text{ and} \\
& \quad (3)\ \sigma \text{ is } \boldsymbol{b}\text{-compatible at } [\![\varphi]\!]^{\mathcal{S}},
\end{array}$$

where $[\![\varphi]\!]^{\mathcal{S}} \stackrel{\text{def}}{=} \{s \mid \mathcal{S}, s \Vdash \varphi\}$. Let $\mathcal{S}$ be the model of Fig. 1, it holds that $\mathcal{S}, s \Vdash \mathsf{Kh}^{6}(\mathsf{p}, \mathsf{q})$, since the plan ab is 6-compatible. Also, we have that $\mathcal{S}, s \nVdash \mathsf{Kh}^{5}(\mathsf{p}, \mathsf{q})$, as ab is not 5-compatible: while the computation leading to u has weight -1, the one leading to t has weight -6, exhausting the available budget of 5.

Let us define the *model-checking problem for* $\mathcal{L}_{kh}(\star)$, written $\mathrm{MC}(\mathcal{L}_{kh}(\star))$ and introduced in [18] (finite models are such that S, Act and Prop are finite). Its budget-free instance (i.e., no need of checking condition (3)) is called $\mathrm{MC}(\mathcal{L}_{kh})$.

Input: a finite model $\mathcal{S}$, a state s and a formula φ.
Question: $\mathcal{S}, s \Vdash \varphi$?

In [18], it is shown that the conditions (1)–(2) can be encoded by a finite-state automaton of exponential size, leading to the PSpace-membership of $\mathrm{MC}(\mathcal{L}_{kh})$.

Proposition 1 ([18]). *Let $\mathcal{S} = (\mathrm{S}, (\mathrm{R}_{\mathsf{a}})_{\mathsf{a}\in\mathsf{Act}}, \mathit{wf}, \mathrm{L})$ be a finite model and $X_1, X_2 \subseteq \mathrm{S}$. The set of plans σ such that $X_1 \subseteq \mathrm{SE}(\sigma)$ and $\mathrm{R}_\sigma(X_1) \subseteq X_2$ can be accepted by a deterministic finite-state automaton $\mathcal{A} = (Q, \mathsf{Act}, \delta, I, F)$ with $|Q| \leq 2^{|\mathrm{S}|(|\mathrm{S}|+2)}$. Moveover, each location in Q can be encoded in polynomial space, and deciding whether $q \xrightarrow{\mathsf{a}} q' \in \delta$ (resp. $q \in I$, $q \in F$) can be done in polynomial space.*

The decision problem $\mathrm{MC}(\mathcal{L}_{kh}(\star))$ witnesses a genuine complexity blow-up, partly due to condition (3) combined with (2). Indeed, adding condition (3) amounts to restrict further the class of plans to be considered to satisfy a knowing-how formula. Recall that the decidability status of $\mathrm{MC}(\mathcal{L}_{kh}(\star))$ is left open in [18] though the subproblem in which the cost of an action is independent of the states, $wf(s_1, a, s_1') = wf(s_2, a, s_2')$ for all $s_1, s_1', s_2, s_2' \in \mathrm{S}$ and all $\mathsf{a} \in \mathsf{Act}$, is shown to be ExpSpace-complete [18, Thm. 4].

The rest of the paper is devoted to filling this gap and showing that the problem $\mathrm{MC}(\mathcal{L}_{kh}(\star))$ is decidable and Ack-complete. As a consequence of our developments, solving $\mathrm{MC}(\mathcal{L}_{kh}(\star))$ requires non-primitive-recursive time, even if restricted to a single resource type. This result is formally stated below.

Theorem 1. $\mathrm{MC}(\mathcal{L}_{kh}(\star))$ *is* Ack*-complete, even in dimension* $r = 1$.

2.2 A new reachability problem on VASS

To characterise the complexity of $\mathrm{MC}(\mathcal{L}_{kh}(\star))$, we introduce a new decision problem `UCReach` on complete VASS, which is structurally less rich than $\mathrm{MC}(\mathcal{L}_{kh}(\star))$ but for which it is quite handy to get complexity results. We show that `UCReach` is Ack-complete (Secs. 4 and 5) and equivalent to $\mathrm{MC}(\mathcal{L}_{kh}(\star))$ modulo exponential-time reductions (the exponential blow up occurs only in one direction), see Sec. 3.

We briefly recall that a *vector addition system with states (VASS)* [23] is a tuple $\mathcal{V} = (Q, \Sigma, r, R)$, where Q is a finite set of *locations*, $r \in \mathbb{N}$ is its *dimension*, and R is a finite set of *transitions* in $Q \times \Sigma \times \mathbb{Z}^r \times Q$ for some finite non-empty alphabet Σ. Herein, the transitions are labelled by letters from the alphabet Σ (also sometimes written Act). A VASS $\mathcal{V}$ is *complete* if for all $q \in Q$ and $\mathsf{a} \in \Sigma$, there is some $(\boldsymbol{u}, q')$ such that $(q, \mathsf{a}, \boldsymbol{u}, q') \in R$. A *configuration* (resp. *pseudo-configuration*) in a VASS $\mathcal{V}$ is a pair $(q, \boldsymbol{x}) \in Q \times \mathbb{N}^r$ (resp. in $Q \times \mathbb{Z}^r$). Given pseudo-configurations $(q, \boldsymbol{x})$, $(q', \boldsymbol{x}')$ and a transition $T = q \xrightarrow{\mathsf{a}, \boldsymbol{u}} q'$, we write $(q, \boldsymbol{x}) \xrightarrow{T} (q', \boldsymbol{x}')$ whenever $\boldsymbol{x}' = \boldsymbol{u} + \boldsymbol{x}$. A *pseudo-run* is a finite sequence $\rho = (q_0, \boldsymbol{x}_0) \xrightarrow{T_1} (q_1, \boldsymbol{x}_1) \xrightarrow{T_2} (q_2, \boldsymbol{x}_2) \cdots \xrightarrow{T_L} (q_L, \boldsymbol{x}_L)$ of pseudo-configurations, where $(q_0, \boldsymbol{x}_0)$ is the *initial* pseudo-configuration. We say that the pseudo-run ρ is labeled by $\mathsf{a}_1 \cdots \mathsf{a}_L$ if T_i is of the form $(\cdot, \mathsf{a}_i, \cdot, \cdot)$ for all $1 \leq i \leq L$. A *run* is a pseudo-run in which only configurations in $Q \times \mathbb{N}^r$ occur (negative values disallowed). We present the *universal control-state reachability problem* `UCReach`:

