Weighted Distributed Systems and Their Logics

Benedikt Bollig[1] and Ingmar Meinecke[2]

[1] LSV, CNRS UMR 8643 & ENS de Cachan
61 Av. du Président Wilson, F-94235 Cachan Cedex, France
bollig@lsv.ens-cachan.fr
[2] Institut für Informatik, Universität Leipzig
Johannisgasse 26, D-04103 Leipzig, Germany
meinecke@informatik.uni-leipzig.de

Abstract. We provide a model of weighted distributed systems and give a logical characterization thereof. Distributed systems are represented as weighted asynchronous cellular automata. Running over directed acyclic graphs, Mazurkiewicz traces, or (lossy) message sequence charts, they allow for modeling several communication paradigms in a unifying framework, among them probabilistic shared-variable and probabilistic lossy-channel systems. We show that any such system can be described by a weighted existential MSO formula and, vice versa, any formula gives rise to a weighted asynchronous cellular automaton.

1 Introduction

Classical automata theory has become an indispensable tool in many modern areas of computer science, supporting, for example, programming languages and specification and verification techniques. In some applications, automata need to cope with quantitative phenomena. Then, taking a transition in an automaton is accompanied by measuring its cost or weight. For example, a system might provide a counter tracking the number of occurrences of a given pattern; or its behavior might depend on probability laws so that the outcome of a transition is generally uncertain and depends on a probability distribution. Actually, automata with weights enjoy manifold applications in numerous areas such as speech recognition [18], probabilistic systems [12, 1], and image compression [4].

Formally, the behavior of a weighted automaton is no longer characterized by the pure existence of an accepting run. Rather, a weighted automaton comes up with a *formal power series* assigning to any possible execution sequence a value from a semiring. More precisely, the values collected along an automaton execution are multiplied, whereas nondeterminism is resolved by summation therewith generalizing the two operations of the two-valued Boolean algebra, cf. [13].

For a long time, the correspondence of automata and logic has been a captivating research direction in computer science. The probably most famous result goes back to Büchi and Elgot, who discovered a precise correspondence between finite automata and the logical formalism of monadic second-order (MSO) formulas [3, 11]. In particular, any system description formalized in the MSO language comes up with an implementation in terms of a finite automaton. Concerning weighted automata, most results establish Kleene-like theorems stating that a formal power series is described by a weighted automaton iff it is rational [21, 6, 15]. A logical characterization of weighted automata has been achieved only recently: Droste and Gastin opened a new research direction by providing a weighted MSO logic to define formal power series over words [7]. Their achievements have been extended, among others, to automata on infinite words [9], trees [10], pictures [16], and traces [17].

In this paper, we deal with a model for weighted distributed systems that unifies many communication paradigms such as shared-memory systems, (lossy) channel systems, etc. It is constituted by asynchronous cellular automata (ACAs) [8] running on directed acyclic graphs (dags) without auto-concurrency. Unlike finite automata, which process their input words in a sequential fashion, ACAs are appropriate to concurrent executions. Accordingly, the assignment of weights does not depend on the order in which independent events are executed. ACAs have already been equipped with weights by Kuske to recognize formal power series over traces [15]. Generalizing results by Ochmański [19] and Droste and Gastin [6], he showed that a series is regular iff it is recognized by some weighted ACA. Actually, we provide an even more general model subsuming Kuske's automata. Running over dags rather than traces, our weighted ACA can cope with many common domains for concurrency, not only traces but also message sequence charts which play a prominent role in telecommunication. As we will discuss in the course of this paper, the latter domain allows an embedding of probabilistic lossy-channel systems. Our main result states that weighted ACAs recognize precisely the formal power series that are definable in an existential fragment of a weighted MSO logic over dags. This result cannot be obtained by a translation of the word setting as it was done for traces [17]. On the other hand, a lot of technical difficulties arise in our setting compared to this of words. Especially, we have to prove an unambiguity result for first-order definable languages before establishing the main theorem. For words such an unambiguity result is for free since deterministic devices suffice to recognize all regular languages. Moreover, the construction of a weighted formula from a given weighted asynchronous cellular automaton is much more tricky than for words.

The paper is structured as follows: in Section 2, we introduce our notion of a dag over a distributed alphabet. Hereby, a distributed alphabet constitutes the system architecture by assigning to any process its supply of actions. Section 3 introduces ACAs first in their classical, then in their weighted form. The behavior of a weighted ACA will be described in terms of a formal power series over (a subset of) the class of dags. Having introduced weighted MSO logic over dags in Section 4, Sections 5 and 6 derive our main result, the precise correspondence between weighted ACAs and the existential fragment of weighted MSO logic.

2 Dags over Distributed Alphabets

We fix a nonempty finite set Ag of *agents*, a *distributed alphabet* $\widetilde{\Sigma}$, which is a tuple $(\Sigma_i)_{i \in Ag}$ of (not necessarily disjoint) alphabets Σ_i, and an alphabet C. Elements from Σ_i are understood to be *actions* that are performed by agent i. Let $\Sigma = \bigcup_{i \in Ag} \Sigma_i$ denote the set of all the actions. The actions will label the nodes of a graph, which we will later refer to as *events*. Elements from C label edges of a graph to provide a kind of control information. For example, they might reflect the type of a message represented by an edge between communicating events. A (directed) *graph* over (Σ, C) is a structure $(V, \{\lhd_\ell\}_{\ell \in C}, \lambda)$ where V is its finite set of *nodes*, $\lhd_\ell \subseteq V \times V$ are disjoint binary relations on V, and $\lambda : V \to \Sigma$ is the *labeling function*. We call $\lhd := \bigcup_{\ell \in C} \lhd_\ell$ the *edge relation* and set $\leq\, = \,\lhd^*$ and $<\, = \,\lhd^+$. For $u, v \in V$, we define the *cover relation* $u \lessdot v$ of $\leq$ by $u < v$ and, for any $w \in V$, $u < w \leq v$ implies $w = v$. A *directed acyclic graph* (dag) over (Σ, C) is a graph $(V, \{\lhd_\ell\}_{\ell \in C}, \lambda)$ over (Σ, C) such that $\lhd$ is irreflexive and $\leq$ is a partial order. The set of all those dags is denoted by $\mathbb{DAG}(\Sigma, C)$. For $a \in \Sigma$, we put $loc(a) := \{i \in Ag \mid a \in \Sigma_i\}$. Then, a and b are *independent*, writing $a\ I_{\widetilde{\Sigma}}\ b$, if $loc(a) \cap loc(b) = \emptyset$. Otherwise, we say a and b are *dependent*, writing $a\ D_{\widetilde{\Sigma}}\ b$.

We now introduce the models representing the behavior of a system of communicating agents. In doing so, we combine and extend the models from [8, 14, 2].

Definition 2.1. *A $(\widetilde{\Sigma}, C)$*-dag *is a dag $(V, \{\lhd_\ell\}_{\ell \in C}, \lambda) \in \mathbb{DAG}(\Sigma, C)$ where*

- *for any $i \in Ag$, $\lambda^{-1}(\Sigma_i)$ is totally ordered by $\leq$ and*
- *for any $\ell \in C$ and $(u, v), (u', v') \in \lhd_\ell$ with $\lambda(u) D_{\widetilde{\Sigma}} \lambda(u')$ and $\lambda(v) D_{\widetilde{\Sigma}} \lambda(v')$, we have $u \leq u'$ iff $v \leq v'$.*

The set of all $(\widetilde{\Sigma}, C)$-dags is denoted by $\mathbb{DAG}(\widetilde{\Sigma}, C)$.

The first condition reflects that a single agent is considered to operate sequentially. Especially, there is no auto-concurrency. The second condition ensures a FIFO architecture of communicating systems. Messages (u, v) and (u', v') of the same type between the same agents are received in the same order as they have been sent. Because of the FIFO-architecture and the absence of auto-concurrency, we conclude that, in a $(\widetilde{\Sigma}, C)$-dag $(V, \{\lhd_\ell\}_{\ell \in C}, \lambda)$, for any $u \in V$, $\ell \in C$, and $a \in \Sigma$, there is at most one vertex $v \in V$ such that both $u \lhd_\ell v$ (or $v \lhd_\ell u$) and $\lambda(v) = a$.[3] If C is a singleton, we actually deal with structures $(V, \lhd, \lambda)$ and we speak of $\widetilde{\Sigma}$-dags.

