Journal of Philosophical Logic (2019) 48:1077–1118
https://doi.org/10.1007/s10992-019-09508-w

Intensional Protocols for Dynamic Epistemic Logic

Hanna S. van Lee[1] · Rasmus K. Rendsvig[1,2] · Suzanne van Wijk[3]

Received: 7 November 2017 / Accepted: 21 March 2019 / Published online: 28 May 2019

Abstract
In dynamical multi-agent systems, agents are controlled by protocols. In choosing a class of formal protocols, an implicit choice is made concerning the types of agents, actions and dynamics representable. This paper investigates one such choice: An intensional protocol class for agent control in dynamic epistemic logic (DEL), called 'DEL dynamical systems'. After illustrating how such protocols may be used in formalizing and analyzing information dynamics, the types of epistemic temporal models that they may generate are characterized. This facilitates a formal comparison with the only other formal protocol framework in dynamic epistemic logic, namely the extensional 'DEL protocols'. The paper concludes with a conceptual comparison, highlighting modeling tasks where DEL dynamical systems are natural.

Keywords Dynamic epistemic logic · Multi-agent systems · Protocols · Epistemic temporal logic · Dynamical systems

1 Introduction

In logically modeling dynamics in multi-agent systems—whether by using global-perspective frameworks like interpreted systems [24] and epistemic temporal logic

✉ Rasmus K. Rendsvig
rasmus@hum.ku.dk

Hanna S. van Lee
hannavanlee@gmail.com

Suzanne van Wijk
scc.vanwijk@gmail.com

[1] Center for Information and Bubble Studies, University of Copenhagen, CIBS/KUA, Karen Blixens Plads 8, Copenhagen 2300S, Denmark

[2] Theoretical Philosophy, Lund University, Lund, Sweden

[3] Diemen, The Netherlands

[51], or by using local-perspective frameworks like dynamic epistemic logic [16]—the dynamics rely on *protocols*: control mechanisms that determine which actions may occur when.

Protocols take a plethora of forms, ranging from natural language descriptions, over pseudo-code renderings, to fully formalized representations. Moreover, protocol specifications may vary in their fundamental structure. Specifically, one may distinguish between *extensional* protocols and *intensional* protocols.[1]

Extensional protocols are *temporal*: They consult an *external clock* to specify which actions are available for execution at a given time of a run of a system. Roughly speaking, an extensional protocol is a set of sequences of actions that allows the execution of action a at time t if a is on the tth position of a sequence in the protocol. Abstractly, think of a function assigning to each natural number a set of allowed actions.

Intensional protocols, in contrast, are *conditional*: They consult the *current state of the system* to specify which actions are available for execution *now*. Roughly speaking, a conditional protocol is a set of "`if` φ`, do` a" statements. Such a statement—or rule—allows the execution of an action a *now* if the current state satisfies the test condition φ. Abstractly, think of a function assigning to each possible state of the system some set of allowed actions.

Both extensional and intensional protocols *qua* protocols have been investigated in the epistemic agency literature, but mainly in different paradigms: Where the interpreted systems literature has favored intensional protocols [24, 45, 62], the literature on protocols in dynamic epistemic logic has favored extensional protocols [15, 20, 37, 38, 58, 61].

There is, however, no formal reason to avoid intensional protocols in the dynamic epistemic logic setting. In fact, such protocols may be both intuitive and compact in representation. Moreover, the mathematical basis and logical theory for intensional protocols enjoys established results, albeit not cast as results concerning protocols (cf. Section 1.2 on related literature).

This paper concerns intensional protocols for dynamic epistemic logic (DEL). In particular, it investigates intensional protocols as represented by *multi-pointed action models* applied iteratively. By doing this, the paper takes a *discrete-time dynamical systems* perspective on protocols for information dynamics. The resulting intensional protocols are referred to as *DEL dynamical systems*.

The overarching question the paper asks is how such intensional protocols relate to their closest extensional relative, namely the DEL protocols of van Benthem, Gerbrandy, Hoshi and Pacuit, [15]. The main motivation for this question is a wish to clarify similarities and differences in the implicit assumptions and restrictions inherent in the two frameworks. This, in turn, is motivated by a desire to understand the up- and downsides of protocol frameworks from a design and modeling perspective.

Methodologically, the main comparison is achieved by characterizing the types of epistemic temporal logic (ETL) models generatable by intensional protocols coded as DEL dynamical systems, and compare the resulting ETL properties with those

[1]The terms and distinction is adopted from Parikh and Ramanujam, see [51, Sec. 2.2].

previously obtained for extensional DEL protocols by van Benthem, Gerbrandy, Hoshi and Pacuit.

This methodology has a two-fold incentive, the first being that ETL models provide an assumption-free common point of reference between the two protocol forms, thus allowing a comparison of induced properties.[2] This is desirable as the fundamental difference between extensional and intensional protocols—that one relies on an external clock whereas the other reacts to the current state—makes it difficult to compare the protocol frameworks directly. In particular, then, the structure of an extensional protocol is not, in general, enough to determine whether the resulting sequence of models may be obtained from an intensional protocol. The second aspect of the motivation is that as a by-product the methodology relates DEL dynamical systems to ETL models, thus yielding results illuminating the former, in which there has been recent interest, cf. Section 1.2 on related literature.

1.1 Structure of the Paper

Section 2 defines core DEL components as well as intensional protocols (“DEL dynamical systems”) and extensional protocols (“DEL protocols”). These are informally compared and contrasted by example. It is then illustrated how DEL dynamics may be seen as producing ETL models.

Section 4 presents ETL models and eight structural properties of key relevance to the paper.

Section 5 formally defines how to generate ETL models from DEL dynamical systems and contains a first result: For an ETL model to be generatable by a DEL dynamical system, it must satisfy specific seven of the eight structural properties, but not necessarily the eighth.

Section 6 concerns the other direction: Constructing DEL dynamical systems that will generate a given ETL model. It will be shown that if an ETL model possesses all eight structural properties, then this is sufficient for a suitable DEL dynamical system to exist.

Together, the results of Sections 5 and 6 *almost* provide a characterization of the ETL models generatable by DEL dynamical systems, but not quite.

Section 7 restricts attention to a subclass of DEL dynamical systems and a subclass of ETL models: When a DEL dynamical system is *image-finite* and *concluding*,it generates an *image-finite* and *concluding* ETL model. In this case, a proper characterization is obtained: The eight properties are both necessary and sufficient.

Section 8 moves the attention to *non-deterministic* intensional protocols, implemented by running several (deterministic) DEL dynamical systems in parallel. The motivation for this is a tighter correspondence with the methodology of extensional DEL protocols, of which only special cases are deterministic. The section presents weaker, necessary properties of ETL models generated by *families* of DEL dynamical systems.

[2]It is also a point of reference for other frameworks, like interpreted systems or extensive games with imperfect information, cf. the motivation in [15].

Section 9 contains the main comparison of intensional and extensional protocols for DEL, based on the differences in structural properties of generatable ETL models. The section thus compares and discusses the present results with those of van Benthem, Gerbrandy, Hoshi and Pacuit [15].

Section 10 concludes with open questions.

1.2 Related Literature

This paper is situated in the literature on epistemic logic in the tradition of Hintikka [36] with focus on the temporal development of knowledge in multi-agent systems. In the DEL approach to this topic, a temporal dynamics is a sequence of self-contained models m_1, m_2, m_3, ... where each model—except for the first—is obtained from the former by some transformation (see e.g. [6, 22, 48] for introductions). This perspective stands in contrast to models representing dynamics internally, from the outset offering a full, unfolded view of time. This is the contrast between the *local* and *global*—or *Grand Stage*—views on dynamics, in the terms of van Benthem [10]. Grand Stage models may typically be envisioned as temporal trees or forests with branches connected by agent-index relations. For illustrations, see the ETL models depicted in Sections 2 and 4.

Grand Stage models have been a go-to in the literature on distributed computing. ETL models are typically attributed to Parikh and Ramanujam [50], while an early source on the *interpreted systems* framework is Halpern and Moses' [32] where protocols and the existence of correct protocols constitute objects of study (see [24] for an introduction). Several of the properties of ETL models of interest in this paper (see Section 6) have previously been studied in Grand Stage models, including synchronous/asynchronous agency and perfect recall, in relation to axiomatizations and complexity [33–35, 44, 46].

In interpreted systems, intensional protocols are commonplace, exemplified by the pioneering *knowledge-based programs* of Fagin, Halpern, Moses and Vardi [24]. There, an agent's possibly non-deterministic behavior is specified by a set of instructions of the form "`if` $K_i\varphi$`, do` a". From an initial state of a system, a joint set of instructions for agents and the environment specifies the next state. Over time, the initial state is thus unfolded to produce a temporal model. As such, knowledge-based programs act as a local mechanism for generating Grand Stage models.

Automata theory provides an alternative approach to generating (and also classifying) Grand Stage models such as epistemic trees and forests. Taking branches as words, a class of forests may possibly correspond to the language accepted by a given class of finite-state machines. This approach is taken by Mohalik and Ramanujam [47] for ETL models (a related work for interpreted systems is [45]). Mohalik and Ramanujam study agents computing asynchronously, but with occasional synchronous full and perfect information exchange. Each agent is locally modeled by a finite transition system, from which a product automaton determines global behavior. The authors show a range of results concerning the language of the class of such product automata. Mohalik and Ramanujam remark on a connection to generating ETL models using DEL action models as undertaken in [15] (see below), but leave the relation an open question. Touching vaguely on such a connection is [54], where Rendsvig shows that Liu, Seligman, and Girard's social network belief dynamics induced by transition system agents [43] may be presented using DEL action models. A direct connection between automata theory, forest generation and DEL may be found in the work of Aucher, Maubert and Pinchinat [4]. The authors show that iterating a finite action model with Boolean pre- and postconditions on an initial finite Kripke model produces a regular structure, admitting representation by a finite-state synchronous transducer. Several other papers on DEL draw connections to automata theory without direct links to forest generation [3, 18, 39]. As the characterization results in Section 7 concern ETL models with finite or repetitive nature, these may possibly be generated by some finite-state mechanism.[3] To explore the relation between automata theory and the present results is future work (see Section 10).

More narrowly within the DEL literature, two strands of research are of particular relevance: work on protocols and DEL and work on DEL and dynamical systems. The first comprises [12, 20, 37, 38, 58, 61] and van Benthem, Gerbrandy, Hoshi and Pacuit's [15]. All papers in this collection use or study extensional protocols in the style of Parikh and Ramanujam [50, 51], with various aims. Of special interest to the current paper is [15]: In [15], the authors investigate which classes of ETL models one may generate using action models, product update and extensional protocols. Their results are illuminating in elucidating epistemic and logical properties inherent in the DEL methodology. The methodology and results of [15] are presented and discussed throughout.

The second strand of DEL research concerns the iterated application of DEL model transformers on sets of pointed Kripke models. In this strand, some works cast such iterations exactly *as* dynamical systems.[4] This idea was first explicitly put in play by Sadzik in 2006 [59]; he investigates frame conditions for action models that guarantee a stabilizing orbit modulo bisimulation, drawing on conceptual ideas from van Benthem, advanced in 2002 [9]. Since then, various works have honed in on long-term behavior of iterated model transformations—with e.g. [1, 8, 30] concerned with aspects of belief change, and [3, 4, 18, 19] describing epistemic planning—without explicit ties to dynamical systems. An interesting approach to *protocol learning* is taken by van Ditmarsch, Ghosh, Verbrugge and Wang in [23], using a variation of action models labeled with a protocol specification language. Closer to the present

[3]We thank an anonymous reviewer for pointing this out.

[4]For the interested reader, there also exists a body of literature taking the converse perspective, using logics to describe qualitative aspects of long-run behavior. On this approach, logic meets dynamical systems by the latter playing the role of semantics to the former. Papers falling in this category, detailing logics of *dynamical topological systems*, include Kremer and Mints' 2007 handbook chapter [42] (on research from 1997 onwards by e.g. Artemov [2] and the authors of [41]) and several recent papers by Fernandéz-Duque [25–27]; Sarenac's paper from 2011 [60] exploring modal logical approaches to describing *iterated function systems*; and finally van Benthem's work in [9, 11], outlining various possible logical approaches to fixed points and limit cycles of dynamical systems by applying fixed-point and oscillation operators galvanized by modal μ-calculus. The latter two papers additionally provide an excellent bridge between the high abstraction level approach to logic and dynamical systems of this note and the micro-perspective literature in the main text.

approach is [53, 55] where information dynamics are modeled using a DEL intensional protocol format akin to the (one-step) planning problems explored by Bolander and Birkegaard [18]. In [56], that same protocol format is cast in dynamical systems terms and is shown to be effectively equivalent with both a DEL-variant of knowledge-based programs and the multi-pointed action models of Baltag and Moss [5]. These results motivate the use of multi-pointed action models both here and in [39, 40]. In [39, 40], iterated dynamics of multi-pointed action models are construed in a topological setting, inducing maps that satisfy the common definition of a dynamical system: A compact, metric space under the action of a continuous function.[5]

In none of the above-mentioned papers is DEL investigated as a dynamical system *qua* its role as protocol defining, and neither have the resulting sequences been related to the Grand Stage ETL models they generate nor to extensional protocols.

2 Protocols for DEL

In this section, standard notions from dynamic epistemic logic are introduced together with intensional and extensional DEL protocols. The reader is referred to the excellent literature on the topic of epistemic logic and DEL for more information and philosophical interpretation: See e.g. [6, 7, 10, 13, 14, 16, 21, 22, 24, 36].

2.1 Pointed Kripke Models and Language

Let a countable, non-empty set of propositional atoms Φ and a finite, non-empty set of agents I be given. Throughout the paper, it will be assumed that these sets remain fixed.

A **Kripke model** is a tuple $M = (\llbracket M \rrbracket, R, \llbracket \cdot \rrbracket)$ where

$\llbracket M \rrbracket$ is a countable, non-empty set of **states**;

$R : I \longrightarrow \mathcal{P}(\llbracket M \rrbracket \times \llbracket M \rrbracket)$ assigns to each agent i an **accessibility relation** $R(i)$, also denoted R_i;

$\llbracket \cdot \rrbracket : \Phi \longrightarrow \mathcal{P}(\llbracket M \rrbracket)$ is a **valuation**, assigning to each atom an extension of states.

A pair (M, s) with $s \in \llbracket M \rrbracket$ is called a **pointed Kripke model** with s called the **designated state**. Throughout, the pair (M, s) is written Ms.

Where $p \in \Phi$ and $i \in I$, define a language $\mathcal{L}_{(\Phi,I)}$ by

$$\varphi := \top \mid p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \Box_i \varphi$$

with non-propositional formulas evaluated over pointed Kripke model Ms by

$$Ms \models \Box_i \varphi \text{ iff for all } t \in \llbracket M \rrbracket, sR_i t \text{ implies } Mt \models \varphi,$$

[5]This paper omits the topological augmentation as moving between concrete DEL models and the abstract quotient models of [39, 40] introduces additional steps in the arguments used to compare sequences of DEL models with ETL models.

and standard propositional semantics. For sets of formulas Δ, Δ', write $\Delta \vDash \Delta'$ if for all pointed Kripke models Ms, $Ms \vDash \varphi$ for all $\varphi \in \Delta$ implies $Ms \vDash \varphi'$ for all $\varphi' \in \Delta'$. Set brackets are omitted when no confusion should arise.

Using standard modal logical semantics makes *bisimulation* the natural notion of equality between pointed Kripke models (see e.g. [17]). With $M = (\llbracket M \rrbracket, R, \llbracket\cdot\rrbracket)$ and $M' = (\llbracket M' \rrbracket, R', \llbracket\cdot\rrbracket')$ two Kripke models, a binary relation $Z \subseteq \llbracket M \rrbracket \times \llbracket M' \rrbracket$ is a **bisimulation** if, for all $i \in I$:

1. If sZs', then satisfy the same atoms: for all $p \in \Phi$, $s \in \llbracket p \rrbracket$ iff $s' \in \llbracket p \rrbracket'$,
2. If sZs' and sR_it, then there exists a $t' \in \llbracket M' \rrbracket$ such that tZt' and $s'R_i't'$, and
3. If sZs' and $s'R_it'$, then there exists a $t \in \llbracket M \rrbracket$ such that tZt' and sR_it.

When Z is a bisimulation and sZs', the pointed Kripke models Ms and $M's'$ are **bisimilar**, denoted $Ms \leftrightarroweq M's'$.

2.2 Action Models and Product Update

In dynamic epistemic logic, dynamics are introduced by transitioning between pointed Kripke models from some set X using a possibly partial map $f : X \longrightarrow X$. Such a map is often referred to as a *model transformer*. Many model transformers have been suggested in the literature, the most well-known being *truthful public announcement*, $!\varphi$ [52]. Truthful public announcements are a special case of a rich class of model transformers, here referred to as the class of *clean maps*.

In essence, a clean map f is given by $f(x) = c(x \otimes a)$ with specific term a, product $\otimes$ and restricting operation c. The term a is based on a *deterministic multi-pointed action model*, defined below.[6] Intuitively, one may think of a clean map as a set of program lines, each of the form "`If` φ_i`,` `do` a_i", where the preconditions φ_i are mutually exclusive. When "run" on a pointed Kripke model x, the program checks if x satisfies any φ_i. If so, it executes action a_i (a sub-action of a) on x, obtaining the result $x \otimes a$. If not, the product of x and a is undefined. Finally, the operation c removes redundant states. Their usage is exemplified in Section 3.2.

Define an **action model** as a tuple $\Sigma = (\llbracket \Sigma \rrbracket, \mathsf{R}, pre, post)$, sharing language $\mathcal{L}_{(\Phi,I)}$ with models in X, where

$\llbracket \Sigma \rrbracket$ is a countable, non-empty set of **actions** σ;

$\mathsf{R} : I \longrightarrow \mathcal{P}(\llbracket \Sigma \rrbracket \times \llbracket \Sigma \rrbracket)$ assigns an **accessibility relation** $\mathsf{R}(i)$ to each index $i \in I$, with $\mathsf{R}(i)$ denoted R_i;

$pre : \llbracket \Sigma \rrbracket \longrightarrow \mathcal{L}_{(\Phi,I)}$ assigns to each action a **precondition**, specifying the conditions under which σ is executable;

$post : \llbracket \Sigma \rrbracket \longrightarrow \mathcal{L}_{(\Phi,I)}$ assigns to each action a **postcondition** (a conjunctive clause[7] over Φ, or $\top$). The postcondition specifies whether σ changes the values of select atoms.