Input: A complete VASS $\mathcal{V} = (Q, \Sigma, r, R)$, $p_0 \in Q$, $\boldsymbol{y_0} \in \mathbb{N}^r$, $Q_F \subseteq Q$.
Question: Does there exist a word $\sigma \in \Sigma^*$ such that for all finite pseudo-runs $\rho = (q_0, \boldsymbol{x}_0) \xrightarrow{T_1} (q_1, \boldsymbol{x}_1) \xrightarrow{T_2} (q_2, \boldsymbol{x}_2) \cdots \xrightarrow{T_L} (q_L, \boldsymbol{x}_L)$ from $(p_0, \boldsymbol{y_0})$ with label σ, we have $q_L \in Q_F$ and $\{\boldsymbol{x}_0, \ldots, \boldsymbol{x}_L\} \subseteq \mathbb{N}^r$ (hence ρ is a run)?

Observe that the input VASS in the definition of `UCReach` is complete because we can restrict ourselves to such VASS to reduce the model-checking problem for the logic $\mathcal{L}_{kh}(\star)$. A positive instance of `UCReach` can be formulated by an existential quantification over plans, followed by a universal quantification over the class of pseudo-runs whose label is precisely the witness plan. Compared to

the standard control-state reachability problem for VASS, there is a universal quantification over all the pseudo-runs labelled by σ (instead of an existential one). Note also that we could equally assume that $|Q_F| = 1$ as $\mathtt{UCReach}$ can be easily reduced to this subproblem but the current formulation is more convenient to us. Though $\mathtt{UCReach}$ is quite natural in view of its standard existential variant, as far as we know, the problem $\mathtt{UCReach}$ was never defined in the literature. Most probably, so far, there was no need for it.

Furthermore, note that many decision problems on VASS for which we do not need to reach a precise configuration are known to be ExpSpace-complete, see e.g. the covering and boundedness problems [35,10,39,19]. The problem $\mathtt{UCReach}$ *seems to be* of the same flavour, as we need to reach the specified control states but the final counter values are not important. By contrast, the reachability problem (between configurations) is Ack-complete [32,13,31]. However, the problem $\mathtt{UCReach}$ has similarities with the fixed initial credit problem for specific blind games [38] (see also [15]). The crucial difference rests on the fact that $\mathtt{UCReach}$ requires a synchronised reachability condition whereas the problem in [38] is related to a safety property (non-termination). Moreover, the fixed initial credit problem in [38] handles a single counter. Below, we show that there is a logspace reduction from $\mathtt{UCReach}$ to its restriction with a single counter (Lemma 2).

3 Relationships between MC($\mathcal{L}_{kh}(\star)$) and UCReach

First, we show that $\mathtt{UCReach}$ is closely related to MC($\mathcal{L}_{kh}(\star)$).

Lemma 1. *There is an exponential-time Turing reduction from* MC($\mathcal{L}_{kh}(\star)$) *to* $\mathtt{UCReach}$.

Proof (Sketch). By using a standard labelling algorithm, the existence of a Turing reduction only requires to show that for all $\mathcal{S}=(\mathrm{S}, (\mathrm{R}_{\mathtt{a}})_{\mathtt{a}\in\mathsf{Act}}, \mathit{wf}, \mathrm{L})$ and $s \in \mathrm{S}$, $\mathcal{S}, s \Vdash \mathsf{Kh}^{\boldsymbol{b}}(\mathtt{p}, \mathtt{q})$ can be reduced to an instance of $\mathtt{UCReach}$. Recall that $\mathcal{S}, s \Vdash \mathsf{Kh}^{\boldsymbol{b}}(\mathtt{p}, \mathtt{q})$ iff there is $\sigma \in \mathsf{Act}^*$ such that (1) $[\![\mathtt{p}]\!]^{\mathcal{S}} \subseteq \mathrm{SE}(\sigma)$, (2) $\mathrm{R}_\sigma([\![\mathtt{p}]\!]^{\mathcal{S}}) \subseteq [\![\mathtt{q}]\!]^{\mathcal{S}}$ and (3) for every $t \in [\![\mathtt{p}]\!]^{\mathcal{S}}$, σ is $\boldsymbol{b}$-compatible at t. Below, w.l.o.g. we can assume that $[\![\mathtt{p}]\!]^{\mathcal{S}} \neq \emptyset$ because this case is obvious to handle. By Prop. 1, we can define a deterministic finite-state automaton $\mathcal{A} = (Q, \mathsf{Act}, \delta, I, F)$ that accepts exactly the plans satisfying (1)–(2) and $\mathcal{A}$ has the following properties:

- $|Q| \leq 2^{|\mathrm{S}|(|\mathrm{S}|+2)}$,
- each location in Q can be encoded in polynomial space and deciding whether $q \xrightarrow{\mathtt{a}} q' \in \delta$ (respectively $q \in I$, $q \in F$) can be done in polynomial space.

Given $X \subseteq \mathrm{S}$, let us define the VASS $\mathcal{V}^\star[X] = (Q^\star, \mathsf{Act}, r, R^\star)$ as follows.

- $Q^\star \stackrel{\text{def}}{=} \mathrm{S} \times Q \uplus \{\mathtt{init}, \bot\}$,
- $(s, q) \xrightarrow{\mathtt{a}, \boldsymbol{u}} (s', q') \in R^\star$ iff $s \xrightarrow{\mathtt{a}} s'$ in $\mathcal{S}$ and $q \xrightarrow{\mathtt{a}} q' \in \delta$ (synchronisation on the action $\mathtt{a}$), and $\boldsymbol{u} = \mathit{wf}(s, \mathtt{a}, s')$. Obviously, the VASS $\mathcal{V}^\star[X]$ can be viewed as a synchronised product between $\mathcal{S}$ and $\mathcal{A}$.

- $(s,q) \xrightarrow{\mathtt{a},\mathbf{0}} \bot \in R^\star$ if there is no s' such that $s \xrightarrow{\mathtt{a}} s'$ in $\mathcal{S}$.
- $\bot \xrightarrow{\mathtt{a},\mathbf{0}} \bot \in R^\star$ for all $\mathtt{a} \in \mathsf{Act}$ and $\mathtt{init} \xrightarrow{\mathtt{a},\mathbf{0}} \bot \in R^\star$ for all $\mathtt{a} \in \mathsf{Act} \setminus \{\mathtt{a}^\dagger\}$ where $\mathtt{a}^\dagger$ is some distinguished action.
- $\mathtt{init} \xrightarrow{\mathtt{a}^\dagger,\boldsymbol{b}} (t,q_0) \in R^\star$ for all $t \in X$ and $q_0 \in I$.