The automaton model as introduced in the next section monitors for every node $u \in V$ of a $(\widetilde{\Sigma}, C)$-dag $(V, \{\lhd_\ell\}_{\ell \in C}, \lambda)$ the direct neighborhood of u. Therefore, we introduce the following abbreviations: For $u \in V$, we denote by $\mathrm{Read}(u) := \{(a, \ell) \in \Sigma \times C \mid \exists v \in V : v \lhd_\ell u \wedge \lambda(v) = a\}$ the *read domain* of u and, given $(a, \ell) \in \mathrm{Read}(u)$, let $(a, \ell)\text{-pred}(u)$ be the unique vertex v such that both $v \lhd_\ell u$ and $\lambda(v) = a$. Similarly, let $\mathrm{Write}(u) := \{(a, \ell) \in \Sigma \times C \mid \exists v \in V : u \lhd_\ell v \wedge \lambda(v) = a\}$ be the *write domain* of u and, for $(a, \ell) \in \mathrm{Write}(u)$, $(a, \ell)\text{-succ}(u)$ denote the unique vertex v such that both $u \lhd_\ell v$ and $\lambda(v) = a$. For $i \in Ag$ and $V_i = \{u \in V \mid \lambda(u) \in \Sigma_i\}$, sequential progress of an agent $i \in Ag$ is reflected by $\lhd_i := \lhd \cap (V_i \times V_i)$ and the total order $\leq_i := \leq \cap (V_i \times V_i)$ (do not mistake relation $\lhd_i$ of agent i for edge relation $\lhd_\ell$ for $\ell \in C$). For $u \in V$ and $i \in Ag$, u is *Σ_i-maximal* if $u \in V_i$ and there is no $v \in V_i$ such that $u < v$. Obviously, there is at most one Σ_i-maximal vertex.

Dags over distributed alphabets subsume popular domains of concurrency:

Example 2.1 (Mazurkiewicz Traces [5]). We consider distributed systems where an action $a \in \Sigma$ is executed simultaneously by any component $i \in loc(a)$. The behavior of

[3] As a consequence, the underlying graph has bounded degree. This property is essential in establishing the coincidence between recognizability and logical definability [2].

such a "shared-memory" system is described naturally by a set of traces. Commonly, traces are defined as congruence classes of words or as dependence graphs. In our setting, we model a trace as the union of the Hasse diagrams of the total orders of the different agents. Moreover, the labeling of an edge between two nodes u and v provides information about which agents execute u and v consecutively. In detail, a *trace* over $\widetilde{\Sigma}$ is a dag $(V, \{\lhd_\ell\}_{\ell \in 2^{Ag}}, \lambda)$ from $\mathbb{DAG}(\widetilde{\Sigma}, 2^{Ag})$ such that both $\lhd = \bigcup_{i \in Ag} \lessdot_i$ and, for any $(u, v) \in \lhd$ and $\ell \in 2^{Ag}$, $u \lhd_\ell v$ iff $\ell = \{i \in Ag \mid u \lessdot_i v\}$ (recall that $\lessdot_i$ is the cover relation of $\leq_i$). This modeling of a trace will turn out to be tremendously helpful when simulating shared-memory systems in terms of asynchronous cellular automata, as the edge relation will be used to access, for any event u and any agent $i \in \mathit{loc}(\lambda(u))$, the current state of $i \in Ag$ right before executing u. A sample trace over $\widetilde{\Sigma} = (\{a, b, c\}, \{a, b, d\}, \{a, b\})$ (with $Ag = \{1, 2, 3\}$) is depicted in Fig. 1(a).

Example 2.2 ((Lossy) Message Sequence Charts). Another communication paradigm is that of channel systems: several components $i \in Ag$ communicate by sending and receiving messages through channels. So let $\mathit{Ch} = (Ag \times Ag) \setminus \mathit{id}_{Ag}$ be the set of channels. To model the behavior of such a system, we need to fix supplies of send and receive actions: for $i \in Ag$, let Γ_i denote $\{i!j \mid (i, j) \in \mathit{Ch}\} \cup \{i?j \mid (i, j) \in \mathit{Ch}\}$, the set of *(communication) actions* of agent i. Action $i!j$ reads as "i sends a message to j". Accordingly, $j?i$ is the complementary receive action. Let $\widetilde{\Gamma}$ be the distributed alphabet $(\Gamma_i)_{i \in Ag}$. A *message sequence chart* (MSC) over Ag is a $\widetilde{\Gamma}$-dag $(V, \lhd, \lambda)$ such that, for any $i \in Ag$, $\lhd_i$ is the cover relation of $\leq_i$, for any $(u, v) \in \lhd$ with $\lambda(u)\ I_{\widetilde{\Gamma}}\ \lambda(v)$, $\lambda(u) = i!j$ and $\lambda(v) = j?i$ for some i, j, and, for any $u \in V$, there is $v \in V$ satisfying both $\lambda(u)\ I_{\widetilde{\Gamma}}\ \lambda(v)$ and either $u \lhd v$ or $v \lhd u$. Observe that, due to the general definition of a $\widetilde{\Gamma}$-dag, we deal with a model for FIFO communication. If we do not require a send event to be followed by a corresponding receive event, we deal with a *lossy* MSC. More precisely, the last condition in the definition of an MSC is weakened as follows: for any $v \in V$ with $\lambda(v)$ a receive action, there is $u \in V$ satisfying $\lambda(u)\ I_{\widetilde{\Gamma}}\ \lambda(v)$ and $u \lhd v$. Figure 1(b) depicts an MSC over $\{1, 2\}$, whereas the structure from Fig. 1(c) is not an MSC but a lossy MSC.

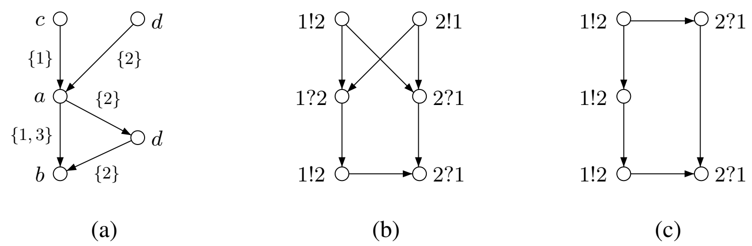


Fig. 1. A trace over $(\{a, b, c\}, \{a, b, d\}, \{a, b\})$, an MSC over $\{1, 2\}$, and a lossy MSC over $\{1, 2\}$ that is not an MSC

3 Weighted Asynchronous Cellular Automata

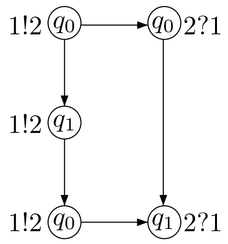
First we provide the unweighted model of an asynchronous cellular automaton, similar to the one proposed in [2]. Actually, we deal with asynchronous cellular automata *with types* (ACATs) over $(\widetilde{\Sigma}, C)$-dags, which have limited access to the future. To express "communication requests", a type function associates with any action a and any state q the set of actions that henceforth "communicate" with a, provided executing a results in state q. Regarding lossy MSCs, for example, we might require an event labeled with a send action $1!2$ to be followed by the suitable receive event, which is then labeled with the communication action $2?1$. For some classes the expressive power of ACAs with and without types coincide. But in general, omitting the type function severely restricts the expressive power of ACATs [2].

Definition 3.1. *An* asynchronous cellular automaton with types *(ACAT) over* $(\widetilde{\Sigma}, C)$ *is a structure* $\mathcal{A} = (Q, \Delta, T, F)$ *where*

- Q *is the nonempty finite set of* states*,*
- $\Delta \subseteq \mathit{Trans}_{(\widetilde{\Sigma},C)}(Q) := (Q \cup \{-\})^{\Sigma \times C} \times \Sigma \times Q$ *is the set of* transitions*,*
- $T : (\Sigma \times Q) \to 2^{\Sigma \times C}$ *is the* type function*, and*
- $F \subseteq (Q \cup \{\imath\})^{Ag}$ *is the set of global* final states.

We often write $(\overline{q}, a, q) \in \Delta$ with $\overline{q} \in (Q \cup \{-\})^{\Sigma \times C}$ as $\overline{q} \longrightarrow (a, q)$. Note that $\overline{q}[(b, \ell)] = -$ means that there is no (b, ℓ)-predecessor. Hence, we will sometimes write $\overline{q}$ as an element from $\mathfrak{P}(\Sigma \times C \times Q)$. The idea of a run of an asynchronous cellular automaton $\mathcal{A}$ on a $(\widetilde{\Sigma}, C)$-dag $\mathcal{D} = (V, \{\lhd_\ell\}_{\ell \in C}, \lambda)$ is an additional labeling of the nodes $u \in V$ with states $q \in Q$ such that the local neighborhoods match the transitions, after executing $\mathcal{D}$ the system is in a final state, and the requests of the type function are satisfied.

First, let us consider the following example: $\mathcal{A} = (Q, \Delta, T, F)$ running on lossy MSCs over agents $\{1, 2\}$, cf. Example 2.2. We put $Q = \{q_0, q_1\}$. Now, the following transitions are in Δ: $\emptyset \to (1!2, q_0)$, $(1!2, q_0) \to (1!2, q_1)$, $(1!2, q_1) \to (1!2, q_0)$, $(1!2, q_0) \to (2?1, q_0)$, $\{(1!2, q_0), (2?1, q_0)\} \to (2?1, q_1)$, and $\{(1!2, q_1), (2?1, q_0)\} \to (2?1, q_1)$. Moreover, we put $T(1!2, q_0) = \{2?1\}$ and $F = \{(1, q_0), (2, q_1)\}$. Then the picture on the left hand side depicts a successful run of $\mathcal{A}$ on the lossy MSC from Figure 1(c). For every node, the node itself together with its read domain is covered by a transition. Furthermore, agent 1 stops in q_0 and agent 2 in q_1. Last but not least, every send event $1!2$ in state q_0 is followed by a receive event $2?1$ as imposed by the type function.