[6] Action models and product update was introduced in [16]. The extension to multi-pointed action models came with [5]. The present version of postconditions is inspired by [21] and the usage of deterministic models by [56].

[7] I.e. a conjunction of literals, where a literal is an atomic proposition or a negated atomic proposition.

A pair (Σ, Γ) with $\emptyset \neq \Gamma \subseteq [\![\Sigma]\!]$ is a **multi-pointed action model** with Γ the set of **designated actions**; (Σ, Γ) is also written $\Sigma\Gamma$. If Γ is a singleton $\{\sigma\}$, then $\Sigma\Gamma$ is called **single-pointed** and is written $\Sigma\sigma$. If $X \models pre(\sigma) \wedge pre(\sigma') \rightarrow \bot$ for each $\sigma \neq \sigma' \in \Gamma$, then $\Sigma\Gamma$ is called **deterministic** over X, for X a set of pointed Kripke models. The term deterministic is used as the requirement ensures that at most one designated action from Γ updates the designated state s when $\Sigma\Gamma$ is applied to a pointed Kripke model $Ms \in X$ using **product update** $\otimes$. The product $Ms \otimes \Sigma\Gamma$ is the pointed Kripke model $([\![M\Sigma]\!], R', [\![\cdot]\!]', s')$ with

$$\begin{aligned}
[\![M\Sigma]\!] &= \left\{(s,\sigma) \in [\![M]\!] \times [\![\Sigma]\!] : Ms \models pre(\sigma)\right\} \\
R' &= \{((s,\sigma),(t,\tau)) : (s,t) \in R_i \text{ and } (\sigma,\tau) \in \mathsf{R}_i\}, \text{ for all } i \in I \\
[\![p]\!]' &= \left\{(s,\sigma) : s \in [\![p]\!], post(\sigma) \not\vDash \neg p\right\} \cup \{(s,\sigma) : post(\sigma) \vDash p\}, \text{ for all } p \in \Phi \\
s' &= (s,\sigma) : \sigma \in \Gamma \text{ and } Ms \models pre(\sigma)
\end{aligned}$$

If Ms does not satisfy the precondition of any action σ in Γ or if $\Sigma\Gamma$ is not deterministic over $\{Ms\}$, then the product is undefined.

In the product $Ms \otimes \Sigma\Gamma$, there may be states that are not reachable from the point (s,σ) via any collection of relations. Such states are, for present purposes, superfluous: They neither affect the formulas satisfied at (s,σ) nor the set of models with which $(Ms \otimes \Sigma\Gamma, (s,\sigma))$ is bisimilar. As it is later convenient to work with correspondence between structures up to isomorphism, in the current paper such superfluous states are always deleted.

Superfluous states are deleted by regarding only the substructure of any pointed Kripke model Ms that is connected to the actual state s. This substructure is denoted $C(Ms)$ and is defined as follows:

Let R^* be the reflexive, transitive and symmetric closure of $\bigcup\{R_i\}_{i\in I}$. Let $R^*(s)$ be the set of states reachable from s via R^*, i.e., $R^*(s) := \{s' \in [\![M]\!] : (s,s') \in R^*\}$. Then the **connected component** of Ms is the *unpointed* substructure $C(Ms) := ([\![M]\!]_{|R^*(s)}, R_{|R^*(s)}, V_{|R^*(s)})$. With $s' \in [\![C(Ms)]\!]$, $C(Ms)s'$ is thus again a pointed Kripke model. In particular, $C(Ms)s$ is bisimilar to Ms.[8,9]

2.3 Intensional Protocols: DEL Dynamical Systems

The most general class of maps—the intensional protocols—of interest in the following may now be defined as follows:

Definition 1 (**Clean Map**) Let X be a set of pointed Kripke models. A **clean map on** X is any possibly partial model transformer $f : X \longrightarrow X$ given by $f(x) = C(x \otimes \Sigma\Gamma)s'$, for all $x \in X$, with $\Sigma\Gamma$ a multi-pointed action model deterministic over X.

[8]This is not a *bisimulation contraction* (cf. [31]): $C(Ms)s$ need not be bisimulation minimal.

[9]The authors apologize for the cumbersome notation: It is useful when later working with connected components in unpointed epistemic, temporal structures.

Defining intensional protocols using mappings, it is required that also their domain and range be specified:

Definition 2 (**DEL Dynamical System**) A **DEL dynamical system** is a pair (X, f) where X is a set of pointed Kripke models and f is a clean map on X. A **pointed DEL dynamical system** (X, f, x) is augmented with an **initial model** $x \in X$, assumed to be connected.

The **orbit** of (X, f, x) is the (possibly finite) sequence $x, f(x), f(f(x)), ...$ The orbit is denoted $\langle f^k(x)\rangle_{k\in\mathbb{N}}$. In case the orbit is a finite sequence, $\langle f^k(x)\rangle_{k\in\mathbb{N}}$ is short for $\langle f^k(x)\rangle_{k=0}^{k'}$ for k' the maximum $k \in \mathbb{N}$ such that $f^k(x)$ is defined.

Remark 1 This definition of a DEL dynamical system is restrictive. A broader definition would allow f to be any bisimulation-preserving map.

2.4 Extensional Protocols: DEL Protocols

In [15], two types of DEL protocols are defined, one allowing the protocol to vary from state to state of the initial model and one where the protocol is "common knowledge":

Definition 3 (**DEL Protocol**) Let $\mathbb{E}$ be the class of all $\mathcal{L}_{(\Phi,I)}$ single-pointed action models. Let $\mathbb{E}^*$ be the class of all finite sequences of elements from $\mathbb{E}$. A set $\mathsf{P} \subseteq \mathbb{E}^*$ is a **DEL protocol** iff P is closed under non-empty prefixes. Let $Ptcl(\mathbb{E})$ denote the class of all DEL protocols.

Let Ms be a pointed Kripke model. A **state-dependent DEL protocol** on Ms is a map

$$\mathsf{p} : [\![M]\!] \longrightarrow Ptcl(\mathbb{E}).$$

If p is constant over $[\![M]\!]$, i.e., if for all $s, t \in [\![M]\!]$, $\mathsf{p}(s) = \mathsf{p}(t)$, then p is a **uniform DEL protocol**.

A DEL protocol specifies which pointed action models *may* be executed at a given time—whether they *can* be executed then again depends on the preconditions of the designated action. Their usage is exemplified in Section 3.2.

2.5 An Initial Comparison

Although DEL protocols and DEL dynamical systems invoke the same rudimentary changes by using action models, they differ vastly in structure. In particular, where every DEL dynamical systems encodes a deterministic[10] protocol—by virtue

[10]In the sense that given any input state (pointed Kripke model), the protocol outputs at most a single resulting state.

of being defined as a mapping—DEL protocols may be non-deterministic. Roughly, DEL dynamical systems may be correlated with state-dependent and uniform DEL protocols in the following manner:

- A DEL dynamical system is analogous to a deterministic, uniform DEL protocol: A DEL protocol $\mathsf{P} \subseteq \mathbb{E}^*$ for which all sequence $\varsigma, \varsigma' \in \mathsf{P}$, either ς is a prefix of ς' or *vice versa*.
- A non-deterministic, uniform DEL protocol is analogous to a family of DEL dynamical system, executed in parallel on the *same* pointed Kripke model.
- A non-deterministic, non-uniform DEL protocol is analogous to a family of DEL dynamical systems, executed in parallel on *different* pointed Kripke models, all of which are identical up to the choice of designated state.

In the present paper, dealing with non-uniform DEL protocols or their DEL dynamical systems counterparts will be omitted.

Without going through ETL models, DEL dynamical systems and uniform DEL protocols may be related, showing that the orbits obtainable from DEL dynamical systems is a sub-class of those obtainable using DEL protocols:

Proposition 1 *Let (X, f, x) be a pointed DEL dynamical system. Then there exists a singleton uniform DEL protocol that produces the orbit of f from x.*

Proof At each iteration, the clean map f is—in effect—going to execute a single-pointed action model. Copying the sequence of the executed action models provides a uniform DEL protocol. For details, see Appendix. □

The converse of Proposition 1 does not hold: There exists pointed Kripke models with associated singleton uniform DEL protocols that produce sequences of pointed Kripke models not duplicatable by any DEL dynamical system.[11] This is a consequence of DEL protocols being *extensional*: Not only do they consult the information inherent in the present model to determine ensuing actions, but also the current *time*, exogenously provided by the sequential nature of the protocol. This information is not available to DEL dynamical systems and can therefore not be used in guiding dynamics.

As mentioned in the introduction, this feature makes it difficult to compare DEL protocols and DEL dynamical systems directly: The structure of the DEL protocol may not be enough to determine whether the resulting sequence of pointed Kripke models may be obtained as the orbit of a DEL dynamical system. Hence the current approach: a comparison using ETL models.

[11] An example is the following: Let a two-state pointed Kripke model Ms with $Ms \models p \wedge q$ and $Mt \models p \wedge \neg q$ be given. Let $\mathsf{p}(s) = \{(!p), (!p, !p \wedge q)\}$ with $!\varphi$ the truthful public announcement of φ. Then p on Ms produces the sequence $(Ms, Ms, M's)$ with $[\![M']\!] = [\![M]\!] \backslash \{t\}$. No clean map can duplicate this sequence: As $f(Ms) = Ms$, the system has reached a fixed point from which it will never deviate to produce $M's$.

3 Examples

To give a flavor of DEL dynamical systems and DEL protocols in use, this section contains two examples. The examples are simple and do not showcase the complex agency representable by DEL protocols and DEL dynamical systems, but offer a basis for comparison. For an example of how DEL dynamical systems may be used to model a complex, multi-agent scenario with a range of agent types, see [55] where Rendsvig models the information dynamics of the *bystander effect* from social psychology.

3.1 Example 1: Blowing the Bulb

A family wishes to switch to LED lighting, but still has an incandescent bulb they consider a waste to throw out unused. They therefore instruct a child to perpetually switch the lights on and off, hoping to blow the bulb. The two possible initial states of the situation are depicted in Fig. 1.

One way of providing instructions to the child, c, is by the intensional protocol of Fig. 2 (left). This protocol may be represented by a DEL dynamical system (X, f), with $X = \{Ms, Nt\}$ and f the clean map of the action model in Fig. 2 (right). Applying f to e.g. Ms will produce Nt: Only the precondition of σ is satisfied at s, so (s, σ) is the only surviving state and the postcondition of σ forces it to satisfy $\neg\mathsf{on}$. Finally, (s, σ) is related to itself by R_c as both s and σ are self-related for c. An additional application of f returns the system to Ms. Executed anywhere on X, f implements the desired protocol.

Alternatively, the instructions may be provided by an extensional protocol, e.g., by telling the child to flip between turning on the light and turning off the light perpetually. With Σ_1 and Σ_2 the restrictions of Σ to, respectively, σ and τ, the sequences

$$P_1 = \Sigma_1\sigma, \Sigma_2\tau, \Sigma_1\sigma, \Sigma_2\tau, \Sigma_1\sigma, \Sigma_2\tau, \Sigma_1\sigma, \Sigma_2\tau, ...$$
$$P_2 = \Sigma_2\tau, \Sigma_1\sigma, \Sigma_2\tau, \Sigma_1\sigma, \Sigma_2\tau, \Sigma_1\sigma, \Sigma_2\tau, \Sigma_1\sigma, ...$$

instructs to perpetually change between executing the turnon, $\mathsf{turnoff}$ actions; P_1 instructs to start with $\mathsf{turnoff}$ while P_2 instructs to start with turnon. The uniform DEL protocol $\mathsf{p} = \{P_1, P_2\}$ will then, when executed anywhere on X, implement the desired protocol: If e.g. Nt is the initial condition, the first instruction of P_1 will fail to be executed, but the first instruction of P_2 will succeed, producing Nt, on which the second instruction of P_2 again will succeed, etc.

Adding to the futility, following either protocol, the bulb never blows.

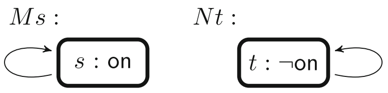


Fig. 1 Two possible states of the light: the pointed Kripke models Ms where the light is on and Nt where the light is off. The child knows the state of the light in either model

```
if on, do turnoff
if ¬on, do turnon
```

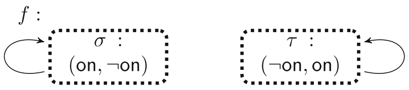


Fig. 2 Left: An intensional protocol. Right: An implementation of the intensional protocol. The clean map f is based on a multi-pointed action model $\Sigma\Gamma$ with two actions. The action σ represents turnoff: it can be executed only when the light is on, and in effect turns it off. The action τ similarly represents turnon. All (both) actions are designated actions of Σ, i.e., $[\![\Sigma]\!] = \{\sigma, \tau\} = \Gamma$, as indicated by their bold dotted boundaries. Which of the designated actions in fact occurs in a given application depends on the pointed Kripke model $\Sigma\Gamma$ is executed on

3.2 Example 2: the Muddy Children Puzzle

To illustrate the differences in use of DEL dynamical systems *qua* intensional protocols and uniform DEL protocols *qua* extensional protocols, two such formal protocols of the classic Muddy Children Puzzle, well-known in the DEL literature (see e.g. [22, 29]), are presented. As a simplified version of the puzzle is sufficient for present purposes, attention is restricted to the case with three children.

The puzzle starts with a partial description of an epistemic state:

> Three brilliant children have been playing outside. While playing, each child may have got their forehead muddy. Each can tell whether or not the others have muddy foreheads, but cannot tell this of themselves. Upon returning home from playing, an adult of unspecified gender informs the children that at least one of them is muddy.

Following standard practice in DEL, this partial description is modeled as an unpointed Kripke model for a language $\mathcal{L}_{(\Phi,I)}$ with the set of agents $I = \{a, b, c\}$ and the set of atoms $\Phi = \{a, b, c\}$ with $i \in \Phi$ read “child i is muddy”. The unpointed model M is illustrated in Fig. 3. A pointed Kripke model results when a designated

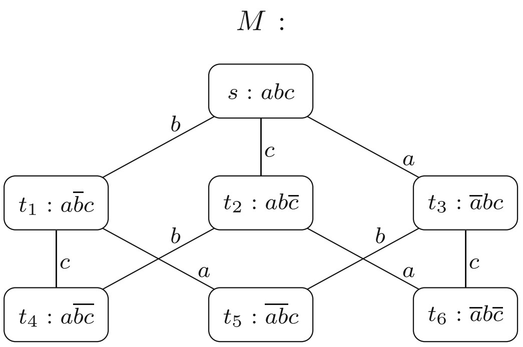


Fig. 3 Unpointed Kripke model M representing the initial situation of the Muddy Children Puzzle. Each state specifies which children are muddy, where for all $i \in \Phi$, $\bar{i} := \neg i$, that is, “child i is not muddy”. Labeled relations between states represent indistinguishability for the children. Reflexive relations are omitted

state is determined: This corresponds to fixing which children became muddy during play. Denote the set of resulting pointed Kripke models X_M.

The puzzle specification continues by the adult detailing a protocol by which the children should update the initial epistemic state:

> "Concurrently with this metronome," the adult instructs, "repeatedly and simultaneously announce aloud whether or not you know whether or not you are muddy."

By means of a suitable model of this protocol, it is desirable to be able to answer the main question of the puzzle, namely:

> If there are n muddy children, how many times does the metronome have to tick before all three children know whether or not they are muddy?

As the uniform DEL protocol and the DEL dynamical systems protocol will share the same informational actions, these will be introduced first.

3.2.1 Muddy Children: Announcements

As standard, each of the announcements made is treated as a truthful public announcement, cf. [52]. A truthful public announcement of the formula φ may be modeled using a single-pointed action model with a single action with φ as precondition.

Each epistemic announcement is modeled using the same singleton single-pointed action model, changing only the precondition. Build the formulas for the group announcements as follows:

1. Interpret the $\Box_i$ modality as reading "child i knows that...", and denote the operator by K_i.
2. Let $know_i$ be short for $K_i i \vee K_i \neg i$. If $know_i$ is true, then child i knows whether he or she is muddy or not.
3. Let $knows_S$ for $S \subseteq I$ be the formula $\bigwedge_{i \in S} know_i \wedge \bigwedge_{i \notin S} \neg know_i$. Then $knows_S$ states that exactly the children in S know their status.
4. For each $S \subseteq I$, let $\Sigma_S \sigma_S = (\{\sigma_S\}, \mathsf{R}_S, pre_S, post_S, \sigma_S)$ be the singleton single-pointed action model with $pre(\sigma_S) = knows_S$, $post(\sigma_S) = \top$ (as the announcement makes no changes to atomic valuations), and $\mathsf{R}_S(i) = \{(\sigma_S, \sigma_S)\}$ for each $i \in I$. As in the initial Kripke model, the epistemic relations are thus equivalence relations.

