A Survey of Dynamic Epistemic Logic

Bryan Renne
Computer Science
CUNY Graduate Center
365 Fifth Avenue, Room 4319
New York NY 10016
USA

http://bryan.renne.org/

July 2, 2008

Abstract

Dynamic Epistemic Logic is the study of how to reason about knowledge, belief, and communication. This paper surveys this fast-growing field, covering syntax, semantics, axiomatics, bisimulation and action emulation, expressivity results, extension and embedding results, and issues of truth persistence and paradox that result from doxastic change. Along the way, we identify a number of open problems and areas for further work.

Contents

1 Introduction **3**

2 Modal Logic: Knowledge, Belief, and Fact **4**

3 Public and Private Communication **8**
3.1 Syntax 8
3.2 Semantics 9
3.3 Hilbert Theory for Public Communication 10

4 BMS Logic: Generalized Communication **10**
4.1 Syntax 12
4.2 Semantics 13
4.3 Special BMS Frames 15
4.4 Hilbert Theory 16

5 Bisimulation and Action Emulation **17**
5.1 Standard Bisimulation 18
5.2 BMS Frame Bisimulation 19
5.3 Action Emulation 19

6 Expressivity **22**
6.1 Definitions 22
6.2 Expressivity Relative to Modal Logic 23
6.3 Relative Expressivity of Differing Communications 25

7 Extensions and Embeddings of BMS Logic **27**
7.1 Relativized Common Knowledge 27
7.2 Epistemic PDL 28
7.3 Temporal Considerations 29
7.3.1 Past-Looking Temporal Modalities 32
7.3.2 Future-Looking Temporal Modalities 33
7.3.3 Connections with Epistemic Temporal Logic 34

8 Issues of Doxastic Change **38**
8.1 Successful Updates 38
8.2 Epistemic Puzzles 40
8.3 The Fitch and Moore Paradoxes 40

9 Other Work **42**
9.1 Valuation Changes 42
9.2 Belief Revision 42
9.3 Updates and Probabilistic Reasoning 43

References **43**

1 Introduction

In using modal logic to reason about the knowledge and belief of agents, we assume that a complete description of a certain moment in time is given by a *pointed Kripke model* [19, 25]. Now a *Kripke model* itself consists of a nonzero number of *worlds*—each having its own truth assignment describing the basic facts of that world—along with a number of binary relations, one for each agent, that may or may not hold between any two worlds. The binary relations represent agent uncertainty: if agent i's relation connects world Γ to world Δ, then agent i will consider it possible that the actual world is Δ whenever the world is in fact Γ. So for agent i to believe something at world Γ, that something must be true at all those worlds i considers to be possible with respect to Γ. This is just the Hintikka-Kripke notion of belief [25, 29].

In this setup, knowledge is identified with correct belief: to say that agent i *knows* a statement φ at world Γ means that agent i believes φ at Γ and this belief is correct (that is, φ is true at Γ) [19].

Now a *pointed Kripke model* is a pair (M, Γ) consisting of a Kripke model M and a particular world Γ in M. The world Γ is to be thought of as the *actual* world. The truth assignment of the actual world Γ tells us the basic facts of the situation represented by (M, Γ). The purpose of the other worlds in M is to represent the agents' beliefs. An agent's beliefs may concern both the basic facts of the situation (M, Γ) and also higher-order beliefs (that is, beliefs about beliefs).

Since we have identified a pointed Kripke model (M, Γ) with a complete description of a certain moment in time, a natural way to represent the passage of time is to consider sequences of moments; that is, we consider sequences

$$(M_1, \Gamma_1), (M_2, \Gamma_2), (M_3, \Gamma_3), \dots, (M_n, \Gamma_n)$$

consisting of pointed Kripke models. This view of time is discrete, with the complete description of the k-th moment in time given by the pointed Kripke model (M_k, Γ_k).

Thinking of our agents as a distributed system, such a sequence of moments represents a certain run of the system, where the $(k+1)$-st moment is generated from the k-th moment as a result of the occurrence of a communication to one or more of the agents. In reasoning about such runs, we often want to consider how the agents' knowledge and belief is affected by a given kind of communication. Here are two examples.

1. If all agents receive a public communication that some basic statement p is true at the actual world in moment (M_k, Γ_k), then it ought to be common knowledge in the next moment (M_{k+1}, Γ_{k+1}) that p is true.

2. If no agent knows whether p is true in moment (M_k, Γ_k), then the private communication to just those agents in group G that p is true ought to bring about a next moment (M_{k+1}, Γ_{k+1}) in which p is common knowledge to the agents in G and yet p is still unknown to the agents not in G.

Dynamic Epistemic Logic (DEL) is the study of how to reason about knowledge, belief, and communication [8, 9, 13, 21, 34, 55]. DEL uses modal logic as the basic language for describing knowledge, belief, and fact. This basic language is then extended in various ways in order to describe what happens as a result of some communication. The most basic such extension is of the following kind: for a group G of agents and statements φ and ψ, we write the statement

$$[\varphi \rightarrow G]\psi$$

to mean that ψ is true after φ is communicated privately to just those agents in group G. We will use A to represent the group consisting of all agents, so the statement

$$[\varphi \rightarrow A]\psi \ ,$$

which we often abbreviate by $[\varphi]\psi$, says that ψ is true after φ is communicated publicly to all agents. Such statements allow us to express how communication affects knowledge and belief. In particular, we can express our example statements above.

1. $[p]C_A p$

   In words: after the public communication of p (to all agents), we have that p is common knowledge (to all agents).

2. $\big(\bigwedge_{i \in A} \neg K_i p\big) \supset [p \rightarrow G]\big(C_G p \wedge \bigwedge_{i \in A \backslash G} \neg K_i p\big)$

   In words: if no agent $i \in A$ knows p, then after the communication of p to just those agents in group G, we have that p is common knowledge to those in G and that no $i \in A \backslash G$ knows p.

   (Note: we always assume that A is finite.)

While we have only mentioned public and private communications, there is a natural way to define much more general kinds of communication that allow for complicated combinations of privacy and deceit [8]. All of this will be described in detail in what is to come.

The aim of this article is to survey the field of Dynamic Epistemic Logic (DEL)—a fast-growing area that has just seen the publication of its first book-length treatment [55]. Both with an eye toward our forthcoming overview of relative expressivity and also to work our way into the complications of the more expressive DEL languages, we will introduce the various DEL languages in order of increasing expressivity. We begin in the next section with an overview of our notation and terminology for modal logic, which serves as the basic language upon which all DEL languages are based.

2 Modal Logic: Knowledge, Belief, and Fact

The language of modal logic allows us to describe basic facts along with the knowledge and beliefs of a finite nonzero number of agents.

Definition 2.1. An *agent set* is a finite nonempty set whose members will be called *agents*.

Once we have fixed our set of agents, then we may define the usual languages of modal logic for this agent set.

Definition 2.2. Let A be an agent set.

- The *language of propositional logic*, written PL, consists of the formulas φ built by the following grammar.

$$\varphi \quad ::= \quad p_k \mid \bot \mid \top \mid \varphi_1 \supset \varphi_2 \qquad k \in \mathbb{N}$$

  $\{p_k : k \in \mathbb{N}\}$ is the set of *propositional letters*. The *atoms* consist of the propositional letters, the propositional constant $\bot$ for falsity, and the propositional constant $\top$ for truth. Formulas written using other logical connectives are understood as abbreviations for formulas in the above language.

- The *language of modal logic (for A)*, written ML^A, is the extension of PL obtained by adding the following rule of formula formation: if φ is a formula and $i \in A$, then $K_i\varphi$ is also a formula.

- The *language of modal logic with common knowledge (for A)*, written ML^A_C, is the extension of ML^A obtained by adding the following rule of formula formation: if φ is a formula and $G \subseteq A$, then $C_G\varphi$ is also a formula.

- For each $G \subseteq A$, we make the following abbreviation:

$$E_G\varphi := \begin{cases} \bigwedge_{i\in G} K_i\varphi & \text{if } G \neq \emptyset, \\ \top & \text{if } G = \emptyset. \end{cases}$$

The modal formulas $K_i\varphi$, $E_G\varphi$, and $C_G\varphi$ allow us to express various kinds of knowledge of our agents in A, all according to the following intuitive readings of these formulas.

- $K_i\varphi$ is read, "(agent) i knows φ."
- $E_G\varphi$ is read, "everyone in G knows φ."
- $C_G\varphi$ is read, "φ is common knowledge to those in G."

The semantics of these languages, due to Kripke [29], begins by specifying a *frame*, which is relational structure that represents agent uncertainty.

Definition 2.3. Let A be an agent set. A *frame (for A)* is a pair (W, R) whose components are given as follows.

- W is a nonempty set whose elements are called *worlds (in M)*.
- $R : A \to 2^{W\times W}$ is a function mapping each $i \in A$ to a binary relation R_i on W.[1]

[1] A *binary relation on (a set) W* is a set $R \subseteq W \times W$. Notation: if R is a binary relation on a set W and $\Gamma, \Delta \in W$, then we write $\Gamma R \Delta$ to mean that $(\Gamma, \Delta) \in R$.

If $F = (W, R)$ is a frame, then we write $\Gamma \in F$ to mean that $\Gamma \in W$. A *pointed frame (for A)* is a pair (F, Γ) consisting of a frame F for A and a world $\Gamma \in F$. To say that a frame $F = (W, R)$ is *finite* means that the set W is finite. To say that a pointed frame (F, Γ) is *finite* means that F is finite.

In a frame (W, R) for A, the relationship $\Gamma R_i \Delta$ between the worlds Γ and Δ represents agent i's thinking that Δ is the actual world in case Γ is in fact the actual world. In this way we can represent agent i's uncertainty as to the actual world. A pointed frame (F, Γ) adds to the uncertainties represented by F an actual world Γ.

We adopt the following notation for the remainder of the paper, which will be useful in forthcoming definitions.

Notation 2.4. Let W be a set, let $R \in 2^{W \times W}$ be a binary relation on W, and let $R_i \in 2^{W \times W}$ be a binary relation on W for each $i \in G$.

- R_G denotes the binary relation $\bigcup_{i \in G} R_i$ on W.
- R^* denotes the reflexive-transitive closure of R.[2]

What remains to interpret formulas is add a *valuation*, which provides a truth assignment for each world in a frame.

Definition 2.5. Let A be an agent set and let $F = (W, R)$ be a frame for A. A *valuation (on F)* is a function $V : \{p_k : k \in \mathbb{N}\} \to 2^W$ that maps each propositional letter p_k to a possibly empty set $V(p_k)$ of worlds in F.

Combining a frame with a valuation gives us a Kripke model. These models are used to interpret formulas in the language of modal logic.

Definition 2.6. Let A be an agent set. A *Kripke model (for A)* is a tuple (W, R, V) consisting of a frame (W, R) for A and a valuation V on (W, R). To say that Γ is a *world in (the) Kripke model M*, written $\Gamma \in M$, means that $\Gamma \in W$. A *pointed Kripke model (for A)* is a pair (M, Γ) consisting of a Kripke model M for A and a world $\Gamma \in M$. To say that pointed Kripke model (M, Γ) for A has a property $\mathcal{P}$ of binary relations—examples include reflexivity, transitivity, euclideanness, or seriality—means that for each $i \in A$, the binary relation R_i has property $\mathcal{P}$.[3] For a pointed Kripke model (M, Γ) and a formula $\varphi \in \mathsf{ML}_C^A$, we write $M, \Gamma \models \varphi$ to mean that φ is *true at* (M, Γ). The negation of $M, \Gamma \models \varphi$ is written $M, \Gamma \not\models \varphi$. Truth of a formula $\varphi \in \mathsf{ML}_C^A$ at a pointed Kripke model (M, Γ) is given by the following induction on the construction of φ.

- $M, \Gamma \models p_k$ means that $\Gamma \in V(p_k)$.

[2]The *reflexive-transitive closure of* R is the smallest set $R' \in 2^{W \times W}$ such that $R \subseteq R'$ and R' is both reflexive and transitive.

[3]Let $R \in 2^{W \times W}$ be a binary relation on a set W. R is *reflexive* iff $\Gamma R \Gamma$ for each $\Gamma \in W$. R is *transitive* iff $\Gamma R \Delta$ and $\Delta R \Omega$ together imply $\Gamma R \Omega$ for each $\Gamma, \Delta, \Omega \in W$. R is *euclidean* iff $\Gamma R \Delta$ and $\Gamma R \Omega$ together imply $\Delta R \Omega$ for each $\Gamma, \Delta, \Omega \in W$. R is *serial* iff for each $\Gamma \in W$, there is a $\Delta \in W$ such that $\Gamma R \Delta$.

- $M, \Gamma \not\models \bot$ and $M, \Gamma \models \top$.
- $M, \Gamma \models \varphi_1 \supset \varphi_2$ means that $M, \Gamma \not\models \varphi_1$ or $M, \Gamma \models \varphi_2$.
- $M, \Gamma \models K_i\varphi$ means that $M, \Delta \models \varphi$ for each $\Delta \in M$ satisfying $\Gamma R_i \Delta$.
- $M, \Gamma \models C_G\varphi$ means that $M, \Delta \models \varphi$ for each $\Delta \in M$ satisfying $\Gamma R_G^* \Delta$.

If $\mathcal{I}$ is a set of pointed Kripke models for A, then to say that a formula $\varphi \in \mathfrak{L}^A$ is *valid for* $\mathcal{I}$, written $\mathcal{I} \models \varphi$, means that $M, \Gamma \models \varphi$ for each pointed Kripke model $(M, \Gamma) \in \mathcal{I}$. To say that a formula $\varphi \in \mathfrak{L}^A$ is *valid*, written $\models \varphi$, means that φ is valid for the set of all pointed Kripke models for A. (Note that the semantics ensures that $\models C_\emptyset\varphi \equiv \varphi$.)

Since most of what follows in this survey is sufficiently flexible to work for a number of normal modal theories, we need not commit ourselves to any particular normal modal theory. But it will nonetheless make the formulation of the coming axiom systems much more readable if we do choose a particular normal modal logic. Thus we will typically choose K, the minimal normal modal logic [14].

Definition 2.7. Let A be an agent set. The modal logic K_C^A is given by the following axiom schemes and rules of inference.

- Basic axiom schemes (in the language ML_C^A)
  1. Axiom schemes for classical propositional logic
  2. $K_i(\varphi \supset \psi) \supset (K_i\varphi \supset K_i\psi)$ "i knows the consequences of his knowledge"
- Common Knowledge (CK) axiom schemes (in the language ML_C^A)
  1. $C_G(\varphi \supset \psi) \supset (C_G\varphi \supset C_G\psi)$ "CK is closed under consequence"
  2. $C_G\varphi \supset (\varphi \wedge E_G C_G\varphi)$ "CK implies truth and group knowledge of CK"
  3. $\varphi \wedge C_G(\varphi \supset E_G\varphi) \supset C_G\varphi$ "CK arises from group knowledge by induction"
- Rules of inference
  - *Modus Ponens:* if φ and $\varphi \supset \psi$ are each provable, then so is ψ.
  - *K_i-Necessitation:* if φ is provable, then so is $K_i\varphi$. "i knows what is provable"
  - *C_G-Necessitation:* if φ is provable, then so is $C_G\varphi$. "what is provable is CK"

The modal logic K^A is given by omitting the common knowledge axiom schemes, the C_G-Necessitation rule, and restricting the remaining axiom schemes to the language ML^A.

Theorem 2.8. Let A be an agent set. For each formula $\varphi \in \mathsf{ML}^A$, we have that φ is a theorem of K^A if and only if $\models \varphi$ [29]. Similarly, for each formula $\varphi \in \mathsf{ML}_C^A$, we have that φ is a theorem of K_C^A if and only if $\models \varphi$ (see, for example, [19]).

Now that we have reviewed our notation and terminology for modal logic, we proceed to our survey of Dynamic Epistemic Logic proper.