To be precise, let $\rho : V \to Q$. We write $(\mathcal{D}, \rho)$ to denote the dag $(V, \{\lhd_\ell\}_{\ell \in C}, (\lambda, \rho))$ over $(\Sigma \times Q, C)$. For $(\mathcal{D}, \rho)$, let $\mathit{trans}_{(\mathcal{D},\rho)} : V \to \mathit{Trans}_{(\widetilde{\Sigma},C)}(Q)$ describe the downward local neighborhood, i.e., for any $u \in V$ let $\mathit{trans}_{(\mathcal{D},\rho)}(u) = (\overline{q}, \lambda(u), \rho(u))$ where, for any $(b, \ell) \in \Sigma \times C$,

$$\overline{q}[(b, \ell)] = \begin{cases} - & \text{if } (b, \ell) \notin \mathrm{Read}(u), \\ \rho((b, \ell)\text{-pred}(u)) & \text{if } (b, \ell) \in \mathrm{Read}(u). \end{cases}$$

Moreover, we define $\mathit{final}_{(\mathcal{D},\rho)} \in (Q \cup \{\imath\})^{Ag}$ by $\mathit{final}_{(\mathcal{D},\rho)}[i] = \imath$ for any agent $i \in Ag$ with $V_i = \emptyset$. Otherwise, $\mathit{final}_{(\mathcal{D},\rho)}[i] = \rho(u)$ where u is Σ_i-maximal in V. Thus, if the system starts in the global state $(\imath)_{i \in Ag}$ and executes $\mathcal{D}$, then it ends up in the global state $\mathit{final}_{(\mathcal{D},\rho)}$. Now a *run* of $\mathcal{A}$ on $\mathcal{D}$ is a mapping $\rho : V \to Q$ such that, for any $u \in V$, $\mathit{trans}_{(\mathcal{D},\rho)}(u) \in \Delta$. Moreover, ρ is *accepting* if both $\mathit{final}_{(\mathcal{D},\rho)} \in F$ and, for any $u \in V$, we have $T(\lambda(u), \rho(u)) \subseteq \mathrm{Write}(u)$. The intuition behind the latter condition is that we require $\mathrm{Write}(u)$ to contain at least the communication requests imposed by the type function of the automaton. The language $L(\mathcal{A})$ is the set of all $\mathcal{D}$ such that there is at least one accepting run of $\mathcal{A}$ on $\mathcal{D}$. We call $\mathcal{A}$ *unambiguous* if, for any $(\widetilde{\Sigma}, C)$-dag $\mathcal{D}$ and any two accepting runs ρ, ρ' of $\mathcal{A}$ on $\mathcal{D}$, we have $\rho = \rho'$.

A set $L \subseteq \mathbb{DAG}(\widetilde{\Sigma}, C)$ is called *recognizable* if $L(\mathcal{A}) = L$ for some ACAT $\mathcal{A}$ over $(\widetilde{\Sigma}, C)$. Similarly, we say that L is *unambiguously recognizable* if $L(\mathcal{A}) = L$ for some unambiguous ACAT $\mathcal{A}$ over $(\widetilde{\Sigma}, C)$.

A weighted automaton is no longer characterized by the set of accepted executions. Rather, it assigns to any possible execution a value from a semiring. A semiring is a structure $\mathbb{K} = (K, \oplus, \circ, \mathbb{0}, \mathbb{1})$ with two binary operations, addition and multiplication, and constants $\mathbb{0}$ and $\mathbb{1}$, such that $(K, \oplus, \mathbb{0})$ is a commutative monoid, $(K, \circ, \mathbb{1})$ is a monoid, multiplication distributes over addition, and $\mathbb{0} \circ k = k \circ \mathbb{0}$ for any $k \in K$. We say $\mathbb{K}$ is *commutative* if the multiplication $\circ$ is commutative. Sample semirings are $(\mathbb{N}, +, \cdot, 0, 1)$, the 2-valued Boolean algebra $\mathbb{B} = (\{\mathbb{0}, \mathbb{1}\}, \vee, \wedge, \mathbb{0}, \mathbb{1})$, and the probabilistic semiring $\mathbb{P} = ([0,1], \max, \cdot, 0, 1)$. Throughout this paper, we fix a commutative semiring $\mathbb{K} = (K, \oplus, \circ, \mathbb{0}, \mathbb{1})$. Commutativity is needed for a proper definition of automata behavior and several closure properties.

Definition 3.2. *A* **weighted** asynchronous cellular automaton with types *(wACAT) over* $\mathbb{K}$ *and* $(\widetilde{\Sigma}, C)$ *is a structure* (Q, μ, T, γ) *where*

- Q *is the nonempty finite set of* states*,*
- $\mu : \mathit{Trans}_{(\widetilde{\Sigma},C)}(Q) \to \mathbb{K}$ *is the* transition weight function*,*
- $T : (\Sigma \times Q) \to 2^{\Sigma \times C}$ *is the* type function*, and*
- $\gamma : (Q \cup \{\imath\})^{Ag} \to \mathbb{K}$ *is the* final weight function*.*

In a wACAT, the values of a semiring that are collected along an execution of the automaton are multiplied, whereas nondeterminism is resolved by summation. The behavior of such an automaton will be a function $S : \mathbb{DAG}(\widetilde{\Sigma}, C) \to \mathbb{K}$, also called a *formal power series*. The collection of all these functions is denoted by $\mathbb{K}\langle\langle \mathbb{DAG}(\widetilde{\Sigma}, C) \rangle\rangle$.

More precisely: let $\mathcal{D} = (V, \{\lhd_\ell\}_{\ell \in C}, \lambda)$ be a $(\widetilde{\Sigma}, C)$-dag. In the weighted setting, every mapping $\rho : V \to Q$ is referred to as a *run*. The *weight* of ρ is the product

$$\mathit{weight}(\mathcal{D}, \rho) := \Big(\prod_{u \in V} \mu(\mathit{trans}_{(\mathcal{D},\rho)}(u)) \Big) \circ \gamma(\mathit{final}_{(\mathcal{D},\rho)}) \,.$$

We call ρ *successful* if $T(\lambda(u), \rho(u)) \subseteq \mathrm{Write}(u)$ for any $u \in V$. We thus can assign to $\mathcal{A}$ a formal power series $\|\mathcal{A}\| \in \mathbb{K}\langle\langle \mathbb{DAG}(\widetilde{\Sigma}, C) \rangle\rangle$ by

$$(\|\mathcal{A}\|, \mathcal{D}) := \bigoplus_{\substack{\rho : V \to Q \\ \rho \text{ successful}}} \mathit{weight}(\mathcal{D}, \rho)$$

for any $\mathcal{D} = (V, \{\lhd_\ell\}_{\ell \in C}, \lambda) \in \mathbb{DAG}(\widetilde{\Sigma}, C)$. Note that, in the context of formal power series, $(\|\mathcal{A}\|, \mathcal{D})$ is a common notation for $\|\mathcal{A}\|(\mathcal{D})$.

For $L \subseteq \mathbb{DAG}(\widetilde{\Sigma}, C)$, the *characteristic series* $\mathbb{1}_L : \mathbb{DAG}(\widetilde{\Sigma}, C) \to \mathbb{K}$ is given by $(\mathbb{1}_L, \mathcal{D}) = \mathbb{1}$ if $\mathcal{D} \in L$ and $(\mathbb{1}_L, \mathcal{D}) = \mathbb{0}$ if $\mathcal{D} \notin L$. We say that $S \in \mathbb{K}\langle\langle \mathbb{DAG}(\widetilde{\Sigma}, C)\rangle\rangle$ is *recognizable* if there is a wACAT $\mathcal{A}$ with $\|\mathcal{A}\| = S$.

Example 3.1 (Probabilistic Lossy-Channel Systems [20]). A probabilistic lossy-channel system is a tuple $\mathcal{P} = ((Q_i, \delta_i)_{i \in Ag}, \overline{q}^{in}, (r_{ij}(q))_{(i,j) \in Ch, q \in Q_i})$: with any agent i, we associate a sequential process, which is composed of a finite state space Q_i and a transition relation $\delta_i \subseteq Q_i \times \Gamma_i \times Q_i$. Recall that Γ_i comprises the set of communication actions executed by agent i, i.e., actions of the form $i!j$ or $i?j$ with $i \neq j$. We shall assume δ_i to be deterministic, i.e., for any $q \in Q_i$ and $\sigma \in \Gamma_i$, there is at most one $q' \in Q_i$ such that $(q, \sigma, q') \in \delta_i$. Moreover, the system is equipped with a global initial state $\overline{q}^{in} \in \prod_{i \in Ag} Q_i$. There is an unreliable channel in between any two agents i and j with $i \neq j$, i.e., depending on a state $q \in Q_i$ in which a message is sent, a channel (i, j) has a reliability $r_{ij}(q) \in [0, 1]$. Thus, the message arrives at agent j with probability $r_{ij}(q)$ and is lost with probability $1 - r_{ij}(q)$.