3.2.2 Muddy Children: Intensional Protocol

Notice that the instructions of the parent in the natural language protocol are already provided in an intensional (conditional) form. Essentially, the parent instructs the children to follow the rules

"`If you know whether you are muddy, then announce so.`", and
"`If you don't know, then announce so.`"

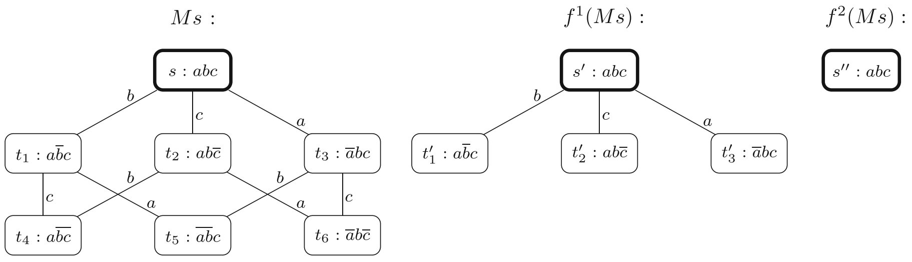


Fig. 4 The three models of the Muddy Children Puzzle in the case of three muddy children. For any $r' \in [\![f^1(Ms)]\!]$, let $r' := (r, \sigma_\emptyset)$ and let $s'' := ((s, \sigma_\emptyset), \sigma_\emptyset)$. Points are distinguished by a thick contour. It can be seen that after two rounds of question and answers all children know whether or not they are muddy. Furthermore, $f^k(Ms)$ is isomorphic to $f^2(Ms)$ for all $k \geq 2$

Aggregated to rules for the group, the antecedents in these conditional rules are exactly the preconditions of the actions in the $\Sigma_S\sigma_S$ models described in point 4 above. As these preconditions are pairwise jointly unsatisfiable over any set of pointed Kripke models and the models are disjoint, their union is a deterministic multi-pointed action model: Let $\Sigma\Gamma = ([\![\Sigma]\!], \mathsf{R}, pre, post, \Gamma)$ with $[\![\Sigma]\!] = \cup_{S\subseteq I}[\![\Sigma_S]\!]$, $\mathsf{R} = \cup_{S\subseteq I}\mathsf{R}_S$, $pre = \cup_{S\subseteq I}pre_S$, $post = \cup_{S\subseteq I}post_S$ and $\Gamma = \cup_{S\subseteq I}\{\sigma_S\}$.

Let X be a superset of the muddy children models X_M of Fig. 3, closed under the operation $\otimes\Sigma\Gamma$. With f the clean map on X based on $\Sigma\Gamma$, (X, f) is a DEL dynamical system. Moreover, applied to any $x \in X_M \subseteq X$, f implements the desired protocol and produces, tractably and in finite time, an answer to the puzzle. Figure 4 illustrates this for the case of three muddy children.

With this implementation, the intensional protocol may straightforwardly be applied to pointed models differing in other respects then the number of muddy children, e.g., with different initial announcements by the parent.

3.2.3 Muddy Children: Extensional Protocol

Constructing an extensional protocol for the Muddy Children given some initial Kripke model is straightforward: Simply run the intensional protocol above on the initial model, taking note which designated actions' preconditions were satisfied when and encode this sequence as an extensional protocol. The resulting extensional protocol will induce the transformations appropriate for the given initial model. However, the protocol will not be useful in answering the question of the puzzle: It is a one-off solution for the given Kripke model only, constructed with knowledge of the answer sought.

A more informative extensional DEL protocol may be constructed, but it requires a countably infinite representation: Assume to construct an extensional DEL protocol that will adequately encode the natural language instructions, is applicable to any model in X_M and presumes no prior knowledge of the developing information dynamics. The set of relevant announcements is, as above, $\{\Sigma_S\sigma_S : S \subseteq I\}$. For the announcement made at the first time step, the protocol must allow $\Sigma_S\sigma_S$ for each

$S \subseteq I$, seeing that no information about the development of the dynamics may be assumed. Similarly, each possible announcement must be allowed to follow the first, etc. Hence, only satisfactory extensional protocol is $\mathsf{P} = \{\Sigma_S\sigma_S \colon S \subseteq I\}^*$. This set is countably infinite.

This protocol facilitates the process of finding an answer to the Muddy Children Puzzle: For a given initial model $Ms \in X_M$, find the actions that the protocol allows to be executed at time 1. These are all the actions models $\Sigma_S\sigma_S$ for which the length 1 sequence $\langle \Sigma_S\sigma_S \rangle$ is in P (i.e., all the actions the protocol allows at time 1). For each of these, calculate the product $Ms \otimes \Sigma_S\sigma_S$. As the preconditions are, in the current example, mutually inconsistent, only one such model will be well-defined. The result is exactly $f(Ms)$, for f the intensional protocol given above. For time 2, take all the models produced at time 1 – in this case $\{f(Ms)\}$—and execute on each of them all the actions in the continuations of the sequence from which that model stems. This produces a second set of pointed Kripke models—in this case $\{f(f(Ms)\}$. This process thus leads to a model in which all children will announce that they know whether they are muddy.

3.2.4 A Remark on Protocol Size

If one is interested in implementing a DEL protocol to seek computational assistance in puzzle solving, the countably infinite representations required for the extensional protocol may prove cumbersome.[12] Represented as a countably infinite set of sequences, an implementation will never run through the first step of all allowed action sequences.[13] This problem does not occur when protocols are implemented as clean maps: They are by construction finite. Set theoretically, the clean map representation of a protocol may thus be vastly smaller than its extensional counterpart.

4 Epistemic Temporal Logic

The run of a pointed DEL dynamical system may be recorded as a sequence of pointed Kripke models. Using information from the action models, an insight of [15] was that this may naturally be regarded as a temporal, modal structure, a so-called *epistemic temporal logic model.*

ETL models form a simple and general framework. Such models allow the representation of epistemic and temporal interplay, and allow it in an assumption-free manner. Hence, in generating ETL models from DEL dynamics, any structural properties (e.g., Synchronicity, Perfect Recall) shared by the generated ETL models are features induced by the DEL operations. Thus, characterizing the classes of

[12]It was not suggested in [15] that DEL protocols be implementable nor that they are suited for modeling purposes.

[13]To produce an implementation, the extensional protocol should at least be represented in a different manner.

generatable ETL models elucidates assumptions implicit in DEL dynamics about epistemic and temporal interplay. This is a main conceptual insight of [15].

An ETL model is a temporal forest with additional modal (epistemic) relations between nodes. With E^* the set of all finite sequences of elements from the set E, an **ETL model** for the language $\mathcal{L}_{(\Phi,I)}$ is a tuple $\mathcal{H} = (E, H, \mathtt{R}, V)$ where

E is a set of **events** e;

$H \subseteq E^*$ is a set of **histories**, closed under non-empty prefixes;[14]

$\mathtt{R} : I \longrightarrow \mathcal{P}(H \times H)$ is a map assigning to each agent an **accessibility relation** $\mathtt{R}(i)$, written $\mathtt{R}_i$;

$V : \Phi \longrightarrow \mathcal{P}(H)$ is a **valuation**.

In contrast with pointed Kripke models, ETL models are not equipped with actual states. To obtain a tighter connection between DEL dynamical system orbits and ETL models, the latter is augmented to include *multiple points*. Figure 5 on the next page illustrates such an augmented ("saturated") ETL model and its relation to the orbit of a DEL dynamical system.

As in Section 2.2, let $\mathtt{R}^*$ be the reflexive, symmetric and transitive closure of $\mathtt{R}$, and let $\mathtt{R}^*(h) := \{h' \in H : (h, h') \in \mathtt{R}^*\}$. Then define the **connected component** of $h \in H$ in $\mathcal{H}$—denoted $C(\mathcal{H}h)$—as the restriction of $\mathcal{H}$ to $\mathtt{R}^*(h)$, i.e., let $C(\mathcal{H}h) := (H_{|\mathtt{R}^*(h)}, \mathtt{R}_{|\mathtt{R}^*(h)}, V_{|\mathtt{R}^*(h)})$. If $\underline{h} \in H_{|\mathtt{R}^*(h)}$, then $C(\mathcal{H}h)\underline{h}$ is a pointed Kripke model.

Finally, define the ETL structures of interest as follows:

Definition 4 (**Saturated ETL Model**) Let $\mathcal{H} = (E, H, \mathtt{R}, V)$ be an ETL model. Let $\underline{H} \subseteq H$ be a set of histories closed under prefixes, called **points**. The pair $(\mathcal{H}, \underline{H})$ is **saturated** iff for all $h \in H$, the connected component $C(\mathcal{H}h)$ contains a unique point $\underline{h}$ from $\underline{H}$.

Remark 2 The addition of points to ETL models is vital to structurally relate ETL models to orbits of DEL dynamical systems. When computing the $k + 1$th element of an orbit of a DEL dynamical system, its clean map reacts to information from the designated state of the kth element. The designated states thus carry information determining the dynamics and are therefore structurally essential. However, nothing plays a similar role in ETL models. Hence, a stronger structural likeness between the two constructions may be obtained by adding a corresponding notion to ETL models: the points.

4.1 ETL Isomorphism

For simplicity of arguments, saturated ETL models are identified up to isomorphism. This allows arguments without repeated references to bisimulation contractions or

[14]It is overall assumed that any ETL model contains no redundant events relative to the model's set of histories. That is, for any ETL model $\mathcal{H}$, it holds that any event $e \in E$ is either a history ($e \in H$) or part of a history ($\exists h \in H$ such that $he \in H$).

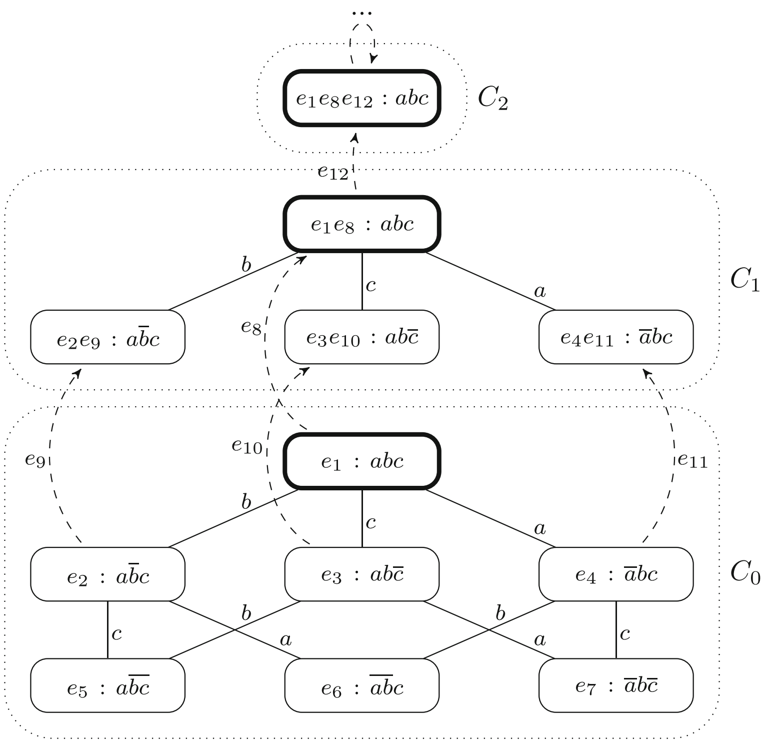


Fig. 5 A saturated ETL model $(\mathcal{H}, \underline{H})$. Time flows upwards where labeled dashed lines represent events. Connected components are marked by dotted circles, points by thick contours. Each pointed connected component is isomorphic to a pointed Kripke model from Fig. 4. Moreover, a history h' is the successor of h in $\mathcal{H}$ iff the Fig. 4 counterpart of h' is a state (s, σ) for s the counterpart of h

other specific representatives. In the definition of isomorphism between ETL models, note that the temporal structure of the models is also preserved:

Definition 5 (ETL Isomorphism) Let saturated ETL models $(\mathcal{H}, \underline{H}) = (E, H, \mathtt{R}, V, \underline{H})$ and $(\mathcal{H}', \underline{H}') = (E', H', \mathtt{R}', V', \underline{H}')$ be given. Let $f : E \to E'$. For $h = e_0 ... e_n \in E^*$, let $f(h) := f(e_0)...f(e_n)$. The map f is an **ETL isomorphism** iff f is a bijection and for all $h \in H$, $h' \in H'$

1. $h \in H$ iff $f(h) \in H'$, and $h \in \underline{H}$ iff $f(h) \in \underline{H}'$,
2. $h\mathtt{R}_i h'$ iff $f(h)\mathtt{R}'_i f(h')$, for all $i \in I$,
3. $h \in V(p)$ iff $f(h) \in V'(p)$, for all $p \in \Phi$.

$(\mathcal{H}, \underline{H})$ and $(\mathcal{H}', \underline{H}')$ are **ETL isomorphic** iff there exists an ETL isomorphism between their domains.

In the remainder, "ETL isomorphism" and "isomorphism" are used interchangeably.

4.2 Eight Properties of Saturated ETL Models

When generating an ETL model from a DEL dynamical system, the resulting forest will inherit a set of properties. Some stem from the graph theoretic nature of action models, product update and the associated pruning to connected components of clean maps, some from the workings of pre- and postconditions, and yet some stock from the functional *modus operandi* of dynamical systems. This section defines the eight properties of main relevance to this paper. In describing their intuitions, the agent relations are interpreted to represent indistinguishability.

Throughout this section, let $\mathcal{H} = (E, H, \mathtt{R}, V, \underline{H})$ be a saturated ETL model. Notationally, $len(h)$ denotes the *length* of history h, he denotes the sequence extending history h with event e and $h \sqsubseteq h'$ denotes that h is a prefix of h'. Agent-quantification is suppressed: The stated properties should all be taken to hold for all agents.

The first three properties are well-known. First, Synchronicity requires that agents know the current time: If two histories are indistinguishable for agent i, then they are of equal length. $\mathcal{H}$ satisfies

Synchronicity iff $\forall h, h' \in H$, if $h\mathtt{R}_i h'$, then $len(h) = len(h')$.

Second, Perfect Recall ensures agents never forget what they have learned (though new uncertainty may be introduced): If agent i cannot distinguish two histories, then neither can i distinguish their predecessors. $\mathcal{H}$ satisfies

Perfect Recall iff $\forall h, h' \in H, \forall e, e' \in E : he, h'e' \in H$, if $he\mathtt{R}_i h'e'$, then $h\mathtt{R}_i h'$.

Third, Local No Miracles enforces that events carry the same information in all states in the same "epistemic context" (connected component): If two events do not carry distinguishing information in one history of the context, then they should not miraculously carry information in another history in the same context. $\mathcal{H}$ satisfies

Local No Miracles iff $\forall h, h', h_1, h_2 \in H, \forall e, e' \in E : he, h'e' \in H$, if $h\mathtt{R}_i h'$, $h_1\mathtt{R}_i h_2$, $h_1 e\mathtt{R}_i h_2 e'$ and $h\mathtt{R}^* h_1$, then $he\mathtt{R}_i h'e'$.

Synchronicity, Perfect Recall and Local No Miracles were identified by van Benthem et al. [15] to be inherited in any ETL model generated using sequences of actions models (see Section 9 for discussion and comparison).

The fourth property is almost the converse of Synchronicity: Connected Time-Steps requires that a time-step contains at most one connected component. ETL models generated by clean maps will satisfy this property as clean maps delete superfluous states. $\mathcal{H}$ satisfies

Connected Time-Steps iff $\forall h, h' \in H$, if $len(h) = len(h')$, then $h\mathtt{R}^* h'$.

The next three properties all concern definability issues. Each in their own way, they ensure that a DEL dynamical system sought to generate $\mathcal{H}$ can invoke the right operation at the right place or time. They are given as existence requirements without listing criteria that ensure their satisfaction.

First, Precondition Describable ensures the definability of the precondition of the action σ_e that will emulate event e in the DEL dynamical system: The formula δ_e required to exist describes exactly those histories h in a connected component on which e is executed. $\mathcal{H}$ satisfies

Precondition Describable iff $\forall e \in E$, there exists a $\delta_e \in \mathcal{L}_{(\Phi,I)}$ such that if there is a $h'e \in H$, then for all $h \in H$, if $h'\mathrm{R}^*h$, then $C(\mathcal{H}h)h \models \delta_e$ iff $he \in H$.

Second, Postcondition Describable ensures the definability of the postcondition of the action σ_e that will emulate event e in the DEL dynamical system: The formula δ_{D_e} describes the propositional change due to e; the set D_e contains the atoms made true by e and the negation of the atoms that e makes false. $\mathcal{H}$ satisfies

Postcondition Describable iff $\forall he \in H$, there exists a $\delta_{D_e} \in \mathcal{L}_{(\Phi,I)}$ such that $\delta_{D_e} \models D_e$ for
$D_e = \{p \in \Phi : h \notin V(p), he \in V(p)\} \cup \{\neg q : q \in \Phi, h \in V(q), he \notin V(q)\}$.

Third, Component Collection Describable ensures the definability of an additional precondition for each point $\sigma \in \Gamma$ of the multi-pointed action model $\Sigma\Gamma$ of the DEL dynamical system. The additional precondition φ is akin to the test conditions in a "`if` φ, `do` a" instruction from knowledge-based programs: It specifies when σ should be the "surviving" designated action. When the points of $(\mathcal{H}, \underline{H})$ are suitably describable by such tests, the tests may be used to control the behavior of the DEL dynamical system. The right degree of describability of points of $(\mathcal{H}, \underline{H})$ for such control turns out to be on the level of collections of connected components.

Identifying each component by its point, a **component collection** for $(\mathcal{H}, \underline{H})$ is a set of points $A \subseteq \underline{H}$ such that

a) all points in A belong to a common history: $\underline{h}, \underline{h'} \in A$ implies $\underline{h} \sqsubseteq \underline{h'}$ or $\underline{h'} \sqsubseteq \underline{h}$, and
b) respecting common histories, A is closed under bisimulation equivalence: if $(\underline{h} \sqsubseteq \underline{h'}$ or $\underline{h'} \sqsubseteq \underline{h})$ and $\underline{h} \leftrightarroweq \underline{h'}$, then $\underline{h} \in A$ iff $\underline{h'} \in A$.

The property Component Collection Describable then requires the existence of a formula φ_A which must be true at the points $A \subseteq \underline{H}$ (the "right times"), while being false at $\underline{H}\backslash A$ (excluding "wrong times"), for every component collection A. $\mathcal{H}$ satisfies

Component Collection Describable iff for every component collection $A \subseteq \underline{H}$, there exists a $\varphi_A \in \mathcal{L}_{(\Phi,I)}$ such that $C(\mathcal{H}\underline{h})\underline{h} \models \varphi_A$ iff $\underline{h} \in A$.