3 Public and Private Communication

In this section, we begin our first step into the study of Dynamic Epistemic Logic by defining extensions of modal logic for reasoning about public and private communication. These extended languages are the simplest languages in the DEL family, in that these languages describe the most basic kinds of communication.

3.1 Syntax

Definition 3.1. Let A be an agent set.

- The *language of public and private communication (for A)*, written COM^A, is the extension of ML^A obtained by adding the following rule of formula formation: if φ and ψ are formulas and $G \subseteq A$, then $[\varphi\!\rightarrow\! G]\psi$ is also a formula.

- The *language of public and private communication (for A) with common knowledge*, written COM^A_C, is the extension of COM^A obtained by adding the following rule of formula formation: if φ is a formula and $G \subseteq A$, then $C_G\varphi$ is also a formula.

Abbreviations: for each $i \in A$, we let $[\varphi_1\!\rightarrow\! i]\varphi_2$ abbreviate $[\varphi_1\!\rightarrow\!\{i\}]\varphi_2$; we also let $[\varphi_1]\varphi_2$ abbreviate $[\varphi_1\!\rightarrow\! A]\varphi_2$.

We read the formula $[\varphi\!\rightarrow\! G]\psi$ as "ψ is true after the communication of φ to just those in G." Note that the semantics will ensure the validity of the scheme $[\varphi\!\rightarrow\!\emptyset]\psi \equiv \varphi \supset \psi$.

It will be useful to define a few fragments of our languages COM^A and COM^A_C, with the particular fragment determined by the various groups of agents that are allowed to receive a communication.

Definition 3.2. Let A be an agent set and let $\mathfrak{G} \subseteq 2^A$ be a possibly empty collection of subsets of A. Then for each $\mathfrak{L} \in \{\mathsf{COM}^A, \mathsf{COM}^A_C\}$, the language $\mathfrak{L}(\mathfrak{G})$ is the fragment of $\mathfrak{L}$ obtained by restricting all subformulas of the form $[\varphi\!\rightarrow\! G]\psi$ so that $G \in \mathfrak{G}$. Notation: for $\mathfrak{L} \in \{\mathsf{COM}^A, \mathsf{COM}^A_C\}$, $G \subseteq A$, and $i \in A$, we let $\mathfrak{L}(G)$ denote $\mathfrak{L}(\{G\})$ and we let $\mathfrak{L}(i)$ denote $\mathfrak{L}(\{i\})$.

We now define a few fragments of COM^A and COM^A_C that are of particular interest in the present paper.

Definition 3.3. Let A be an agent set and $G \subseteq A$.

- The *language of public communication (for A)*, written PUB^A, is $\mathsf{COM}^A(A)$.

- The *language of private communication (for A)*, written PRI^A, is $\mathsf{COM}^A\big(2^A \backslash \{A\}\big)$.

- The *language of single-recipient private communication (for A)*, written $\mathsf{PRI1}^A$, is

$$\mathsf{COM}^A\Big(\big\{\{i\} : i \in A\big\}\Big) \ .$$

- For each $\mathfrak{L} \in \{\mathsf{PUB}^A, \mathsf{PRI}^A, \mathsf{PRI1}^A\}$, the extension of $\mathfrak{L}$ *with common knowledge*, written $\mathfrak{L}_C$, that is obtained from $\mathfrak{L}$ by adding the following rule of formula formation: if φ is a formula and $G \subseteq A$, then $C_G\varphi$ is also a formula.

3.2 Semantics

COM^A_C-formulas are interpreted using an extension of Kripke's semantics for modal logic [29]. This extension is due to Baltag, Moss, and Solecki [8, 9].

Definition 3.4 ([8, 9])**.** Let A be an agent set. Truth of a formula $\varphi \in \mathsf{COM}^A_C$ at a pointed Kripke model (M, Γ) is given by extending the induction in the definition of truth for formulas in ML^A_C (Definition 2.6) by adding the following inductive clause: $M, \Gamma \models [\varphi_1 \to G]\varphi_2$ means that either we have $M, \Gamma \not\models \varphi_1$ or else we have both $M, \Gamma \models \varphi_1$ and $M[\varphi_1 \to G], (\Gamma, 0) \models \varphi_2$, where the Kripke model $M[\varphi_1 \to G]$ is the tuple

$$\big(W[\varphi_1 \to G], R[\varphi_1 \to G], V[\varphi_1 \to G]\big)$$

whose components are given as follows.

- $W[\varphi_1 \to G] \;:=\; \big\{(\Delta, 0) \in W \times \{0\} : M, \Delta \models \varphi_1\big\} \cup \big\{(\Delta, 1) \in W \times \{1\} : M, \Delta \models \top\big\}$
- For each $i \in G$: $R_i[\varphi_1 \to G]$ is the set
$$\Big\{\big((\Delta, a), (\Omega, b)\big) \in (W[\varphi_1 \to G])^2 : (\Delta R_i \Omega) \wedge (a = b)\Big\}$$
- For each $j \in A \backslash G$: $R_j[\varphi_1 \to G]$ is the set
$$\Big\{\big((\Delta, a), (\Omega, b)\big) \in (W[\varphi_1 \to G])^2 : (\Delta R_j \Omega) \wedge (b = 1)\Big\}$$
- $V[\varphi_1 \to G](p_k) := \big\{(\Delta, a) \in W[\varphi_1 \to G] : \Delta \in V(p_k)\big\}$

The various notions of validity from Definition 2.6 carry over directly to COM^A_C-formulas.

The idea behind the construction of the Kripke model $M[\varphi \to G]$ may be understood as follows. The worlds in $M[\varphi \to G]$ of the form $(\Gamma, 0)$ are just those worlds of M at which φ is true, while the worlds in $M[\varphi \to G]$ of the form $(\Gamma, 1)$ make up a copy of the Kripke model M. The binary relations in $M[\varphi \to G]$ are then defined so that from a world $(\Gamma, 0)$, agents in G will only consider possible worlds of the form $(\Delta, 0)$ while agents in $A \backslash G$ will only consider possible worlds of the form $(\Delta, 1)$. Thus the agents in G jointly eliminate from consideration all worlds in M at which φ is not true—and in this sense it becomes common knowledge among G that φ was communicated—while the agents in $A \backslash G$ are effectively unaware that the communication of φ to G ever occurred. So in case we have that $M, \Gamma \models \varphi$, then the construction of $M[\varphi \to G]$ takes us from the moment in time given by the pointed Kripke model (M, Γ) to a next moment in time given by the pointed Kripke model $\big(M[\varphi \to G], (\Gamma, 0)\big)$. It is in this way that communication moves time from one moment to the next in this framework.

3.3 Hilbert Theory for Public Communication

In this subsection, we will examine the axiomatization of the validities for a fragment of COM^A_C. The fragment in question is PUB^A_C, the language of public communication. Recall that we write $[\varphi]\psi$ as an abbreviation for the formula $[\varphi \!\rightarrow\! A]\psi$.

Definition 3.5. Let A be an agent set. The *theory for* PUB^A_C is given by the following axiom schemes and rules of inference.

- Axiom schemes and rules for K^A_C
- Axiom schemes for communication (in the language PUB^A_C)
  1. $[\varphi]p \equiv (\varphi \supset p)$, for each atom p
     "facts are unchanged by announcements"
  2. $[\varphi](\psi \supset \chi) \equiv \big([\varphi]\psi \supset [\varphi]\chi\big)$
     "announcements commute with Boolean connectives"
  3. $[\varphi]K_i\psi \equiv \varphi \supset K_i[\varphi]\psi$
     "knowledge of ψ after a public announcement comes from having knowledge that the announcement will bring about ψ"
  4. $[\varphi][\psi]\chi \equiv [\varphi \wedge [\varphi]\psi]\chi$
     "iterated announcements may be combined into a single announcement"
- Rules for communication
  - *Announcement-Necessitation:* if ψ is provable, then so is $[\varphi]\psi$.
    "what is provable holds after an announcement"
  - *CK Rule:* if each of $\chi \supset [\varphi]\psi$ and $\chi \wedge \varphi \supset E_G\chi$ is provable, then so is $\chi \supset [\varphi]C_G\psi$.

The *theory for* PUB^A is obtained by replacing the axiom schemes and rules for K^A_C by the axiom schemes and rules for K^A, omitting the CK Rule, and restricting the remaining schemes to the language PUB^A.

Theorem 3.6. Let A be an agent set. For each $\varphi \in \mathsf{PUB}^A$, we have that φ is a theorem of the theory for PUB^A if and only if $\models \varphi$ [21, 34, 55]. Also, for each $\varphi \in \mathsf{PUB}^A_C$, we have that φ is a theorem of the theory for PUB^A_C if and only if $\models \varphi$ [8, 10, 55].

4 BMS Logic: Generalized Communication

The work of Baltag, Moss, and Solecki [8, 9, 10] was a watershed in the study of Dynamic Epistemic Logic. The key insight of their work is that an agent's uncertainty as to the particular formula that is communicated can be represented in the same way as the agent's uncertainty as to the actual world. Let us sketch out in a bit more detail what it is that we mean by this.

To begin, we take a finite frame (U, S) and label each world in this frame by a formula in some fixed language $\mathfrak{L}$. Formally, this is accomplished by introducing a labeling function $l : U \to \mathfrak{L}$ that assigns to each world $v \in U$ a formula $l(v) \in \mathfrak{L}$. We will call the combined tuple $B = (U, S, l)$ a *BMS frame*.

The BMS frame $B = (U, S, l)$ represents a number of possible communications, one for each world $v \in U$, with the world $v \in U$ representing the communication of the formula $l(v)$. The frame (U, S) then encodes the agents' uncertainty as to which formula was actually communicated: if vS_iv', then agent i will consider it possible that $l(v')$ is communicated whenever $l(v)$ is in fact communicated.

For the purpose of our exposition here, to say that the formula φ is *communicated* in the Kripke model M means that we delete from M all worlds that contradict φ; that is, we delete all $\Gamma \in M$ such that $M, \Gamma \not\models \varphi$, obtaining a Kripke model $M|\varphi$ as the result of this operation. Thus to communicate a formula φ is simply to have the agents jointly eliminate all $\neg\varphi$-worlds from consideration, and it is in this sense that φ is communicated.

Communicating the entire BMS frame $B = (U, S, l)$ in a model M then amounts to making all possible communications of a formula $l(v)$ in M for each $v \in U$ and then combining these various possible communications according to the agent uncertainties encoded in the BMS frame B. So, in a bit more detail, we take the disjoint union

$$\dot{\bigcup}_{v\in U} M|l(v) = \big\{(\Gamma, v) : M, \Gamma \models l(v)\big\}$$

of all the various formulas in $B = (U, S, l)$, giving us a copy of the submodel $M|l(v)$ of M for each possible communication $v \in U$. The agents' uncertainty as to which was the actual communication is then encoded according to S:

$$(\Gamma, v)R'_i(\Gamma', v') \text{ means } \Bigg(\Gamma R_i \Gamma' \text{ and } vS_iv'\Bigg) \ .$$

Here $R' : A \to 2^{W\times W}$ is the function representing agent uncertainty after the communication of the entire BMS frame B. Notice that agent i's uncertainty after the communication comes from two sources: his uncertainty as to the actual world before the communication (represented by the binary relation R_i) and his uncertainty as to the actual formula communicated (represented by the binary relation S_i). Thus for i to know that the communication of $l(w)$ is occurring, we must have the equality

$$\{v \in U : wS_iv\} = \{w\} \ ,$$

which says that the only communication i considers possible is the communication $l(w)$ itself. But we can in general represent much more subtle combinations of agent uncertainty over communications by varying the structure of the BMS frame $B = (U, S, l)$. We will see some specific examples of this in §4.3. First let us go over the formal definitions of the syntax and semantics.

4.1 Syntax

We now formalize our introductory comments. We begin by defining a labeling function for finite sets.

Definition 4.1. Let U be a finite set and let $\mathfrak{L}$ be a language. A *labeling for* U is a function $l : U \to \mathfrak{L}$ that maps each $v \in U$ to a formula $l(v) \in \mathfrak{L}$.

Combining a labeling function with a finite frame gives us a BMS frame.

Definition 4.2. Let A be an agent set. A *BMS frame (for A)* is a tuple (U, S, l) consisting of a finite frame (U, S) for A and a labeling l for U.

- The BMS frame (U, S, l) is said to be *based on* the frame (U, S).
- To say that v is a *world in* B, written $v \in B$, means that $v \in U$.
- For each world $v \in B$, we let $B(v)$ denote the formula $l(v)$.
- To say that the BMS frame B is *in (the) language* $\mathfrak{L}$, written $B \in \mathfrak{L}$, means that $B(v) \in \mathfrak{L}$ for each $v \in B$.

A *pointed BMS frame (for A)* is a pair (B, v) consisting of a BMS frame B for A and a world $v \in B$. To say that the pointed BMS frame (B, v) is *based on* the frame (U, S) for A means that B is based on (U, S). To say that the pointed BMS frame (B, v) is *in (the) language* $\mathfrak{L}$, written $(B, v) \in \mathfrak{L}$, means that $B \in \mathfrak{L}$.

The language of BMS Logic is then obtained by admitting pointed BMS frames (B, w) as modals. The reason that it makes sense for us to do this is that a pointed BMS frame is a finite structure and thus can be written down using a finite number of symbols.[4]

Definition 4.3. Let A be an agent set.

- The *language of BMS Logic (for A)*, written BMS^A, consists of the set of formulas built using the rules of formula formation for ML^A in addition to the following rule: if we have that (B, w) is a pointed BMS frame for A, that $B(v)$ is a formula for each $v \in B$, and that φ is a formula, then $[B, w]\varphi$ is also a formula.
- The *language of BMS Logic (for A) with common knowledge*, written BMS^A_C, is the extension of BMS^A obtained by adding the following rule of formula formation: if φ is a formula and $G \subseteq A$, then $C_G\varphi$ is also a formula.

In the languages BMS^A and BMS^A_C, the modals $[B, w]$ are called *BMS modals*.

Finally, it is useful to have a notion of composition for BMS frames. This allows us to combine the communications and uncertainties given by first communicating the BMS frame B and then communicating the BMS frame B' into a single BMS frame $B \circ B'$.

[4]Some work has been done to bring the structure of communications explicitly into the language, whether by a hybrid extension [16] or by the use of fixed-points [6].

Definition 4.4 (Composition)**.** Let A be an agent set and let $B = (U, S, l)$ and $B' = (U', S', l')$ be labeled BMS frames for A in the language BMS_C^A. The *composition of* B *and* B', written $B \circ B'$, is the labeled BMS frame (U^c, S^c, l^c) given as follows.

- $U^c := U \times U'$
- For each $i \in A$, we let S_i^c be the set
$$\Big\{ \big((v, v'), (w, w')\big) \in U^c : vS_i w \wedge v' S_i' w' \Big\}$$
- For each $(v, v') \in U$, set $l^c(v, v') := \neg[B, v]\neg B'(v')$

4.2 Semantics

BMS_C^A-formulas are interpreted using an extension of Kripke's semantics for modal logic [29]. This extension is due to Baltag, Moss, and Solecki [8, 9].

Definition 4.5 ([9, 8])**.** Let A be an agent set. Truth of a formula $\varphi \in \mathsf{BMS}_C^A$ at a pointed Kripke model (M, Γ) for A is given by extending the induction in the definition of truth for formulas in ML_C^A (Definition 2.6) by adding the following inductive clause: $M, \Gamma \models [B, w]\varphi$ means that either we have $M, \Gamma \not\models B(w)$ or else we have both $M, \Gamma \models B(w)$ and $M[B], (\Gamma, w) \models \varphi$, where for the BMS frame $B = (U, S, l)$ for A, we define the components of the Kripke model
$$M[B] = \big(W[B], R[B], V[B]\big)$$
as follows.