We will give the probabilistic lossy-channel system $\mathcal{P}$ a semantics in terms of a wACAT $\mathcal{A}_{\mathcal{P}} = (Q, \mu, T, \gamma)$ over $\mathbb{P} = ([0, 1], \max, \cdot, 0, 1)$ and $\widetilde{\Gamma}$ reading lossy MSCs where $Q = (\bigcup_{i \in Ag} Q_i) \times \{\mathsf{success}, \mathsf{failure}, \mathsf{rec}\}$. Here, we give just the idea of the construction. Roughly speaking, we shift the reliabilities of the channels to the sequential processes. Then a state with second component $\mathsf{success}$ is assigned to a send event that succeeds in delivering a message, which is guaranteed by the type function, i.e., T maps a pair of the form $(i!j, (q, \mathsf{success}))$ to $\{j?i\}$ and any other pair to the empty set. Such a $\mathsf{success}$-state is entered with the probability that the transmission succeeds. In contrast, a send event that is equipped with a state that carries the attribute $\mathsf{failure}$ is entered with the probability that the transmission fails. Thus, it cannot be followed by a corresponding receive. Any other event will carry rec to indicate that we deal with a receive event. As we do not explicitly deal with final states, γ maps any possible final configuration to 1. For a lossy MSC $\mathcal{M}$, $(\|\mathcal{A}_{\mathcal{P}}\|, \mathcal{M}) \in [0, 1]$ might now be interpreted to be the probability of acceptance of $\mathcal{M}$ by $\mathcal{P}$.

Example 3.2 (Probabilistic Asynchronous Automata [12]). The model of asynchronous automata [22] over Mazurkiewicz traces represents shared-memory systems rather than channel systems. In an asynchronous automaton running on traces, any action a has to be executed simultaneously by any component $i \in loc(a)$. Probabilistic asynchronous automata have been introduced by Jesi, Pighizzini, and Sabadini [12]. In a probabilistic asynchronous automaton, the outcome of a transition depends on a probability distribution on the set of global states of the system. Formally, a *probabilistic asynchronous automaton* over $\widetilde{\Sigma}$ is a structure $\mathcal{B} = ((S_i)_{i \in Ag}, (\mathbf{P}_a)_{a \in \Sigma}, \overline{q}_0, \eta)$ where

- for each $i \in Ag$, S_i is a nonempty finite set of (i-)*local states*,
- for each $a \in \Sigma$, $\mathbf{P}_a$ is a mapping $S_a \times S_a \to [0, 1]$ such that, for any $\overline{s} \in S_a$, $\mathbf{P}_a(\overline{s}, .)$ is a probability distribution on S_a where $S_a := \{\overline{s} \in \prod_{i \in Ag}(S_i \cup \{*\}) \mid$ for any $i \in Ag,\ \overline{s}[i] = *$ iff $i \notin loc(a)\}$,
- $\overline{q}_0 \in \prod_{i \in Ag} S_i$ is the *global initial state*, and
- $\eta : \prod_{i \in Ag} S_i \to \{0, 1\}$ assigns a weight to any possible final configuration.

A probability distribution $\mathbf{P}_a(\overline{s})$ reflects that, in a global configuration from $\prod_{i \in Ag} S_i$ that coincides with $\overline{s}$ with respect to the locations from $loc(a)$, executing an a will alter at most the local states $\overline{s}$ of agents from $loc(a)$.

We provide the reader with a rather intuitive semantics of $\mathcal{B}$ and refer to [12] for details. Roughly speaking, $\mathcal{B}$ assigns to any trace a probability of acceptance. To determine the acceptance probability of a trace $\mathcal{T} = (V, \{\lhd_\ell\}_{\ell \in 2^{Ag}}, \lambda)$ over $\widetilde{\Sigma}$ (see Example 2.1), $\mathcal{B}$ will fix an arbitrary linear extension $w = (V, \leq', \lambda)$ of $\mathcal{T}$, i.e., $\leq'$ is a total-order relation containing $\leq$. As usual, w can be seen as a word $a_1 \ldots a_n \in \Sigma^*$ with $n = |V|$. Then, starting in the global initial state $\overline{q}_0$, $\mathcal{B}$ reads w letter by letter and assigns to any position $k = 1, \ldots, n$ a global state $\overline{q}_k \in \prod_{i \in Ag} S_i$ such that going from $\overline{q}_{k-1}$ to $\overline{q}_k$ changes at most the components from $loc(a_k)$, i.e., $\overline{q}_{k-1}[i] = \overline{q}_k[i]$ for any $i \notin loc(a_k)$. A step from $\overline{q}_{k-1}$ to $\overline{q}_k$ uniquely determines a pair $(\overline{s}_{k-1}, \overline{s}_k) \in S_{a_k} \times S_{a_k}$ with $\overline{s}_{k-1}[i] = \overline{s}_k[i] = *$ for any $i \notin loc(a_k)$ and $\overline{s}_{k-1}[i] = \overline{q}_{k-1}[i]$ and $\overline{s}_k[i] = \overline{q}_k[i]$ for any other i. The sequence $\overline{q}_0, \ldots, \overline{q}_n$ might be called a run of $\mathcal{B}$ on w. The weight of this particular run is the product $\prod_{k=1,\ldots,n} \mathbf{P}_{a_k}(\overline{s}_{k-1}, \overline{s}'_k) \cdot \eta(\overline{q}_n)$ (if $n = 0$, then we set its weight to be $\eta(\overline{q}_0)$). Summing up the weights of all possible runs of $\mathcal{B}$ on w determines the value $\mathbf{P}_{\mathcal{B}}(\mathcal{T}) \in [0, 1]$, the probability that $\mathcal{T}$ is accepted by $\mathcal{B}$.

Lemma 3.1. *There is a wACAT $\mathcal{A} = (Q, \mu, T, \gamma)$ over $(\mathbb{R}_{\geq 0}, +, \cdot, 0, 1)$ and $(\widetilde{\Sigma}, 2^{Ag})$ such that $|Q| \leq |\Sigma| \times |\prod_{i \in Ag} S_i|$ and $(\|\mathcal{A}\|, \mathcal{T}) = \mathbf{P}_{\mathcal{B}}(\mathcal{T})$ for any trace $\mathcal{T}$.*[4]

Proof. Let $Q = \bigcup_{a \in \Sigma} S_a$ and $T(a, \overline{s}) = \emptyset$ for any $(a, \overline{s}) \in \Sigma \times Q$.

- Suppose $t = \{((a_1, \overline{s}_1), \ell_1), \ldots, ((a_n, \overline{s}_n), \ell_n)\} \longrightarrow (a, \overline{s}) \in \mathit{Trans}_{(\widetilde{\Sigma}, 2^{Ag})}(Q)$. If $\overline{s}_k \in S_{a_k}$, $k = 1, \ldots, n$, $\overline{s} \in S_a$, and the sets $\ell_k \in 2^{Ag}$ are pairwise disjoint, then $\mu(t)$ is set to be $\mathbf{P}_a(\overline{s}', \overline{s})$ where $\overline{s}'$ is determined as follows: for any $i \in loc(a)$, $\overline{s}'[i] = \overline{q}_0[i]$ if $i \notin \bigcup_{k=1,\ldots,n} \ell_k$, and, otherwise, $\overline{s}'[i] = \overline{s}_k[i]$ for the unique $k \in \{1, \ldots, n\}$ with $i \in \ell_k$. Any other transition is mapped to 0.
- Suppose $\overline{q} \in (Q \cup \{\imath\})^{Ag}$. If there is $\overline{q}' \in \prod_{i \in Ag} S_i$ such that, for any $i \in Ag$, $\overline{q}[i] = \imath$ implies $\overline{q}'[i] = \overline{q}_0[i]$ and $\overline{q}[i] \in Q$ implies $\overline{q}'[i] = \overline{q}[i][i]$, then set $\gamma(\overline{q})$ to be $\eta(\overline{q}')$. Otherwise, set $\gamma(\overline{q})$ to be 0. $\square$

Note that (weighted) ACATs relative to traces can actually do without types. By Lemma 3.1 and Theorem 4.2, we will give, as a byproduct, a weighted formula defining the behavior of a probabilistic asynchronous automaton $\mathcal{B}$.

We collect some closure properties of recognizable series needed to show that definable series are recognizable. Let $S, S' \in \mathbb{K}\langle\langle \mathbb{DAG}(\widetilde{\Sigma}, C) \rangle\rangle$. Then, we define $k \circ S$ for $k \in \mathbb{K}$, $S + S'$, and $S \odot S'$ by $(k \circ S, \mathcal{D}) = k \circ (S, \mathcal{D})$, $(S + S', \mathcal{D}) = (S, \mathcal{D}) \oplus (S', \mathcal{D})$, and $(S \odot S', \mathcal{D}) = (S, \mathcal{D}) \circ (S', \mathcal{D})$ for any $\mathcal{D} \in \mathbb{DAG}(\widetilde{\Sigma}, C)$.

Proposition 3.1. *Let $S, S' : \mathbb{DAG}(\widetilde{\Sigma}, C) \to \mathbb{K}$ be recognizable and $k \in \mathbb{K}$. Then, $k \circ S$, $S + S'$, and $S \odot S'$ are recognizable.*

[4] Note that we calculate values in the interval $[0, 1]$ only. But unfortunately, $([0, 1], +, \cdot, 0, 1)$ is not a semiring. Therefore, we turn to $(\mathbb{R}_{\geq 0}, +, \cdot, 0, 1)$.