The final property, Point Bisimulation Invariance, ensures that the temporal structure of $(\mathcal{H}, \underline{H})$ is mimicable by a mapping: Identical conditions must be followed by identical effects. Point Bisimulation Invariance reflects this slogan for two aspects of clean maps. First, that clean maps are mappings: When applied to identical elements (pointed Kripke models that have bisimilar points), identical images result. Second, the property reflects the slogan in the workings of the preconditions of action models: If the same action model is executed on any two pointed Kripke models, then if *any* points in those two models are bisimilar, they will be treated equally under the product with the action model.[15] $\mathcal{H}$ satisfies

Point Bisimulation Invariance iff $\forall h_1, h_2, h_3, h_4 \in H$, if $C(\mathcal{H}h_1)\underline{h} \leftrightarrows C(\mathcal{H}h_2)\underline{h'}$ and $C(\mathcal{H}h_1)h_3 \leftrightarrows C(\mathcal{H}h_2)h_4$, then $h_3e \in H$ iff $h_4e \in H$.

In words: Take two connected components, $C(\mathcal{H}h_1)$ and $C(\mathcal{H}h_2)$ from $(\mathcal{H}, \underline{H})$. Each component will have a single designated point; let $\underline{h}$ be the designated point of $C(\mathcal{H}h_1)$ and $\underline{h}'$ that of $C(\mathcal{H}h_2)$. Point Bisimulation Invariance then states the following: If the points $\underline{h}$ and $\underline{h}'$ are bisimilar, then if two other histories, say h_3 and h_4, from respectively $C(\mathcal{H}h_1)$ and $C(\mathcal{H}h_2)$ are also bisimilar, then h_3 and h_4 will be extended by exactly the same events. In short: If the points are bisimilar, then history bisimilarity implies event effect invariance.

5 Generated ETL Models and Their Properties

A saturated ETL model is generated from an initial pointed Kripke models x and a clean map f by, essentially, recording the orbit of f from x as a temporal structure: The states of x become histories of length 1 and states of $f^k(x)$ become histories of length $k+1$; the actual state in each $f^k(x)$ becomes an ETL model point; epistemic relations and valuations are directly transferred. Formally:

Definition 6 (**Generated Structure**) For any pointed DEL dynamical system (X, f, x), its **generated structure** is the tuple $(E, H, \mathrm{R}, V, \underline{H})$ given by

$$\begin{aligned}
E &:= \{e_\sigma : \sigma \in \nabla_k \text{ for some } k \in \mathbb{N}\}\\
&\quad\ \text{for } \nabla_0 := [\![x]\!] \text{ and } \nabla_{k+1} := \{\sigma : (s, \sigma) \in [\![f^{k+1}(x)]\!]\}\\
H &:= \{\gamma(s) : s \in [\![f^k(x)]\!] \text{ for some } k \in \mathbb{N}\}\\
&\quad\ \text{with } \gamma : \bigcup_{k\in\mathbb{N}} \nabla_k \longrightarrow E \text{ given by } \gamma(\sigma) = e_\sigma\\
&\quad\ \text{and for } s = ((\sigma_1, \sigma_2), ..., \sigma_n) \text{ use } \gamma(s) := \gamma(\sigma_1)\gamma(\sigma_2)...\gamma(\sigma_n)\\
\mathrm{R}_i &:= \{(h, h') \in H \times H : \gamma^{-1}(h)R_i\gamma^{-1}(h')\} \text{ for all } i \in I\\
V(p) &= \{h \in H : \exists k \in \mathbb{N}, \gamma^{-1}(h) \in [\![p]\!]_k := \{t \in [\![f^k(x)]\!] : t \vDash p\}\}\\
\underline{H} &:= \{\underline{h} : \exists k \in \mathbb{N},\ f^k(x) = Ms \text{ and } \underline{h} = \gamma(s)\}
\end{aligned}$$

If $(E, H, \mathrm{R}, V, \underline{H})$ is isomorphic to a saturated ETL model $(\mathcal{H}', \underline{H}')$, then (X, f, x) **generates** $(\mathcal{H}', \underline{H}')$.

Property 1 For any DEL dynamical system, the structure generated is a saturated ETL model: H is indeed closed under prefixes and it is saturated as for all $h \in H$, $C(\mathcal{H}h)$ shares a unique $\underline{h}$ with $\underline{H}$.

[15] The second aspect is the content of the weaker property Local Bisimulation Invariance of [15], to which Point Bisimulation Invariance is related in Section 9.

The first main result furnishes a set of properties that any DEL dynamical system generated ETL model will necessarily satisfy:

Proposition 2 *If saturated ETL model* $(\mathcal{H}, \underline{H})$ *is generated by a pointed DEL dynamical system, then* $(\mathcal{H}, \underline{H})$ *satisfies seven of the eight properties of* Section 4.2, *namely Synchronicity, Perfect Recall, Local No Miracles, Connected Time-Steps, Precondition Describable, Postcondition Describable, and Point Bisimulation Invariance.*

Proof All proofs may be found in Appendix. □

The second result shows that the last property of Section 4.2 is indeed only a contingent feature of some generated ETL models:

Proposition 3 *Not all saturated ETL models generated by pointed DEL dynamical systems are Component Collection Describable.*

Proof See Appendix. □

6 From ETL Model to Dynamical System

For certain ETL models, there exists DEL dynamical systems that will generate them. The following result lists sufficient conditions of an ETL model for it to be generatable by a DEL dynamical system:

Proposition 4 *If* $(\mathcal{H}, \underline{H})$ *is a saturated ETL model that satisfies all eight properties of Section 4.2, then there exists a pointed DEL dynamical system that generates* $(\mathcal{H}, \underline{H})$.

The proof, which may be found in the Appendix, rests on the idea of regarding an ETL model as a collection of *saturated component branches*, illustrated in Fig. 6. Each such branch is a sequence of pointed Kripke models and hence potentially the orbit of a DEL dynamical system.

To obtain the notion of a saturated component branch, decompose ETL model into *branches*, lump these together in connected components and saturate:

Definition 7 (**Branches**) A **branch** of an ETL model $\mathcal{H} = (E, H, \mathtt{R}, V)$ is a set $b \subseteq H$ that

1. has a unique **root**, i.e., contains a unique history that has length 1;
2. is maximal with unique extension: If $h \in b$ and $he \in H$ for some $e \in E$, then $|\{he' : he' \in b\}| = 1$;
3. is closed under finite prefixes.

The **component branch** of b is the sequence $\mathbf{b} = \mathbf{b}_1, \mathbf{b}_2, \ldots$ of connected components that *a*) is ordered according to history length, *b*) has prefix $(C(\mathcal{H}h))_{h \in b}$, and *c*) is

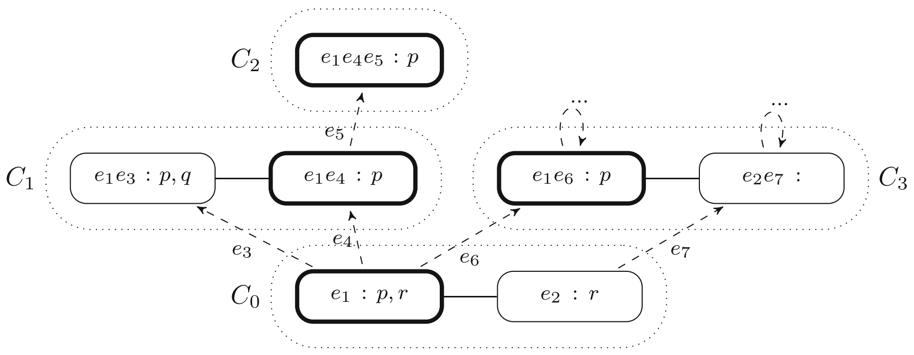


Fig. 6 An ETL model $(\mathcal{H}, \underline{H})$ with two saturated component branches. Connected components C_0, C_1 and C_2 form one component branch $\mathbf{b}$. With $\underline{H}_1 = \{e_1, e_1e_4, e_1e_4e_5\}$, $(\mathbf{b}, \underline{H}_1)$ is a saturated component branch. The infinite set consisting of components C_0, C_3, etc. form another component branch $\mathbf{b}'$. Notice that for the branch $b = \{e_1, e_1e_3\}$, the component branch $\mathbf{b}$ is longer than b itself, as $\mathbf{b}$ includes the component C_2

extended to be either **maximal** in H ($\exists k \in \mathbb{N} \forall h' \in \mathbf{b}_k \neg \exists e \in E : h'e \in H$) or **infinite** ($\forall k \in \mathbb{N} \exists h' \in \mathbf{b}_k \exists e \in E : h'e \in \mathbf{b_{k+1}}$). A **saturated component branch** is a pair $(\mathbf{b}, \underline{H})$ with $\underline{H} \subseteq H$ a set of **points** closed under finite prefixes such that every component in $\mathbf{b}$ has exactly one point.

Enumerating $\underline{H}$ by history length, the following link to pointed Kripke models is obtained:

Property 2 For saturated component branch $(\mathbf{b}, \underline{H})$, the pair $(\mathbf{b}_k, \underline{h}_k)$ is a pointed Kripke model.

Moreover, the construction emphasizes how specific ETL models have a strong resemblance to DEL dynamical system orbits:

Property 3 If saturated ETL model $(\mathcal{H}, \underline{H})$ has property Connected Time-Steps, then $\mathcal{H}$ has a unique component branch $\mathbf{b}$.

Jointly, these two properties allow us to illustrate the proof methodology of Proposition 4: Take an ETL model $\mathcal{H}$ that has Connected Time-Steps and is saturated by points $\underline{H}$. Envision the model as a component branch $\mathbf{b}$ saturated by $\underline{H}$. From this, extract the sequence of pointed Kripke models $(\mathbf{b}_k, \underline{h}_k)_{k \in \mathbb{N}}$. For each k, find an action model that transforms $(\mathbf{b}_k, \underline{h}_k)$ into $(\mathbf{b}_{k+1}, \underline{h}_{k+1})$. Join all these action models into a deterministic multi-pointed action model and construct its clean map f. Then $(\{(\mathbf{b}_k, \underline{h}_k) : k \in \mathbb{N}\}, f, (\mathbf{b}_1, \underline{h}_1))$ is a pointed DEL dynamical system that generates $(\mathcal{H}, \underline{H})$. Full details may be found in the Appendix.

The relation between DEL dynamical systems and ETL models that do not have Connected Time-Steps is discussed in Section 8.

7 Characterization: Image-Finite and Concluding

Propositions 2 and 4 do not quite yield a characterization result pertaining to the ETL model generatable by DEL dynamical systems. This is due to the fact that Component Collection Describable is not implied for ETL models generated by an DEL dynamical system when working with a normal, finitary modal logical language, as shown by Proposition 3.

Imposing two restrictions on ETL models and DEL dynamical systems yields a characterization result. Both are finiteness assumptions. The first an assumption of *image-finiteness* for the modal relations:

A binary relation $B \subseteq A \times A$ is **image-finite** iff the set $\{y : (x, y) \in B\}$ is finite for all $x \in A$. On sets of image-finite structures, the **Hennessy-Milner Theorem** ensures that bisimilarity and modal equivalence relate exactly the same models, cf. e.g. [17, 31]. The assumption is therefore natural from a modal logical point of view. The notion may be applied to DEL dynamical systems and ETL models: Call a pointed DEL dynamical system (X, f, x) image-finite if both x and the action model of f are image-finite for all I-indexed relations. This ensures that $f^k(x)$ is image-finite for all $i \in I$, all $k \in \mathbb{N}$. An ETL model is image-finite if all its I-indexed relations are image-finite.

The second restriction concerns the temporal evolution, which is required to show finite variety:

Definition 8 (**Concluding DEL Dynamical System**) A pointed DEL dynamical system (X, f, x) is **periodic** iff $f^k(x) = f^{k+m}(x)$ for some $k \geq 0, m > 0$. It **terminates** iff for some $k \in \mathbb{N}$, $f^k(x)$ is undefined. If it does either, it is said to **conclude**.

Definition 9 (**Concluding ETL Model**) A point $\underline{h} \in \underline{H}$ of a saturated ETL model $(\mathcal{H}, \underline{H})$ is **repeating** if there exists points $\underline{h}', \underline{h}'' \in \underline{H}$ with $\underline{h} \sqsubseteq \underline{h}' \sqsubset \underline{h}''$ and $C(\mathcal{H}\underline{h}')\underline{h}' \leftrightarroweq C(\mathcal{H}\underline{h}'')\underline{h}''$. A point $\underline{h}$ is **finite** if there exists a point $\underline{h}'$ with $\underline{h} \sqsubseteq \underline{h}'$ while there is no $e \in E$ for which $\underline{h}'e \in H$. The model $(\mathcal{H}, \underline{H})$ **concludes** if every point in $\underline{H}$ is either repeating or finite.

Restricting attention to the classes of image-finite and concluding DEL dynamical systems and ETL models, a proper characterization result exists:

Theorem 1 *A saturated ETL model $(\mathcal{H}, \underline{H})$ is image-finite, concluding and satisfies all eight properties of* Section 4.2 *if, and only if, it is generatable by an image-finite and concluding pointed DEL dynamical system.*

Proof *Left-to-right:* The existence of a generating DEL dynamical system is guaranteed by Proposition 4. The constructions in the proof of Prop. 4 moreover ensure that the DEL dynamical system is both image-finite and concluding.

Right-to-left: Proposition 2 ensures that the model will satisfy all eight properties, except maybe Component Collection Describable. Lemma 1 ensures the model is image-finite and concluding, which by Lemma 2 ensures that it does satisfy Component Collection Describable. Both lemma's are proven in the Appendix.

Lemma 1 *If there exists an image-finite and concluding pointed DEL dynamical system that generates $(\mathcal{H}, \underline{H})$, then $(\mathcal{H}, \underline{H})$ is image-finite and concluding.*

Lemma 2 *If a saturated ETL model $(\mathcal{H}, \underline{H})$ is image-finite, concluding and satisfies Connected Time-Steps, then $(\mathcal{H}, \underline{H})$ is Component Collection Describable.* □

Remark 3 The converse of Lemma 2 does not hold, as Component Collection Describable does not imply image-finiteness.[16]

8 Non-deterministic Intensional Protocols

In the previous sections, the ETL models contained only single component branches as this is a requirement to be generatable from a DEL dynamical system—or a deterministic extensional DEL protocol. Extensional DEL protocols are in general non-deterministic and may therefore generate ETL models with multiple component branches, as e.g. the ETL model in Fig. 6. To facilitate comparison, this section is dedicated to non-deterministic intensional protocols, implemented as families of DEL dynamical systems running in parallel.

Definition 10 (**Component Branch Sub-model**) Let $\mathcal{H} = (E, H, \mathtt{R}, V)$ be an ETL model and let $(\mathbf{b}, \underline{H})$ be a saturated component branch obtained from $\mathcal{H}$. The **component branch sub-model** of $\mathcal{H}$ given by $(\mathbf{b}, \underline{H})$ is then $(\mathcal{H}_{\mathbf{b}}, \underline{H}) = (E_{\mathbf{b}}, H_{\mathbf{b}}, \mathtt{R}_{|H_{\mathbf{b}}}, V_{|H_{\mathbf{b}}}, \underline{H})_{i\in I}$ such that $E_{\mathbf{b}} = \{e \in E : e \in \mathbf{b} \text{ or } \exists h \in \mathbf{b}, he \in \mathbf{b}\}$, $H_{\mathbf{b}} = \{h \in H : h \in \mathbf{b}\}$ and $_{|H_{\mathbf{b}}}$ denotes restriction.

Property 4 If $\mathcal{H}$ is an ETL model and $(\mathbf{b}, \underline{H})$ a saturated component branch obtained from $\mathcal{H}$, then $(\mathcal{H}_{\mathbf{b}}, \underline{H})$ is a saturated ETL model.

To (re-)produce ETL models that consist of more than one component branch, a family of dynamical systems each generating a component branch of the ETL model is used. The complete ETL model is obtained by taking the union of all ETL component branches.

Definition 11 (**ETL Model Union**) Given a countable family of saturated ETL models $\{(\mathcal{H}_j, \underline{H}_j)\}_{j\in J}$ with each $(\mathcal{H}_j, \underline{H}_j) = (E_j, H_j, \mathtt{R}_j, V_j, \underline{H}_j)$ for $j \in J$, their (unpointed) **union model** is $U_J = (E_J, H_J, \mathtt{R}_J, V_J)$ with $\star_J := \bigcup_{j\in J} \star_j$ for $\star \in \{E, H, \mathtt{R}, V\}$.

[16]Being Component Collection Describable does not imply being image-finite: Let the model have two components in one component branch. Let the root component $\mathbf{b}_0$ be image-infinite and satisfy p at $\mathbf{b}_0\underline{h}$. Let $\mathbf{b}_1\underline{h'}$ satisfy $\neg p$. Then the model satisfies Component Collection Describable with $\varphi_{\{\underline{h}\}} := p$, $\varphi_{\{\underline{h'}\}} := \neg p$ and $\varphi_{\{\underline{h},\underline{h'}\}} := p \wedge \neg p$, but it is not image-finite.

An ETL model $\mathcal{H}$ is **generated by a family of pointed DEL dynamical systems** $\{(X_j, f_j, x_j)\}_{j\in J}$ iff each (X_j, f_j, x_j) generates a saturated ETL model $(\mathcal{H}_j, \underline{H}_j)$ such that $\mathcal{H}$ is the union model of$\{(\mathcal{H}_j, \underline{H}_j)\}_{j\in J}$. The family $\{(X_j, f_j, x_j)\}_{j\in J}$ is **minimal** in generating $\mathcal{H}$ iff no proper subset of the family also generates $\mathcal{H}$.

Lemma 3 *Let $\{(X_j, f_j, x_j)\}_{j\in J}$ be minimal in generating $\mathcal{H}$ and let (X_j, f_j, x_j) generate the saturated ETL model $(\mathcal{H}_j, \underline{H}_j)$. Then $(\mathcal{H}_j, \underline{H}_j)$ is the component branch sub-model for some saturated component branch* $(\mathbf{b}, \underline{H})$ *of $\mathcal{H}$.*

Proof See Appendix. □

Theorem 2 *Let an image-finite and concluding ETL model $\mathcal{H}$ be given. $\mathcal{H}$ is generatable up to ETL isomorphism by a family of image-finite and concluding pointed DEL dynamical systems, if, and only if, there exists a saturation of each component branch* $\mathbf{b}$ *of $\mathcal{H}$ such that $(\mathcal{H}_{\mathbf{b}}, \underline{H})$ satisfies all eight properties of* Section 4.2.