The edges in $\mathcal{V}^\star[X]$ involving the sink location $\bot$ are designed so that $\mathcal{V}^\star[X]$ is complete but without affecting the correctness of the reduction. Indeed, one can show that there is a plan σ satisfying (1)–(3) iff $\mathcal{V}^\star[[\![\mathtt{p}]\!]^{\mathcal{S}}]$, $\mathtt{init}$, $\mathbf{0}$, $\mathrm{S} \times F$ is a positive instance of the problem $\mathtt{UCReach}$.

The size of $\mathcal{V}^\star[[\![\mathtt{p}]\!]^{\mathcal{S}}]$ is at most exponential in the size of $\mathcal{S}$. So, the above equivalence provides a many-one reduction for checking $\mathcal{S}, s \Vdash \mathsf{Kh}^{\boldsymbol{b}}(\mathtt{p},\mathtt{q})$, whence the Turing reduction for the full model-checking problem. □

Note that the reduction in the proof of Lemma 1 preserves the number r of resource types. Below, we show that $\mathtt{UCReach}$ restricted to a single counter (denoted by $\mathtt{UCReach}(1)$) is actually as hard as $\mathtt{UCReach}$. Indeed, due to the synchronisation on pseudo-runs with the same plan σ, each dimension can be handled separately thanks to the universal quantification.

Lemma 2. *There is a logarithmic-space reduction from* $\mathtt{UCReach}$ *to* $\mathtt{UCReach}(1)$.

Thanks to Lemmas 1 and 2, it is enough to show that $\mathtt{UCReach}(1)$ is in Ack to prove that $\mathrm{MC}(\mathcal{L}_{kh}(\star))$ is in Ack. Eventually, we get the Ack-completeness of $\mathtt{UCReach}$, $\mathtt{UCReach}(1)$, $\mathrm{MC}(\mathcal{L}_{kh}(1))$ and $\mathrm{MC}(\mathcal{L}_{kh}(\star))$.

4 UCReach is in Ackermannian time

This section is devoted to showing that $\mathtt{UCReach}(1)$, the universal control-state reachability problem restricted to one counter, can be solved in Ackermannian time, following the definition of the complexity class Ack from [42]. To do so, we take advantage of the notion of belief functions from [38] as well as of the well-structured properties of the underlying transition system (WSTS) and the analysis about the length of bad sequences as done in [38, pp. 94-95] (essentially relying on the results from [20], see also [41]). However, unlike what is done in [38] to characterise the complexity of a non-termination problem, we use a backward algorithm along the lines of [21,7,28], since $\mathtt{UCReach}$ is essentially a covering problem, as shown below.

Let $\mathcal{V} = (Q, \mathsf{Act}, 1, R)$, $p_0 \in Q$, $y_0 \in \mathbb{N}$, $Q_F \subseteq Q$ be an instance of $\mathtt{UCReach}(1)$, that is fixed for the rest of this section. A *belief function* is a map $f : Q \to \mathbb{N} \cup \{\infty\}$ [38]. Given $X \subseteq Q \times \mathbb{N}$, we define its belief function f_X as: for all $q \in Q$,

$$f_X(q) \stackrel{\text{def}}{=} \min(\{\infty\} \cup \{v \mid (q,v) \in X\}).$$

Observe that q does not occur in X iff $f_X(q) = \infty$. Moreover, f_X can be understood as a finite abstraction of X. What matters in X are the configurations (q,v) with minimal value v. We write $\mathit{supp}(f) \stackrel{\text{def}}{=} \{q \mid f(q) \in \mathbb{N}\}$ to denote the

support set of f. Belief functions are introduced in [15, Sec. 3], [38, Sec. 3] and [22, Sec. 5] and though we use such belief functions with a backward algorithm that differs from the forward algorithm in [38, Sec. 3.1], we can still take advantage of results introduced in [38].

Given two belief functions f and g, we define the binary relation $\preceq$ such that $f \preceq g$ iff $supp(f) = supp(g)$ and for all $q \in supp(f)$, we have $f(q) \leq g(q)$ [38, Sec. 3.1]. The relation $\preceq$ on belief functions behaves like the component-wise less than relation on $\mathbb{N}^{|Q|}$. In particular, as a consequence of Dickson's Lemma, the relation $\preceq$ on belief functions can be shown to be a well-quasi-ordering (wqo) too (basic definitions about well-quasi-orderings can be found in [21, Sec. 2.1]). Given a belief function f and a set $\mathcal{X}$ of belief functions, we write $\uparrow f$ to denote the set $\{g \mid f \preceq g\}$ and $\uparrow\mathcal{X}$ to denote the set $\{g \mid f \preceq g,\ f \in \mathcal{X}\}$. The set $\uparrow f$ (resp. $\uparrow\mathcal{X}$) is known as the *upward closure* of f (resp. $\mathcal{X}$). *Upward closed sets* $\mathcal{X}$ of belief functions are sets of belief functions satisfying $\mathcal{X} = \uparrow\mathcal{X}$.

We define the transition system $\mathcal{T}_{\mathcal{V}} \stackrel{\text{def}}{=} (\{f \mid f : Q \to \mathbb{N} \cup \{\infty\}\}, (\stackrel{\mathsf{a}}{\to})_{\mathsf{a}\in\mathsf{Act}})$ built over the set of belief functions such that $f \stackrel{\mathsf{a}}{\to} g$ iff the conditions (1)-(2) hold:

(1) for all $q \in supp(f)$ and all $q \stackrel{\mathsf{a},u}{\to} q' \in R$, we have $f(q) + u \geq 0$ ("all the a-transitions are fireable", i.e. no negative budget value is reached),
(2) for all $q' \in Q$,

$$g(q') \stackrel{\text{def}}{=} \min(\{v' \mid \exists\ q \in supp(f) \text{ and } q \stackrel{\mathsf{a},u}{\to} q' \in R \text{ s.t. } v' = f(q) + u\} \cup \{\infty\}).$$