Now let Σ_i, Γ_i be arbitrary alphabets for $i \in Ag$ with $\Sigma = \bigcup_{i \in Ag} \Sigma_i$ and $\Gamma = \bigcup_{i \in Ag} \Gamma_i$. Moreover, let $\pi_v : \Sigma \to \Gamma$ such that $\pi_v(\Sigma_i) \subseteq \Gamma_i$ for all $i \in Ag$ and $(a, b) \in D_{\widetilde{\Sigma}}$ iff $(\pi_v(a), \pi_v(b)) \in D_{\widetilde{\Gamma}}$. Then, we call $\pi : \mathbb{DAG}(\widetilde{\Sigma}, C) \to \mathbb{DAG}(\widetilde{\Gamma}, C)$ with $\pi(\mathcal{D}) = (V, \{\lhd_l\}_{l \in C}, \pi_v \circ \lambda)$ for $\mathcal{D} = (V, \{\lhd_l\}_{l \in C}, \lambda) \in \mathbb{DAG}(\widetilde{\Sigma}, C)$ a *projection* from $\mathbb{DAG}(\widetilde{\Sigma}, C)$ to $\mathbb{DAG}(\widetilde{\Gamma}, C)$. Note that $\pi(\mathcal{D})$ is indeed a $(\widetilde{\Gamma}, C)$-dag because of the properties of π_v. For $S \in \mathbb{K}\langle\langle \mathbb{DAG}(\widetilde{\Sigma}, C)\rangle\rangle$, let $\pi(S)$ be the series defined for every $\mathcal{D}' \in \mathbb{DAG}(\widetilde{\Gamma}, C)$ by $(\pi(S), \mathcal{D}') = \bigoplus_{\mathcal{D} \in \pi^{-1}(\mathcal{D}')} (S, \mathcal{D})$.

Proposition 3.2. *Let $S \in \mathbb{K}\langle\langle \mathbb{DAG}(\widetilde{\Sigma}, C)\rangle\rangle$ and $\pi : \mathbb{DAG}(\widetilde{\Sigma}, C) \to \mathbb{DAG}(\widetilde{\Gamma}, C)$ be a projection. If S is recognizable, then $\pi(S) \in \mathbb{K}\langle\langle \mathbb{DAG}(\widetilde{\Gamma}, C)\rangle\rangle$ is recognizable.*

Proposition 3.3. *Let $L \subseteq \mathbb{DAG}(\widetilde{\Sigma}, C)$ be an unambiguously recognizable language. Then, the characteristic series $\mathbb{1}_L$ over $\mathbb{K}$ is recognizable.*

4 Weighted Monadic Second-Order Logic

We fix sets $Var = \{x, y, \ldots\}$ of *first-order* and $VAR = \{X, Y, \ldots\}$ of *second-order variables*. Still, we assume the semiring $\mathbb{K}$ being commutative.

Definition 4.1. *The set* wMSO$(\mathbb{K}, (\widetilde{\Sigma}, C))$ *of* weighted monadic second-order (wMSO) *formulas over $\mathbb{K}$ and $(\widetilde{\Sigma}, C)$ is given by (let $k \in K$, $a \in \Sigma$, and $\ell \in C$):*

$$\begin{aligned}\varphi ::= & k \mid \lambda(x) = a \mid \neg(\lambda(x) = a) \mid x \lhd_\ell y \mid \neg(x \lhd_\ell y) \mid x = y \mid \neg(x = y) \mid \\ & x \in X \mid \neg(x \in X) \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \wedge \varphi_2 \mid \exists x.\varphi \mid \exists X.\varphi \mid \forall x.\varphi \mid \forall X.\varphi\end{aligned}$$

The formulas k, $\lambda(x) = a$, $x \lhd_\ell y$, $x = y$ and $x \in X$ are called *atomic*. Negation of k has no reasonable semantics for general semirings. Thus, to obtain an intuitive interpretation of negation in terms of $\mathbb{0}$ and $\mathbb{1}$, it is pushed to the atomic level, omitting k. In exchange, we have to enrich the syntax by conjunction and universal quantification, cf. [7]. Let $Free(\varphi)$ be the set of free variables of φ and $\mathcal{V} \in 2^{Var \cup VAR}$ a finite set of variables. We say that $\mathcal{D} = (V, \{\lhd_\ell\}_{\ell \in C}, (\lambda, \rho))$ with $\rho : V \to \{0,1\}^{\mathcal{V}}$ is *valid* if, for any first-order variable $x \in \mathcal{V}$, there is a unique node $u \in V$ such that $\rho(u)[x] = 1$. In that case, $\rho(x)$ shall refer to u. Given $x \in \mathcal{V}$ and $u \in V$, we define the *update* $\rho[x/u] = \rho' : V \to \{0,1\}^{\mathcal{V}}$ such that $\rho'(u)[x] = 1$, $\rho'(v)[x] = 0$ for any $v \in V \setminus \{u\}$, and $\rho'(v)[\boldsymbol{x}] = \rho(v)[\boldsymbol{x}]$ for any $v \in V$ and $\boldsymbol{x} \in \mathcal{V} \setminus \{x\}$. Similarly, $\rho[X/V']$ is defined for $X \in \mathcal{V}$ and $V' \subseteq V$.

Note that, given a set $\mathcal{V}$ of variables, (weighted) ACATs can be extended to run on dags over $(\Sigma \times \{0,1\}^{\mathcal{V}}, C)$: We define a distributed alphabet $\widetilde{\Sigma}^{\mathcal{V}} = (\widetilde{\Sigma}_i^{\mathcal{V}})_{i \in Ag}$ by $\widetilde{\Sigma}_i^{\mathcal{V}} = \Sigma_i \times \{0,1\}^{\mathcal{V}}$.

Definition 4.2. *Suppose $\varphi \in$ wMSO$(\mathbb{K}, (\widetilde{\Sigma}, C))$ and suppose $\mathcal{V} \in 2^{Var \cup VAR}$ is finite with $Free(\varphi) \subseteq \mathcal{V}$. The semantics of φ wrt. $\mathcal{V}$ is a series $[\![\varphi]\!]_{\mathcal{V}} \in \mathbb{K}\langle\langle \mathbb{DAG}(\widetilde{\Sigma}^{\mathcal{V}}, C)\rangle\rangle$, given as follows: if $\mathcal{D} = (V, \{\lhd_\ell\}_{\ell \in C}, (\lambda, \rho)) \in \mathbb{DAG}(\widetilde{\Sigma}^{\mathcal{V}}, C)$ is not valid, we set $[\![\varphi]\!]_{\mathcal{V}}(\mathcal{D}) = \mathbb{0}$. Otherwise, $[\![\varphi]\!]_{\mathcal{V}}(\mathcal{D})$ is determined inductively as shown in Table 1.*

$$\llbracket k \rrbracket_{\mathcal{V}}(\mathcal{D}) = k$$

$$\llbracket \lambda(x) = a \rrbracket_{\mathcal{V}}(\mathcal{D}) = \begin{cases} \mathbb{1} & \text{if } \lambda(\rho(x)) = a \\ \mathbb{0} & \text{otherwise} \end{cases}$$

$$\llbracket x \lhd_\ell y \rrbracket_{\mathcal{V}}(\mathcal{D}) = \begin{cases} \mathbb{1} & \text{if } \rho(x) \lhd_\ell \rho(y) \\ \mathbb{0} & \text{otherwise} \end{cases}$$

$$\llbracket x = y \rrbracket_{\mathcal{V}}(\mathcal{D}) = \begin{cases} \mathbb{1} & \text{if } \rho(x) = \rho(y) \\ \mathbb{0} & \text{otherwise} \end{cases}$$

$$\llbracket x \in X \rrbracket_{\mathcal{V}}(\mathcal{D}) = \begin{cases} \mathbb{1} & \text{if } \rho(x) \in \rho(X) \\ \mathbb{0} & \text{otherwise} \end{cases}$$

$$\llbracket \neg\varphi \rrbracket_{\mathcal{V}}(\mathcal{D}) = \begin{cases} \mathbb{1} & \text{if } \llbracket \varphi \rrbracket_{\mathcal{V}}(\mathcal{D}) = \mathbb{0} \\ \mathbb{0} & \text{if } \llbracket \varphi \rrbracket_{\mathcal{V}}(\mathcal{D}) = \mathbb{1} \end{cases}$$

$$\llbracket \varphi_1 \vee \varphi_2 \rrbracket_{\mathcal{V}}(\mathcal{D}) = \llbracket \varphi_1 \rrbracket_{\mathcal{V}}(\mathcal{D}) \oplus \llbracket \varphi_2 \rrbracket_{\mathcal{V}}(\mathcal{D})$$

$$\llbracket \varphi_1 \wedge \varphi_2 \rrbracket_{\mathcal{V}}(\mathcal{D}) = \llbracket \varphi_1 \rrbracket_{\mathcal{V}}(\mathcal{D}) \circ \llbracket \varphi_2 \rrbracket_{\mathcal{V}}(\mathcal{D})$$

$$\llbracket \exists x.\varphi \rrbracket_{\mathcal{V}}(\mathcal{D}) = \bigoplus_{u \in V} \llbracket \varphi \rrbracket_{\mathcal{V}}(\mathcal{D}[x/u]])$$

$$\llbracket \forall x.\varphi \rrbracket_{\mathcal{V}}(\mathcal{D}) = \prod_{u \in V} \llbracket \varphi \rrbracket_{\mathcal{V}}(\mathcal{D}[x/u])$$

$$\llbracket \exists X.\varphi \rrbracket_{\mathcal{V}}(\mathcal{D}) = \bigoplus_{V' \subseteq V} \llbracket \varphi \rrbracket_{\mathcal{V}}(\mathcal{D}[X/V'])$$

$$\llbracket \forall X.\varphi \rrbracket_{\mathcal{V}}(\mathcal{D}) = \prod_{V' \subseteq V} \llbracket \varphi \rrbracket_{\mathcal{V}}(\mathcal{D}[X/V'])$$