- $W[B] := \big\{(\Delta, v) \in W \times U : M, \Delta \models B(v)\big\}$
- For each $i \in A$: $R_i[B]$ is the set
$$\Big\{ \big((\Delta, v), (\Delta', v')\big) \in \big(W[B]\big)^2 : (\Delta R_i \Delta') \wedge (vS_i v') \Big\}$$
- $V[B](p_k) := \big\{(\Delta, v) \in W[B] : \Delta \in V(p_k)\big\}$

The various notions of validity from Definition 2.6 carry over directly to BMS_C^A-formulas.

The idea behind the construction of the Kripke model $M[B]$ may be understood as follows. For each world $v \in B$ in the BMS frame B, we make a copy of the submodel $M|B(v)$ of M consisting of the worlds in M at which the formula $B(v)$ holds. This gives us the disjoint union
$$\dot{\bigcup}_{v \in B} M|B(v) = \big\{(\Gamma, v) : M, \Gamma \models B(v)\big\}$$
consisting of the copies of submodels of M, one for each $v \in B$ that brings about a communication. The submodel $M|B(v)$ represents the communication of the formula $B(v)$ in the

sense that every world in M that is inconsistent with the formula $B(v)$ is eliminated from consideration. But the agents are in general uncertain as to which formula was actually communicated. Thus if we have that vS_iv', then agent i thinks it possible the formula $B(v')$ was communicated when the formula $B(v)$ was in fact communicated. So the agents' resulting uncertainty in $M[B]$ reflects the two sources of agent uncertainty, the first being uncertainty as to the actual world in M (represented by the function R) and the second being uncertainty as to the actual communication (represented by the function S). Note that the facts true at a world before and after a communication do not change, since $V'(\Gamma, v) = V(\Gamma)$.

If we are given a pointed Kripke model (M, Γ) and a pointed BMS frame (B, w) such that $M, \Gamma \models B(w)$, then we may construct the pointed Kripke model $\big(M[B], (\Gamma, w)\big)$. It is in this way that communication moves time from one moment to the next in this framework. Note that this movement of time requires that $M, \Gamma \models B(w)$, an often-encountered condition called *executability*.

Definition 4.6. Let A be an agent set, let (B, u) be a pointed BMS frame for A with $B \in \mathsf{BMS}^A$, and let (M, Γ) be a pointed Kripke model for A.

- To say that (B, u) *is executable at (the pointed Kripke model)* (M, Γ) means that $M, \Gamma \models B(u)$.
- To say that B *is executable in (the Kripke model)* M means that there is a $v \in B$ and a $\Delta \in M$ such that (B, v) is executable at (M, Δ).

Fixing a BMS frame B, we may define the set S of all Kripke models such that B is executable in M. From this, we can construct a function f that maps each Kripke model $M \in S$ to the Kripke model $M[B]$. A partial function f that may be constructed in this way is called a *BMS update*.

Definition 4.7. Let A be an agent set and let B be a BMS frame for A.

- The *update induced by* B is the partial function

$$\big\{(M, M[B]) : B \text{ is executable in } M\big\}$$

  that maps pointed Kripke models for A to pointed Kripke models for A.

- A *BMS update (for A)* is any partial function f such that there is a BMS frame B' for A satisfying the property that the update induced by B is f.

This definition gives us another way to understand BMS Logic: BMS Logic is the logic for reasoning about how knowledge and belief change as the result of a BMS update.[5]

[5]Note that BMS updates are in general *partial* functions. In [41], the authors define a different notion of public communication so as to guarantee that the updates induced by these public communications are *total* functions.

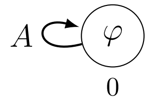

Figure 1. The BMS frame Pub^A_φ, the public communication of φ, from Definition 4.8.

4.3 Special BMS Frames

While we have seen in general how BMS frames interact syntactically and semantically in the extension BMS^A_C of modal logic, there are certain BMS frames that model well-known kinds of communication. We identify a few of these BMS frames here.

Definition 4.8 ([8])**.** Let A be an agent set and let $G \subseteq A$. Then the *public communication of* φ (also called the *public announcement of* φ), written Pub^A_φ, is the BMS frame $\big(U, S, l\big)$ whose components are defined as follows: $U := \{0\}$, $S_i := \big\{(0,0)\big\}$ for each $i \in A$, and $l(0) := \varphi$. See Figure 1 for a picture of the BMS frame Pub^A_φ.

The public communication Pub^A_φ of φ is structured in such a way that it is common knowledge among all the agents that φ is in fact what is communicated. Thus if we have that Pub^A_φ is executable in a Kripke model M, then the function $M \mapsto M[\mathrm{Pub}^A_\varphi]$ simply restricts M to the submodel obtained by deleting those worlds $\Gamma \in M$ that are inconsistent with φ (meaning $M, \Gamma \not\models \varphi$). Thus the agents in A all jointly eliminate from consideration those $\neg\varphi$-worlds in M, and it is in this sense that the operation $M \mapsto M[\mathrm{Pub}^A_\varphi]$ is a public communication of φ. Observe that for each atom p, we have that $\models [\mathrm{Pub}^A_p, 0]C_A p$; that is, the public announcement of p makes it common knowledge that p holds.

Let $\mathfrak{L}$ be the fragment of BMS^A_C obtained by restricting BMS modals to the form $[\mathrm{Pub}^A_\varphi, 0]$ for some $\varphi \in \mathfrak{L}$. Then $\mathfrak{L}$ is equivalent to PUB^A_C: replacing each BMS modal $[\mathrm{Pub}^A_\varphi, 0]$ in a formula $\chi \in \mathfrak{L}$ by the modal $[\varphi]$ produces a formula $\chi' \in \mathsf{PUB}^A_C$ such that $\models \chi \equiv \chi'$, and replacing each modal $[\varphi]$ in a formula $\theta \in \mathsf{PUB}^A_C$ by the BMS modal $[\mathrm{Pub}^A_\varphi, 0]$ produces a formula $\theta' \in \mathfrak{L}$ such that $\models \theta \equiv \theta'$. So we see that BMS^A_C generalizes PUB^A_C.

Another important BMS frame is the BMS frame for private communication.

Definition 4.9 ([8])**.** Let A be an agent set and let $G \subseteq A$. Then the *private communication of* φ *to* G, written $\mathrm{Pri}^A_{\varphi\to G}$, is the BMS frame $\big(U, S, l\big)$ whose components are defined as follows: $U := \{0,1\}$, $S_i := \big\{(0,0),(1,1)\big\}$ for each $i \in G$, $S_i := \big\{(0,1),(1,1)\big\}$ for each $i \in A\backslash G$, $l(0) := \varphi$, and $l(1) := \top$. See Figure 2 for a picture of the BMS frame $\mathrm{Pri}^A_{\varphi\to G}$.

The private communication $\mathrm{Pri}^A_{\varphi\to G}$ of φ to G is structured in such a way that it is common knowledge to the agents in G that φ is communicated, whereas the agents in $A\backslash G$ all think that $\top$ is communicated. Note that communicating $\top$ is the same as having no communication at all: since we have identified the communication of a formula with the operation that restricts a Kripke model to just those worlds in which the formula is

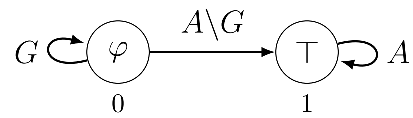

Figure 2. The BMS frame $\mathrm{Pri}^{A}_{\varphi\to G}$, the private communication of φ to G, from Definition 4.9.

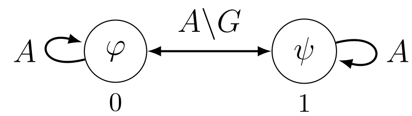

Figure 3. The BMS frame $\frac{1}{2}\mathrm{Pri}^{A}_{\varphi,\psi\to G}$, the semi-private communication of φ or ψ to G, from Definition 4.10.

true, it follows from the validity of $\top$ that a communication of $\top$ leaves every Kripke model unchanged; thus a communication of $\top$ is equivalent to having no communication at all. So in the private communication $\mathrm{Pri}^{A}_{\varphi\to G}$ of φ to G, it is common knowledge to the agents in G that φ is communicated, whereas the agents in $A\backslash G$ all think that there was no communication at all.

Letting $\mathfrak{L}$ be the fragment of BMS^{A}_{C} obtained by restricting BMS modals to the form $[\mathrm{Pri}^{A}_{\varphi\to G}, 0]$ for some $\varphi \in \mathfrak{L}$ and $G \subseteq A$, we see that $\mathfrak{L}$ and COM^{A}_{C} are equivalent: replacing each BMS modal $[\mathrm{Pri}^{A}_{\varphi\to G}, 0]$ in a formula $\chi \in \mathfrak{L}$ by the modal $[\varphi\to G]$ produces a formula $\chi' \in \mathsf{COM}^{A}_{C}$ such that $\models \chi \equiv \chi'$, and replacing each modal $[\varphi\to G]$ in a formula $\theta \in \mathsf{COM}^{A}_{C}$ by the BMS modal $[\mathrm{Pri}^{A}_{\varphi\to G}, 0]$ produces a formula $\theta' \in \mathfrak{L}$ such that $\models \theta \equiv \theta'$.

The BMS frame for private communication is quite flexible. As an example, pointed BMS frame $(\mathrm{Pri}^{A}_{K_i\varphi\to G}, 0)$ may be used to represent agent i sending a private message of φ to the group G (with $i \in G$), since the content of this private communication is i's knowledge of φ [6].

Our last important BMS frame is the BMS frame for semi-private communication.

Definition 4.10 ([8])**.** Let A be an agent set and let $G \subseteq A$. Then the *semi-private communication of* φ *or* ψ *to* G, written $\frac{1}{2}\mathrm{Pri}^{A}_{\varphi,\psi\to G}$, is the BMS frame $\big(U, S, l\big)$ whose components are defined as follows: $U := \{0, 1\}$, $S_i := \big\{(0,0), (1,1)\big\}$ for each $i \in G$, $S_i := U \times U$ for each $i \in A\backslash G$, $l(0) := \varphi$, and $l(1) := \psi$. See Figure 3 for a picture of the BMS frame $\frac{1}{2}\mathrm{Pri}^{A}_{\varphi,\psi\to G}$.

In the BMS frame $\frac{1}{2}\mathrm{Pri}^{A}_{\varphi,\psi\to G}$, exactly one of φ or ψ is communicated to the agents in the group G. The agents in $A\backslash G$ know this but they do not know which of φ or ψ was in fact communicated.

4.4 Hilbert Theory

Definition 4.11 ([8, 10])**.** Let A be an agent set. The *theory for* BMS^{A}_{C} is given by the following axiom schemes and rules of inference.

- Axiom schemes and rules for K_C^A
- Axiom schemes for communication (in the language BMS_C^A)
    1. $[B,w]p \equiv \big(B(w) \supset p\big)$, for each atom p
    "facts are unchanged by communication"
    2. $[B,w](\psi \supset \chi) \equiv \big([B,w]\psi \supset [B,w]\chi\big)$
    "communication commutes with Boolean connectives"
    3. $[B,w]K_i\psi \equiv \big(B(w) \supset \bigwedge_{wS_iv} K_i[B,v]\psi\big)$
    "knowledge of ψ after a communication comes from having knowledge that the communication will bring about ψ"
    4. $[B,w][B',w']\psi \equiv [B \circ B',(w,w')]\psi$
    "iterated communications are equivalent to their composition"
- Rules for communication
    - $[B,w]$-*Necessitation:* if ψ is provable, then so is $[B,w]\psi$.
    "what is provable holds after a communication"
    - *Action Rule:* given a BMS frame (B,S,l), a world $w \in B$, and a set $G \subseteq A$, suppose that we have a formula $\theta_v \in \mathsf{BMS}_C^A$ for each $v \in B$ satisfying wS_G^*v and, further, that each of the following is provable for each $v \in B$ satisfying wS_G^*v.
        * $\theta_v \supset [B,v]\psi$
        * $B(v) \wedge \theta_v \supset \bigwedge_{i\in G} \bigwedge_{vS_iv'} K_i\theta_{v'}$

    Then we have that $\theta_w \supset [B,w]C_G\psi$ is also provable.

The *theory for* BMS^A is obtained by replacing the axiom schemes and rules for K_C^A by the axiom schemes and rules for K^A, omitting the Action Rule, and restricting the remaining schemes and the Action Rule to the language BMS^A.

Theorem 4.12 ([10, 55])**.** Let A be an agent set. Then for each $\varphi \in \mathsf{BMS}_C^A$, we have that φ is a theorem of the theory for BMS_C^A if and only if $\models \varphi$.

5 Bisimulation and Action Emulation

In this section, we examine bisimulation and a bisimulation-like notion called (action) emulation [36, 57]. These notions give us sufficient conditions for two BMS-type communications to be indistinguishable by formulas in the language BMS_C^A, something we will explain in more detail in a moment. Let us first review the standard definitions for bisimulation in modal logic.

5.1 Standard Bisimulation

For convenience in formulating forthcoming definitions, we begin by defining the notion of *frame-bisimulation*.

Definition 5.1. Let A be an agent set and let $F = (W, R)$ and $F' = (W', R')$ be frames for A. A *frame-bisimulation between (the) frames F and F'* is a nonempty relation $\mathcal{B} \subseteq W \times W'$ satisfying each of the following schemes.

- *Back.* For each $i \in A$: $\Gamma\mathcal{B}\Gamma'$ and $\Gamma' R'_i \Delta'$ together imply that there is a $\Delta \in W$ such that $\Gamma R_i \Delta$ and $\Delta\mathcal{B}\Delta'$.
- *Forth.* For each $i \in A$: $\Gamma\mathcal{B}\Gamma'$ and $\Gamma R_i \Delta$ together imply that there is a $\Delta' \in W'$ such that $\Gamma' R'_i \Delta'$ and $\Delta\mathcal{B}\Delta'$.

A *frame-bisimulation between (the) pointed frames (F, Γ) and (F', Γ') (for A)* is a frame-bisimulation $\mathcal{B}$ between frames F and F' satisfying $\Gamma\mathcal{B}\Gamma'$. We write $(F, \Gamma) \simeq (F', \Gamma')$ to mean that there exists a frame-bisimulation between the pointed frames (F, Γ) and (F', Γ').

If there is a frame-bisimulation between two pointed frames, then any path taken in one frame can be simulated by a path in the other frame. Adding a condition of propositional agreement to this then gives us the usual notion of bisimulation between Kripke models [14].

Definition 5.2. Let A be an agent set and let $M = (W, R, V)$ and $M' = (W', R', V')$ be Kripke models for A. A *bisimulation between (the) Kripke models M and M'* is a nonempty relation $\mathcal{B} \subseteq W \times W'$ satisfying each of the following schemes.

- *Propositional Agreement.* $\Gamma\mathcal{B}\Gamma'$ implies that $\Gamma \in V(p_k)$ if and only if $\Gamma' \in V'(p_k)$ for each $k \in \mathbb{N}$.
- *Frame Bisimulation.* $\mathcal{B}$ is a frame-bisimulation between the frames (W, R) and (W', R').

To say that the bisimulation $\mathcal{B}$ between M and M' is *total* means we have that each $\Gamma \in M$ has a $\Gamma' \in M'$ such that $\Gamma\mathcal{B}\Gamma'$ and that each $\Delta' \in M'$ has a $\Delta \in M$ such that $\Delta\mathcal{B}\Delta'$. To say that the Kripke models M and M' are *bisimilar* means that there is a bisimulation between M and M'; to say that M and M' are *total-bisimilar* means that there is a total bisimulation between M and M'. A *bisimulation between (the) pointed Kripke models (M, Γ) and (M', Γ')* is a bisimulation $\mathcal{B}$ between Kripke models M and M' satisfying $\Gamma\mathcal{B}\Gamma'$. To say that the pointed Kripke models (M, Γ) and (M', Γ') are *bisimilar*, written $(M, \Gamma) \simeq (M', \Gamma')$, means that there exists a bisimulation between (M, Γ) and (M', Γ').

By adapting the standard arguments for modal logic [14] to the language BMS^A_C, we can show that bisimilar pointed Kripke models are indistinguishable to BMS^A_C-formulas and bisimulations are preserved under BMS updates.