Proof See Appendix. □

8.1 Persistence Under Union

The properties in Theorem 2 are associated with the component branches of the ETL model, instead of the ETL model itself. However, several of the eight properties are not inherited from component branches to the union model: Some are not defined for unpointed structures, and some are simply not robust under union. In the following final set of results linking DEL dynamical systems and ETL models, properties definable for general, unpointed ETL models are detailed.

Lemma 4 *The saturated ETL model properties Synchronicity, Perfect Recall and Postcondition Describable persist under ETL model union. I.e.: Let $\{(\mathcal{H}_j, \underline{H}_j)\}_{j\in J}$ be a countable set of saturated ETL models. If all $(\mathcal{H}_j, \underline{H}_j)$ satisfy either of the mentioned properties, then the (unsaturated) union model U_J satisfies that property.*

Proof See Appendix. □

Property 5 Local No Miracles, Precondition Describable, Point Bisimulation Invariance and Connected Time-Steps do not persist under union.

Proof See Appendix. □

Though neither Local No Miracles nor Point Bisimulation Invariance persist under union, weaker versions of each property do recur in the union model: See Proposition 5 below. Both properties persist as they are independent of structure outside a given component. They are therefore not affected by union. Local Bisimulation Invariance originates from [15] and is further discussed in Section 9.

Definition 12 (ETL Model Properties) An unsaturated ETL model $\mathcal{H} = (E, H, \mathtt{R}, V)$ satisfies

Very Local No Miracles iff $\forall h, h', h_1, h_2 \in H, \forall e, e' \in E : he, h'e' \in H$, if $h\mathtt{R}_i h'$, $h_1 e \mathtt{R}_i h_2 e'$, $he\mathtt{R}^* h_1 e$ and $h\mathtt{R}^* h_1$, then $he\mathtt{R}_i h'e'$;
Local Bisimulation Invariance iff for all $h, h' \in H$, $e \in E$, if h and h' are bisimilar, $h\mathtt{R}^* h'$ and $he \in H$, then $h'e \in H$.

Proposition 5 *If an ETL model $\mathcal{H}$ is generated by a family of pointed DEL dynamical systems (possibly neither image-finite nor concluding), then $\mathcal{H}$ satisfies Synchronicity, Perfect Recall and Postcondition Describable and Very Local No Miracles and Local Bisimulation Invariance.*

Proof See Appendix. □

9 Protocol Comparison

This paper is in line with the approach of van Benthem et al. [15] in investigating the generative power of DEL dynamical systems with respect to the class of ETL models. In this section, the above results are compared to those obtained in [15] relating DEL protocols to ETL models.

9.1 Generating ETL Models from DEL Protocols

Generating ETL models from DEL protocols is somewhat simpler than from sets of DEL dynamical systems. Unsaturated ETL forests are generated directly from a DEL protocol, without e.g. first defining saturated component branches. For the special case of uniform DEL protocols, an ETL model is generated from an initial pointed Kripke model as follows, cf. [15]:[17]

Definition 13 (ETL Model Generated from a Uniform DEL Protocol) Let p be a uniform DEL protocol for the pointed Kripke model Ms, let $\rho = \rho_1 ... \rho_n \in \mathsf{p}(s)$ and let $(Ms)^\rho := (Ms \otimes \rho_1) ... \otimes \rho_n$. The generated ETL model of Ms and p is $\mathcal{H} = (E, H, \mathtt{R}, V)$ with $(H, \mathtt{R}, V) = \bigcup_{\rho \in \mathsf{p}} (Ms)^\rho$.

Remark 4 Notice that no restriction to connected components is required posterior to taking products.

[17]The method for generating an ETL model from a state-dependent DEL protocol has a slightly more complex definition. As uniform DEL protocols is the case closest to the cases for DEL dynamical systems dealt with in this paper, the reader is referred to [15] for the definition for state-dependent DEL protocols.

9.2 ETL Properties from DEL Protocols

The properties of ETL models generated from DEL protocols [15] results in a list of properties not identical to that of Section 4.2. But there is overlap: Synchronicity, Perfect Recall and Local No Miracles. The remaining properties from [15] are Local Bisimulation Invariance, Propositional Stability and Finite Executions:

Definition 14 (ETL Model Properties of [15]**)** An ETL model $\mathcal{H} = (E, H, \mathtt{R}, V)$ satisfies

Propositional Stability iff for all propositional formulas p and for all $h \in H, e \in E$ such that $he \in H$, it holds that $h \in V(p)$ iff $he \in V(p)$;

Finite Executions iff for each n, for each $e \in E$, the set $\{h : he \in H$ and $len(h) = n\}$ is finite.

Remark 5 The property Propositional Stability is required as [15] concerns action models *without postconditions*.[18] There is a comment on the resulting difference below.

Before relating DEL protocols to DEL dynamical systems and the ETL model properties they induce, recall the main results of [15].

Theorem (Main Representation Theorem of [15]**)**

1) *If an ETL model is generated by a uniform DEL protocol, then it satisfies the five properties Propositional Stability, Local Bisimulation Invariance, Synchronicity, Perfect Recall and Local No Miracles.*
2) *If an ETL model satisfies the six properties Finite Executions, Propositional Stability, Local Bisimulation Invariance, Synchronicity, Perfect Recall and Local No Miracles, then it is generatable by some uniform DEL protocol.*

Theorem (Theorem 2 of [15]**)** An ETL model is generatable by a state-dependent DEL protocol iff it satisfies Propositional Stability, Synchronicity, Perfect Recall and Local No Miracles.

9.3 Discussion and Comparison of DEL Protocols and DEL Dynamical Systems

With results established for both DEL dynamical systems and extensional DEL protocols, these may now be compared, first on a technical level concerning the induced properties, and second from a modeling perspective.

For both DEL dynamical systems and DEL protocols the generated ETL model satisfies the core DEL properties Synchronicity, Perfect Recall and Local No Miracles. This comes as no surprise, as these properties—as was mentioned in

[18]Equivalently in the current setting would be action models with $\mathsf{post}(\sigma) = \top$ for all events σ.

Section 4.2—stem from the very nature of product update. Beyond these, however, differences emerge:

Connected Time-Steps:

An ETL model generated using a single DEL dynamical system has connected time-steps as a consequence of using the restriction to connected components. This property does not survive model union, and is therefore not inherited by ETL models generated by families of DEL dynamical systems, cf. Remark 5. DEL protocols do not induce the property in generated ETL models, irrespective of whether such are defined using a restriction to connected components or not: DEL protocols may contain several sequences of action models, producing disjoint new time steps.

Conceptually, as an additional requirement on ETL models, Connected Time-Steps adds nothing not already inherent in the standard modal logical approach to agency: Using relational semantics, nothing disconnected from the designated state impacts the satisfaction of formulas, and hence neither does it affect the modeled agents.

Finite Executions vs. Precondition Describable:

Finite Executions (referred to as "the finiteness assumption" in [15]) is meant to ensure the existence of the precondition formula of the action model event σ_e for each ETL event e. Thus, it shares a role with the abstract Precondition Describable, but is weaker than this direct existence requirement. It is conjectured that a compilation error occurred post-submission of [15], omitting further requirements.[19]

Propositional Stability vs. Postcondition Describable:

That the theorems of van Benthem and co-authors include Propositional Stability is a result of their use of action models without postconditions. DEL dynamical systems limited to complex model transformers built over the same class of action models would generate ETL models also satisfying this property. Conversely, it is hypothesized that any ETL model generated by a DEL protocol defined over action models with postconditions would satisfy the abstract requirement of being Postcondition Describable by exhibiting only finite atomic change between successive histories.

Component Collection Describable:

The Component Collection Describable requirement ensures the existence of suitable preconditions for the designated actions of the multi-pointed action model underlying the clean map, which control the temporal flow of the dynamical system when

[19]Finite Executions is not enough to guarantee the existence of suitable preconditions formulas: Let a single-agent ETL model $\mathcal{H}$ be given with histories of length 1 divided into two disconnected $\mathtt{R}_1$-components, H_1 and H_1' with $e \in H_1$, and $e' \in H_1'$. Let the sub-model H_1, H_1' be non-image-finite and non-bisimilar but let (H_1, e) (H_1', e') be modally equivalent. Such pointed Kripke model exist, cf. e.g. [17, Ex. 2.23, p. 68]. Let the set of histories of length 2 be given by $\{ee^*\}$ and let $\mathcal{H}$ contain no further histories. Then $\mathcal{H}$ satisfies Finite Executions (and the other properties), but there exists no suitable precondition formula for σ_{e^*} as e and e' are modally equivalent, but e^* only executed on e. An additional requirement of image-finiteness would solve this problem.

seeking to build a particular ETL model. This is not needed when working with DEL protocols, as the temporal occurrence of events is exogenously given. The requirement is not inherited by every ETL model build from a DEL dynamical system, but is implied when the system is aptly finite.

Conceptually, Component Collection Describable may be seen as an *internalization requirement* on dynamic development: It requires that any transformation that is executed can be given an "explanation" within the model, in the sense that there exists a formula describing exactly those connected components in the ETL model where that transformation occurs. As the existence of such "explanations" for behavior is a prerequisite for any form of rationalizability, we find that this additional restriction is, in spirit, implied by the very idea of working with logical (rational) agents.

Local vs. Point Bisimulation Invariance:

Whether generated by a uniform DEL protocol, single DEL dynamical system or a family of DEL dynamical systems, the resulting ETL model satisfies Local Bisimulation Invariance. This is due to the nature of preconditions in product update. Any saturated ETL model generated by a single DEL dynamical system satisfies the stronger property of Point Bisimulation Invariance, which also involves a temporal component, reflecting the fact that clean maps are mappings acting on the points of pointed Kripke models, and hence output equivalent values given equivalent inputs. As the temporal invariance is defined on points, it is lost when moving to unsaturated models, exactly as these are unpointed. In contrast, as DEL protocols react to an external clock rather than to the structure of the current pointed Kripke model, such protocols do not induce this strong version of bisimulation invariance.

Conceptually, the temporal invariance of Point Bisimulation Invariance represents a *behavioral uniformity assumption*: An agent defined by an intensional protocol will perform the same action in any two bisimilar situations. With bisimilarity implying modal equivalence and agents' reasoning capabilities given by the modal language used to describe their circumstances, the uniformity assumption then enforces that agents base their decisions fully on information they can explicitly reason about. Contraposed: If an agent varies its action, this must be caused by a change in circumstances perceivable by the agent (i.e., expressible in its language). In modeling agency, we do not find the assumption that circumstances determine decision unduly strong. Rather, we would argue, the assumption is a prerequisite for categorizing the resulting behavior as rational.[20]

10 Conclusion

Logical modeling of dynamics in multi-agent systems relies on protocols as control mechanisms. With multiple protocol frameworks available, the question naturally

[20]Compare to extensive games with imperfect information, where it is, as standard, assumed that agents have *knowledge of their own actions*, i.e., that if an agent cannot distinguish between two nodes, the agent will choose the same action in the two nodes. In [49], it is assumed by definition. For a discussion, see [28] where the requirement is denoted the *Ex Interim Condition*.

arises which, if any, is better suited for a given modeling task. In choosing a class of protocols in which to cast a model, an implicit choice of agency, actions and dynamics is thus made. In this paper, an implementation of intensional protocols as DEL dynamical systems has been investigated. On the technical side, the type of epistemic temporal models that DEL dynamical systems may generate have been characterized. Conceptually, DEL dynamical systems *qua* intensional protocols have been compared to the main protocol framework in dynamic epistemic logic, namely the extensional DEL protocols of [15]. In summary, extensional DEL protocols are convenient for encoding extensional protocols: In cases where one wishes to answer a question concerning how a particular sequence of actions will influence a given initial model, then directly specifying that sequence of actions is a straightforward formalization. In contrast, it is not possible to run dynamics on an external clock using DEL dynamical systems.[21] In cases where one seeks to model an intensional protocol, possibly applicable to more than a single initial model, then DEL dynamical systems enjoys a particular benefit: As exemplified, intensional natural language protocols may in a natural way be encoded as clean maps. Further, as shown by the Muddy Children example, a clean map may be a vastly smaller representation of the intended protocol than any extensional DEL protocol counterpart. Finally, in relation to ETL model generation, then DEL dynamical systems do not impose restrictions over and above what one may expect from an intensional protocol framework. The new, main conceptual restriction—that circumstances determine decision—may even be desirable when modeling agents that base their decisions fully on information they can explicitly reason about through the formal language. Given the popularity of intensional protocols in other multi-agent paradigms, it is surprising that they have not previously been systematically investigated for dynamic epistemic logic.

There is a range of open questions that we find highly interesting. The relation to automata theory (cf. Section 1.2) seems a leading candidate for wider and deeper semantic appreciation. Results on when (and how) DEL dynamical systems can obtain various automata representations could possibly allow for a comparison both to orbit results on topological DEL dynamical systems [39, 40, 57], but also to work on Grand Stage models [45, 47]. As that branch of literature is rich with results on axiomatizations and complexity results, a tighter semantic connection may possibly facilitate a partial result transfer, with one aim being axiomatizations of epistemic temporal logics for classes of ETL models generated by particular types of DEL dynamical systems, akin to the result on temporal public announcement logic of [15].

Acknowledgments The Center for Information and Bubble Studies is funded by the Carlsberg Foundation. The contribution of R.K. Rendsvig to the research reported in this article was also funded by the Swedish Research Council through the framework project 'Knowledge in a Digital World' (Erik J. Olsson, PI): A previous version of this paper occurs in the thesis [57]. We thank Alexandru Baltag, Johan van Benthem, Thomas Bolander, Vincent F. Hendricks and Dominik Klein for valuable comments on elements of this paper, as well as the participants of the 2016 CADILLAC and the 2016 LogiCIC workshops, held in Christiania, Denmark, and Amsterdam, the Netherlands, respectively.

[21] A clock may however be build into DEL dynamical system: This may be done by working in an extended language with atomic propositions denoting the current time and using postconditions to make time run. For finite time sequences, this may be encoded using a finite model.

Appendix: Proofs

Proposition 1 Let (X, f, x) be a pointed DEL dynamical system given by multi-pointed action model $\Sigma\Gamma$. Then there exists a singleton uniform DEL protocol that produces the orbit of f from x.

Proof For each $k \in \mathbb{N}$, let $\sigma_k \in \Gamma$ be such that $f^k(x) \models \mathsf{pre}(\sigma_k)$. As $\Sigma\Gamma$ is X-deterministic, for each k there is at most one such σ_k. Define a uniform DEL protocol p as the smallest protocol for which $\mathsf{p}_k(s) = \Sigma\sigma_k$ (for all $s \in x$) whenever σ_k exists. Then when p is sequentially applied to x using product update, it produces the sequence $\langle f^k(x)\rangle_{k\in\mathbb{N}}$ of pointed Kripke models (up to the deletion of redundant states not connected to the designated states, cf. Section 2). □

Proposition 2 If saturated ETL model $(\mathcal{H}, \underline{H})$ is generated by a pointed DEL dynamical system, then $(\mathcal{H}, \underline{H})$ satisfies seven of the eight properties of Section 4.2, namely Synchronicity, Connected Time-Steps, Perfect Recall, Local No Miracles, Precondition Describable, Precondition Describable, and Point Bisimulation Invariance.

Proof Let (X, f, x) be a pointed DEL dynamical system with orbit $(f^k(x))_{k\in\mathbb{N}}$. The **length** of a state s in $f^k(x)$ is $len(s) := k + 1$.

Let $(\mathcal{H}, \underline{H})$ be the saturated ETL model generated by (X, f, x). Given the construction of γ in Def. 6, there exists a family of isomorphisms $\{g_k\}_{k\in\mathbb{N}}$ with each g_k mapping $[\![f^k(x)]\!]$ to $H_k := \{h \in H : len(h) = k\}$ satisfying $g_1(s) = e_s$ and $g_{k+1}((s,\sigma)) = g_k(s)e_\sigma$. Using this family, it is shown that $(\mathcal{H}, \underline{H})$ satisfies the listed properties in order:

Synchronicity. Assume for arbitrary $h, h' \in H$ that $h\mathtt{R}_i h'$. Then by the construction of the generated $\mathtt{R}_i$ (Def. 6), $\exists k \in \mathbb{N} : g_k^{-1}(h)R_i g_k^{-1}(h')$. Hence $g_k^{-1}(h), g_k^{-1}(h') \in [\![f^k(x)]\!]$. Thus, $len(g_k^{-1}(h)) = len(g_k^{-1}(h'))$. Hence, by the construction of g, $len(h) = len(h')$.

Perfect Recall. Assume for arbitrary $he, h'e' \in H$ that $he\mathtt{R}_i h'e'$. Then $\exists k \in \mathbb{N} : g_k^{-1}(he)R_i g_k^{-1}(h'e')$. By construction of f, $f^k(x) = C(f^{k-1}(x) \otimes \Sigma\Gamma)s'$ for $\Sigma\Gamma$ the multi pointed action model. As $g_k^{-1}(he)R_i g_k^{-1}(h'e')$, by definition of $\otimes$ and clean maps, $g_{k-1}^{-1}(h)R_i g_{k-1}^{-1}(h')$. Hence, by definition of $\mathtt{R}_i$, it follows that $h\mathtt{R}_i h'$.

Local No Miracles. Assume that 1) $h\mathtt{R}_i h'$, 2) $h_1 e\mathtt{R}_i h_2 e'$ and 3) $h\mathtt{R}^* h_1$ for arbitrary $he, h'e', h_1 e, h_2 e' \in H$. 1) implies that 1*) $g_k^{-1}(h)R_i g_k^{-1}(h')$ for $k = len(h)$. 3) implies that $len(h) = len(h_1)$ by Synchronicity. In conjunction with 2), this implies that 2*) $g_{k+1}^{-1}(h_1 e)R_i g_{k+1}^{-1}(h_2 e')$.