Table 1. The semantics of wMSO-formulas

We abbreviate $\llbracket \varphi \rrbracket_{Free(\varphi)}$ by $\llbracket \varphi \rrbracket$. For $\mathbb{K}$ being the 2-valued Boolean algebra $\mathbb{B} = \{\mathbb{0}, \mathbb{1}\}$, wMSO$(\mathbb{B}, (\widetilde{\Sigma}, C))$ reduces to the usual MSO logic. Accordingly, $L \subseteq \mathbb{DAG}(\widetilde{\Sigma}, C)$ is *FO-definable* if its support is definable in FO$(\mathbb{B}, (\widetilde{\Sigma}, C))$, i.e., in the fragment of wMSO$(\mathbb{B}, (\widetilde{\Sigma}, C))$ in which no second-order quantifier occurs. We say that the series $S \in \mathbb{K}\langle\langle \mathbb{DAG}(\widetilde{\Sigma}, C) \rangle\rangle$ is an *FO-definable step function* if $S = \bigoplus_{i=1}^{n} k_i \circ \mathbb{1}_{L_i}$ for some $n \in \mathbb{N}$, $k_i \in K$, and FO-definable languages L_i. We call $\varphi \in$ wMSO$(\mathbb{K}, (\widetilde{\Sigma}, C))$ *restricted* if it contains no universal second-order quantification and, for any subformula $\forall x.\psi$ of φ, $\llbracket \psi \rrbracket$ is an FO-definable step function. We denote the set of restricted wMSO-formulas over $\mathbb{K}$ and $(\widetilde{\Sigma}, C)$ by wRMSO$(\mathbb{K}, (\widetilde{\Sigma}, C))$. Finally, let wREMSO$(\mathbb{K}, (\widetilde{\Sigma}, C))$ be the *existential* fragment of wRMSO$(\mathbb{K}, (\widetilde{\Sigma}, C))$, which contains the formulas of the form $\exists X_1 \ldots \exists X_n.\varphi$ where the kernel formula $\varphi \in$ wRMSO$(\mathbb{K}, (\widetilde{\Sigma}, C))$ contains no second-order quantifier.[5]

Even for words, wMSO has to be restricted because, otherwise, definability exceeds recognizability. While, in their logic, Droste and Gastin [7] deal with *recognizable* step functions exploiting the notion of determinism for finite automata, we have to cope with *FO-definable* functions in the context of dags. Fortunately, we can show unambiguity of $\mathbb{1}_L$ for FO-definable L, which is a cornerstone in establishing a logical characterization of wACATs.

Theorem 4.1. *Any FO-definable set of* $(\widetilde{\Sigma}, C)$*-dags is unambiguously recognizable.*

Proof (Sketch). It is well-known that any first-order formula can be written as the Boolean combination of statements "the pattern P occurs at least n times". Here, P is meant to be the (isomorphism type of the) environment of a node bounded by some radius $R \in \mathbb{N}$, also called an R-sphere. In [2], an ACAT $\mathcal{A}_R$ over dags detects the R-environment of any node. To transform the formula into an equivalent ACAT, we need

[5] It is not trivial to rewrite every wRMSO-formula into a wREMSO-formula. The problem is that it is not clear if for an FO-definable language L (as used in an FO-definable step function) the characteristic series $\mathbb{1}_L$ is again wFO-definable (see also the discussion in Section 6).

to equip $\mathcal{A}_R$ with a (deterministic) threshold counting procedure to count how often a sphere is used in a run. However, $\mathcal{A}_R$ from [2] is not unambiguous due to some coloring of spheres that is not unique. Such a coloring can be performed unambiguously so that any first-order formula can be simulated by an unambiguous ACAT. □

Corollary 4.1. *$\{\mathcal{D} \in \mathbb{DAG}(\widetilde{\Sigma}^{\mathcal{V}}, C) \mid \mathcal{D}$ valid$\}$ is unambiguously recognizable.*

Proof. It suffices to show FO-definability. In fact, it is easy to provide an FO formula requiring that, for any first-order variable x, there is exactly one node whose labeling is 1 in the component that corresponds to x. □

Example 4.1. Consider the ring $\mathbb{Z} = (\mathbb{Z}, +, \cdot, 0, 1)$ and the class of lossy message sequence charts with $Ag = \{1, 2\}$, cf. Example 2.2. Then the formula

$$\varphi = (\exists x.\lambda(x) = 1!2) \vee (\exists y. -1 \wedge \lambda(y) = 2?1)$$

defines a series $[\![\varphi]\!]$ which maps every lossy MSC $\mathcal{M}$ to the number of messages from process 1 to 2 that are lost.

The remainder of this paper is dedicated to the proof of our main theorem:

Theorem 4.2. *Let $\mathbb{K}$ be a commutative semiring and $S \in \mathbb{K}\langle\langle \mathbb{DAG}(\widetilde{\Sigma}, C)\rangle\rangle$. Then, the following are equivalent:*

1. *S is recognizable,*
2. *S is wRMSO-definable, and*
3. *S is wREMSO-definable.*

5 Definable Series are Recognizable

In this section we show that series defined by restricted formulas are recognizable. Due to Corollary 4.1 and Propositions 3.1 and 3.3, we can restrict to valid $(\widetilde{\Sigma}, C)$-dags.

By the closure properties of wACATs as stated in Propositions 3.1 and 3.2, we get:

Proposition 5.1. *Let $\varphi, \psi \in \mathrm{wMSO}(\mathbb{K}, (\widetilde{\Sigma}, C))$.*

(a) If φ is atomic or the negation of an atomic formula, then $[\![\,\varphi\,]\!]$ is recognizable.
(b) If $[\![\,\varphi\,]\!]$ and $[\![\,\psi\,]\!]$ are recognizable, then $[\![\,\varphi \vee \psi\,]\!]$ and $[\![\,\varphi \wedge \psi\,]\!]$ are recognizable.
(c) If $[\![\,\varphi\,]\!]$ is recognizable, then $[\![\,\exists x.\varphi\,]\!]$ and $[\![\,\exists X.\varphi\,]\!]$ are recognizable.

Proposition 5.2. *Let $\varphi \in \mathrm{wMSO}(\mathbb{K}, (\widetilde{\Sigma}, C))$ with $[\![\,\varphi\,]\!] = \sum_{i=1}^{n} k_i \circ \mathbb{1}_{L_i}$ an FO-definable step function. Then, $[\![\,\forall x.\varphi\,]\!]$ is recognizable.*

Proof (Sketch). Let $\mathcal{W} = \mathrm{free}(\varphi)$ and $\mathcal{V} = \mathrm{free}(\forall x.\varphi) = \mathcal{W} \setminus \{x\}$. Furthermore, $[\![\,\varphi\,]\!] = \sum_{i=1}^{n} k_i \mathbb{1}_{L_i}$ with $k_i \in \mathbb{K}$ and $L_i \subseteq \mathbb{DAG}(\widetilde{\Sigma}^{\mathcal{W}}, C)$ FO-definable languages for $i = 1, \ldots, n$. By Theorem 4.1, every L_i is recognized by an unambiguous ACAT. FO-definable languages are closed under union, complement and intersection. Therefore, $\{L_i \mid i = 1, \ldots, n\}$ can be assumed being a partition of $\mathbb{DAG}(\widetilde{\Sigma}^{\mathcal{W}}, C)$ with $k_i \neq k_j$

for $i \neq j$. First, let $x \in \mathcal{W}$. We put $\Gamma = \Sigma \times \{1, \ldots, n\}$ and consider $(\widetilde{\Gamma}^{\mathcal{V}}, C)$-dags $\widetilde{\mathcal{D}} = (V, \{\lhd_l\}_{l \in C}, (\lambda, \sigma, \rho))$ with $\lambda : V \to \Sigma$, $\sigma : V \to \{1, \ldots, n\}$, and $\rho : V \to \{0,1\}^{\mathcal{V}}$. Let $\widetilde{L} \subseteq \mathbb{DAG}(\widetilde{\Gamma}^{\mathcal{V}}, C)$ such that, for any $\widetilde{\mathcal{D}} \in \widetilde{L}$, any $v \in V$, and any $i \in \{1, \ldots, n\}$, we have $(\sigma(v) = i) \iff (V, \{\lhd_l\}_{l \in C}, (\lambda, \rho[x/v])) \in L_i$. Note that $\rho[x/v] : V \to \{0,1\}^{\mathcal{W}}$.