Theorem 5.3 ([10, 55])**.** Let A be an agent set and let (M, Γ) and (M', Γ') be bisimilar pointed Kripke models for A. Then each of the following holds.

- For each $\varphi \in \mathsf{BMS}^A_C$, we have that $M, \Gamma \models \varphi$ if and only if $M', \Gamma' \models \varphi$.
- For each pointed BMS frame $(B, v) \in \mathsf{BMS}^A_C$ that is executable at (M, Γ), we have that $\big(M[B], (\Gamma, v)\big) \simeq \big(M'[B], (\Gamma', v)\big)$.

5.2 BMS Frame Bisimulation

Since BMS frames themselves have an underlying frame structure, it is natural to examine a notion of bisimulation for BMS frames.

Definition 5.4. Let A be an agent set and let $B = (U, S, l)$ and $B' = (U', S', l')$ be BMS frames for A such that $B \in \mathsf{BMS}^A_C$ and $B' \in \mathsf{BMS}^A_C$. A *bisimulation between (the) BMS frames B and B'* is a nonempty relation $\mathcal{B} \subseteq U \times U'$ satisfying each of the following schemes.

- *Formula Agreement.* $v\mathcal{B}v'$ implies that $\models B(v) \equiv B'(v')$.
- *Frame Bisimulation.* $\mathcal{B}$ is a frame-bisimulation between the frames (U, S) and (U', S').

To say that the labeled BMS frames B and B' are *bisimilar* means that there is a bisimulation between B and B'. A *bisimulation between (the) pointed BMS frames (B, v) and (B', v')* is a bisimulation $\mathcal{B}$ between BMS frames B and B' satisfying $v\mathcal{B}v'$. To say the the pointed BMS frames (B, v) and (B', v') are *bisimilar*, written $(B, v) \simeq (B', v')$, means that there exists a bisimulation between (B, v) and (B', v').

As one would expect, bisimilar BMS frames preserve bisimulations between Kripke models.

Theorem 5.5 ([10, 55])**.** Let A be an agent set, let (B, v) and (B', v') be bisimilar pointed BMS frames for A in the language BMS^A_C, and let (M, Γ) and (M, Γ') be bisimilar pointed Kripke models for A. Then each of the following holds.

- (B, v) is executable at (M, Γ) if and only if (B', v') is executable at (M', Γ').
- If (B, v) is executable at (M, Γ), then we have $\big(M[B], (\Gamma, v)\big) \simeq \big(M'[B'], (\Gamma', v')\big)$.

5.3 Action Emulation

We saw in Theorem 5.5 that bisimilar BMS frames preserve bisimulations between Kripke models. It follows that if B and B' are bisimilar BMS frames, then we have that B and B' are *equivalent* in the sense of the following definition.

Definition 5.6 ([36, 57])**.** Let A be an agent set and let B and B' be BMS frames for A. To say that B and B' are *equivalent* means that for each Kripke model M for A such that B and B' are each executable in M, we have that $M[B]$ and $M[B']$ are bisimilar.

So bisimilar BMS frames are equivalent by Theorem 5.5. But it is not the case that equivalent BMS frames are bisimilar: letting p be a propositional letter, the BMS frames $\text{Pub}^A_\top$ (Definition 4.8) and $\frac{1}{2}\text{Pri}^A_{p,\neg p\to\emptyset}$ (Definition 4.10) are equivalent but not bisimilar [36, 57]. As a step toward providing a bisimulation-like connection that holds between equivalent BMS frames, the authors of [36, 57] introduced the notion of *(action) emulation*.

Definition 5.7 (Adapted from reformulation in [55] of [36, 57])**.** Let A be an agent set and let $B = (U, S, l)$ and $B' = (U', S', l')$ be BMS frames for A in the language BMS^A_C. An *(action) emulation between (the) BMS frames B and B'* is a nonempty relation $\mathcal{E} \subseteq U \times U'$ satisfying each of the following schemes.

- *Disjunctive Agreement Back.* $v\mathcal{E}v'$ implies there is a set $T' \subseteq U'$ satisfying
  - $v' \in T'$,
  - $v\mathcal{E}w'$ for each $w' \in T'$,
  - $\models B(v) \supset \bigvee_{w'\in T} B'(w')$, and
  - for each $i \in A$ and $u' \in U'$, we have that $\{u'\} \times T' \subseteq S'_i$ or $\{u'\} \times T' \subseteq \big(U'^2 \backslash S'_i\big)$.
- *Disjunctive Agreement Forth.* $v\mathcal{E}v'$ implies there is a set $T \subseteq U$ satisfying
  - $v \in T$,
  - $w\mathcal{E}v'$ for each $w \in T$,
  - $\models B'(v') \supset \bigvee_{w\in T} B(w)$, and
  - for each $i \in A$ and $u \in U$, we have that $\{u\} \times T \subseteq S_i$ or $\{u\} \times T \subseteq \big(U^2 \backslash S_i\big)$.
- *Frame Bisimulation.* $\mathcal{E}$ is a frame-bisimulation between the frames (U, S) and (U', S').

To say that the emulation $\mathcal{E}$ between M and M' is *total* means we have that each $v \in B$ has a $v' \in B'$ such that $v\mathcal{E}v'$ and that each $w' \in B'$ has a $w \in B$ such that $w\mathcal{E}w'$. To say that the BMS frames B and B' are *emulous* means that there is an emulation between B and B'; to say that B and B' are *total-emulous* means that there is a total emulation between B and B'. An *emulation between (the) pointed BMS frames (B, v) and (B', v')* is an emulation $\mathcal{E}$ between BMS frames B and B' satisfying $v\mathcal{E}v'$. To say that the pointed BMS frames (B, v) and (B', v') are *emulous*, written $(B, v) \simeq_e (B', v')$, means that there is an emulation between (B, v) and (B', v').

The following theorem, whose proof is straightforward, says that emulation is weaker than bisimulation.

Theorem 5.8 ([36, 57])**.** Let A be an agent set and let $\mathcal{B}$ be a bisimulation between the BMS frames $B \in \mathsf{BMS}^A_C$ and $B' \in \mathsf{BMS}^A_C$. Then $\mathcal{B}$ is an emulation between B and B'.

Like bisimilar BMS frames, emulous BMS frames preserve bisimulations between Kripke models.

Theorem 5.9 (Adapted from [36, 57])**.** Let A be an agent set, let $B = (U, S, l) \in \mathsf{BMS}_C^A$ and $B' = (U', S', l') \in \mathsf{BMS}_C^A$ be emulous BMS frames for A, and let $M = (W, R, V)$ and $M' = (W', R', V')$ be bisimilar Kripke models for A such that B is executable in M and B' is executable in M'. Now suppose that at least one of the following items is true.

1. There is a $(\Delta, u) \in M[B]$ and a $(\Delta', u') \in M'[B']$ such that $(M, \Delta) \simeq (M', \Delta')$ and $(B, u) \simeq_e (B', u')$.

2. M and M' are total-bisimilar and B and B' are total-emulous.

Then it follows that $M[B]$ and $M'[B']$ are bisimilar.

Proof. We define the relation $\mathcal{B} \subseteq W[B] \times W'[B']$ as follows: $(\Gamma, v)\mathcal{B}(\Gamma', v')$ means that $(M, \Gamma) \simeq (M', \Gamma')$ and $(B, v) \simeq_e (B', v')$. We will show that $\mathcal{B}$ is a bisimulation between $M[B]$ and $M[B']$.

We first show that $\mathcal{B}$ is nonempty. This follows immediately from Item 1. Let us see that it also follows from Item 2. To say that B is executable in M means that there is a $\Gamma \in M$ and $v \in B$ such that $M, \Gamma \models B(v)$. By the totality of the bisimulation between M and M', it follows that there is a $\Gamma' \in M'$ such that $(M, \Gamma) \simeq (M', \Gamma')$. By the totality of the emulation between B and B', it follows that there is a $v' \in M'$ such that $(B, v) \simeq_e (B', v')$. It follows from $M, \Gamma \models B(v)$ and $(B, v) \simeq_e (B', v')$ that there is a $w' \in B'$ such that $(B, v) \simeq_e (B', w')$ and $M, \Gamma \models B'(w')$. Since $(M, \Gamma) \simeq (M', \Gamma')$, we then have that $M', \Gamma' \models B'(w')$ by Theorem 5.3. We thus have that $(\Gamma', w') \in M'[B']$ and, since $(M, \Gamma) \simeq (M', \Gamma')$ and $(B, v) \simeq_e (B', w')$, it again follows that $\mathcal{B}$ is nonempty.

Now that we have shown that $\mathcal{B}$ is nonempty, we must show that $\mathcal{B}$ satisfies the properties of bisimulation: Propositional Agreement, Back, and Forth.

- *Propositional Agreement.* Suppose that $(\Gamma, v)\mathcal{B}(\Gamma', v')$. This implies that $(M, \Gamma) \simeq (M', \Gamma')$. Now $p_k \in V[B](\Gamma, v)$ means that $p_k \in V(\Gamma)$, which is equivalent to $p_k \in V'(\Gamma')$ because $(M, \Gamma) \simeq (M', \Gamma')$. But $p_k \in V'(\Gamma')$ is equivalent to $p_k \in V'[B'](\Gamma', v')$.

- *Back.* Suppose that $(\Gamma, v)\mathcal{B}(\Gamma', v')$ and $(\Gamma', v')R'_i[B'](\Delta', w')$. Now $(\Gamma, v)\mathcal{B}(\Gamma', v')$ means that $(M, \Gamma) \simeq (M', \Gamma')$ and $(B, v) \simeq_e (B', v')$; also, $(\Gamma', v')R'_i[B'](\Delta', w')$ means that $\Gamma' R'_i \Delta'$ and $v' S'_i w'$. From $\Gamma' R'_i \Delta'$ and $(M, \Gamma) \simeq (M', \Gamma')$, it follows that there is a $\Delta \in M$ such that $(M, \Delta) \simeq (M', \Delta')$ and $\Gamma R_i \Delta$. From $v' S'_i w'$, it follows that there is a $w \in B$ such that $(B, w) \simeq_e (B', w')$ and $v S_i w$. Since $(\Delta', w') \in M'[B']$, we have that $M', \Delta' \models B'(w')$. It follows from $M', \Delta' \models B'(w')$ and $(B, w) \simeq_e (B', w')$ that there is a $u \in B$ such that $(B, u) \simeq_e (B', w')$ and $M', \Delta' \models B(u)$. Since $(M, \Delta) \simeq (M', \Delta')$, it follows that $M, \Delta \models B(u)$ by Theorem 5.3. Hence $(\Delta, u) \in M[B]$. Further, since we have that $v S_i w$, that $(B, w) \simeq_e (B', w')$, and that $(B, u) \simeq_e (B', w')$, we then have that $v S_i u$ and thus that $(\Gamma, v)R_i[B](\Delta, u)$. Since $(M, \Delta) \simeq (M', \Delta')$ and $(B, u) \simeq_e (B', w')$, we then have that $(M, \Delta)\mathcal{B}(M', \Delta')$ by the definition of $\mathcal{B}$. Thus the Back condition holds.

- *Forth.* The argument is quite similar to the that for the Back condition. □

Emulation was proposed as a bisimulation-like characterization of equivalent BMS frames [36, 57]. While it remains open whether all emulous BMS frames are equivalent, it has been shown that emulous BMS frames in the language PL of propositional logic are indeed equivalent.

Theorem 5.10 ([36, 57])**.** Let A be an agent set and let $B \in \mathsf{PL}$ and $B' \in \mathsf{PL}$ be emulous BMS frames for A. Then B and B' are equivalent.

6 Expressivity

Expressivity is the comparative study of the propositions expressible in two languages that share a common semantics. The intuitive question this study attempts to answer is the following: can one language say everything that the other language can say?

6.1 Definitions

Definition 6.1. Let $\mathfrak{L}_1$ and $\mathfrak{L}_2$ be languages with a common semantics and let $\mathcal{I}$ be a set of interpretations for $\mathfrak{L}_1$ and $\mathfrak{L}_2$. A *translation function (from $\mathfrak{L}_1$ to $\mathfrak{L}_2$ over $\mathcal{I}$)* is a function $u : \mathfrak{L}_1 \to \mathfrak{L}_2$ that maps each formula $\varphi \in \mathfrak{L}_1$ to a formula $\varphi^u \in \mathfrak{L}_2$ such that for each $\psi \in \mathfrak{L}_1$ and each $I \in \mathcal{I}$, we have $I \models \psi$ if and only if $I \models \psi^u$. We write $\mathfrak{L}_1 \hookrightarrow_{\mathcal{I}} \mathfrak{L}_2$ to mean that there exists a translation function $u : \mathfrak{L}_1 \to \mathfrak{L}_2$ over $\mathcal{I}$. The negation of $\mathfrak{L}_1 \hookrightarrow_{\mathcal{I}} \mathfrak{L}_2$ is written $\mathfrak{L}_1 \not\hookrightarrow_{\mathcal{I}} \mathfrak{L}_2$.

Our informal reading of $\mathfrak{L}_1 \hookrightarrow_{\mathcal{I}} \mathfrak{L}_2$ is "$\mathfrak{L}_2$ can say at least as much as $\mathfrak{L}_1$." This reading leads us to the following definition of relative expressivity.

Definition 6.2 (Relative Expressivity)**.** We adopt the notation of Definition 6.1.

- To say that $\mathfrak{L}_1$ is *more expressive (for $\mathcal{I}$)* than $\mathfrak{L}_2$ means that $\mathfrak{L}_1 \not\hookrightarrow_{\mathcal{I}} \mathfrak{L}_2$ and $\mathfrak{L}_2 \hookrightarrow_{\mathcal{I}} \mathfrak{L}_1$.
- To say that $\mathfrak{L}_1$ and $\mathfrak{L}_2$ are *equally expressive (for $\mathcal{I}$)* means that $\mathfrak{L}_1 \hookrightarrow_{\mathcal{I}} \mathfrak{L}_2$ and $\mathfrak{L}_2 \hookrightarrow_{\mathcal{I}} \mathfrak{L}_1$.
- To say that $\mathfrak{L}_1$ and $\mathfrak{L}_2$ are *expressively incomparable (for $\mathcal{I}$)* means that $\mathfrak{L}_1 \not\hookrightarrow_{\mathcal{I}} \mathfrak{L}_2$ and $\mathfrak{L}_2 \not\hookrightarrow_{\mathcal{I}} \mathfrak{L}_1$.

The definition of $\mathfrak{L}_1 \hookrightarrow_{\mathcal{I}} \mathfrak{L}_2$ is a formalization for the notion of $\mathfrak{L}_2$ saying at least as much as $\mathfrak{L}_1$. This gives us a partial ordering on languages from which we defined the strict partial ordering that is relative expressivity. But note that these definitions all depend on a given set $\mathcal{I}$ of interpretations. In particular, we will see shortly (in Theorems 6.8 and 6.10) how the outcome of an expressivity result hinges on the choice of $\mathcal{I}$.

$$
\begin{array}{lll}
p^u & := & p, \text{ for each atom } p \\
(\varphi \supset \psi)^u & := & \varphi^u \supset \psi^u \\
(K_i\varphi)^u & := & K_i\varphi^u \\
\big([\varphi]p\big)^u & := & \varphi^u \supset p, \text{ for each atom } p \\
\big([\varphi](\psi \supset \chi)\big)^u & := & \big([\varphi]\psi\big)^u \supset \big([\varphi]\chi\big)^u \\
\big([\varphi]K_i\psi\big)^u & := & \varphi^u \supset K_i\big([\varphi]\psi\big)^u \\
\big([\varphi][\psi]\chi\big)^u & := & \big([\varphi \wedge [\varphi]\psi]\chi\big)^u
\end{array}
$$

Figure 4. Inductive definition of the translation function $u : \mathsf{PUB}^A \to \mathsf{ML}^A$ used in the proof of the Plaza-Gerbrandy Theorem (Theorem 6.3).