By construction of f, $f^{k+1}(x) = C(f^k(x) \otimes \Sigma\Gamma)s'$. By 2*) and the definition of $\otimes$ and clean maps, there must be 4) $\sigma_e, \sigma_{e'} \in \Sigma\Gamma$ such that $\sigma_e \mathsf{R}_i \sigma_{e'}$, for σ_e the σ such that $g_{k+1}((s,\sigma)) = h_1 e$, for some $s \in [\![f^k(x)]\!]$, and $\sigma_{e'}$ the σ' such that $g_{k+1}((t,\sigma')) = h_2 e'$, for some $t \in [\![f^k(x)]\!]$.

Now assume that $(g_k^{-1}(h), \sigma_e), (g_k^{-1}(h'), \sigma_{e'}) \in [\![f^{k+1}(x)]\!]$. Then 1*), 4) and Def. $\otimes$ jointly imply that $(g_{k+1}^{-1}(h), \sigma_e) R_i (g_{k+1}^{-1}(h'), \sigma_{e'})$. By the definition of the generated $\mathtt{R}_i$, it thus follows that $he\mathtt{R}_i h'e'$.

Connected Time-Steps. For arbitrary $h, h' \in H$ assume $len(h) = len(h')$. Let $k \in \mathbb{N}$ such that $h, h' \in H_k$. Then $g_k^{-1}(h), g_k^{-1}(h') \in [\![f^k(x)]\!]$. By definition of product update $\otimes$, clean maps and the fact that x is connected, $g_k^{-1}(h) R^* g_k^{-1}(h')$. By definition of $(\mathcal{H}, \underline{H})$ it follows that $h\mathtt{R}^*h'$.

Precondition Describable. For arbitrary $e \in E$, let $\delta_e = \mathsf{pre}(\sigma_e)$. Recall that by definition of $\otimes$, $\forall k \in \mathbb{N}$, $(g_k^{-1}(h), \sigma_e) \in [\![f^{k+1}(x)]\!]$ iff $g_k^{-1}(h) \models \mathsf{pre}(\sigma_e)$ and $\sigma_e \in \Sigma^{k+1}$ $(*)$. Assume $\exists h' \in H : h'e \in H$ and let $h \in H$ such that $h'\mathtt{R}^*h$.

$\Rightarrow$: Assume for some $k \in \mathbb{N}$ that $(H_k, h) \models \delta_e$. Then $g_k^{-1}(h) \models \mathsf{pre}(\sigma_e)$. By assumption, $\sigma_e \in \Sigma_{k+1}$. By $(*)$, thus $(g_k^{-1}(h), \sigma_e) \in [\![f^{k+1}(x)]\!]$. Therefore, $he \in H$.

$\Leftarrow$: Assume $he \in H$. Then for some $k \in \mathbb{N}$, $g_{k+1}^{-1}(he) = (g_k^{-1}(h), \sigma_e) \in [\![f^{k+1}(x)]\!]$. By $(*)$, $g_k^{-1}(h) \models \mathsf{pre}(\sigma_e)$. And thus $h \models \delta_e$.

Precondition Describable. For arbitrary $he \in H$ (in specific for some $k \in \mathbb{N}$, $he \in H_{k+1}$) let $\delta_{D_e} = \mathsf{post}(\sigma_e)$ where $D_e = D_1 \cup D_2$, for $p, q \in \Phi$ such that $D_1 = \{p : h \notin V(p), he \in V(p)\}$ and $D_2 = \{\neg q : h \in V(q), he \notin V(q)\}$.

Consider an arbitrary $p \in D_1$. Then by definition of the generated ETL model $(\mathcal{H}, \underline{H})$, $g_k^{-1}(h) \notin [\![p]\!]_k$ and $g_{k+1}^{-1}(he) \in [\![p]\!]_{k+1}$ $(*)$. By construction, $g_{k+1}^{-1}(he) = (g_k^{-1}(h), \sigma)$ for some $\sigma \in \nabla_{k+1}$ $(**)$. By definitions of $(\mathcal{H}, \underline{H})$ and $\otimes$, $(g_k^{-1}(h), \sigma) \in [\![p]\!]_{k+1}$ iff $\mathsf{post}(\sigma) \models p$. Then, by $(*)$ and $(**)$, $\mathsf{post}(\sigma) \models p$. As $p \in D_1$ was arbitrary, $\mathsf{post}(\sigma) \models D_1$.

The argument for $\mathsf{post}(\sigma) \models D_2$ is identical. Conclude that $\mathsf{post}(\sigma) \models D_e$ and thus $\delta_{D_e} \models D_e$.

Point Bisimulation Invariance. Let arbitrary $C(\mathcal{H}h)\underline{h} = (H_k, \underline{h})$ and $C(\mathcal{H}h')\underline{h'} = (H_l, \underline{h'})$ be such that $(H_k, \underline{h}) \leftrightarroweq (H_l, \underline{h'})$. Hence $f^k(x) \leftrightarroweq f^l(x)$ $(*)$.

Further, assume for arbitrary $h \in H_k$ and $h' \in H_l$ that $(H_k, h) \leftrightarroweq (H_l, h')$.

$\Rightarrow$: Assume $he \in H$. By construction of g and definition of clean maps, both H_k and H_l are connected components, i.e, $\forall h, h' \in H_k : h\mathtt{R}^*h'$ and idem for H_l. By the Hennessy-Milner Theorem (see Section 7), it follows that h and h' satisfy exactly the same modal formulas. Hence, by construction of g and the definition of $\mathcal{H}$, $g_k^{-1}(h)$ and $g_l^{-1}(h')$ satisfy exactly the same modal formulas as well. Now as $he \in H$, $g_{k+1}^{-1}(he) \in [\![f^{k+1}(x)]\!]$ and thus $g_k^{-1}(h) \models \mathsf{pre}(\sigma_e)$. Hence $g_l^{-1}(h') \models \mathsf{pre}(\sigma_e)$ $(**)$. By $(*)$ and $(**)$, it follows that $(g_l^{-1}(h'), \sigma_e) \in [\![f^{l+1}(x)]\!]$. Hence $h'e \in H$.

$\Leftarrow$: By the same argument, $h'e \in H$ implies $he \in H$.

This concludes the proof of Proposition 2. □

Proposition 3 Not all saturated ETL models generated by DEL dynamical systems are Component Collection Describable.

Proof Let Ms and f be as in Fig. 7 (cf. [56]) and X be the orbit of f from Ms. Then the ETL model generated by the DEL dynamical system (X, f) from initial model Ms is not Component Collection Describable.
Consider $A = \{f^n(Ms) : n \text{ is even}\}$. There does not exist a φ such that for all $x \in X$, $x \models \varphi$ iff $x \in A$: Assume the modal depth of φ is k. Let $m > k$. Then $f^m(Ms) \models \varphi$ iff $f^{m+1}(Ms) \models \varphi$ as two such models will not differ in the first $m + 1$ relational steps from the point. Hence the ETL model generated by (X, f) from Ms is not Component Collection Describable. □

Proposition 4 If $(\mathcal{H}, \underline{H})$ is a saturated ETL model that satisfies all eight properties of Section 4.2, then there exists a pointed DEL dynamical system that generates $(\mathcal{H}, \underline{H})$.

Proof Proposition 4 is shown by constructing a DEL dynamical system (X, f) with f the clean map of a X-deterministic multi-pointed action model $\Sigma\Gamma$ and an initial Kripke model $x \in X$ such that the saturated ETL model $(\mathcal{H}', \underline{H}') = (E', H', \mathrm{R}', V', \underline{H}')$ generated by (X, f) from x is ETL isomorphic to $(\mathcal{H}, \underline{H})$. The latter is shown by induction on $len(h)$ of $h \in \mathcal{H}$ for a map $\gamma^* : E \longrightarrow E'$. As in Def. 5, for $h = e_0...e_n$, write $\gamma^*(h) := \gamma^*(e_0)...\gamma^*(e_n)$.

As $(\mathcal{H}, \underline{H})$ satisfies property Connected Time-Steps, the ETL model is a saturated component branch. To emphasize this, in this proof $(\mathcal{H}, \underline{H})$ is written $(\mathcal{H}_{\mathbf{b}}, \underline{H})$.

1. **Initial Kripke model**

To obtain a practical and consistent naming of states, the initial Kripke model $x = (\llbracket x \rrbracket, R, \llbracket\cdot\rrbracket, s)$ is set to be a re-naming of the initial component of $\mathbf{b}$: Let $\llbracket x \rrbracket = \{\sigma_e : e \in \mathbf{b}_0\underline{h}\}$. For the relations and valuation of the initial model, simply copy over the relations and valuation from the initial component of $\mathbf{b}$: For all $i \in \mathcal{A}$, let $\sigma_e R_i \sigma_{e'}$ iff $e\mathrm{R}_i e'$, and for all $p \in \Phi$, let $\sigma_e \in \llbracket p \rrbracket$ iff $e \in V(p)$. Finally, let the point of x be the copy of the point of $\mathbf{b}_0\underline{h}$: Let $s = \sigma_{\underline{h}}$.

2. **Constructing** (X, f)

To define the DEL dynamical system (X, f), first construct a multi-pointed action model $\Sigma\Gamma = (\llbracket\Sigma\rrbracket, \mathsf{R}, pre, post, \Gamma)$. In words, construct $\Sigma\Gamma$ so that for each time-step $\mathbf{b}_k$ of the component branch $\mathbf{b}$, Γ contains a designated action $\underline{\sigma}_k$ connected to a set of actions $\llbracket\Sigma_k\rrbracket \subseteq \llbracket\Sigma\rrbracket$ such that the single -pointed action model $\Sigma'_{\{\underline{\sigma}_k\}}$ obtained

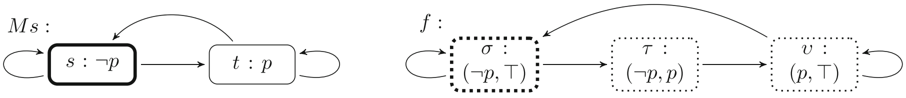
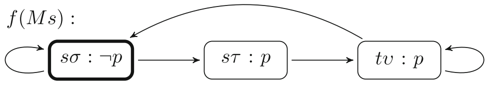


Fig. 7 Initial Kripke model Ms and pointed action model. The orbit of f from Ms produces non-bisimilar models forever: the unique state not satisfying p will split, inserting a new p-state as it's child with τ; any other state gets exactly one child:



from restricting Σ to $[\![\Sigma_k]\!]$ produces the equivalent of $\mathbf{b}_{k+1}\underline{h}$ from the Kripke model-equivalent of $\mathbf{b}_k\underline{h}$. In the precondition of $\underline{\sigma}_k$, include a formula $\delta_{\mathbf{b}_k\underline{h}}$ characterizing $\mathbf{b}_k\underline{h}$. As $(\mathcal{H}_\mathbf{b}, \underline{H})$ is Component Collection Describable by assumption, such a formula exists.

Formally, construct $\Sigma\Gamma$ piece-wise as follows: Let $\mathbf{b}_k\underline{h}$ and $\mathbf{b}_{k+1}\underline{h}$ be given. $\Sigma_{k+1}\sigma$ is constructed such that $C(f^k(x) \otimes \Sigma_k\sigma)s'$ mirrors the structure of $\mathbf{b}_{k+1}\underline{h}$:

Let the single-pointed action model $\Sigma_{k+1}\underline{\sigma}_{k+1}$ be $([\![\Sigma_{k+1}]\!], \mathsf{R}_{k+1}, \mathsf{pre}_{k+1}, \mathsf{post}_{k+1}, \underline{\sigma}_{k+1})$, given by

$[\![\Sigma_{k+1}]\!] = \{\sigma_e : he \in \mathbf{b}_{k+1}\}$ with $\underline{\sigma}_{k+1} = \sigma_e$ such that $\exists h : he \in \mathbf{b}_{k+1} \cap \underline{H}$.

$(\sigma_e, \sigma_{e'}) \in \mathsf{R}_i$ iff $\exists he, h'e' \in \mathbf{b}_{k+1} : (he, h'e') \in \mathrm{R}_i$.

$$pre(\sigma_e) = \begin{cases} \delta_e & \text{if } \sigma_e \neq \underline{\sigma}_{k+1} \\ \delta_e \wedge \delta_{\mathbf{b}_k\underline{h}} & \text{else} \end{cases}$$

with δ_e and $\delta_{\mathbf{b}_k\underline{h}}$ given by Precondition Describable and Component Collection Describable, respectively.

$\mathsf{post}(\sigma_e) = \delta_{D_e}$ as given Postcondition Describable.

Let the multi-pointed action model $\Sigma\Gamma = ([\![\Sigma]\!], \mathsf{R}, pre, post, \Gamma)$ be given by, for $\ast \in \{[\![\Sigma]\!], \mathsf{R}, pre, post, \}$, $\ast = \bigcup_{k:\mathbf{b}_k\in\mathbf{b}} \ast_k$ and $\Gamma = \bigcup_{k:\mathbf{b}_k\in\mathbf{b}}\{\underline{\sigma}_k\}$. This is well-defined: For pre and $post$, this follows from Precondition Describable and Postcondition Describable.

Let X be the closure of $\{x\}$ under the operation $\otimes\Sigma\Gamma$. On X, $\Sigma\Gamma$ is guaranteed to be deterministic as any two characteristic formulas $\delta_{\mathbf{b}_k\underline{h}}$ and $\delta'_{\mathbf{b}_{k'}\underline{h'}}$ are not simultaneously satisfiable. Finally, let f be the clean map of $\Sigma\Gamma$ on X. Then (X, f) is a DEL dynamical system with $x \in X$.

3. **Constructing the Isomorphism**

Let $(\mathcal{H}', \underline{H}') = (E', H', \mathrm{R}', V', \underline{H}')$ be the saturated ETL model generated by (X, f) from x. Define the two mappings: $\gamma^\dagger : E \longrightarrow [\![\Sigma]\!]$ with $\gamma^\dagger(e) = \sigma_e$ and $\gamma : [\![\Sigma]\!] \longrightarrow E'$ for $\gamma(\sigma) = e_\sigma$ cf. Def. 6. From these, define $\gamma^* : E \longrightarrow E'$ as $\gamma^* := \gamma \circ \gamma^\dagger$.

Define subsets of E based on history length: For all $k \in \mathbb{N}$ let $E_0 = \{e : e \in \mathbf{b}_0\}$ and $E_{k+1\geq 1} = \{e : h \in \mathbf{b}_k, he \in \mathbf{b}_{k+1}\} \subseteq E$. Let $E'_k, k \in \mathbb{N}$, be given *mutatis mutandis*. Let $\gamma_k^\dagger : E_k \longrightarrow [\![\Sigma_k]\!]$, $\gamma_k : [\![\Sigma_k]\!] \longrightarrow E'_k$ and $\gamma_k^* : E_k \longrightarrow E'_k$ be the restrictions of $\gamma^\dagger, \gamma$ and γ^* to $E_k \times [\![\Sigma_k]\!]$, $[\![\Sigma_k]\!] \times E'_k$ and $E_k \times E'_k$, respectively. Then, of course, $\gamma^* = \bigcup_{k\in\mathbb{N}} \gamma_k^*$, whereby $\gamma^*(e) = e'_{\sigma_e}$. By induction on $len(h)$, $h \in \mathcal{H}$, it is now shown that γ^* is an ETL isomorphism.

Claim: The map γ^* is a bijection. By the construction of $\gamma^\dagger$ and γ, each γ_k^* is an injection: if $e \neq e'$, then $\gamma_k^*(e) \neq \gamma_k^*(e')$. By construction, it is also guaranteed that γ_k^* is a surjection: $\forall e' \in E' \exists e \in E : \gamma_k^*(e) = e'$. Hence for each $k \in \mathbb{N}$, γ_k^* is a bijection.

Furthermore, γ^* is a total map: if $e \in E_k \cap E_m$, then $\gamma_k^\dagger(e) = \gamma_m^\dagger(e)$ (i.e., $\sigma_e \in [\![\Sigma_k]\!] \cap [\![\Sigma_m]\!]$) and $\gamma_k(\sigma_e) = \gamma_m(\sigma_e)$. Thus $\gamma \circ \gamma^\dagger(e)$ is well-defined and in $E'_k \cap E'_m$.

Finally, γ^* inherits injectivity and surjectivity from its restrictions. Hence, the map γ^* is a bijection.

Claim: The map γ^* is an ETL isomorphism. The claim is shown by 4 inductive sub-proofs.

1) *Domain and Temporal Structure.*
Base. Let $h \in H_0$. This is the case iff $\gamma^\dagger(h) \in f^0(x)$ (by construction of initial Kripke model) iff $\gamma \circ \gamma^\dagger(h) \in H'_0$ (by Def. 6).
Step. It is shown that $he \in H_{k+1}$ iff $\gamma^*(he) \in H'_{k+1}$.
$\Rightarrow$: Assume $he \in H_{k+1}$. Then $h \in H_k$. By the induction hypothesis, $\gamma^\dagger(h) \in \llbracket f^k(x) \rrbracket$. By construction of Σ_{k+1}, $\gamma^\dagger(e) \in \llbracket \Sigma_{k+1} \rrbracket$. By the same construction and Precondition Describable, $\gamma^\dagger(h) \models \mathsf{pre}(\sigma_e)$. Hence $(\gamma^\dagger(h), \gamma^\dagger(e)) \in \llbracket f^{k+1}(x) \rrbracket$. By Def. 6, in particular the construction of H_{k+1}, $\gamma((\gamma^\dagger(h), \gamma^\dagger(e))) \in H'_{k+1}$.
$\Leftarrow$: Assume $\gamma((\gamma^\dagger(h), \gamma^\dagger(e))) \in H'_{k+1}$. Then $(\gamma^\dagger(h), \gamma^\dagger(e)) \in \llbracket f^{k+1}(x) \rrbracket$ by Def. 6, so $\gamma^\dagger(h) \in \llbracket f^k(x) \rrbracket$ and $\gamma^\dagger(e) \in \llbracket \Sigma_{k+1} \rrbracket$. By the induction hypothesis, $h \in H_k$. If $he \notin H_{k+1}$, a contradiction is reached: $\mathsf{pre}(\gamma^\dagger(e))$ is satisfied by exactly those $\gamma^\dagger(h) \in \llbracket f^k(x) \rrbracket$ such that $(\gamma^\dagger(h), \gamma^\dagger(e)) \in \llbracket f^{k+1}(x) \rrbracket$ – by the construction of action models in this proof and as $(\mathcal{H}_\mathbf{b}, \underline{H})$ is Precondition Describable. So $he \in H_{k+1}$.