Since the L_i are FO-definable, one can build an FO-formula $\widetilde{\varphi}$ defining $\widetilde{L}$ (we omit the details). As $\widetilde{\varphi}$ is an FO-formula, the language $\widetilde{L} = L(\widetilde{\varphi})$ is recognizable by an unambiguous ACAT $\widetilde{\mathcal{A}} = (Q, \Delta, T, F)$ by Theorem 4.1. Let $t = (\overline{q}, (a, \sigma_t, \rho_t), q) \in \mathit{Trans}_{(\widetilde{\Gamma}^{\mathcal{V}}, C)}(Q)$ with $a \in \Sigma$, $\sigma_t \in \{1, \ldots, n\}$, and $\rho_t \in \{0,1\}^{\mathcal{V}}$. We transform $\widetilde{\mathcal{A}} = (Q, \Delta, T, F)$ into a wACAT $\mathcal{A} = (Q, \mu, T, \gamma)$ by adding weights as follows: set $\mu(t)$ to be k_i if $t \in \Delta$ and $\sigma_t = i$. Otherwise, $\mu(t) = \mathbb{0}$. Moreover, $\gamma(q_1, \ldots, q_{|Ag|}) = \mathbb{1}$ if $(q_1, \ldots, q_{|Ag|}) \in F$, and $\gamma(q_1, \ldots, q_{|Ag|}) = \mathbb{0}$ otherwise. Since $\widetilde{\mathcal{A}}$ is unambiguous and recognizes $\widetilde{L}$, the weight of an extended dag $\widetilde{\mathcal{D}} \in \widetilde{L}$ in $\mathcal{A}$ is $\prod_{1 \leq i \leq n} k_i^{|\sigma^{-1}(i)|}$ and for $\widetilde{\mathcal{D}} \notin \widetilde{L}$ we have $(\| \mathcal{A} \|, \widetilde{\mathcal{D}}) = \mathbb{0}$. Now we consider the projection $h : \mathbb{DAG}(\widetilde{\Gamma}^{\mathcal{V}}, C) \to \mathbb{DAG}(\widetilde{\Sigma}^{\mathcal{V}}, C)$ mapping $\widetilde{\mathcal{D}} = (V, \{\lhd_l\}_{l \in C}, (\lambda, \sigma, \rho))$ to $\mathcal{D} = (V, \{\lhd_l\}_{l \in C}, (\lambda, \rho))$. Note that for $\mathcal{D} \in \mathbb{DAG}(\widetilde{\Sigma}^{\mathcal{V}}, C)$ there is a unique $\widetilde{\mathcal{D}} \in \widetilde{L}$ with $h(\widetilde{\mathcal{D}}) = \mathcal{D}$. Hence, we have

$$\begin{aligned}
(h(\| \mathcal{A} \|), \mathcal{D}) &= \bigoplus_{\widetilde{\mathcal{D}} \in h^{-1}(\mathcal{D}) \cap \widetilde{L}} (\| \mathcal{A} \|, \widetilde{\mathcal{D}}) = (\| \mathcal{A} \|, \widetilde{\mathcal{D}}) = \prod_{1 \leq i \leq n} k_i^{|\sigma^{-1}(i)|} \\
&= \prod_{v \in V} (\llbracket \varphi \rrbracket, (\mathcal{D}, \rho[x/v])) = (\llbracket \forall x.\varphi \rrbracket, \mathcal{D}) .
\end{aligned}$$

By Proposition 3.2, $\llbracket \forall x.\varphi \rrbracket$ is recognizable. The case $x \notin \mathcal{W}$ is derived easily. □

By Propositions 5.1 and 5.2, we have immediately:

Theorem 5.1. *Let $\mathbb{K}$ be a commutative semiring and let $S \in \mathbb{K}\langle\langle \mathbb{DAG}(\widetilde{\Sigma}, C) \rangle\rangle$. If S is wRMSO-definable, then S is also recognizable.*

6 Recognizable Series are Definable

For $S \in \mathbb{K}\langle\langle \mathbb{DAG}(\widetilde{\Sigma}, C) \rangle\rangle$, let $\mathrm{Supp}(S) = \{\mathcal{D} \in \mathbb{DAG}(\widetilde{\Sigma}, C) \mid (S, \mathcal{D}) \neq \mathbb{0}\}$. We adopt the notion of an *unambiguous* $\mathrm{FO}(\mathbb{K}, (\widetilde{\Sigma}, C))$-formula [7]:

- All atomic formulas apart from k and their negations are unambiguous.
- If φ and ψ are unambiguous, then so are $\varphi \wedge \psi$, $\forall x.\varphi$, and $\forall X.\varphi$.
- If φ and ψ are unambiguous and $\mathrm{Supp}(\llbracket \varphi \rrbracket) \cap \mathrm{Supp}(\llbracket \psi \rrbracket) = \emptyset$, then $\varphi \vee \psi$ is unambiguous.
- If, finally, φ is unambiguous and, for any $(\mathcal{D}, \rho)$ with $\rho : V \to \{0,1\}^{\mathit{Free}(\varphi)}$, there is at most one vertex u of $\mathcal{D}$ such that $\llbracket \varphi \rrbracket_{\mathit{Free}(\varphi) \cup \{x\}}(\mathcal{D}, \rho[x/u]) \neq \mathbb{0}$, then $\exists x.\varphi$ is unambiguous.

Observe that, though, syntactically, we deal with ordinary MSO formulas, an unambiguous formula is primarily a weighted formula, as unambiguousness is defined in terms of its series. Let $\varphi \in \mathrm{FO}(\mathbb{K}, (\widetilde{\Sigma}, C))$. If φ is unambiguous, then $\llbracket \varphi \rrbracket$ is an FO-definable step function. We know from [7] that certain simple formulas can be made unambiguous:

Proposition 6.1 ([7]). *Let* $\varphi \in \mathrm{FO}(\mathbb{K}, (\widetilde{\Sigma}, C))$ *be a (positive) Boolean combination of atomic formulas apart from* k *and their negations. Then, there is an unambiguous formula* $\varphi^+ \in \mathrm{FO}(\mathbb{K}, (\widetilde{\Sigma}, C))$ *such that* $[\![\varphi^+]\!] = [\![\varphi]\!]$.

Proof. We proceed by induction and simultaneously define formulas φ^+ and φ^-. If $\varphi \in \mathrm{FO}(\mathbb{K}, (\Sigma, C))$ is atomic or the negation of an atomic formula, we set $\varphi^+ = \varphi$ and $\varphi^- = \neg\varphi$ (where $\neg\neg\psi$ is reduced to ψ). Moreover, we let

- $(\varphi \vee \psi)^+ = \varphi^+ \vee (\varphi^- \wedge \psi^+)$,
- $(\varphi \vee \psi)^- = \varphi^- \wedge \psi^-$,
- $(\varphi \wedge \psi)^- = \varphi^- \vee (\varphi^+ \wedge \psi^-)$, and
- $(\varphi \wedge \psi)^+ = \varphi^+ \wedge \psi^+$. $\square$

In the context of words and an (E)MSO logic that employs the predicate $\leq$ instead of the direct successor relation, Droste and Gastin need to transform an ordinary MSO formula φ into an unambiguous weighted MSO formula φ' such that $[\![\varphi']\!]$ is the characteristic series of the language of φ [7]. To this aim, they identify the unique least position of a word (wrt. $\leq$) that satisfies a given property. In our logic, such an identification is no longer feasible. Nevertheless, we can transform any wACAT into an equivalent weighted formula.

Theorem 6.1. *Let* $\mathcal{A}$ *be a wACAT over commutative* $\mathbb{K}$ *and* $(\widetilde{\Sigma}, C)$*. There is a sentence* ψ *from* $\mathrm{wREMSO}(\mathbb{K}, (\widetilde{\Sigma}, C))$ *such that* $[\![\psi]\!] = \|\mathcal{A}\|$.

Proof. Let $\mathcal{A} = (Q, \mu, T, \gamma)$ be a wACAT over $\mathbb{K}$ and $(\widetilde{\Sigma}, C)$. In the following, t and t' will range over $\mathit{Trans}_{(\widetilde{\Sigma},C)}(Q)$. Set $\overline{X}$ to be a collection (X_t) of second-order variables and suppose $Ag = \{1, \ldots, N\}$ for some $N \in \mathbb{N}$. The construction of a wREMSO sentence from $\mathcal{A}$ follows the route of transforming a finite automaton into a formula where an interpretation of second-order variables reflects an assignment of vertices to transitions. We first provide some building blocks of the desired wREMSO formula.

The unambiguous formula

$$\mathrm{Partition}(\overline{X}) := \forall x. \bigvee_t (x \in X_t \wedge \bigwedge_{t' \neq t} \neg(x \in X_{t'}))$$

claims that $\overline{X}$ actually represents a run, i.e., an assignment of vertices to transitions.

Given $a \in \Sigma$ and $q \in Q$, $\varphi^+_{(a,q)}(x)$ ($\varphi^-_{(a,q)}(x)$) shall denote the disjunction (conjunction) of formulas $x \in X_t$ ($\neg(x \in X_t)$) such that $t = (\overline{q}, a, q)$ for some $\overline{q}$, respectively.