Observe that each relation $\stackrel{\mathsf{a}}{\to}$ is deterministic but not necessarily total. Moreover, for every action $\mathsf{a} \in \mathsf{Act}$, there is a self-loop on the unique belief function with empty support set. We write $\stackrel{*}{\to}$ to denote the reflexive and transitive closure of $(\bigcup_{\mathsf{a}\in\mathsf{Act}} \stackrel{\mathsf{a}}{\to})$. Given a set $\mathcal{X}$ of belief functions, we write $pre_{\mathsf{a}}(\mathcal{X})$ to denote the set $\{f' \mid \exists\ f \in \mathcal{X} \text{ s.t. } f' \stackrel{\mathsf{a}}{\to} f\}$ and $pre(\mathcal{X})$ to denote $\bigcup_{\mathsf{a}\in\mathsf{Act}} pre_{\mathsf{a}}(\mathcal{X})$ (sets of *predecessors*). The set $pre^\star(\mathcal{X})$ of belief functions is defined similarly. Formally, $f \in pre^\star(\mathcal{X})$ iff there are $f_0, f_1, \ldots, f_n$ and $\mathsf{a}_1, \ldots, \mathsf{a}_n$ $(n \geq 0)$ such that $f_n = f$, $f_0 \in \mathcal{X}$ and $f_i \in pre_{\mathsf{a}_i}(\{f_{i-1}\})$ for all $i \in [1, n]$. This means that we have

$$f = f_n \stackrel{\mathsf{a}_n}{\to} f_{n-1} \cdots f_1 \stackrel{\mathsf{a}_1}{\to} f_0 \in \mathcal{X}.$$

In order to establish that $\mathcal{T}_{\mathcal{V}}$ is a well-structured transition system (WSTS) in the sense of [21, Def. 2.5], we need to establish the monotony property below.

Lemma 3 (Monotony). *If $f \stackrel{\mathsf{a}}{\to} g$ and $f \preceq f'$, then there is $g \preceq g'$ s.t. $f' \stackrel{\mathsf{a}}{\to} g'$.*

Lemma 3 entails that $\mathcal{T}_{\mathcal{V}}$ is a WSTS, see [21, Def. 2.5] (the main properties being monotony and $\preceq$ is a well-quasi-ordering). As a consequence, we also get (see [21, Prop. 3.1]):

Lemma 4. *For all upward closed sets $\mathcal{X}$ of belief functions, the set $pre^\star(\mathcal{X})$ is upward closed, i.e., $pre^\star(\mathcal{X}) = \uparrow pre^\star(\mathcal{X})$.*

Furthermore, Lemma 5 below states that solving $\texttt{UCReach}(1)$ amounts to solving a covering problem on $\mathcal{T}_{\mathcal{V}}$, which can be decided using [21, Thm. 3.6]. It states that $\texttt{UCReach}(1)$ and the covering problem for the WSTS $\mathcal{T}_{\mathcal{V}}$ can be used to solved each other. After proving it, it remains to show that the effectivity hypotheses in [21, Thm. 3.6] hold, which requires further developments.

Lemma 5. *The elements $\mathcal{V} = (Q, \mathsf{Act}, 1, R)$, $p_0 \in Q$, $y_0 \in \mathbb{N}$, $Q_F \subseteq Q$ form a positive instance of $\texttt{UCReach}(1)$ iff $f_0 \xrightarrow{*} f$ for some $f_0, f \in \mathcal{T}_{\mathcal{V}}$ such that $supp(f_0) = \{p_0\}$, $f_0(p_0) = y_0$, and $\emptyset \neq supp(f) \subseteq Q_F$.*

By Lemma 5, in order to solve $\texttt{UCReach}(1)$, we need to design a decision procedure to check whether $f_0 \in pre^\star(\{f \mid \emptyset \neq supp(f) \subseteq Q_F\})$, where $\{f \mid \emptyset \neq supp(f) \subseteq Q_F\}$ is an upward closed set whose basis is $\{f \mid \emptyset \neq supp(f) \subseteq Q_F$ and $\forall q \in supp(f), f(q) = 0\}$, i.e.,

$$\{f \mid \emptyset \neq supp(f) \subseteq Q_F\} =$$

$$\uparrow\{f \mid \emptyset \neq supp(f) \subseteq Q_F \text{ and } \forall q \in supp(f), f(q) = 0\}.$$

Given $f : Q \to \mathbb{N} \cup \{\infty\}$ and $\mathtt{a} \in \mathsf{Act}$, we define a finite set $pb(\mathtt{a}, f)$ of belief functions obtained from the belief function $root(\mathtt{a}, f) : Q \to \mathbb{N} \cup \{\infty\}$ defined as follows. Note that 'pb' stands for "pred-basis" by reference to the notation in [21, Sec. 3]. Given $q \in Q$, the value $root(\mathtt{a}, f)(q)$ is ∞ if there is $q \xrightarrow{\mathtt{a},u} q' \in R$ such that $f(q') = \infty$ or if there is no transition $q \xrightarrow{\mathtt{a},u} q'$ in R, for some u, q'; otherwise,

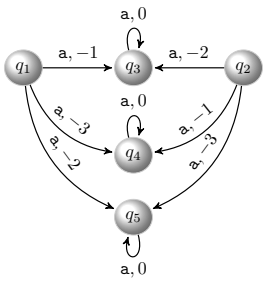
$$root(\mathtt{a}, f)(q) \stackrel{\text{def}}{=} \max(\{f(q') - u \mid q \xrightarrow{\mathtt{a},u} q' \in R\} \cup \{0\}).$$

Let $pb(\mathtt{a}, f)$ be the set of belief functions g obtained from $root(\mathtt{a}, f)$ by satisfaction of the three conditions below.

(PB1) $supp(g) \subseteq supp(root(\mathtt{a}, f))$.
(PB2) For all $q \in supp(g)$, we have $g(q) = root(\mathtt{a}, f)(q)$.
(PB3) There is g' such that $g \xrightarrow{\mathtt{a}} g'$ and $f \preceq g'$.

Viewing $root(\mathtt{a}, f)$ as the *seed* from which the elements g in $pb(\mathtt{a}, f)$ are defined, the conditions (PB1) and (PB2) simply state that g must be a restriction of $root(\mathtt{a}, f)$. However, (PB3) enforces that the restriction must not be too limited so that a belief function g' greater than f can be reached from g with $\xrightarrow{\mathtt{a}}$. It is also easy to prove that $pb(\mathtt{a}, f)$ is finite and actually $|pb(\mathtt{a}, f)| \leq 2^{|Q|}$.