2) *Epistemic relations.*
Base. It follows by construction of initial Kripke model and Def. 6.
Step. It is shown that $\forall he, h'e' \in \mathbf{b}_{k+1}$, $he\mathrm{R}_i h'e'$ iff $\gamma^*(he)\mathrm{R}'_i\gamma^*(h'e')$.
$\Rightarrow$: Assume that $he\mathrm{R}_i h'e'$. By Perfect Recall, $h\mathrm{R}_i h'$. By the induction hypothesis, $\gamma^\dagger(h) R_i \gamma^\dagger(h')$. By construction of Σ_{k+1}, $\gamma^\dagger(e)\mathsf{R}_i\gamma^\dagger(e')$. By definition of $\otimes$, $(\gamma^\dagger(h), \gamma^\dagger(e)) R_i (\gamma^\dagger(h'), \gamma^\dagger(e'))$. By Def. 6, $\gamma((\gamma^\dagger(h), \gamma^\dagger(e)))\mathrm{R}_i\gamma((\gamma^\dagger(h'), \gamma^\dagger(e')))$.
$\Leftarrow$: Assume $\gamma((\gamma^\dagger(h), \gamma^\dagger(e)))\mathrm{R}_i\gamma((\gamma^\dagger(h'), \gamma^\dagger(e')))$. By Def. 6, $(\gamma^\dagger(h), \gamma^\dagger(e)) R_i (\gamma^\dagger(h'), \gamma^\dagger(e'))$. By definition of $\otimes$, both $\gamma^\dagger(h) R_i \gamma^\dagger(h')$ and $\gamma^\dagger(e)\mathsf{R}_i\gamma^\dagger(e')$. So by construction of Σ_{k+1}, $\exists h_1 e, h_2 e' \in \mathbf{b}_{k+1} : h_1 e\mathrm{R}_i h_2 e'$. By the induction hypothesis, $h\mathrm{R}_i h'$. Further, note that $\forall h, h' \in \mathbf{b}_k : h\mathrm{R}^* h'$. Hence, by Local No Miracles, $he\mathrm{R}_i h', e'$.

3) *Valuation.*
Base. It follows by construction of initial Kripke model and Def. 6.
Step. It is shown that $\forall he \in \mathbf{b}_{k+1}$, $he \in V(p)$ iff $\gamma(he) \in V'(p)$.
$\Rightarrow$: Assume $he \in V(p)$. Either i) $h \in V(p)$ or ii) $h \notin V(p)$. If i), then by the induction hypothesis, $\gamma \circ \gamma^\dagger(h) \in V'(p)$. By construction of Σ_{k+1}, $\mathsf{post}(\gamma^\dagger(e)) \not\models \neg p$. Hence $(\gamma^\dagger(h), \gamma^\dagger(e)) \in \llbracket p \rrbracket_{k+1}$. By Def. 6, $\gamma((\gamma^\dagger(h), \gamma^\dagger(e))) \in V'(p)$. If ii), then by the induction hypothesis, $\gamma^\dagger(h) \notin \llbracket p \rrbracket_k$. As $(\mathcal{H}_\mathbf{b}, \underline{H})$ is Postcondition Describable, by construction of Σ_{k+1}, $\mathsf{post}(\gamma^\dagger(e)) \models p$. Thus $(\gamma^\dagger(h), \gamma^\dagger(e)) \in \llbracket p \rrbracket_{k+1}$. By Def. 6, $\gamma((\gamma^\dagger(h), \gamma^\dagger(e))) \in V'(p)$.
$\Leftarrow$: Assume $\gamma((\gamma^\dagger(h), \gamma^\dagger(e))) \in V'(p)$. Then $(\gamma^\dagger(h), \gamma^\dagger(e)) \in \llbracket p \rrbracket_{k+1}$ by Def. 6. Again, either i) $\gamma^\dagger(h) \in \llbracket p \rrbracket_k$ or ii) $\gamma^\dagger(h) \notin \llbracket p \rrbracket_k$. If i), then by the

induction hypothesis, $h \in V(p)$. For a contradiction, suppose $he \notin V(p)$. Then $\mathsf{post}(\gamma^{\dagger}(e)) \models \neg p$. But by construction of Σ_{k+1}, $\mathsf{post}(\gamma^{\dagger}(e)) \not\models \neg p$. This is a contradiction. Hence $he \in V(p)$. If ii), then by the definition of $\otimes$, $\mathsf{post}(\gamma^{\dagger}(e)) \models p$. By the induction hypothesis, $h \notin V(p)$. If it was the case that $he \notin V(p)$, then $\mathsf{post}(\gamma^{\dagger}(e)) \not\models p$. Contradiction. Thus $he \in V(p)$.

4) *Points.*
Base. It follows by construction of initial Kripke model and Def. 6.
Step. It is shown that $\underline{he} \in \underline{H}_{k+1}$ iff $\gamma^{*}(\underline{he}) \in \underline{H}'_{k+1}$. Let $f^{k}(x) = Nt$ and $f^{k+1}(x) = Ms$.

$\Rightarrow$: Assume $\underline{he} \in \underline{H}_{k+1}$. By the induction hypothesis, $\gamma^{\dagger}(\underline{h}) = t$. By saturation of $\mathbf{b}_k$, $\exists e \in E : \underline{h}e \in \mathbf{b}_{k+1} \cap \underline{H}$. By construction of Σ_{k+1}, $\gamma^{\dagger}(e) \in [\![\Sigma_{k+1}]\!]$ and, as $(\mathcal{H}_{\mathbf{b}}, \underline{H})$ is Precondition Describable, $\gamma^{\dagger}(\underline{h}) \models \mathsf{pre}(\gamma^{\dagger}(e))$. By construction of f, $f^{k+1}(x) = C(Nt \otimes (\Sigma_{k+1}, \gamma^{\dagger}(e))) = C(N \otimes \Sigma_{k+1}, (t, \gamma^{\dagger}(e))) = Ms$. By Def. 6, $\gamma(s) = \gamma(\gamma^{\dagger}(\underline{he})) \in \underline{H}'_{k+1}$.

$\Leftarrow$: Assume $\gamma(\gamma^{\dagger}(\underline{he})) \in \underline{H}'_{k+1}$. By Def. 6, $f^{k+1}(x) = (C(N \otimes \Sigma_{k+1}), (t, \gamma^{\dagger}(\underline{e})))$. By induction hypothesis, $\gamma^{\dagger-1}(t) = h \in \underline{H}_k$. By construction of the Action Model, $\gamma^{\dagger}(\underline{e}) = \gamma^{-1}(e)$ for e such that $\exists h' : h'e \in \underline{H}_{k+1}$ and $h' \in \underline{H}_k$. As points in H_m are unique for all m by Def. 7, h' must be h. Thus $he \in \underline{H}_{k+1}$.

This concludes the proof of Proposition 4.[22] □

Lemma 1 If there exists an image-finite and concluding pointed DEL dynamical system that generates $(\mathcal{H}, \underline{H})$, then $(\mathcal{H}, \underline{H})$ is image-finite and concluding.

Proof As the DEL dynamical system is image-finite, for all $k \in \mathbb{N}$ with $f^{k}(x) = Ms$, R_i is image-finite (for all $i \in I$). The construction of the generated ETL model (see Definition 6) ensures that for all H_k, R_i is image-finite (for all $i \in I$,). Hence $(\mathcal{H}, \underline{H})$ is image-finite.

If the DEL dynamical system terminates, then there is a $k \in \mathbb{N}$ such that $f^{k}(x)$ is undefined. Let $f^{k-1}(x) = Ms$ and $\gamma(s) = \underline{h}$. As $f^{k}(x)$ is undefined, there is no σ such that $\gamma(s)\gamma(\sigma) \in H$. Hence, there is no e such that $\underline{h}e \in H$. Note that by construction of $\underline{\underline{H}}$, for all $\underline{h}' \in \underline{H}$: $\underline{h}' \sqsubseteq \underline{h}$. Thus, all $\underline{h}' \in \underline{H}$ are finite and hence $(\mathcal{H}, \underline{H})$ concludes.

If the DEL dynamical system is periodic, $f^{k}(x) = f^{k+m}(x)$ for some $k \geq 0$ and $m > 0$. Let $f^{k}(x) = Ms$ and $f^{k+m}(x) = M's'$, and let $\gamma(s) = \underline{h}$ and $\gamma(s') = \underline{h}'$. By construction, $\underline{h} \sqsubseteq \underline{h}'$. From $Ms = M's'$ it follows that $C(\mathcal{H}\underline{h})\underline{h} \leftrightarrows C(\mathcal{H}\underline{h}')\underline{h}'$. Thus, all $\underline{h}'' \in \underline{H}$ such that $\underline{h}'' \sqsubseteq \underline{h}$ are repeating. Furthermore, note that for all $n \in \mathbb{N}$, $f^{k+n}(x) = f^{k+m+n}(x)$. Now for arbitrary $n \in \mathbb{N}$, let $f^{k+n}(x) = M^{n}s^{n}$ and $f^{k+m+n}(x) = M^{n\prime}s^{n\prime}$, and let $\gamma(s^{n}) = \underline{h}^{n}$ and $\gamma(s^{n\prime}) = \underline{h}^{n\prime}$. By same argument as

[22]When constructing an action model that produces a pointed Kripke model isomorphic to a particular level in the to-be-generated ETL model, Proposition 3.2 of [21] (which states, roughly, that for almost any two pointed Kripke models, there exists an action model with postconditions that will produce one from the other using product update) is not applicable. The proposition is not applicable as the transforming action model allows only the designated state to survive, from which the desired Kripke model is unfolded. The resulting generated ETL models would therefore (most often) not be ETL isomorphic to the original ETL model, as ETL isomorphisms require that the temporal structure is preserved.

above, $\underline{h}^n$ is repeating. As n is arbitrary, all $\underline{h}'' \in \underline{H}$ such that $\underline{h}'' \sqsupseteq \underline{h}$ are repeating. Thus, all $\underline{h}'' \in \underline{H}$ are repeating and hence $(\mathcal{H}, \underline{H})$ concludes. □

Lemma 2 If a saturated ETL model $(\mathcal{H}, \underline{H})$ is image-finite, concluding and satisfies Connected Time-Steps, then $(\mathcal{H}, \underline{H})$ is Component Collection Describable.

Proof Let $(\mathcal{H}, \underline{H})$ be an image-finite and concluding saturated ETL model. Set $\mathcal{B} := \{C(\mathcal{H}h) : h \in \underline{H}\}$, the set of all connected components in $(\mathcal{H}, \underline{H})$. As all $C(\mathcal{H}h) \in \mathcal{B}$ are image-finite, by the Hennessy-Milner Theorem (see Section 7), for each pair $\underline{h}_1, \underline{h}_2 \in \underline{H}$ if $\underline{h}_1 \not\leftrightarrows \underline{h}_2$, there exists a formula $\varphi_{\underline{h}_1,\underline{h}_2}$ distinguishing between $\underline{h}_1$ and $\underline{h}_2$: $\underline{h}_1 \models \varphi_{\underline{h}_1,\underline{h}_2}$ while $\underline{h}_2 \not\models \varphi_{\underline{h}_1,\underline{h}_2}$.

Let $[C(\mathcal{H}\underline{h})\underline{h}]_{\leftrightarrows}$ be the equivalence class $\{C(\mathcal{H}\underline{h}')\underline{h}' \colon C(\mathcal{H}\underline{h}') \in \mathcal{B} \text{ and } \underline{h}' \leftrightarrows \underline{h}\}$. Then, since $(\mathcal{H}, \underline{H})$ is concluding, the set $\mathcal{B}_{\leftrightarrows} := \{[C(\mathcal{H}\underline{h})\underline{h}]_{\leftrightarrows} : \underline{h} \in \underline{H}, C(\mathcal{H}\underline{h}) \in \mathcal{B}\}$ is finite. Therefore, the conjunction $\bigwedge_{\underline{h}_2 \in \{\underline{h} \in \underline{H} \colon \underline{h} \not\leftrightarrows \underline{h}_1\}} \varphi_{\underline{h}_1,\underline{h}_2}$ is well-defined for any $\underline{h}_1 \in \underline{H}$. This conjunction distinguishes $\underline{h}_1$ from any point in $\underline{H}$ that is not bisimilar to $\underline{h}_1$. Denote this formula $\varphi_{\underline{h}_1}$.

Moreover, as $\mathcal{B}_{\leftrightarrows}$ is finite, all sets $A \subseteq \underline{H}$ for which $\underline{h} \in A$ and $\underline{h}' \leftrightarrows \underline{h}$ implies $\underline{h}' \in A$ are finite. Hence, for any such A, the disjunction $\bigvee_{\underline{h}_1 \in A} \varphi_{\underline{h}_1}$ is well-defined. This disjunction distinguishes the connected components in A from those not in A. □

Lemma 3 Let $\{(X_j, f_j)\}_{j\in J}$ be minimal in generating $\mathcal{H}$ and let (X_j, f_j, x_j) generate the saturated ETL model $(\mathcal{H}_j, \underline{H}_j)$. Then $(\mathcal{H}_j, \underline{H}_j)$ is the component branch sub-model for some saturated component branch $(\mathbf{b}, \underline{H})$ of $\mathcal{H}$.

Proof Let $\mathcal{H} = (E, H, \mathrm{R}, V)$ be generated by $\{(X_j, f_j)\}_{j\in J}$. Let $\mathbf{b}_j$ be the sequence of connected components $\mathbf{b}_{j,k} = C(\mathcal{H}, h)$ for all $h_k \in \underline{H}_j$, ordered on history length k. Suppose for proof by contradiction that $\mathbf{b}_j$ is both non-maximal and finite (cf. Def. 7). As $\mathbf{b}_j$ is finite, there is a $k \in \mathbb{N}$ such that for all $h \in \mathbf{b}_{j,k}$ there is no $e \in E$ such that $he \in \mathbf{b}_{j,k+1}$. However, as $\mathbf{b}_j$ is non-maximal, there is a $h \in \mathbf{b}_{j,k}$ and some $e \in E$ such that $\underline{he} \in H$. This implies that there is an ETL model $(\mathcal{H}_{j'}, \underline{H}_{j'})$ that extends $(\mathcal{H}_j, \underline{H}_j)$. Hence there is a DEL dynamical system $(X_{j'}, f_{j'})$, $j' \in J$, such that $f_j^n(x_j) = f_{j'}^n(x_j)$ for all $n \leq k-1$. Hence, (X_k, f_k) is redundant in generating $\mathcal{H}$, contradicting the assumption that $\{(X_j, f_j)\}_{j\in J}$ is minimal. Thus, $(\mathbf{b}_j, \underline{H}_j)$ is a saturated component branch that gives $(\mathcal{H}_j, \underline{H}_j)$. □

Theorem 2 Let an image-finite and concluding ETL model $\mathcal{H}$ be given. $\mathcal{H}$ is generatable up to ETL isomorphism by a family of image-finite and concluding pointed DEL dynamical systems, if, and only if, there exists a saturation of each component branch $\mathbf{b}$ of $\mathcal{H}$ such that $(\mathcal{H}_{\mathbf{b}}, \underline{H})$ satisfies all eight properties of Section 4.2.

Proof Left-to-right: Suppose $\mathcal{H}$ is generatable by a family of image-finite and concluding DEL dynamical systems $\{(X_j, f_j)\}_{j\in J}$. Let $(\mathcal{H}_j, \underline{H}_j)$ be the saturated ETL model generated by (X_j, f_j) (cf. Def. 6). By Lemma 3, $(\mathcal{H}_j, \underline{H}_j)$ is a component branch sub-model of $\mathcal{H}$ for some saturated component branch $(\mathbf{b}, \underline{H}_k)$ of $\mathcal{H}$. As $(\mathcal{H}_j, \underline{H}_j)$ was generated by an image-finite and concluding DEL dynamical system, the saturation $\underline{H}_j$ of $\mathcal{H}_j$ makes $(\mathcal{H}_j, \underline{H}_j)$ satisfy all 8 properties of Section 4.2 by Proposition 2 and Lemma 2.

Right-to-left: Let $\mathbf{B}$ be the set of all component branches of $\mathcal{H}$ and assume that for each $\mathbf{b} \in \mathbf{B}$, there exists a saturation such that $(\mathcal{H}_\mathbf{b}, \underline{H}_\mathbf{b})$ satisfies all eight properties of Section 4.2. As $\mathcal{H}$ is image-finite and concluding, also each $(\mathcal{H}_\mathbf{b}, \underline{H}_\mathbf{b})$ is image-finite and concluding. By the constructions in the proof of Prop. 4, it follows that each $(\mathcal{H}_\mathbf{b}, \underline{H}_\mathbf{b})$ is generatable up to ETL isomorphism by a DEL dynamical system that is image-finite and concluding.

Let $(X_\mathbf{b}, f_\mathbf{b}, x_\mathbf{b})$ be the pointed DEL dynamical system that generates $(\mathcal{H}_\mathbf{b}, \underline{H}_\mathbf{b})$ up to isomorphism, as given by the construction in the proof of Prop. 4. Let $(\mathcal{H}'_\mathbf{b}, \underline{H}'_\mathbf{b})$ be the specific ETL model generated by $(X_\mathbf{b}, f_\mathbf{b}, x_\mathbf{b})$, as given by Def. 6. It will now be shown that the union structure $U_\mathbf{B} = (E_\mathbf{B}, H_\mathbf{B}, \mathrm{R}_{\mathbf{B},i}, V_\mathbf{B})_{i\in I}$ of $\{(\mathcal{H}'_\mathbf{b}, \underline{H}'_\mathbf{b})\}_{\mathbf{b}\in\mathbf{B}}$ is isomorphic to $\mathcal{H} = (E, H, \mathrm{R}_i, V)_{i\in I}$. To this end, the existence of a bijection $g : H \longrightarrow H_\mathbf{B}$ will be shown.