Now let $t = \{((a_1, q_1), \ell_1), \ldots, ((a_m, q_m), \ell_m)\} \longrightarrow (a, q)$. To ensure that x is contained in X_t only if the transition taken at x corresponds to t, we use

$$\begin{aligned}\mathrm{Trans}_t(x, \overline{X}) := \quad & x \in X_t \wedge \lambda(x) = a \wedge \bigwedge_{k \in \{1,\ldots,m\}} \exists y. \Big[y \lhd_{\ell_k} x \wedge \varphi^+_{(a_k,q_k)}(y) \Big]^+ \\ & \wedge \forall y. \Big[\bigwedge_{\ell \in C} \neg(y \lhd_\ell x) \vee \bigvee_{k \in \{1,\ldots,m\}} \Big(y \lhd_{\ell_k} x \wedge \lambda(y) = a_k \Big) \Big]^+ .\end{aligned}$$

Another difficulty is to determine the weight of a global final state $\overline{q}$ with respect to an extended dag. We would like to identify, for any agent $i \in Ag$ with $\overline{q}[i] \neq \imath$, the Σ_i-maximal node. For this purpose, we demand the unique upwards-closed set of nodes Y that contains a single minimal element x such that x is the only node executed by agent i. Then, x is Σ_i-maximal and shall be contained in X_t for some transition $t = \overline{q} \longrightarrow (a, \overline{q}[i])$. Therefore, we define $\max_i(x, Y)$

$$
\begin{aligned}
\max_i(x,Y) := \;& \Big[\bigvee_{a\in\Sigma_i} \lambda(x) = a\Big]^+ \wedge x \in Y \\
& \wedge \forall y.\forall z.\Big[\neg(y \in Y) \vee \neg(y \lhd z) \vee z \in Y\Big]^+ \\
& \wedge \forall y.\Big[\neg(y \in Y) \vee \bigwedge_{a\in\Sigma_i} \neg(\lambda(y) = a) \vee y = x\Big]^+ \\
& \wedge \forall y.(\neg\varphi_1 \vee (\varphi_1 \wedge \varphi_2) \vee (\varphi_1 \wedge \neg\varphi_2 \wedge \varphi_3))
\end{aligned}
$$

where $\varphi_1 = (y \in Y)$, $\varphi_2 = (y = x)$, and $\varphi_3 = \exists z.(z \in Y \wedge [\bigvee_{(a,\ell)\in\Sigma\times C} (z \lhd_\ell y \wedge \lambda(z) = a)]^+)$. Hereby, the last conjunct ensures unambiguity of $\max_i(x, Y)$ by requiring that x is the *only* minimal event in Y.

To collect the weights of global final states, we require, for any $\overline{q} \in (Q \cup \{\imath\})^{Ag}$, a formula $\mathrm{Final}_{\overline{q}}(\overline{X}, Y_1, \ldots, Y_N) :=$

$$
\bigwedge_{\substack{i\in Ag\\ \overline{q}[i]\in Q}} \exists x.\Big[\max_i(x, Y_i) \wedge \Big(\bigvee_{a\in\Sigma} \varphi^+_{(a,\overline{q}[i])}(x)\Big)^+\Big] \wedge \bigwedge_{\substack{i\in Ag\\ \overline{q}[i]=\imath}} \forall x. \bigwedge_{a\in\Sigma_i} \neg(\lambda(x) = a)\,.
$$

To simulate the type function T, we make use of the unambiguous formula $\mathrm{Type}(\overline{X}) :=$

$$
\forall x. \bigwedge_{(a,q)\in\Sigma\times Q} \Big[\varphi^-_{(a,q)}(x) \vee \Big([\varphi^+_{(a,q)}(x)]^+ \wedge \bigwedge_{(b,\ell)\in T(a,q)} \exists y.(x \lhd_\ell y \wedge \lambda(y) = b)\Big)\Big]^+ .
$$

We are now prepared to specify the desired formula ψ. Namely, setting

$$
\begin{aligned}
\psi'(\overline{X}) =& \exists Y_1 \ldots \exists Y_N. \\
& \quad \mathrm{Partition}(\overline{X}) \wedge \bigwedge_t \forall x.(\neg(x \in X_t) \vee \mathrm{Trans}_t(x, \overline{X})) \\
& \wedge \bigwedge_{i\in Ag} \Big(\exists x.\max_i(x, Y_i)\Big) \vee \forall x.\Big(\neg(x \in Y_i) \wedge \bigwedge_{a\in\Sigma_i} \neg(\lambda(x) = a)\Big) \\
& \wedge \bigwedge_t \forall x.(\neg(x \in X_t) \vee ((x \in X_t) \wedge \mu(t))) \\
& \wedge \mathrm{Type}(\overline{X}) \wedge \bigvee_{\overline{q}\in F} \Big(\mathrm{Final}_{\overline{q}}(\overline{X}, Y_1, \ldots, Y_N) \wedge \gamma(\overline{q})\Big)\,,
\end{aligned}
$$

we finally let $\psi = \exists\overline{X}.\psi' \in \mathrm{wREMSO}(\mathbb{K}, (\widetilde{\Sigma}, C))$. Observe that the subformula $\neg(x \in X_t) \vee ((x \in X_t) \wedge \mu(t))$ of ψ is an FO-definable step function.

In fact, for any $\mathcal{D} = (V, \{\lhd_\ell\}_{\ell\in C}, \lambda) \in \mathbb{DAG}(\widetilde{\Sigma}, C)$, we have $[\![\psi]\!](\mathcal{D}) = (\|\mathcal{A}\|, \mathcal{D})$. Thus, we obtain $[\![\psi]\!] = \|\mathcal{A}\|$. □

References

1. C. Baier and M. Größer. Recognizing omega-regular languages with probabilistic automata. In *Proceedings of LICS 2005*. IEEE Computer Society Press, 2005.
2. B. Bollig. On the expressiveness of asynchronous cellular automata. In *Proceedings of FCT 2005*, volume 3623 of *Lecture Notes in Comp. Sc.*, pages 528–539. Springer, 2005.
3. J. Büchi. Weak second order arithmetic and finite automata. *Z. Math. Logik, Grundlag. Math.*, 5:66–62, 1960.
4. K. Culik and J. Kari. Image compression using weighted finite automata. *Computer and Graphics*, 17(3):305–313, 1993.
5. V. Diekert and G. Rozenberg, editors. *The Book of Traces*. World Scientific, Singapore, 1995.
6. M. Droste and P. Gastin. The Kleene-Schützenberger theorem for formal power series in partially commuting variables. *Inform. and Comp.*, 153:47–80, 1999.
7. M. Droste and P. Gastin. Weighted automata and weighted logics. In *Proceedings of ICALP 2005*, volume 3580 of *Lecture Notes in Comp. Sc.*, pages 513–525. Springer, 2005.
8. M. Droste, P. Gastin, and D. Kuske. Asynchronous cellular automata for pomsets. *Theoret. Comp. Sc.*, 247(1-2):1–38, 2000.
9. M. Droste and G. Rahonis. Weighted automata and weighted logics on infinite words. In *10th Int. Conf. on Developments in Language Theory (DLT)*, volume 4036 of *Lecture Notes in Comp. Sc.*, pages 49–58. Springer, 2006.
10. M. Droste and H. Vogler. Weighted tree automata and weighted logics. *Theoret. Comp. Sc.*, 366:228–247, 2006.
11. C. C. Elgot. Decision problems of finite automata design and related arithmetics. *Trans. Amer. Math. Soc.*, 98:21–52, 1961.
12. S. Jesi, G. Pighizzini, and N. Sabadini. Probabilistic asynchronous automata. *Mathematical Systems Theory*, 29(1):5–31, 1996.
13. W. Kuich. Semirings and Formal Power Series. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*, volume 1, chapter 9, pages 609–677. Springer, 1997.
14. D. Kuske. Emptiness is decidable for asynchronous cellular machines. In *Proceedings of CONCUR 2000*, volume 1877 of *Lecture Notes in Comp. Sc.*, pages 536–551. Springer, 2000.
15. D. Kuske. Weighted asynchronous cellular automata. In *Proceedings of STACS 2006*, volume 3884 of *Lecture Notes in Comp. Sc.*, pages 685–696. Springer, 2006.
16. I. Mäurer. Weighted picture automata and weighted logics. In *Proceedings of STACS 2006*, volume 3884 of *Lecture Notes in Comp. Sc.*, pages 313–324. Springer, 2006.
17. I. Meinecke. Weighted logics for traces. In *Proceedings of CSR 2006*, volume 3967 of *Lecture Notes in Comp. Sc.*, pages 235–246. Springer, 2006.
18. M. Mohri. Finite-state transducers in language and speech processing. *Computational Linguistics*, 23(2):269–311, 1997.
19. E. Ochmański. Regular behaviour of concurrent systems. *Bulletin of the EATCS*, 27:56–67, 1985.
20. Ph. Schnoebelen. The verification of probabilistic lossy channel systems. In *Valid. of Stochastic Systems*, volume 2925 of *Lecture Notes in Comp. Sc.*, pages 445–465. Springer, 2004.
21. M.P. Schützenberger. On the definition of a family of automata. *Information and Control*, 4:245–270, 1961.
22. W. Zielonka. Notes on finite asynchronous automata. *R.A.I.R.O. — Informatique Théorique et Applications*, 21, 1987.