In order to illustrate the construction of $pb(\mathtt{a}, f)$, consider the complete VASS with one counter below and the belief function f such that $supp(f) = \{q_3, q_4, q_5\}$ and $f(q_3) = f(q_4) = f(q_5) = 0$.



The belief function $root(\mathsf{a}, f)$ is such that $supp(root(\mathsf{a}, f)) = \{q_1, \dots q_5\}$, $root(\mathsf{a}, f)(q_3) = root(\mathsf{a}, f)(q_4) = root(\mathsf{a}, f)(q_5) = 0$ and both $root(\mathsf{a}, f)(q_1)$ and $root(\mathsf{a}, f)(q_2)$ are equal to 3. We have $\{root(\mathsf{a}, f), f_1, f_2\} \subseteq pb(\mathsf{a}, f)$ with $supp(f_1) = \{q_1\}$, $supp(f_2) = \{q_2\}$ and $f_1(q_1) = f_2(q_2) = 3$. Observe that $root(\mathsf{a}, f)$ and f_i $(i = 1, 2)$ are incomparable because the two belief functions have distinct support sets.

We present the main technical lemma to get decidability (and it is also useful to get the Ack upper bound, as explained further below).

Lemma 6. *$\uparrow pb(\mathsf{a}, f) = \uparrow pre_{\mathsf{a}}(\uparrow f)$ for all actions $\mathsf{a} \in \mathsf{Act}$ and belief functions f.*

As a consequence of Lemma 6, for all finite sets $\mathcal{Z}$ of belief functions, we have

$$\uparrow \bigcup \{pb(\mathsf{a}, f) \mid f \in \mathcal{Z}\} = \uparrow pre_{\mathsf{a}}(\uparrow \mathcal{Z}),$$

and therefore $\uparrow \bigcup \{pb(\mathsf{a}, f) \mid f \in \mathcal{Z}, \mathsf{a} \in \mathsf{Act}\} = pre(\uparrow \mathcal{Z})$. To solve the covering problem from Lemma 5, we proceed as follows using the proof method from [21]. We define a family $(\mathcal{Z}_i)_{i \in \mathbb{N}}$ made of finite sets of belief functions such that

- $\mathcal{Z}_0 \stackrel{\text{def}}{=} \{f \mid \emptyset \neq supp(f) \subseteq Q_F \text{ and for all } q \in supp(f),\ f(q) = 0\}$,
- $\mathcal{Z}_{i+1} \stackrel{\text{def}}{=} \mathcal{Z}_i \cup \bigcup \{pb(\mathsf{a}, f) \mid \mathsf{a} \in \mathsf{Act},\ f \in \mathcal{Z}_i\}$.

By [21, Lemma 2.4], there is m such that $\uparrow \mathcal{Z}_m = \uparrow \mathcal{Z}_{m+1}$ ($\mathcal{T}_{\mathcal{V}}$ is a WSTS) and therefore the family $(\uparrow \mathcal{Z}_i)_{i \in \mathbb{N}}$ stabilises at some point. One can show that a positive instance of the covering problem holds iff $f_0 \in \uparrow \mathcal{Z}_m$ (since $\uparrow \mathcal{Z}_m = pre^\star(\uparrow \mathcal{Z}_0)$). Since computing $pb(\mathsf{a}, f)$ is effective and checking whether $\uparrow \mathcal{Z} \subseteq \uparrow \mathcal{Z}'$ can be done effectively too, this provides a decision procedure. For instance, (A) $\uparrow \mathcal{Z} \subseteq \uparrow \mathcal{Z}'$ iff (B) for all $f \in \mathcal{Z}$, there is $f' \in \mathcal{Z}'$ such that $f' \preceq f$. First suppose that (A) holds. Let $f \in \mathcal{Z}$ and therefore $f \in \uparrow \mathcal{Z}'$ by (A). There is $f' \in \mathcal{Z}'$ such that $f' \preceq f$ and therefore (B) holds. Conversely, suppose that (B) holds and let $f \in \uparrow \mathcal{Z}$. There is $f' \in \mathcal{Z}$ such that $f' \preceq f$ and by (B), there is $f'' \in \mathcal{Z}'$ such that $f'' \preceq f'$. Consequently, $f'' \preceq f$ by transitivity of $\preceq$ and therefore $f \in \uparrow \mathcal{Z}'$.

Since we can effectively build the sequence $\mathcal{Z}_0, \dots, \mathcal{Z}_{m+1}$ and each $\mathcal{Z}_{i+1}$ can be built in exponential-time in the respective sizes of the instance of UCReach(1) and of $\mathcal{Z}_i$, UCReach(1) can be solved in Ack as concluded below. Indeed, this holds as soon as we can reasonably bound the length m depending on the size N of the instance. Since $\uparrow \mathcal{Z}_0 \subset \uparrow \mathcal{Z}_1 \subset \cdots \subset \uparrow \mathcal{Z}_m = \uparrow \mathcal{Z}_{m+1}$, for all $i \in [1, m]$, there is a belief function f_i such that $f_i \in \mathcal{Z}_i$ and $f_i \notin \uparrow \mathcal{Z}_{i-1}$. The sequence $f_1, \dots, f_m$ is a bad sequence in the sense that for all $i < i'$, we have $f_i \not\preceq f_{i'}$. As we use the notion of belief functions from [38], we can take advantage of the analysis about bad sequences therein, which is itself essentially based on [20]. Hence, we have $m \leq h(N)$ for some function h in $\mathcal{F}_{|Q|+3}$, where $\mathcal{F}_i$ is the ith level of Grzegorczyk hierarchy, see e.g. [42, Sec. 2]. The level $|Q| + 3$ in the class $\mathcal{F}_{|Q|+3}$ is shown in [38, page 94]. Thus, $\mathcal{Z}_{m+1}$ can be computed in time $F_\omega(N)$ where F_ω is the fast-growing function at the level ω, see [42, Sec. 2.2.1].

Theorem 2. UCReach(1) *in* Ack.

By putting together Lemmas 1 and 2 and Thm. 2, we get:

Corollary 1. UCReach *and* MC($\mathcal{L}_{kh}(\star)$) *are in* Ack.

5 Ack-hardness of the problem MC($\mathcal{L}_{kh}(\star)$)

Below, we show that MC($\mathcal{L}_{kh}(1)$) is Ack-hard by reducing the control-state reachability problem for VASS with resets, known to be Ack-hard [43, Sec. 6]. By Lemma 1, we get that UCReach(1) is also Ack-hard. Our proof strategy differs from the one for Ack-hardness of the fixed initial credit problem for blind games in [38, Theo. 2] that reduces some halting problem for Minsky machines.