Let $g_\mathbf{b}$ be the isomorphism between $(\mathcal{H}_\mathbf{b}, \underline{H}_\mathbf{b})$ and $(\mathcal{H}'_\mathbf{b}, \underline{H}'_\mathbf{b})$, guaranteed to exist by Prop. 4 and specifically given by 1) the construction of a DEL dynamical system from a saturated component branch of the proof of Prop. 4 and 2) the construction for generating an ETL model from a DEL dynamical system of Def. 6. Combining the state-history naming schemes used in these two constructions yield $g_\mathbf{b}$ given by

$$g_\mathbf{b}(h) = \begin{cases} e'_{\sigma_h} & \text{if } len(h) = 1 \\ g_\mathbf{b}(h^\dagger)e'_{\sigma_e} \text{ with } h = h^\dagger e & \text{else} \end{cases}$$

The history names from $\mathcal{H}_\mathbf{b}$ are hereby carried over as indices to the histories of $\mathcal{H}'_\mathbf{b}$.

Define the mapping $g : H \longrightarrow H_\mathbf{B}$ by $g(h) = g_\mathbf{b}(h)$ for $h \in \mathcal{H}_\mathbf{b}$. This mapping is well-defined as either *i)* for exactly one $\mathbf{b} \in \mathbf{B}$, $h \in \mathcal{H}_\mathbf{b}$ (in which case $g(h)$ is well-defined), or *ii)* if $h \in \mathcal{H}_\mathbf{b}$ and $h \in \mathcal{H}_{\mathbf{b}'}$, then $g_\mathbf{b}(h) = g_{\mathbf{b}'}(h)$. The latter is ensured as history names are carried over in exactly the same way by $g_\mathbf{b}$ and $g_{\mathbf{b}'}$, by construction. Hence $g = \bigcup_{\mathbf{b}\in\mathbf{B}} g_\mathbf{b}$ is a well-defined map. It is an injection: For $h \in H$ and $h' \in H$, if $h \neq h'$, then $g(h) \neq g(h')$ by the unique naming convention of the maps $g_\mathbf{b}$, $\mathbf{b} \in \mathbf{B}$. It is also a surjection: it is well-defined and $|H| = |H_\mathbf{B}|$ as by construction $|H'_\mathbf{b}| = |H_\mathbf{b}|$ for all $\mathbf{b} \in \mathbf{B}$, and $H = \bigcup_{\mathbf{b}\in\mathbf{B}} H_\mathbf{b}$ and $H_\mathbf{B} = \bigcup_{\mathbf{b}\in\mathbf{B}} H'_\mathbf{b}$. Hence g is a bijection.

That g is also the sought ETL isomorphism follows as $(\mathcal{H}_\mathbf{b}, \underline{H}_\mathbf{b})$ is isomorphic to $(\mathcal{H}'_\mathbf{b}, \underline{H}'_\mathbf{b})$, for all $\mathbf{b} \in \mathbf{B}$, cf. Prop. 4. This completes the proof. □

Lemma 4 The saturated ETL model properties Synchronicity, Perfect Recall and Postcondition Describable persist under ETL model union.

I.e.: Let $\{(\mathcal{H}_j, \underline{H}_j)\}_{j\in J}$ be a countable set of saturated ETL models. If all $(\mathcal{H}_j, \underline{H}_j)$ satisfy either of the mentioned properties, then the (unsaturated) union structure U_J satisfies that property.

Proof Synchronicity, Perfect Recall and Postcondition Describable persist because they are defined on histories which occur uniquely in the ETL forest, which makes them local by nature. Hence these properties are evaluated locally within a branch to ensure that there will be no conflicts when taking the union of different ETL sub-models. □

Property 5 Local No Miracles, Precondition Describable, Point Bisimulation Invariance and Connected Time-Steps do not persist under union.

Proof Local No Miracles does not persist under union: Consider a family of two DEL dynamical systems (that each individually satisfy property Local No Miracles.) with multi-pointed action models f and g with equal initial Kripke model $x = \{h, h', h_1, h_2\}$ where $h\mathtt{R}_i h'$ and $h\mathtt{R}^* h_1$ (by default, as initial models are connected), but disjoint Kripke models at the next level: $f^1(x) = \{he, h'e'\}$ and $g^1(x) = \{h_1 e, h_2 e'\}$ where $h_1 e \mathtt{R}_i h_2 e'$ while not $he\mathtt{R}_i h'e'$. In this example, Local No Miracles fails because not $he\mathtt{R}^* h_1 e$.

Precondition Describable does not persist under union: Consider a family of two DEL dynamical systems (that each individually satisfy property Precondition Describable) with multi-pointed action models f and g with equal initial Kripke model $x = \{h, h'\}$ with $h\mathtt{R}^* h'$ (by default as initial models are connected), but disjoint Kripke models at the next level: $f^1(x) = \{he\}$ and $g^1(x) = \{h'e\}$. Then it is possible that $h' \models \delta_e$ while $h \not\models \delta_e$, which breaks property Precondition Describable. It is left as an open question whether a suitably weakened version of Precondition Describable exists.

That Point Bisimulation Invariance is not preserved under union follows as the property is stated based on a saturation, but the union structure is unpointed, and hence unsaturated. If the union structure would be pointed with the set of points chosen as the union of the sets of points from the united ETL models, then the resulting set of points need not be a saturation, as the united ETL models may overlap, but have distinct points. In that case, the union structure would be "oversaturated".

Connected Time-Steps does not persist under union as histories within the same time-step are no longer necessarily epistemically connected in a union of disconnected ETL sub-models. □

Proposition 5 If an ETL model $\mathcal{H}$ is generated by a family of pointed DEL dynamical systems (possibly neither image-finite nor concluding), then $\mathcal{H}$ satisfies Synchronicity, Perfect Recall, Postcondition Describable and Very Local No Miracles and Local Bisimulation Invariance.

Proof That $\mathcal{H}$ satisfies Synchronicity, Perfect Recall and Postcondition Describable follows from Prop. 2 and Lemma 4.

That $\mathcal{H}$ satisfies Very Local No Miracles follows directly by the additional requirement compared to Local No Miracles that $he\mathtt{R}^* h_1 e$.

That $\mathcal{H}$ satisfies Local Bisimulation Invariance follows as 1) any ETL model that satisfies Point Bisimulation Invariance also satisfies Local Bisimulation Invariance, 2) Local Bisimulation Invariance is a property local to a connected component, and 3) connected components remain untouched under union. □

References

1. Ågotnes, T., van Ditmarsch, H., Wang, Y. (2017). True lies. *Synthese*, *195*(10), 4581–4615.

2. Artemov, S., Davoren, J., Nerode, A. (1997). Modal logics and topological semantics for hybrid systems. Technical report, Cornell University.
3. Aucher, G., & Bolander, T. (2013). Undecidability in epistemic planning. In *Proceedings of the twenty-third international joint conference on artificial intelligence* (pp. 27–33). AAAI Press.
4. Aucher, G., Maubert, B., Pinchinat, S. (2014). Automata techniques for epistemic protocol synthesis. In *Proceedings 2nd international workshop on strategic reasoning, SR 2014, Grenoble, France, April 5–6, 2014* (pp. 97–103).
5. Baltag, A., & Moss, L.S. (2004). Logics for epistemic programs. *Synthese*, *139*(2), 165–224.
6. Baltag, A., & Renne, B. (2016). Dynamic epistemic logic. In E.N. Zalta (Ed.), *The Stanford encyclopedia of philosophy*. Fall 2016 edition.
7. Baltag, A., & Smets, S. (2008). A qualitative theory of dynamic interactive belief revision. In G. Bonanno, W. van der Hoek, M. Wooldridge (Eds.), *Logic and the foundations of game and decision theory (LOFT 7), Texts in logic and games* (Vol. 3, pp. 9–58). Amsterdam University Press.
8. Baltag, A., & Smets, S. (2009). Group belief dynamics under iterated revision: fixed points and cycles of joint upgrades. In *Proceedings of the 12th conference on theoretical aspects of rationality and knowledge, TARK '09* (pp. 41–50). New York: ACM.
9. van Benthem, J. (2002). "One is a lonely number": logic and communication. In Z. Chatzidakis, P. Koepke, W. Pohlers (Eds.), *Logic colloquium '02*. Lecture notes in logic, 27 (pp. 95–128). Association for Symbolic Logic.
10. van Benthem, J. (2011). *Logical dynamics of information and interaction*. Cambridge: Cambridge University Press.
11. van Benthem, J. (2016). Oscillations, logic, and dynamical systems. In S. Ghosh, & J. Szymanik (Eds.) *The facts matter* (pp. 9–22). College Publications.
12. van Benthem, J., & Dégremont, C. (2010). Bridges between dynamic doxastic and doxastic temporal logics. In *Logic and the foundations of game and decision theory–LOFT 8* (pp. 151–173). Berlin: Springer.
13. van Benthem, J., & Smets, S. (2015). Dynamic logics of belief change. In H. van Ditmarsch, J.Y. Halpern, W. van der Hoek, B. Kooi (Eds.), *Handbook of logics for knowledge and belief* (pp. 299–368). College Publications.
14. van Benthem, J., van Eijck, J., Kooi, B. (2006). Logics of communication and change. *Information and Computation*, *204*(11), 1620–1662.
15. van Benthem, J., Gerbrandy, J., Hoshi, T., Pacuit, E. (2009). Merging frameworks for interaction. *Journal of Philosophical Logic*, *38*(5), 491–526.
16. Baltag, A., Moss, L.S., Solecki, S. (1998). The logic of public announcements, common knowledge, and private suspicions (extended abstract). In *Proceedings of the international conference of TARK 1998* (pp. 43–56). Morgan Kaufmann Publishers.
17. Blackburn, P., de Rijke, M., Venema, Y. (2001). *Modal logic*. Cambridge: Cambridge University Press.
18. Bolander, T., & Birkegaard, M. (2011). Epistemic planning for single- and multi-agent systems. *Journal of Applied Non-Classical Logics*, *21*(1), 9–34.
19. Bolander, T., Jensen, M.H., Schwarzentruber, F. (2015). Complexity results in epistemic planning. In *Proceedings of IJCAI 2015 (24th international joint conference on artificial intelligence).*
20. Dégremont, C. (2010). The temporal mind: observations on the logic of belief change in interactive systems. PhD thesis, University of Amsterdam.
21. van Ditmarsch, H., & Kooi, B. (2008). Semantic results for ontic and epistemic change. In G. Bonanno, W. van der Hoek, M. Wooldridge (Eds.), *Logic and the foundations of game and decision theory (LOFT 7).* Texts in logic and games (Vol. 3, pages 87–117). Amsterdam University Press.
22. van Ditmarsch, H., van der Hoek, W., Kooi, B. (2008). *Dynamic epistemic logic*. Berlin: Springer.
23. van Ditmarsch, H., Ghosh, S., Verbrugge, R., Wang, Y. (2014). Hidden protocols: modifying our expectations in an evolving world. *Artificial Intelligence*, *208*, 18–40.
24. Fagin, R., Halpern, J.Y., Moses, Y., Vardi, M.Y. (1995). *Reasoning about knowledge*. Cambridge: The MIT Press.
25. Fernández-Duque, D. (2012). A sound and complete axiomatization for Dynamic Topological Logic. *Journal of Symbolic Logic*, *77*(3), 1–26.
26. Fernández-Duque, D. (2012). Dynamic topological logic of metric spaces. *Journal of Symbolic Logic*, *77*(1), 308–328.
27. Fernández-Duque, D. (2014). Non-finite axiomatizability of dynamic topological logic. *ACM Transactions on Computational Logic*, *15*(1), 1–18.

28. Galeazzi, P., & Lorini, E. (2016). Epistemic logic meets epistemic game theory: a comparison between multi-agent Kripke models and type spaces. *Synthese*, *193*(7), 2097–2127.
29. Gerbrandy, J., & Groeneveld, W. (1997). Reasoning about information change. *Journal of Logic, Language and Information*, *6*(2), 147–169.
30. Gierasimczuk, N. (2010). Knowing one's limits. Phd thesis, Institute for Logic, Language and Computation, University of Amsterdam.
31. Goranko, V., & Otto, M. (2008). Model theory of modal logic. In P. Blackburn, J. van Benthem, F. Wolter (Eds.) , *Handbook of modal logic*. Elsevier.
32. Halpern, J.Y., & Moses, Y. (1990). Knowledge and common knowledge in a distributed environment. *Journal of the ACM*, *37*(3), 549–587.
33. Halpern, J.Y., & Vardi, M.Y. (1988). Reasoning about knowledge and time in asynchronous systems. In *Proceedings of the 20th ACM symposium on theory of computing, STOC '88* (pp. 53–65). ACM.
34. Halpern, J.Y., & Vardi, M.Y. (1989). The complexity of reasoning about knowledge and time. I. Lower bounds. *Journal of Computer and System Sciences*, *38*(1), 195–237.
35. Halpern, J.Y., van der Meyden, R., Vardi, M.Y. (2004). Complete axiomatizations for reasoning about knowledge and time. *SIAM Journal of Computation*, *33*(3), 674–703.
36. Hintikka, J. (1962). *Knowledge and belief: an introduction to the logic of the two notions*, 2nd, 2005 edition. London: College Publications.
37. Hoshi, T. (2009). Epistemic dynamics and protocol information. PhD thesis, ILLC, Universiteit van Amsterdam.
38. Hoshi, T. (2010). Merging DEL and ETL. *Journal of Logic, Language and Information*, *19*(4), 413–430.
39. Klein, D., & Rendsvig, R.K. (2017). Convergence, continuity and recurrence in dynamic epistemic logic. In A. Baltag, & J. Seligman (Eds.) , *Logic and rational interaction (LORI VI).* Lecture Notes in Computer Science. Berlin: Springer.
40. Klein, D., & Rendsvig, R.K. (2017). Metrics for formal structures, with an application to dynamic epistemic logic. arXiv:1704.00977.
41. Kremer, P., & Mints, G. (1997). Dynamical topological logic. *Bulleting of Symbolic Logic*, *3*, 371–372.
42. Kremer, P., & Mints, G. (2007). Dynamic topological logic. In M. Aiello, I. Pratt-Hartmann, J. van Benthem (Eds.), *Handbook of spatial logics*, chapter 10 (pp. 565–606).
43. Liu, F., Seligman, J., Girard, P. (2014). Logical dynamics of belief change in the community. *Synthese*, *191*(11), 2403–2431.
44. van der Meyden, R. (1994). Axioms for knowledge and time in distributed systems with perfect recall. In *Proceedings ninth annual IEEE symposium on logic in computer science* (pp. 448–457).
45. van der Meyden, R. (1997). Constructing finite state implementations of knowledge-based programs with perfect recall. In *Intelligent agent systems theoretical and practical issues* (pp. 135–151). Berlin: Springer.
46. van der Meyden, R., & Wong, K.-S. (2003). Complete axiomatizations for reasoning about knowledge and branching time. *Studia Logica*, *75*(1), 93–123.
47. Mohalik, S., & Ramanujam, R. (2010). Automata for epistemic temporal logic with synchronous communication. *Journal of Logic, Language and Information*, *19*(4), 451–484.
48. Moss, L.S. (2015). Dynamic epistemic logic. In van Ditmarsch, H., J.Y. Halpern, W. van der Hoek, B. Kooi (Eds.), *Handbook of epistemic logic*. College Publications.
49. Osbourne, M.J., & Rubinstein, A. (1994). *A course in game theory*. Cambridge: The MIT Press.
50. Parikh, R., & Ramanujam, R. (1985). Distributed processes and the logic of knowledge. In Parikh, R. (Ed.), *Logics of programs* (pp. 256–268). Heidelberg: Berlin.
51. Parikh, R., & Ramanujam, R. (2003). A knowledge based semantics of messages. *Journal of Logic, Language and Information*, *12*(4), 453–467.
52. Plaza, J.A. (1989). Logics of public communications. In M.L. Emrich, M.S. Pfeifer, M. Hadzikadic, Z.W. Ras (Eds.), *Proceedings of the 4th international symposium on methodologies for intelligent systems* (pp. 201–216).
53. Rendsvig, R.K. (2013). Aggregated beliefs and informational cascades. In D. Grossi, O. Roy, R.K. Rendsvig (Eds.), *Logic, rationality, and interaction.* Lecture Notes in Computer Science (pp. 337–341). Berlin: Springer.
54. Rendsvig, R.K. (2014). Diffusion, influence and best-response dynamics in networks: an action model approach. In R. de Haan (Ed.), *Proceedings of the ESSLLI 2014 student session* (pp. 63–75). arXiv:1708.01477.

55. Rendsvig, R.K. (2014). Pluralistic ignorance in the bystander effect: informational dynamics of unresponsive witnesses in situations calling for intervention. *Synthese*, *191*(11), 2471–2498.
56. Rendsvig, R.K. (2015). Model transformers for dynamical systems of dynamic epistemic logic. In W. van der Hoek, W.H. Holliday, W.F. Wang (Eds.), *Logic, rationality, and interaction (LORI 2015, Taipei), LNCS* (pp. 316–327). Berlin: Springer.
57. Rendsvig, R.K. (2018). Logical dynamics and dynamical systems. PhD thesis, Lund University.
58. Rodenhäuser, B. (2011). A logic for extensional protocols. *Journal of Applied Non-Classical Logics*, *21*(3–4), 477–502.
59. Sadzik, T. (2006). Exploring the iterated update universe. ILLC Report PP-2006-263, pp. 1–34.
60. Sarenac, D. (2011). Modal logic for qualitative dynamics. In O. Roy, P. Girard, M. Marion (Eds.), *Dynamic formal epistemology, volume 351 of Synthese Library* (pp. 75–101). Berlin: Springer.
61. Wang, Y. (2010). Epistemic modelling and protocol dynamics. Phd thesis, Universiteit van Amsterdam.
62. Wooldridge, M., & Lomuscio, A. (1999). Reasoning about visibility, perception, and knowledge. In *International workshop on agent theories, architectures and languages* (pp. 1–12). Berlin: Springer.