6.2 Expressivity Relative to Modal Logic

The Plaza-Gerbrandy Theorem marks the beginning of Dynamic Epistemic Logic as an independent area of study.

Theorem 6.3 (Plaza-Gerbrandy [21, 34])**.** Let A be an agent set. Then $\mathsf{PUB}^A \hookrightarrow_{\mathcal{I}} \mathsf{ML}^A$ for each set $\mathcal{I}$ of pointed Kripke models for A.

The translation function $u : \mathsf{PUB}^A \to \mathsf{ML}^A$ used in the proof of the Plaza-Gerbrandy Theorem is given in Figure 4. Since $\mathsf{ML}^A \hookrightarrow_{\mathcal{I}} \mathsf{PUB}^A$ for each set $\mathcal{I}$ of pointed Kripke models for A, the Plaza-Gerbrandy Theorem implies that PUB^A and ML^A are equally expressive for any class of pointed Kripke models for A. Thus public communication does not add expressive power to the language of modal logic without common knowledge. But this does not tell us that the language PUB^A is useless; in fact, PUB^A is *exponentially more succinct* than ML^A, as the following theorem explains.

Theorem 6.4 ([30])**.** Let A be an agent set satisfying $|A| \geq 2$, let $\mathcal{I}$ be the set of all pointed Kripke models for A, and let $t : \mathsf{PUB}^A \to \mathsf{ML}^A$ be a translation function over $\mathcal{I}$. Abbreviations: let $\langle\psi\rangle$ abbreviate $\neg[\psi]\neg$ and let $\hat{K}_i$ abbreviate $\neg K_i \neg$. Choosing $i \in A$ and $j \in A$ such that $i \neq j$, define the sequence $\{\varphi_k\}_{k\in\mathbb{N}}$ of PUB^A-formulas by the following induction.

$$
\varphi_k := \begin{cases} \top & \text{if } k = 0, \\ \langle\langle\varphi_{k-1}\rangle\hat{K}_i\top\rangle\hat{K}_j\top & \text{if } k > 0. \end{cases}
$$

Then it follows that for each $k \in \mathbb{N}$, the number of symbols in φ_k^t is at least 2^k.

It remains open whether Theorem 6.4 holds if we choose for $\mathcal{I}$ the set of all reflexive, transitive, and euclidean pointed Kripke models (for A).

The next theorem lifts the Plaza-Gerbrandy Theorem to the language of BMS Logic.

Theorem 6.5 ([8, 10])**.** Let A be an agent set. Then $\mathsf{BMS}^A \hookrightarrow_{\mathcal{I}} \mathsf{ML}^A$ for each set $\mathcal{I}$ of pointed Kripke models for A.

$$
\begin{array}{lcl}
p^u & := & p, \text{ for each atom } p \\
(\varphi \supset \psi)^u & := & \varphi^u \supset \psi^u \\
(K_i\varphi)^u & := & K_i\varphi^u \\
\big([B,v]p\big)^u & := & B(v)^u \supset p, \text{ for each atom } p \\
\big([B,v](\psi \supset \chi)\big)^u & := & \big([B,v]\psi\big)^u \supset \big([B,v]\chi\big)^u \\
\big([B,v]K_i\psi\big)^u & := & B(v)^u \supset \bigwedge_{vS_iw} K_i\big([B,w]\psi\big)^u \\
\big([B,v][B',v']\chi\big)^u & := & \big([B \circ B',(v,v')]\chi\big)^u
\end{array}
$$

Figure 5. Inductive definition of the translation function $u : \mathsf{BMS}^A \to \mathsf{ML}^A$ used in the proof of the Theorem 6.5. Here $B = (U, S, l)$. The composition $B \circ B'$ is defined in Definition 4.4.

The translation function $u : \mathsf{BMS}^A \to \mathsf{ML}^A$ used in the proof of Theorem 6.5 is given in Figure 5. Since $\mathsf{ML}^A \hookrightarrow_{\mathcal{I}} \mathsf{BMS}^A$ for each set $\mathcal{I}$ of pointed Kripke models for A, Theorem 6.5 implies that BMS^A and ML^A are equally expressive for each set $\mathcal{I}$ of pointed Kripke models for A. Thus generalized communication does not add expressive power to the language of modal logic without common knowledge.

We now survey results concerning the relative expressivity of fragments of COM^A_C.

Theorem 6.6 ([10, 55])**.** Let A be an agent set satisfying $|A| = 1$ and let $\mathcal{I}$ be the set of all pointed Kripke models for A. Then $\mathsf{PUB}^A_C \not\hookrightarrow_{\mathcal{I}} \mathsf{ML}^A_C$.

Proof Hint. The PUB^A_C-formula $[p_0]\neg C_A \neg p_1$ cannot be expressed in ML^A_C [10]. □

Since $\mathsf{ML}^A_C \hookrightarrow_{\mathcal{I}} \mathsf{PUB}^A_C$ for each set $\mathcal{I}$ of pointed Kripke models for A, Theorem 6.6 tells us that public communication strictly increases the expressivity of the language ML^A_C of modal logic with common knowledge for the class of all pointed Kripke models for A. Contrasting this theorem with the Plaza-Gerbrandy Theorem, we see that common knowledge is necessary for this expressivity increase to occur.

Theorem 6.7 ([10])**.** Let A be an agent set satisfying $|A| \geq 2$ and let $\mathcal{I}$ be the set of all reflexive, transitive, and euclidean pointed Kripke models for A. Then $\mathsf{PUB}^A_C \not\hookrightarrow_{\mathcal{I}} \mathsf{ML}^A_C$.

Proof Hint. The PUB^A_C-formula $[p_0]\neg C_G \neg p_1$ with $|G| = 2$ cannot be expressed in ML^A_C. □

Theorem 6.7 tells us that if there are at least two agents in the agent set A, then public communication strictly increases the expressivity of the language ML^A_C of modal logic with common knowledge for the class of all pointed Kripke models for A that are reflexive, transitive, and euclidean.[6] The latter trio of properties characterizes the class of frames valid for the logic $\mathsf{S5}$, a logic generally thought of as the *logic of knowledge* [14, 19]. Thus public communication strictly increases the expressivity of the logic of knowledge when common

[6] Theorem 6.7 fails in the case $|A| = 1$, since $|A| = 1$ implies that $\mathsf{PUB}^A_C \hookrightarrow_{\mathcal{I}} \mathsf{ML}^A_C$ for any set $\mathcal{I}$ of transitive pointed Kripke models for A [10].

$$
\begin{array}{lcl}
p^u & := & p, \text{ for each atom } p \\
(\varphi \supset \psi)^u & := & \varphi^u \supset \psi^u \\
(K_i\varphi)^u & := & K_i\varphi^u \\
(C_G\varphi)^u & := & C_G\varphi^u \\
\big([\varphi\!\rightarrow i]p\big)^u & := & \varphi^u \supset p, \text{ for each atom } p \\
\big([\varphi\!\rightarrow i](\psi \supset \chi)\big)^u & := & \big([\varphi\!\rightarrow i]\psi\big)^u \supset \big([\varphi\!\rightarrow i]\chi\big)^u \\
\big([\varphi\!\rightarrow i]K_j\psi\big)^u & := & \begin{cases} \varphi^u \supset K_j\psi^u & \text{if } j \neq i \\ \varphi^u \supset K_i\big([\varphi\!\rightarrow i]\psi\big)^u & \text{if } j = i \end{cases} \\
\big([\varphi\!\rightarrow i]C_G\psi\big)^u & := & \begin{cases} \big([\varphi\!\rightarrow i]\psi\big)^u \wedge (\varphi^u \supset E_G C_G\psi^u) & \text{if } i \notin G \\ C_i\big([\varphi\!\rightarrow i]\psi\big)^u \wedge C_i(\varphi^u \supset E_{G\setminus\{i\}} C_G\psi^u) & \text{if } i \in G \end{cases} \\
\big([\varphi\!\rightarrow i][\psi\!\rightarrow j]\chi\big)^u & := & \big([\varphi\!\rightarrow i]([\psi\!\rightarrow j]\chi)^u\big)^u
\end{array}
$$

Figure 6. Inductive definition of the translation function $u : \mathsf{PRI1}_C^A \to \mathsf{ML}_C^A$ used in the proof of Theorem 6.8.

knowledge is present. Contrasting this theorem with Theorem 6.5, we again see that common knowledge is necessary for the increase in expressive power.

Finally, we show that single-recipient private communication does not add expressivity to the language of modal logic with common knowledge for any class of *transitive* pointed Kripke models for A.

Theorem 6.8 ([35])**.** Let A be an agent set and let $\mathcal{I}$ be any set of transitive pointed Kripke models for A. Then $\mathsf{PRI1}_C^A \hookrightarrow_{\mathcal{I}} \mathsf{ML}_C^A$.

The translation function $u : \mathsf{PRI1}_C^A \to \mathsf{ML}_C^A$ used in the proof of Theorem 6.8 is given in Figure 6. By choosing $\mathcal{I}$ as a set of transitive pointed Kripke models for A, we choose those Kripke models for which the agents' beliefs are *introspective*, meaning each agent believes his own beliefs. Since each single-recipient communication of agent i's knowledge to a group G having $i \in G$ is in fact a communication received only by i himself, Theorem 6.8 provides a sense in which believing our own beliefs imposes a kind of self-dialog.

6.3 Relative Expressivity of Differing Communications

We begin with a straightforward result that lets us view the language COM_C^A of public and private communication as a fragment of BMS_C^A.

Theorem 6.9. Let A be an agent set. Then $\mathsf{COM}_C^A \hookrightarrow_{\mathcal{I}} \mathsf{BMS}_C^A$ for each set $\mathcal{I}$ of pointed Kripke models for A.

Proof. Let $u : \mathsf{COM}_C^A \to \mathsf{BMS}_C^A$ be the translation function that maps $\varphi \in \mathsf{COM}_C^A$ to the formula $\varphi^u \in \mathsf{BMS}_C^A$ obtained from φ by replacing each instance of a φ-subformula of the form $[\varphi\!\rightarrow G]\psi$ by $[\mathrm{Pri}^A_{\varphi\rightarrow G}, 0]\psi$. (The BMS frame $\mathrm{Pri}^A_{\varphi\rightarrow G}$ is defined in Definition 4.9 and pictured in Figure 2.) □

By identifying COM^A_C with its image in BMS^A_C under the translation function $u : \mathsf{COM}^A_C \to \mathsf{BMS}^A_C$ defined in the proof of Theorem 6.9, we may view COM^A_C as a fragment of BMS^A_C.

Theorem 6.10 ([10])**.** Let A be an agent set satisfying $|A| \geq 2$ and let $\mathcal{I}$ be the set of all pointed Kripke models for A. Then $\mathsf{PRI1}^A_C \not\hookrightarrow_{\mathcal{I}} \mathsf{PUB}^A_C$.[7]

Proof Hint. The $\mathsf{PRI1}^A_C$-formula $[p_0 \to i]\neg C_i K_j p_0$ with $i \neq j$ cannot be expressed in PUB^A_C. □

Theorem 6.10 tells us that if there are at least two agents in the agent set A, then the language PUB^A_C of public communication with common knowledge cannot say everything that can be said by the language $\mathsf{PRI1}^A_C$ of single-recipient private communication with common knowledge. Comparing this result with Theorem 6.5, we again see the necessity of common knowledge for an increase in expressive power.

Theorem 6.8 may look as though it implies the negation of Theorem 6.10: since $\mathsf{ML}^A_C \hookrightarrow_{\mathcal{I}} \mathsf{PUB}^A_C$ for any set $\mathcal{I}$ of transitive pointed Kripke models for A, it follows by Theorem 6.8 that $\mathsf{PRI1}^A_C \hookrightarrow_{\mathcal{I}} \mathsf{PUB}^A_C$ for any set $\mathcal{I}$ of transitive pointed Kripke models for A. But notice that here we require $\mathcal{I}$ to be a set of *transitive* pointed Kripke models, so Theorems 6.8 and 6.10 are not negations of each other—they are simply different statements. We thus see how the outcome of an expressivity result hinges on the choice of $\mathcal{I}$.

Compare the next theorem with Theorem 6.10.

Theorem 6.11 ([35])**.** Let A be an agent set. Then $\mathsf{PUB}^A_C \not\hookrightarrow_{\mathcal{I}} \mathsf{PRI}^A_C$ for the set $\mathcal{I}$ of all pointed Kripke models for A.

Proof Hint. If $|A| = 1$, then $\mathsf{PRI}^A_C = \mathsf{ML}^A_C$ (Definition 3.3) and so the result follows from Theorem 6.6. If $|A| > 1$, then the PUB^A_C-formula $[p_0]\neg C_A \neg p_1$ cannot be expressed in PRI^A_C. □

Theorem 6.11 tells us that the language PRI^A_C of private communication with common knowledge cannot say everything that can be said in the language PUB^A_C of public communication with common knowledge. Since $\mathsf{PRI1}^A_C \hookrightarrow_{\mathcal{I}} \mathsf{PRI}^A_C$ for each set $\mathcal{I}$ of pointed Kripke models for A, combining Theorems 6.10 and 6.11 yields the following result.

Theorem 6.12 ([10, 35])**.** Let A be an agent set satisfying $|A| \geq 2$ and let $\mathcal{I}$ be the set of all pointed Kripke models for A. Then the languages PUB^A_C and PRI^A_C are expressively incomparable for $\mathcal{I}$.

Theorem 6.12 uses the notion of relative expressivity to provide a formal proof that validates our intuition that public and private communication are fundamentally different communicative types.

[7] Theorem 6.10 fails in the case $|A| = 1$ for a trivial reason: $|A| = 1$ implies that $\mathsf{PRI1}^A_C = \mathsf{PUB}^A_C$.

7 Extensions and Embeddings of BMS Logic

7.1 Relativized Common Knowledge

We have seen that public communication does not add expressivity to the language ML^A of modal logic without common knowledge (Theorem 6.3), but that this result fails for the language ML^A_C of modal logic with common knowledge (Theorem 6.6). In [48], it is observed that the failure of the result in the latter case results from the inability of the language ML^A_C to express the concept that the authors of [48] call *relativized common knowledge*.

Definition 7.1. Let A be an agent set. For each of the languages $\mathfrak{L} \in \{\mathsf{ML}^A, \mathsf{PUB}^A\}$, the language of $\mathfrak{L}$ *with relativized common knowledge*, written $\mathfrak{L}_{RC}$, is the extension of $\mathfrak{L}$ obtained by adding the following rule of formula formation: if φ and ψ are formulas and $G \subseteq A$, then $C_G(\varphi, \psi)$ is also a formula.

The new formula $C_G(\varphi, \psi)$ is read, "ψ is common knowledge to the group G relative to φ." The idea is that the formula ψ is common knowledge to the group G subject to the group G making the joint assumption that φ is true. The semantics of the relativized common knowledge modal is given by the following definition.

Definition 7.2. Let A be an agent set. For each $\mathfrak{L} \in \{\mathsf{ML}^A_{RC}, \mathsf{PUB}^A_{RC}\}$, truth of a $\mathfrak{L}$-formula at a pointed Kripke model (M, Γ) for A with $M = (W, R, V)$ is given by an induction on the construction of φ, with the cases of this induction obtained by adding to the cases for $\mathfrak{L}$ (found either in Definition 2.6 or in Definition 3.4) the following inductive case: $M, \Gamma \models C_G(\varphi, \psi)$ means that for each $n \in \mathbb{N}$, if $\{\Gamma_k\}_{k=0}^n$ is a sequence of worlds in M such that $M, \Gamma_k \models \varphi$ for each $k \in \mathbb{N}$ with $k \leq n$, $\Gamma_0 = \Gamma$, and each $k \in \mathbb{N}$ satisfying $k < n$ has an $i \in G$ such that $\Gamma_k R_i \Gamma_{k+1}$, then $M, \Gamma_n \models \psi$.