A *VASS with resets* is an extension of the model of VASS by allowing resets on counters. The notions of pseudo-runs, runs, etc. are defined similarly to VASS once the one-step relation is defined (see Sec. 2.2). More precisely, a VASS with resets (RVASS for short) is a structure $\mathcal{V} = (Q, r, R)$, where Q is a finite set of *locations*, $r \in \mathbb{N}$ is its *dimension*, and R is a finite set of *transitions* in $Q \times OP[r] \times Q$ where $OP[r] \stackrel{\text{def}}{=} \mathbb{Z}^r \cup \{\mathsf{reset}(i) \mid i \in [1, r]\}$. Note that there is no finite alphabet Σ in $\mathcal{V}$ as it is of no use in this part. Given a transition $T = q \xrightarrow{\boldsymbol{u}} q'$ with $\boldsymbol{u} \in \mathbb{Z}^r$, we write $(q, \boldsymbol{x}) \xrightarrow{T} (q', \boldsymbol{x}')$ whenever $\boldsymbol{x}' = \boldsymbol{u} + \boldsymbol{x}$ (as for VASS). By contrast, given a transition $T = q \xrightarrow{\mathsf{reset}(i)} q'$, we write $(q, \boldsymbol{x}) \xrightarrow{T} (q', \boldsymbol{x}')$ whenever for all $j \neq i$, we have $\boldsymbol{x}'[j] = \boldsymbol{x}[j]$ and $\boldsymbol{x}'[i] = 0$. Since the operations in $OP[r]$ are deterministic, any pseudo-run $(q_0, \boldsymbol{x}_0) \xrightarrow{T_1} (q_1, \boldsymbol{x}_1) \cdots \xrightarrow{T_N} (q_N, \boldsymbol{x}_N)$ can be represented by the initial configuration $(q_0, \boldsymbol{x}_0)$ and the sequence of transitions $T_1 \cdots T_N$. The *control-state reachability problem for RVASS* (CReach(RVASS)) defined below is Ack-hard [43, Sec. 6]:

Input: an RVASS $\mathcal{V}$ and two locations q_0, q_f.
Question: does there exist a finite run $(q_0, \mathbf{0}) \xrightarrow{*} (q_f, \boldsymbol{x})$ for some $\boldsymbol{x} \in \mathbb{N}^r$?

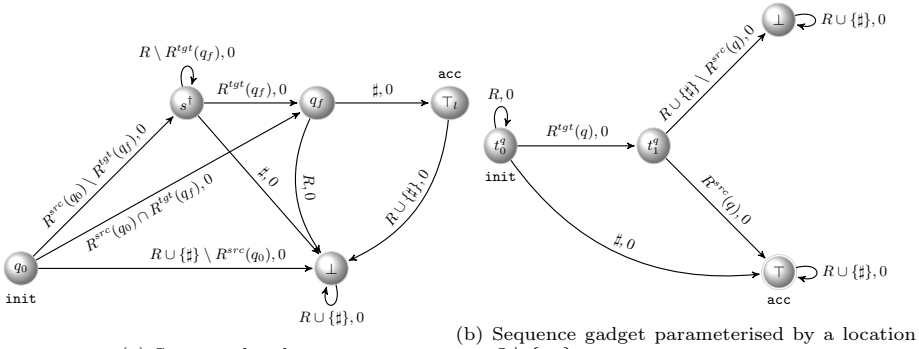
In order to prepare the proof of Thm. 3 below, we introduce several useful notations. Let $\mathcal{V} = (Q, r, R)$ be an RVASS.

- For each $q \in Q$, we write $R^{src}(q)$ to denote the set of transitions $T \in R$ such that q is the source location (i.e. T is of the form $(q, \cdot, \cdot)$). We also write $R^{tgt}(q)$ to denote the set of transitions $T \in R$ such that q is the target location (i.e. T is of the form $(\cdot, \cdot, q)$).
- For each counter $i \in [1, r]$, we write $R(\mathsf{reset}(i))$ to denote the set of transitions T in R of the form $(\cdot, \mathsf{reset}(i), \cdot)$.

The run witnessing $(q_0, \mathbf{0}) \xrightarrow{*} (q_f, \boldsymbol{x})$ can be represented by the sequence of transitions $T_1 \cdots T_N$ since the initial configuration $(q_0, \mathbf{0})$ is fixed. Observe that $T_1 \in R^{src}(q_0)$, $T_N \in R^{tgt}(q_f)$ and for all $i \in [1, N-1]$ and $q \in Q$, $T_i \in R^{tgt}(q)$ implies $T_{i+1} \in R^{src}(q)$.

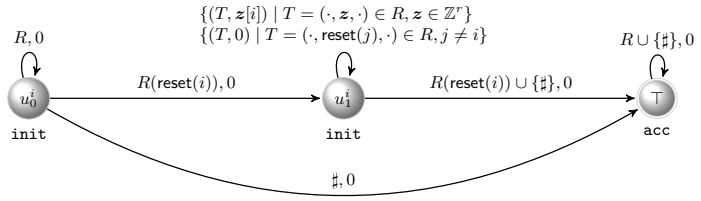
Theorem 3. MC($\mathcal{L}_{kh}(1)$) *is* Ack*-hard.*

Proof (Sketch). We present a reduction from CReach(RVASS) to MC($\mathcal{L}_{kh}(1)$). Given an RVASS $\mathcal{V} = (Q, r, R)$ and $q_0, q_f \in Q$, we construct a model $\mathcal{S}$ and a formula $\varphi = \mathsf{Kh}^0(\mathtt{init}, \mathtt{acc})$ where init, acc are atomic, and we show that for all states s, we have $\mathcal{S}, s \Vdash \varphi$ iff there is a run of $\mathcal{V}$ from $(q_0, \mathbf{0})$ to some configuration $(q_f, \boldsymbol{x})$ with $\boldsymbol{x} \in \mathbb{N}^r$. W.l.o.g., we can assume that $q_0 \neq q_f$ (otherwise the



(a) Structural gadget.

(b) Sequence gadget parameterised by a location $q \in Q \setminus \{q_f\}$.



(c) Reset gadget parameterised by a counter $i \in [1, r]$.

empty run does the job) and q_f is not the source location of any transition in R (otherwise, we duplicate q_f, one copy having no outgoing transitions).

The model $\mathcal{S} = (\mathrm{S}, (\mathrm{R}_{\mathsf{a}})_{\mathsf{a} \in \mathsf{Act}}, \mathrm{L})$ has action set $\mathsf{Act} \stackrel{\text{def}}{=} R \cup \{\sharp\}$ ($\sharp$ plays the role of an end marker), and we construct it so that there exists a plan $\sigma \in \mathsf{Act}^*$ that witnesses the satisfaction of φ in $\mathcal{S}$ if and only if $\sigma = \rho\sharp$, where $\rho \in R^*$ corresponds to a run of the RVASS $\mathcal{V}$ from the initial configuration $(q_0, \mathbf{0})$ to a configuration with location q_f. The model $\mathcal{S}$ consists of the union of three types of gadgets ($\bot$ and $\top$ are shared states). In Figs. 2a–2c, an edge of the form $s \xrightarrow{\mathsf{a},u} s'$ denotes $(s, s') \in \mathrm{R}_{\mathsf{a}}$ with $\mathit{wf}(s, \mathsf{a}, s') = u$. An edge of the form $s \xrightarrow{X,u} s'$ with $X \subseteq \mathsf{Act}$ means that for all $\mathsf{a} \in X$, we have $(s, s') \in \mathrm{R}_{\mathsf{a}}$ with $\mathit{wf}(s, \mathsf{a}, s') = u$.

In the construction, we wish to guarantee that from any state, any action is fireable (so that strong executability is immediate), which may overcomplicate the definition in a few places, in particular for the structural gadget below.

- The *structural gadget* (Fig. 2a) checks that the first action of the plan is a transition in $R^{src}(q_0)$, the penultimate action is a transition in $R^{tgt}(q_f)$ and the plan is a non-empty sequence followed by $\sharp$.
- The *sequence gadget* (Fig. 2b), parameterised by $q \in Q \setminus \{q_f\}$, checks that consecutive actions in the plan form a path in the control graph of the RVASS: whenever a transition in $\mathrm{R}^{tgt}(q)$ occurs in a plan, it must be followed by a transition in $\mathrm{R}^{src}(q)$.

– The *reset gadget* (Fig. 2c) checks that the i-th counter ($i = 1, \ldots, r$) remains non-negative between two resets: it has two states labelled by `init`, and guesses either a reset transition (from the leftmost state u_0^i), or the beginning of the run (from the middle state u_1^i) and tracks the value of the i-th counter until another reset transition or the final state is reached (witnessed by $\sharp$).

The model $\mathcal{S}$ consists of the union of (1) the structural gadget, (2) the sequence gadgets instantiated with all $q \in Q \setminus \{q_f\}$ and (3) the reset gadgets instantiated with $i \in [1, r]$. Using the construction above, one can show that $\mathcal{S} \Vdash \mathsf{Kh}^0(\mathtt{init}, \mathtt{acc})$ iff there is a plan $\sigma = \mathtt{a}_1 \cdots \mathtt{a}_L$ such that for all computations $s_0 \xrightarrow{\mathtt{a}_1} s_1 \cdots \xrightarrow{\mathtt{a}_L} s_L$ such that $\mathcal{S}, s_0 \Vdash \mathtt{init}$, we have: (a) $\mathcal{S}, s_L \Vdash \mathtt{acc}$; (b) for all $i \in [1, L]$, $\mathit{wf}(\lambda_{\leq i}) \geq 0$ (the condition about SE trivially holds, since for all states s and actions $\mathtt{a}$, there is a state s' such that $(s, s') \in \mathrm{R}_{\mathtt{a}}$).

It is easy to see that the size of $\mathcal{S}$ is polynomial in the size of the RVASS, and Ack-hardness of the model-checking problem for $\mathcal{L}_{kh}(1)$ follows from the fact that the control-state reachability problem for RVASS is Ack-hard [43]. □

Then, from Corollary 1 and Theorem 3 we conclude that Theorem 1 follows. By Lemma 1, `UCReach` is Ack-hard too.

6 Final Remarks

We studied the effects of adding budget-like constraints to the knowing how logic from [44], following a fruitful paradigm already used in formalisms like energy games and ATL-like logics. We proved that the model-checking problem for $\mathcal{L}_{kh}(\star)$ is Ack-complete (Thm. 1). The proof relies on the new universal control-state reachability problem over complete VASS (`UCReach`). To get Ack-membership, we designed an underlying well-structured transition system and showed the effectivity of a backward algorithm for some covering problem, following developments from [21,38]. The Ack-hardness is obtained by reducing the control-state reachability problem for VASS with resets. Hopefully, our work enriches the set of interesting problems on VASS and the scope of their applications. As future work, it would be interesting to find further applications of `UCReach`.

Acknowledgments. We would like to thank the anonymous reviewers for their comments and suggestions that helped us to improve the quality of the document. R. Fervari is supported by Agencia I+D+i grant PICT 2021-00400, the EU H2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement 101008233 (MISSION), the IRP SINFIN, SeCyT-UNC grant 33620230100178CB, and the France 2030 program ANR-11-IDEX-0003.

References

1. Alechina, N., Bulling, N., Demri, S., Logan, B.: On the Complexity of Resource-Bounded Logics. TCS **750**, 69–100 (2018)