Note that unlike common knowledge, the statement $M, \Gamma \models C_G(\varphi, \psi)$ of φ being common knowledge relative to ψ requires that the sequence $\{\Gamma_k\}_{k=0}^n$ of worlds in M all satisfy φ. It turns out that this is the key concept that expressively differentiates ML^A_C and PUB^A_C.

Theorem 7.3 ([48])**.** Let A be an agent set. Then $\mathsf{PUB}^A_{RC} \hookrightarrow_{\mathcal{I}} \mathsf{ML}^A_{RC}$ and $\mathsf{PUB}^A_C \hookrightarrow_{\mathcal{I}} \mathsf{ML}^A_{RC}$ for each set $\mathcal{I}$ of pointed Kripke models for A.

The translation function $u : \left(\mathsf{PUB}^A_{RC} \cup \mathsf{PUB}^A_C\right) \to \mathsf{ML}^A_{RC}$ used in the proof of Theorem 7.3 is given in Figure 7. In addition, it is shown in [48] that the expressive relationship that Theorem 7.3 states holds between PUB^A_C and ML^A_{RC} is a strict relationship, which is the content of the following theorem.

Theorem 7.4 ([48])**.** Let A be an agent set. Then ML^A_{RC} is strictly more expressive than PUB^A_C for the set $\mathcal{I}$ of all pointed Kripke models for A.

Proof Hint. The ML^A_{RC}-formula $C_G(p_0, \neg K_i p_0)$ cannot be expressed in PUB^A_C. □

$$
\begin{array}{lcl}
p^u & := & p\text{, for each atom } p \\
(\varphi \supset \psi)^u & := & \varphi^u \supset \psi^u \\
(K_i\varphi)^u & := & K_i\varphi^u \\
(C_G\varphi)^u & := & C_G(\top, \varphi^u) \\
\big([\varphi]p\big)^u & := & \varphi^u \supset p\text{, for each atom } p \\
\big([\varphi](\psi \supset \chi)\big)^u & := & \big([\varphi]\psi\big)^u \supset \big([\varphi]\chi\big)^u \\
\big([\varphi]K_i\psi\big)^u & := & \varphi^u \supset K_i\big([\varphi]\psi\big)^u \\
\big([\varphi]C_G\psi\big)^u & := & \big([\varphi]C_G(\top, \psi)\big)^u \\
\big([\varphi]C_G(\psi, \chi)\big)^u & := & C_G\big(\varphi^u \wedge ([\varphi]\psi)^u, ([\varphi]\chi)^u\big) \\
\big([\varphi][\psi]\chi\big)^u & := & \big([\varphi \wedge [\varphi]\psi]\chi\big)^u
\end{array}
$$

Figure 7. Inductive definition of the translation function $u : \mathsf{PUB}^A_{RC} \to \mathsf{ML}^A_{RC}$ used in the proof of Theorem 7.3.

7.2 Epistemic PDL

A fundamental result in the study of BMS Logic shows that the language of BMS^A_C is a fragment of PDL [48]. This tells us that the expressivity results we surveyed all concern the expressive relationships between various fragments of *epistemic* PDL [48].

Definition 7.5 ([48])**.** Let A be an agent set. The *language of epistemic propositional dynamic logic (for A)*, written PDL^A, consists of the *formulas* φ and the *programs* α built by the following grammar.

$$
\begin{array}{lcl}
\varphi & ::= & p_k \mid \bot \mid \top \mid \varphi_1 \supset \varphi_2 \mid [\alpha]\varphi \\
\alpha & ::= & i \mid ?\varphi \mid \alpha_1;\alpha_2 \mid \alpha_1 \cup \alpha_2 \mid \alpha^* \\
 & & \qquad\qquad k \in \mathbb{N},\ i \in A
\end{array}
$$

Formulas written using other logical connectives are understood as abbreviations for the appropriate formulas in the above language.

PDL^A formulas are interpreted at pointed Kripke models for A according to the usual semantics for PDL [24].

Definition 7.6. Let A be an agent set and let $M = (W, R, V)$ be a Kripke model for A with $\Gamma \in M$ a world in M. By the following simultaneous induction on the construction of PDL^A-formulas and -programs, we both define what it means for a PDL^A-formula to be true at the pointed Kripke model (M, Γ) for A and also associate to each PDL^A-program α a binary relation $[\![\alpha]\!] \in 2^{W \times W}$ on W.

- $M, \Gamma \models p_k$ means that $\Gamma \in V(p_k)$.
- $M, \Gamma \not\models \bot$ and $M, \Gamma \models \top$.

- $M, \Gamma \models \varphi_1 \supset \varphi_2$ means that $M, \Gamma \not\models \varphi_1$ or $M, \Gamma \models \varphi_2$.
- $M, \Gamma \models [\alpha]\varphi$ means that $M, \Delta \models \varphi$ for each $\Delta \in M$ satisfying $\Gamma[\![\alpha]\!]\Delta$.
- $[\![i]\!] := R_i$
- $[\![?\varphi]\!] := \big\{(\Delta, \Delta') \in W^2 : (\Delta = \Delta') \wedge (M, \Delta \models \varphi)\big\}$
- $[\![\alpha_1; \alpha_2]\!] := \big\{(\Delta, \Delta') \in W^2 : (\exists \Omega \in W)(\Delta[\![\alpha_1]\!]\Omega \wedge \Omega[\![\alpha_2]\!]\Delta')\big\}$
- $[\![\alpha_1 \cup \alpha_2]\!] := [\![\alpha_1]\!] \cup [\![\alpha_2]\!]$
- $[\![\alpha^*]\!] := [\![\alpha]\!]^*$, the reflexive-transitive closure of $[\![\alpha]\!]$

The various notions of validity from Definition 2.6 carry over directly to PDL^A-formulas.

The following theorem provides a sense in which BMS^A_C is a fragment of PDL^A.

Theorem 7.7 ([48])**.** Let A be an agent set. Then PDL^A is more expressive than BMS^A_C for the set of all pointed Kripke models for A.

7.3 Temporal Considerations

In the introduction to this paper, we described how a run in a distributed system made up of the agents in the agent set A may be viewed as a sequence

$$(M_1, \Gamma_1), (M_2, \Gamma_2), (M_3, \Gamma_3), \ldots, (M_n, \Gamma_n)$$

of pointed Kripke models for A, where the full description of the k-th state of the run is given by the pointed Kripke model (M_k, Γ_k). We can extend our semantics so that it more closely matches this view of a distributed system.

Definition 7.8. Let A be an agent set. A *run (for A)* is a finite nonempty sequence of pointed Kripke models for A. If $r = \big\{(M_k, \Gamma_k)\big\}_{k=1}^n$ and r' are runs for A and (M, Γ) is a pointed Kripke model for A, then we adopt the following notation.

- $|r|$ denotes the length of the sequence r.
- For each $m \in \mathbb{N}$ with $m \leq n$, we let $r^{(m)}$ denote the prefix $\big\{(M_k, \Gamma_k)\big\}_{k=1}^{n-m}$ of the sequence r. (We let $\big\{(M_k, \Gamma_k)\big\}_{k=1}^{0}$ denote the empty sequence.)
- $\pi_1(r)$ is the model M_n.
- $\pi_2(r)$ is the world Γ_n.
- W^r is the set W' in the pointed Kripke model $\big((W', R', V'), \Gamma'\big) := (M_n, \Gamma_n)$.
- R^r is the function R' in the pointed Kripke model $\big((W', R', V'), \Gamma'\big) := (M_n, \Gamma_n)$.

- V^r is the function V' in the pointed Kripke model $\big((W', R', V'), \Gamma'\big) := (M_n, \Gamma_n)$.
- $r; r'$ is the run consisting of the enumeration of the sequence r followed by the enumeration of the sequence r'.
- $r; (M, \Gamma)$ is the run consisting of the enumeration of r followed by (M, Γ).
- For each $i \in A$, we write $r \to_i r'$ to mean we have that $r^{(1)} = r'^{(1)}$, that $\pi_1(r) = \pi_1(r')$, and that $\pi_2(r) R^r \pi_2(r')$.

We will sometimes identify the pointed Kripke model (M, Γ) for A with the run for A consisting of just the pointed model (M, Γ) itself. But we will only make this identification when doing so ought not cause confusion.

We may interpret BMS^A_C-formulas at a run r for A by a straightforward adaptation of the semantics for BMS^A_C-formulas at pointed Kripke models for A (Definition 4.5).

Definition 7.9. Let A be an agent set. Truth of a formula $\varphi \in \mathsf{BMS}^A_C$ at a run r for A is given by the following induction on the construction of BMS^A_C-formulas.

- $r \models p_k$ means that $\pi_2(r) \in V^r(p_k)$.
- $r \not\models \bot$ and $r \models \top$.
- $r \models \varphi_1 \supset \varphi_2$ means that $r \not\models \varphi_1$ or $r \models \varphi_2$.
- $r \models K_i\varphi$ means that $r' \models \varphi$ for each run r' for A satisfying $r \to_i r'$.
- $r \models C_G\varphi$ means that $r' \models \varphi$ for each run r' for A satisfying $r \to^*_G r'$.
- $r \models [B, w]\varphi$ means that either we have $r \not\models B(w)$ or else we have both $r \models B(w)$ and $r[B, w] \models \varphi$, where the run $r[B, w]$ for $B = (U, S, l)$ is defined as $r; \big(M', (r, w)\big)$ with the Kripke model $M' = (W', R', V')$ for A defined as follows.
  - $W' := \big\{(r', v) \in \mathcal{R}_r \times U : r' \models B(v)\big\}$
    Here $\mathcal{R}_r$ is the set of all runs r' for A such that $r'^{(1)} = r^{(1)}$ and $\pi_1(r') = \pi_1(r)$.
  - For each $i \in A$: R'_i is the set
    $$\Big\{\big((r_1, v_1), (r_2, v_2)\big) \in W'^2 : (r_1 \to_i r_2) \wedge (v_1 S_i v_2)\Big\}$$
  - $V'(p_k) := \big\{(r', v) \in W' : r' \models p_k\big\}$

In this semantics, BMS frames can be used to explicitly generate runs.

Definition 7.10. Let A be an agent set and let $\mathfrak{B} = \big\{(B_k, v_k)\big\}_{k=1}^n$ be a finite nonempty sequence of pointed BMS frames for A such that $(B_k, v_k) \in \mathsf{BMS}^A_C$ for each $k \in \mathbb{N}^+$ satisfying $k \leq n$.

- Let r be a run for A. We define a sequence $\{r_k\}_{k=0}^m$ of runs for A by the following induction.

  - r_0 is defined as the run r.
  - For each $k \in \mathbb{N}^+$ satisfying $k \leq n$, if we defined the run r_{k-1} and we have that $r_{k-1} \models B_k(v_k)$, then the run r_k is defined as $r_{k-1}; r_{k-1}[B_k, v_k]$ (see Definition 7.9).

  The *run generated from* r *by* $\mathfrak{B}$, written $\text{Gen}(\mathfrak{B}, r)$, is the run r_m at the end of the above-defined sequence $\{r_k\}_{k=0}^m$, where $m \in \mathbb{N}$ satisfies $m \leq n$ (with the particular value of m determined by how far the above-specified induction could be carried out).

- To say that the run r for A is *generated* means that there is a finite nonempty sequence $\mathfrak{B}'$ of pointed BMS frames for A and a run r' for A such that $r = \text{Gen}(\mathfrak{B}', r')$.

Thus we see that when we interpret the language BMS_C^A at runs for A, the language itself describes the explicit generation of runs: to evaluate the formula

$$\varphi := [B_1, v_1][B_2, v_2][B_3, v_3] \cdots [B_n, v_n]\psi$$

at a run r, the semantics has us construct $\text{Gen}(\mathcal{B}, r)$, where $\mathcal{B} := \big\{(B_k, v_k)\big\}_{k=1}^n$. We then call φ *true at* the run r if and only if we have either that $|\text{Gen}(\mathcal{B}, r)| < |r| + n$ or else that $|\text{Gen}(\mathcal{B}, r)| = |r| + n$ and ψ is true at $\text{Gen}(\mathcal{B}, r)$. In this way, we see that formulas of the form

$$[B_1, v_1][B_2, v_2][B_3, v_3] \cdots [B_n, v_n]\psi$$

allow us to reason about the possible runs that can be generated from r.

In modeling situations where communication must happen according to a certain prescribed protocol, we must restrict the finite nonempty sequences $\big\{(B_k, v_k)\big\}_{k=1}^n$ of pointed BMS frames for A that may be used in generating a run so as to agree with the protocol. This is the purpose of the next definition.

Definition 7.11 ([47, 46])**.** Let A be an agent set and $\mathfrak{L}$ be a language. A *BMS protocol (for* A *in language* $\mathfrak{L}$*)* is a set $\mathfrak{P}$ of finite nonempty sequences of pointed BMS frames for A in language $\mathfrak{L}$ such that $\mathfrak{P}$ is *prefix-closed*: if $\mathfrak{B} \in \mathfrak{P}$ and the nonempty sequence $\mathfrak{B}'$ of pointed BMS frames for A is a prefix of the sequence $\mathfrak{B}$, then $\mathfrak{B}' \in \mathfrak{P}$.

By choosing a particular BMS protocol, we can specify just those sequences of pointed BMS frames that may be used in generating a run.

Definition 7.12 ([47, 46])**.** Let A be an agent set, let $\mathfrak{P}$ be a BMS protocol for A, and let r be a run for A.

- The *tree generated from* r *by (the) BMS protocol* $\mathfrak{P}$, written $\text{Tree}(\mathfrak{P}, r)$, is defined by

$$\text{Tree}(\mathfrak{P}, r) := \big\{\text{Gen}(\mathfrak{B}, r) : \mathfrak{B} \in \mathfrak{P}\big\} \ .$$

- To say that a set S of runs for A is *generated (by a BMS protocol)* means that there is a BMS protocol $\mathfrak{P}'$ for A and a run r' for A such that $S = \text{Tree}(\mathfrak{P}', r')$.

In viewing the language BMS_C^A in terms of its ability to describe possible generated runs, there are various temporal modalities that we might like to add to our language in order to increase the ability of the language to express properties of generated runs. Let us briefly survey the work that has been done along these lines.

7.3.1 Past-Looking Temporal Modalities

Often our interest is not so much in the runs that may be generated from the current run but instead on the prefixes of this current run, which are just those runs we encountered in the past as the system evolved toward the current run. So if we suppose that the system is in state (M_n, Γ_n) after the generated run

$$(M_1, \Gamma_1), (M_1, \Gamma_1), (M_2, \Gamma_2), \dots, (M_n, \Gamma_n) \enspace ,$$

then we are interested in the *predecessors* of this run, which consist of those runs

$$(M_1, \Gamma_1), (M_1, \Gamma_1), (M_2, \Gamma_2), \dots, (M_j, \Gamma_j)$$

with $j \in \mathbb{N}^+$ satisfying $j < n$. In particular, if for the current run r we have that $|r| > 1$, then we are interested in the *yesterday of* r, which is the nonempty prefix $r^{(1)}$ of H [38]. We will call a modal Y a *yesterday modal* when the semantic condition for truth of a formula $Y\varphi$ at a run r satisfying $|r| > 1$ involves evaluating the truth of φ at the yesterday $r^{(1)}$ of r.

Yap [58] and Sack [37, 38] each consider adding yesterday modals to fragments of BMS_C^A. In particular, they consider two kinds of languages, each of which is defined in terms of a fixed finite frame (U, S) and an agent set A.

- $\mathsf{yBMS}_C^A(U, S)$: *BMS Logic with yesterday for* (U, S) *and* A.

  The formulas of this language are formed using the rules of formula formation for ML_C^A in addition to the following rules.

  - If φ is a formula, then so is $Y\varphi$.
  - If we have that (B, w) is a pointed BMS frame based on (U, S), that $B(v)$ is a formula for each $v \in B$, and that φ is a formula, then $[B, w]\varphi$ is also a formula.