2. Alechina, N., Bulling, N., Logan, B., Nguyen, H.: The virtues of idleness: A decidable fragment of resource agent logic. Artificial Intelligence **245**, 56–85 (2017)
3. Areces, C., Cassano, V., Castro, P., Fervari, R., Saravia, A.R.: How Easy it is to Know How: An Upper Bound for the Satisfiability Problem. In: JELIA'23. LNCS, vol. 14281, pp. 405–419. Springer (2023)
4. Blockelet, M., Schmitz, S.: Model-checking coverability graphs of vector addition systems. In: MFCS'11. LNCS, vol. 6907, pp. 108–119. Springer (2011)
5. Bojańczyk, M., David, C., Muscholl, A., Schwentick, T., Segoufin, L.: Two-variable logic on data words. ACM Transactions on Computational Logic **12**(4), 27 (2011)
6. Bouyer, P., Fahrenberg, U., Larsen, K., Markey, N., Srba, J.: Infinite Runs in Weighted Timed Automata with Energy Constraints. In: FORMATS'08. LNCS, vol. 5215, pp. 33–47. Springer (2008)
7. Bozzelli, L., Ganty, P.: Complexity Analysis of the Backward Coverability Algorithm for VASS. In: RP'11. LNCS, vol. 6945, pp. 96–109. Springer (2011)
8. Bulling, N., Goranko, V.: Combining quantitative and qualitative reasoning in concurrent multi-player games. Autonomous Agents and Multi-Agent Systems **36**(2), 1–33 (2022)
9. Cao, R., Naumov, P.: Budget-Constrained Dynamics in Multiagent Systems. In: IJCAI'17. pp. 915–921. ijcai.org (2017)
10. Cardoza, E., Lipton, R., Meyer, A.: Exponential space complete problems for petri nets and commutative semigroups: Preliminary report. In: STOC'76. pp. 50–54. ACM (1976)
11. Chatterjee, K., Doyen, L., Henzinger, T.: The Cost of Exactness in Quantitative Reachability. In: Models, Algorithms, Logics and Tools - Essays Dedicated to Kim Guldstrand Larsen on the Occasion of His 60th Birthday. LNCS, vol. 10460, pp. 367–381. Springer (2017)
12. Cimatti, A., Pistore, M., Roveri, M., Traverso, P.: Weak, strong, and strong cyclic planning via symbolic model checking. Artificial Intelligence **147**(1-2), 35–84 (2003)
13. Czerwinski, W., Orlikowski, L.: Reachability in vector addition systems is Ackermann-complete. In: STOC'21. pp. 1229–1240. IEEE (2021)
14. David, C.: Analyse de XML avec données non-bornées. Ph.D. thesis, LIAFA, Université Paris VII (2009)
15. Degorre, A., Doyen, L., Gentilini, R., Raskin, J., Torunczyk, S.: Energy and Mean-Payoff Games with Imperfect Information. In: CSL'10. LNCS, vol. 6247, pp. 260–274. Springer (2010)
16. Delzanno, G., Sangnier, A., Zavattaro, G.: Parameterized verification of ad hoc networks. In: CONCUR'10. LNCS, vol. 6269. Springer (2010)
17. Demri, S.: On selective unboundedness of VASS. JCSS **79**(5), 689–713 (2013)
18. Demri, S., Fervari, R.: Model-Checking for Ability-Based Logics with Constrained Plans. In: AAAI'23. pp. 6305–6312. AAAI Press (2023)
19. Esparza, J.: Decidability and Complexity of Petri Net Problems — An Introduction. In: Advances in Petri Nets 1998. LNCS, vol. 1491, pp. 374–428. Springer (1998)
20. Figueira, D., Figueira, S., Schmitz, S., Schnoebelen, P.: Ackermannian and Primitive-Recursive Bounds with Dickson's Lemma. In: LiCS'11. pp. 269–278 (2011)
21. Finkel, A., Schnoebelen, P.: Well-structured transitions systems everywhere! TCS **256**(1–2), 63–92 (2001)
22. Hofman, P., Totzke, P.: Trace inclusion for one-counter nets revisited. TCS **735**, 50–63 (2018)

23. Hopcroft, J., Pansiot, J.: On the reachability problem for 5-dimensional vector addition systems. TCS **8**, 135–159 (1979)
24. Kaiser, A., Kroening, D., Wahl, T.: Dynamic cutoff detection in parameterized concurrent programs. In: CAV'10. LNCS, vol. 6174, pp. 645–659. Springer (2010)
25. Karp, R.M., Miller, R.E.: Parallel Program Schemata. JCSS **3**(2), 147–195 (1969)
26. Kosaraju, R.: Decidability of reachability in vector addition systems. In: STOC'82. pp. 267–281 (1982)
27. Lambert, J.: A structure to decide reachability in petri nets. TCS **99**, 79–104 (1992)
28. Lazić, R., Schmitz, S.: The ideal view on Rackoff's coverability technique. Information and Computation **277**, 104582 (2021)
29. Leroux, J.: The general vector addition system reachability problem by Presburger inductive invariants. In: LiCS'09. pp. 4–13. IEEE (2009)
30. Leroux, J.: Vector Addition System Reachability Problem (A Short Self-Contained Proof). In: POPL'11. pp. 307–316 (2011)
31. Leroux, J.: The Reachability Problem for Petri Nets is Not Primitive Recursive. In: FOCS'21. pp. 1241–1252. IEEE (2021)
32. Leroux, J., Schmitz, S.: Reachability in vector addition systems is primitive-recursive in fixed dimension. In: LiCS'19. pp. 1–13. IEEE (2019)
33. Li, Y.: Knowing what to do: a logical approach to planning and knowing how. Ph.D. thesis, University of Groningen (2017)
34. Li, Y., Wang, Y.: Achieving While Maintaining: - A Logic of Knowing How with Intermediate Constraints. In: ICLA'17. LNCS, vol. 10119, pp. 154–167. Springer (2017)
35. Lipton, R.: The Reachability Problem Requires Exponential Space. Tech. Rep. 62, Department of Computer Science, Yale University (1976)
36. Mayr, E.: An algorithm for the general petri net reachability problem. SIAM Journal of Computing **13**(3), 441–460 (1984)
37. Minsky, M.: Computation, Finite and Infinite Machines. Prentice Hall (1967)
38. Pérez, G.: The fixed initial credit problem for partial-observation energy games is Ack-complete. IPL **118**, 91–99 (2017)
39. Rackoff, C.: The covering and boundedness problems for vector addition systems. TCS **6**(2), 223–231 (1978)
40. Reutenauer, C.: The mathematics of Petri nets. Masson and Prentice (1990)
41. Schmitz, S.: Algorithmic complexity of well-quasi-orders (November 2017), habilitation thesis
42. Schmitz, S.: Complexity Hierarchies beyond Elementary. ACM Transactions on Computation Theory **8**(1), 3:1–3:36 (2016)
43. Schnoebelen, P.: Revisiting Ackermann-Hardness for Lossy Counter Machines and Reset Petri Nets. In: MFCS'10. LNCS, vol. 6281, pp. 616–628. Springer (2010)
44. Wang, Y.: A Logic of Knowing How. In: LORI'15. LNCS, vol. 9394, pp. 392–405. Springer (2015)
45. Wang, Y.: A logic of goal-directed knowing how. Synthese **195**(10), 4419–4439 (2018)
46. Wang, Y.: Beyond knowing that: a new generation of epistemic logics. In: van Ditmarsch, H., Sandu, G. (eds.) Jaakko Hintikka on knowledge and game theoretical semantics, pp. 499–533. Springer (2018)