- $\mathsf{ynBMS}_C^A(U, S)$: *BMS Logic with yesterday and nominals for* (U, S) *and* A.

  The formulas of this language are formed using the rules of formula formation for ML_C^A in addition to the following rules.

  - If $w \in U$, then w is a formula.
  - If φ is a formula and $w \in U$, then $Y_w\varphi$ is a formula.
  - If we have that (B, w) is a pointed BMS frame based on (U, S), that $B(v)$ is a formula for each $v \in B$, and that φ is a formula, then $[B, w]\varphi$ is also a formula.

For a finite frame (U, S), let $\mathfrak{L} \in \{\mathsf{yBMS}_C^A(U,S), \mathsf{ynBMS}_C^A(U,S)\}$. Define the BMS protocol $\mathfrak{P}$ as the set of all finite nonempty sequences $\{(B_k, v_k)\}_{k=1}^n$ of pointed BMS frames for A such that each (B_k, v_k) is both in $\mathfrak{L}$ and also based on (U, S). We then interpret $\mathfrak{L}$-formulas at runs coming from the set

$$\bigcup_{(M,\Gamma)} \mathrm{Tree}\big(\mathfrak{P}, (M,\Gamma)\big) \ ,$$

which is just the set of all runs r that are generated both by the length-one run consisting of a pointed Kripke model (M, Γ) for A and by the BMS protocol $\mathfrak{P}$. Truth of an $\mathfrak{L}$-formula φ at a run $r \in \bigcup_{(M,\Gamma)} \mathrm{Tree}\big(\mathfrak{P}, (M,\Gamma)\big)$ is given by extending the induction on formula construction in Definition 4.5 to include the following cases.

- $r \models Y\varphi$ means that either we have $|r| = 1$ or else we have both $|r| > 1$ and $r^{(1)} \models \varphi$.
- $r \models w$ means that $|r| > 1$ and $\pi_2(r) = (\Gamma', w)$.
- $r \models Y_w\varphi$ means that either we have $|r| = 1$ or else we have that $|r| > 1$, that $\pi_2(r) = (\Gamma', w)$, and that $r^{(1)} \models \varphi$.

Another yesterday modal that is not considered in detail in the work of Yap or of Sack is the iterated yesterday modal Y^*, whose interpretation is obtained by adding the following case to the inductive definition of truth of a formula at a run $r \in \bigcup_{(M,\Gamma)} \mathrm{Tree}\big(\mathfrak{P}, (M,\Gamma)\big)$.

- $r \models Y^*\varphi$ means that for each non-negative integer $j \in \mathbb{N}$ with $j < |r|$, we have that $r^{(j)} \models \varphi$.

Since each run is finite, it seems likely that the validities of the language of BMS Logic with the iterated yesterday modal Y^* are finitely axiomatizable, though this remains an open question.

The interested reader is advised to consult [37, 38, 58] for detailed accounts of the yesterday modals Y and Y_w in BMS Logic.

7.3.2 Future-Looking Temporal Modalities

Yesterday modals let us evaluate what was true at the prefix of our current run r. By identifying prefixes of r with past states of our distributed system, yesterday modals let us ask what was true of the system in the past. Following analogy, for us to evaluate what is true of the system in the future, we may introduce modals that evaluate what is true at runs that extend the current run r. In particular, we are interested in those runs r' that are a *tomorrow of* r, by which we mean that r' satisfies the equality $r' = r[B, w]$ (see Definition 7.9) for some pointed BMS frame (B, w). We will call a modal F a *tomorrow modal* when the semantic condition for truth of a formula $F\varphi$ at a run r involves evaluating the truth of φ at at least one of the tomorrows of r.

The language BMS_C^A already has many tomorrow modals: for each pointed BMS frame (B, w) with $r \models B(w)$, the modal $[B, w]$ is a tomorrow modal whose truth condition involves exactly one tomorrow of r, the tomorrow $r[B,w]$. But we may be interested in other tomorrow modals with more flexible temporal structure. We list two interesting tomorrow modals F by describing the inductive condition for the truth of a formula $F\varphi$ at a run r for the agent set A. Terminology: to *add a modal K to a language* $\mathfrak{L}$ means to obtain the extension of $\mathfrak{L}$ given by adding the rule of formula formation that produces a formula $K\varphi$ from each formula φ.

- *The iterated tomorrow modal $[B,w]^*$.*

  Definition. $r \models [B,w]^*\varphi$ means that $r \models [B,w]^n\varphi$ for each $n \in \mathbb{N}$, where the formula $[B,w]^n\varphi$ is defined by the following induction.

  $$[B,w]^n\varphi := \begin{cases} \varphi & \text{if } n = 0, \\ [B,w][B,w]^{n-1}\varphi & \text{if } n > 0. \end{cases}$$

  The first work on iterated tomorrow modals is [31]. Let the language $\mathfrak{L}$ be the extension of ML^A obtained by adding the iterated tomorrow modal $[\mathrm{Pub}^A_\varphi, 0]^*$ (see Definition 4.8) for each formula $\varphi \in \mathfrak{L}$, and let $\mathfrak{L}_C$ be the analogous extension of ML^A_C. It is shown in [31] that the satisfiability problem for each of $\mathfrak{L}$ and $\mathfrak{L}_C$ is Σ^1_1-complete.

  Whether languages with other interesting iterated tomorrow modals are finitely axiomatizable is open, though the work in [31] suggests that the answer for many iterated tomorrow modals ought to be negative.

- *The branching tomorrow modal $F_{\mathfrak{P}}$ for a BMS protocol $\mathfrak{P}$.*

  Definition. $r \models F_{\mathfrak{P}}\varphi$ means that $r' \models \varphi$ for each $r' \in \mathrm{Tree}(\mathfrak{P}, r)$.

  The work [5] studies the following notion of *arbitrary public communication*. Let $\mathfrak{P}$ be the BMS protocol consisting of all length-one runs made up of a pointed BMS frame $(\mathrm{Pub}^A_\varphi, 0)$ for some $\varphi \in \mathsf{BMS}^A_C$. Then $F_{\mathfrak{P}}\varphi$ says that φ is true after each possible public communication, which we may take to be the meaning of the statement "φ is true after an arbitrary public communication." Defining the *language of arbitrary public communication (for A)*, written aPUB^A, as the extension of ML^A obtained by adding the branching tomorrow modal $F_{\mathfrak{P}}$ and the tomorrow modal $[\mathrm{Pub}^A_\varphi, 0]$ for each $\varphi \in \mathsf{aPUB}^A$, it is shown in [5] that the validities of aPUB^A are finitely axiomatizable.

  It is open whether the branching tomorrow modal for other interesting BMS protocols is finitely axiomatizable.

7.3.3 Connections with Epistemic Temporal Logic

We have discussed a temporal view of runs: if r is the current run, then a prefix of r represents a past state of the system and an extension of r represents a possible future state of the system. In the language BMS^A_C of BMS Logic, a future run is generated from a given run r by a specific sequence $\{(B_k, v_k)\}^n_{k=1}$ of pointed BMS frames. We can think of the k-th step in this generation as the occurrence of the communication event described by (B_k, v_k), which is an event whose occurrence takes us from the run

$$r[B_1, v_1][B_2, v_2][B_3, v_3] \cdots [B_{k-1}, v_{k-1}]$$

to the run

$$r[B_1, v_1][B_2, v_2][B_3, v_3] \cdots [B_{k-1}, v_{k-1}][B_k, v_k] \ .$$

To understand what it means for the event described by (B_k, v_k) to occur, we follow Parikh and Ramanujam: identify the meaning of an event with the possible occurrences of that event [32, 33]. Thus if we let a nonempty prefix-closed set $\mathcal{H}$ of generated runs represent the space of possible runs for a given distributed system, then we define the *meaning in $\mathcal{H}$ of the event described by* (B_k, v_k), written $\mathrm{Sem}_{\mathcal{H}}(B_k, v_k)$, by setting

$$\mathrm{Sem}_{\mathcal{H}}(B_k, v_k) := \big\{(r_1, r_2) \in \mathcal{H} \times \mathcal{H} : r_2 = r_1[B_k, v_k]\big\} \ .$$

As we can identify a run with a timeline that provides a discrete, moment-by-moment description of our distributed system—a system whose running constraints are given by $\mathcal{H}$—the meaning in $\mathcal{H}$ of the event e described by (B_k, v_k) is just the changes in the system that are brought about by an occurrence of the event e.

This line of thinking leads to a more general definition: if $\mathcal{H}$ is a nonempty prefix-closed set of runs, then a *semantic event (in $\mathcal{H}$)* is a binary relation on $\mathcal{H}$ [32, 33]. From this perspective, the semantic events described by pointed BMS frames make up only a subset of the set of all semantic events. The language BMS^A_C of BMS Logic is thus the logic of *BMS events*, which are just the semantic events that can be described by a pointed BMS frame. But there are other languages that more directly address the notion of (semantic) event, and so it is fruitful to look at connections between BMS^A_C and these event-based languages.

One event-based language that has been studied in this vein is the language of epistemic temporal logic [4, 46, 47]. Defining this language begins by specifying a set of names for semantic events.

Definition 7.13. An *event set* is a nonempty, at-most–countable set. We refer to the members of an event set as *events*.

With an event set and agent set in hand, we follow [46, 47] and define the basic languages of epistemic temporal logic as follows.

Definition 7.14. Let A be an agent set and Σ be an event set.

- The *language of epistemic temporal logic (for A with events Σ)*, written $\mathsf{ETL}^A(\Sigma)$, is the extension of ML^A obtained by adding the following rule of formula formation: if φ is a formula and $e \in \Sigma$ is an event, then $N_e\varphi$ is also a formula.

- The *language of epistemic temporal logic (for A with events Σ) with common knowledge*, written $\mathsf{ETL}^A_C(\Sigma)$, is the extension of $\mathsf{ETL}^A(\Sigma)$ obtained by adding the following rule of formula formation: if φ is a formula and $G \subseteq A$, then $C_G\varphi$ is also a formula.

We base our semantics for the language of epistemic temporal logic on the work of Parikh and Ramanujam [32, 33].

Definition 7.15 (Adapted from [32, 33])**.** Let A be an agent set and Σ be an event set. A *history model (for A with events Σ)* is a pair $(\mathcal{H}, E)$ whose components are as follows.

- $\mathcal{H}$ is a nonempty prefix-closed set of runs for A.[8] The members of $\mathcal{H}$ are called *histories*.
- $E : \Sigma \to 2^{\mathcal{H}\times\mathcal{H}}$ is a function mapping each event $e \in \Sigma$ to a binary relation $E(e)$ on $\mathcal{H}$. (This function maps each event $e \in \Sigma$ to a semantic event $E(e)$ in $\mathcal{H}$.)

A *pointed history model (for A and Σ)* is a tuple $(\mathcal{H}, E, r)$ consisting of a history model $(\mathcal{H}, E)$ for A with events Σ and a history $r \in \mathcal{H}$. Each formula $\varphi \in \mathsf{ETL}^A_C(\Sigma)$ is interpreted at a pointed history model $(\mathcal{H}, E, r)$ for A according to the following induction on formula construction.

- $\mathcal{H}, E, r \models p_k$ means that $r \models p_k$ (see Definition 7.9).
- $\mathcal{H}, E, r \not\models \bot$ and $\mathcal{H}, E, r \models \top$.
- $\mathcal{H}, E, r \models \varphi_1 \supset \varphi_2$ means that $\mathcal{H}, E, r \not\models \varphi_1$ or $\mathcal{H}, E, r \models \varphi_2$.
- $\mathcal{H}, E, h \models K_i\varphi$ means that $\mathcal{H}, E, r' \models \varphi$ for each $r' \in \mathcal{H}$ satisfying $r \to_i r'$.
- $\mathcal{H}, E, r \models C_G\varphi$ means that $\mathcal{H}, E, r' \models \varphi$ for each $r' \in \mathcal{H}$ satisfying $r \to^*_G r'$.
- $\mathcal{H}, E, r \models N_e\varphi$ means that $\mathcal{H}, E, r' \models \varphi$ for each $r' \in \mathcal{H}$ satisfying $rE(e)r'$.

To connect ETL^A_C with BMS^A_C, we only need observe that a run r for A may itself be viewed as a history model for A: given a BMS protocol $\mathfrak{P}$, the tree $\text{Tree}(\mathfrak{P}, r)$ generated from r is a nonempty prefix-closed set of runs for A. Taking the event set Σ to be the set of all pointed BMS frames for A, we may then define E as the function that maps a BMS frame for A to its semantics.

Definition 7.16. Let A be an agent set, let $\mathfrak{P}$ be a BMS protocol for A, let r be a run for A, and let Σ be the set of all pointed BMS frames for A. The *pointed history model (for A) generated from* $(\mathfrak{P}, r)$, written $\text{His}(\mathfrak{P}, r)$, is defined by

$$\text{His}(\mathfrak{P}, r) := \big(\text{Tree}(\mathfrak{P}, r), E, r\big) \ ,$$

where the function $E : \Sigma \to 2^{\text{Tree}(\mathfrak{P},r)^2}$ is defined by setting

$$E(B, w) := \big\{(r_1, r_2) \in \text{Tree}(\mathfrak{P}, r)^2 : r_2 = r_1[B, w]\big\}$$

for each $(B, w) \in \Sigma$. Observe that $E(B, w) = \text{Sem}_{\text{Tree}(\mathfrak{P},r)}(B, w)$.

[8] To say that a set $\mathcal{H}$ of runs for A is *prefix-closed* means that for each run $r \in \mathcal{H}$, if r' is a run for A such that the (nonempty) sequence r' is a prefix of r, then $r' \in \mathcal{H}$. Note in particular that a prefix-closed set $\mathcal{H}$ of runs does not contain the empty sequence. This makes sense: while the empty sequence is a sequence of pointed Kripke models for A, the empty sequence is not itself a run because it is not nonempty.

$$
\begin{array}{lcl}
p^u & := & p, \text{ for each atom } p \\
(\varphi \supset \psi)^u & := & \varphi^u \supset \psi^u \\
(K_i\varphi)^u & := & K_i\varphi^u \\
(C_G\varphi)^u & := & C_G\varphi^u \\
\big([B,v]\varphi\big)^u & := & N_{(B,v)}\varphi
\end{array}
$$

Figure 8. Inductive definition of a function $u : \mathsf{BMS}_C^A \to \mathsf{ETL}_C^A(\Sigma)$ from Theorem 7.18, where Σ is the set of all pointed BMS frames for A. Note that $[B, v]$ is a modal in the language BMS_C^A, whereas (B, v) is an event in the language $\mathsf{ETL}_C^A(\Sigma)$.

Truth of an $\mathsf{ETL}_C^A(\Sigma)$-formula in a pointed history model $(\mathcal{H}, E, r)$ depends highly on the structure of the set $\mathcal{H}$. As an example: $\mathcal{H}$ need not be closed under all possible BMS events, by which we mean that if $r \in \mathcal{H}$ is a run for A and (B, w) is a pointed BMS frame for A such that $r \models B(w)$, then it need not be the case that $r[B, w] \in \mathcal{H}$. It is as a result of this example that we define a notion of truth for BMS_C^A-formulas that is *relative* to a given BMS protocol for A.

Definition 7.17. Let A be an agent set and let $\mathfrak{P}$ be a BMS protocol for A. For each formula $\varphi \in \mathsf{BMS}_C^A$, the notion of φ being *true relative to* $\mathfrak{P}$ *at* a run r for A, written $r \models_{\mathfrak{P}} \varphi$, is defined by the following induction on BMS_C^A-formula construction.

- If φ is not of the form $[B, w]\varphi$, then the inductive case for φ is obtained from the appropriate inductive case in Definition 7.9 by replacing each occurrence of the turnstile "$\models$" by the turnstile "$\models_{\mathfrak{P}}$".

- $r \models_{\mathfrak{P}} [B, w]\varphi$ means that either $r \not\models_{\mathfrak{P}} B(w)$ or else $r \models_{\mathfrak{P}} B(w)$, $r[B, w] \in \mathfrak{P}$, and $r[B, w] \models_{\mathfrak{P}} \varphi$.

If $\mathcal{I}$ is a set of runs for A, then to say that a formula $\varphi \in \mathsf{BMS}_C^A$ is *valid for* $\mathcal{I}$ *relative to* $\mathfrak{P}$, written $\mathcal{I} \models_{\mathfrak{P}} \varphi$, means that $r' \models_{\mathfrak{P}} \varphi$ for each $r' \in \mathcal{I}$. To say that a formula $\varphi \in \mathsf{BMS}_C^A$ is *valid relative to* $\mathfrak{P}$, written $\models_{\mathfrak{P}} \varphi$, means that φ is valid for the set of all runs for A relative to $\mathfrak{P}$.

Having a notion of protocol-relative truth for BMS_C^A allows for a straightforward connection between the languages ETL_C^A and BMS_C^A.

Theorem 7.18 ([4, 47, 46])**.** Let A be an agent set, let $\mathfrak{P}$ be a BMS protocol for A, and let Σ be the set of all pointed BMS frames for A. Define the function $u : \mathsf{BMS}_C^A \to \mathsf{ETL}_C^A(\Sigma)$ according to the induction in Figure 8. Then for each formula $\varphi \in \mathsf{BMS}_C^A$ and each run r for A, we have $r \models_{\mathfrak{P}} \varphi$ if and only if $\mathrm{His}(\mathfrak{P}, r) \models \varphi^u$.

Theorem 7.18 essentially says that ETL_C^A is at least as expressive as BMS_C^A, though we need to massage some of the details in order to make this precise.

Corollary 7.19 ([4, 47, 46])**.** Let A be an agent set, let Σ be the set of all pointed BMS frames for A, and let $\mathfrak{P}$ be the set of all finite nonempty sequences of pointed BMS frames for A. For each $r \in \mathfrak{P}$ and formula $\chi \in \mathsf{ETL}^A_C(\Sigma)$, we let $r \models \chi$ abbreviate $\mathrm{His}(\mathfrak{P}, r) \models \chi$, thereby providing us with a common semantics for BMS^A_C-formulas and $\mathsf{ETL}^A_C(\Sigma)$-formulas. Then it follows from Theorem 7.18 that we have $\mathsf{BMS}^A_C \hookrightarrow_{\mathcal{I}} \mathsf{ETL}^A_C(\Sigma)$ for each set $\mathcal{I}$ of generated runs for A.

The most comprehensive works studying connections between BMS^A_C and $\mathsf{ETL}^A_C(\Sigma)$ are [46, 47]. In the former work, the authors introduce a notion of frame correspondence that connects the languages BMS^A_C and $\mathsf{ETL}^A_C(\Sigma)$ by identifying structural properties of BMS protocols with validities of $\mathsf{ETL}^A_C(\Sigma)$. This points to the possibility of using epistemic temporal logic as the primary tool for a more general study of the relationship between BMS protocols and their generated trees.

In addition, the work [46] opens up a new study of axiomatization for relative validity by proving the following theorem.

Theorem 7.20 ([46])**.** Let A be an agent set, let Σ be the set of all pointed BMS frames of the form $(\mathrm{Pub}^A_\psi, 0)$ for some $\psi \in \mathsf{BMS}^A_C$, and let S be the set of all BMS protocols $\mathfrak{P}$ for A such that each $\mathfrak{B} \in \mathfrak{P}$ is a finite nonempty sequence over Σ. Then the set

$$\big\{\varphi \in \mathsf{BMS}^A_C : (\forall \mathfrak{P} \in S)(\models_{\mathfrak{P}} \varphi)\big\}$$

of BMS^A_C-validities relative to all BMS protocols for public communication is finitely axiomatizable.

An interesting next-step along the lines of Theorem 7.20 is to identify other interesting sets S of BMS protocols (example: BMS protocols for private communication) and axiomatize the BMS^A_C-validities relative to all of those BMS protocols in S.

Carrying out the work in this section for other natural temporal extensions of $\mathsf{ETL}^A_C(\Sigma)$ (both past-looking and future-looking) and for the corresponding extensions of BMS^A_C would likely be a fruitful area of further research. Suggested extensions may be found in [46].

8 Issues of Doxastic Change

An *update (for a language $\mathfrak{L}$)* is a partial function μ that maps an interpretation I for $\mathfrak{L}$-formulas to another interpretation $\mu(I)$ for $\mathfrak{L}$-formulas. In the language BMS^A_C of BMS Logic, a pointed BMS frame (B, w) induces an update $\mu_{(B,w)}$ that maps a pointed Kripke model (M, Γ) satisfying $M, \Gamma \models B(w)$ to the pointed Kripke model $\mu_{(B,w)}(M, \Gamma) := \big(M[B], (\Gamma, w)\big)$. It is in this sense that the language BMS^A_C of BMS Logic is the logic of the *BMS updates*, the updates induced by pointed BMS frames.

8.1 Successful Updates

BMS updates do not affect the basic facts of the situation we are modeling: a propositional letter p_k is true at a situation (M, Γ) if and only if p_k is true at the updated situation

$\mu_{(B,w)}(M,\Gamma)$. But a BMS update can change the beliefs and knowledge of the agents in our fixed agent set A. As an example, if we have that $M,\Gamma \models p_0 \wedge \left(\bigwedge_{i\in A} \neg K_i p_0\right)$ ("p_0 is true but no one knows it"), then we have that $\mu_{(\mathrm{Pub}^A_{p_0},0)}(M,\Gamma) \models C_A p_0$ ("the public communication that p_0 is true makes it common knowledge that p_0 is true"). In this example, while the update $\mu_{(\mathrm{Pub}^A_{p_0},0)}$ did not affect the truth of the propositional letter p_0, this update did affect the agents' knowledge of the truth of p_0.

Thus while the truth of purely Boolean formulas in BMS^A_C does not change as a result of a BMS update, BMS^A_C-formulas containing a modal K_i sometimes do. A number of authors have looked into the question the formulas that, when true, remain true after certain BMS updates [2, 5, 21, 22, 43, 44, 52]. Most of this work has focused on the *successful* BMS^A_C-formulas, which are the BMS^A_C-formulas that, when true, remain true after they are communicated publicly.

Definition 8.1 ([21, 52])**.** Let A be an agent set and $\varphi \in \mathsf{BMS}^A_C$ be a formula. To say that φ is *successful* means that $[\varphi]\varphi$ is valid.

For the language ML^A of modal logic without common knowledge, there is a syntactic characterization for the set of successful ML^A-formulas.

Theorem 8.2 ([2, 44])**.** Let A be an agent set. Define the fragment $\mathfrak{L}$ of ML^A as the formulas φ given by the following grammar.

$$\begin{aligned} \varphi \;\; ::= \;\; & \neg p_k \mid p_k \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid K_i\varphi \\ & k \in \mathbb{N},\ i \in A \end{aligned}$$

Then a formula $\varphi \in \mathsf{ML}^A$ is successful if and only if there is a $\psi \in \mathfrak{L}$ such that $\models \psi \equiv \varphi$.

But for extensions of ML^A, and in particular for the extensions ML^A_C and PUB^A_C, the best results up to now identify certain successful fragments but do not prove that these fragments contain all of the successful formulas in the language. So these results set out sufficient conditions for a formula in a particular language to be successful, but these results do not set out necessary conditions for a formula in the language to be successful.

Theorem 8.3 ([21])**.** Let A be an agent set. Define the fragment $\mathfrak{L}$ of PUB^A as the formulas $\varphi \in \mathsf{PUB}^A$ such that each subformula $K_i\psi$ of φ appears positively in φ. Then for each formula $\varphi \in \mathsf{PUB}^A$ such that there is a $\psi \in \mathfrak{L}$ with $\models \psi \equiv \varphi$, we have that φ is successful.

Theorem 8.4 ([52])**.** Let A be an agent set. Define the fragment $\mathfrak{L}$ of PUB^A_C as the formulas $\varphi \in \mathsf{PUB}^A_C$ given by the following grammar.

$$\begin{aligned} \varphi \;\; ::= \;\; & \neg p_k \mid p_k \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid K_i\varphi \mid C_G\varphi \mid [\neg\varphi_1]\varphi_2 \\ & k \in \mathbb{N},\ i \in A,\ G \subseteq A \end{aligned}$$

Then for each formula $\varphi \in \mathsf{PUB}^A_C$ such that there is a $\psi \in \mathfrak{L}$ with $\models \psi \equiv \varphi$, we have that φ is successful.

Theorem 8.5 ([5])**.** Let A be an agent set and let aPUB^A be the language of arbitrary public announcements for A (defined in §7.3.2). Define the fragment $\mathfrak{L}$ of aPUB^A by the following grammar.

$$\varphi \ ::= \ \neg p_k \mid p_k \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid K_i\varphi \mid [\neg\varphi_1]\varphi_2 \mid F_{\mathfrak{B}}\varphi \qquad k \in \mathbb{N},\ i \in A$$

Then for each formula $\varphi \in \mathsf{aPUB}^A$ such that there is a $\psi \in \mathfrak{L}$ with $\models \psi \equiv \varphi$, we have that φ is successful.

Providing a syntactic characterization of the successful PUB^A_C-formulas has been an open problem for some time. And the more general question—which formulas that, when true, remain true under other BMS updates—has not yet been addressed at all. For the more general question, a generalized definition of *successful formula for a BMS update* needs to be worked out. The latter definition needs to take into account the fact that a pointed BMS frame $B = (U, S, l)$ with $|U| > 1$ has labelings on its underlying frame (U, S) that vary at more than one world, requiring the notion of *successful formula* for BMS update given by $[B, w]$ to take into account each of the $|U|$ parameters. But for some special BMS frames, such as $\mathrm{Pri}^A_{\varphi\to G}$, the definition is straightforward: to say that $\varphi \in \mathsf{BMS}^A_C$ is *successful for private communication to* G means that $[\varphi \to G]\varphi$ is valid. Nonetheless, these issues have yet to be investigated.

8.2 Epistemic Puzzles

PUB^A_C, the first language of Dynamic Epistemic Logic, was introduced in [34] to reason about knowledge change. It was shown by example in this work how we can properly reason about epistemic puzzles having the following form: some true statement φ is communicated publicly a number of times, after which the statement φ becomes false. What is "puzzling" about these puzzles is fact that announcing a true statement can make that very statement false. Put another way, we are to be puzzled that there are *unsuccessful* formulas (defined as those formulas that are not successful).

There are a number of well-known epistemic puzzles, each of which generally goes by a number of names (of varying political correctness). Since this author is too pained to retell these often-told puzzles, he will simply name a few along with some canonical (but not necessarily original) sources for further reading: the Muddy Children Puzzle [21, 34, 52], the Sum-and-Product Puzzle [34, 53], the Surprise Examination Paradox [22, 52], and card game puzzles [49, 50, 56].

8.3 The Fitch and Moore Paradoxes

Fitch's Paradox of Knowability [15] concerns the derivability of contradiction in epistemic logic if we accept the *verificationist principle*, which we express modally as the *verificationist scheme*

$$\varphi \supset \Diamond K_i\varphi \ .$$

Here the modal $\Diamond$ is a future-looking temporal modal that represents the passage of time—during which we presumably learn more than we already know—leading us to read the verificationist scheme as "any truth will eventually be known."

If we assume that there is some unknown truth, then a contradiction can be derived from the verificationist scheme in the fusion logic $\mathsf{T} \times \mathsf{K}$ consisting of the (epistemic) modal logic T for the modal K_i and the (minimal) modal logic K for the modal $\Box := \neg\Diamond\neg$. To derive the contradiction, assume $p_0 \wedge \neg K_i p_0$ ("p_0 is an unknown truth"), substitute $p_0 \wedge \neg K_i p_0$ for each occurrence of φ in the verificationist scheme, and then reason in $\mathsf{T} \times \mathsf{K}$ [15, 43].

The assumption $p_0 \wedge \neg K_i p_0$ in this derivation is known as *Moore's Paradox (or Problem)*: the sentence $p_0 \wedge \neg K_i p_0$ is a sentence that can be true but cannot be known (by agent i) because i's knowing the first conjunct p_0 makes the second conjunct $\neg K_i p_0$ false [39, 43].[9] Since public communication can be seen as a kind of learning (in that it eliminates from consideration some possible worlds, decreasing the agents' uncertainty as to which is the actual world), we would expect that this *Moore sentence* cannot be learned through public communication. And this is indeed correct: if $p_0 \wedge \neg K_i p_0$ is true at the pointed Kripke model (M, Γ), then the public communication of this formula makes the formula false at the resulting pointed Kripke model:

$$M, \Gamma \models [p_0 \wedge \neg K_i p_0]\neg(p_0 \wedge \neg K_i p_0) \ .$$

So we see that $p_0 \wedge \neg K_i p_0$ is an *unsuccessful* BMS_C^A-formula.

Researchers have recently began to study the reasoning in the Fitch and Moore Paradoxes using tools from Dynamic Epistemic Logic [5, 43]. In particular, the language aPUB^A of arbitrary public announcements (defined in §7.3.2) from [5] has been used to introduce two notions of truth-preservation that ought to be useful in reasoning about the Fitch Paradox.

Definition 8.6. Let A be an agent set and let $\varphi \in \mathsf{aPUB}^A$ be a formula (see §7.3.2 for the definition of aPUB^A).

- To say that φ is *preserved* means that $\models \varphi \supset F_{\mathfrak{B}}\varphi$.
- To say that φ is *knowable* means that $\models \varphi \supset \neg F_{\mathfrak{B}}\neg E_A\varphi$.

So φ is *preserved* exactly when its truth is preserved under all possible public communications of a formula in PUB^A, and φ is *knowable* exactly when the truth of φ implies that the public communication of a formula in PUB^A makes it so that everyone knows φ. The following theorem concerns the known relationships between preserved, successful, and knowable formulas.

Theorem 8.7 ([5])**.** Let A be an agent set and let $\varphi \in \mathsf{aPUB}^A$ be a formula. Recall that φ is *successful* if and only if $\models [\varphi]\varphi$.

[9] The original formulation of Moore's Problem concerns belief ("p but I don't believe it"), though many authors—including van Benthem [43] and Hintikka [25]—consider the knowledge version ("p but I don't know it").

- For each formula φ such that there is a formula ψ in the language $\mathfrak{L}$ of Theorem 8.5 with $\models \varphi \equiv \psi$, we have that φ is preserved.
- If φ is preserved, then φ is successful.
- If φ is successful, then φ is knowable.

Providing an exact syntactic characterization for each of the preserved and knowable formulas is an open problem. Solving the latter might shed some light on the structure of what is knowable, perhaps providing a revised verificationist principle that has the essential spirit of the original verificationist principle without the pathology of the Moore sentence.

9 Other Work

Broadly construed, Dynamic Epistemic Logic is the study of how to reason about model change. Most of this survey has focused on the changes to Kripke models and runs of Kripke models that arise from BMS updates. There is, however, no reason to restrict attention BMS updates on these models. Accordingly, much recent activity in Dynamic Epistemic Logic research has focused on other types of model change. We will briefly mention a few these, directing the reader elsewhere for further reading.

9.1 Valuation Changes

BMS updates do not affect the truth of basic facts; that is, for each pointed Kripke model (M, Γ) for a fixed agent set A, each pointed BMS frame (B, w) for A such that (B, w) is executable at (M, Γ), and each formula $\varphi \in \mathsf{PL}$ in the language of propositional logic, we have that $M, \Gamma \models \varphi$ if and only if $M[B], (\Gamma, w) \models \varphi$. But sometimes we want to model changes of fact. A number of authors have proposed extended languages for reasoning about BMS updates or other updates with valuation changes, and many of these authors consider other kinds of change as well. We invite the reader to consult [7, 17, 28, 48] and the references therein.

9.2 Belief Revision

Belief Revision is the study of how to consistently incorporate new beliefs, even if these new beliefs contradict old beliefs [1, 20]. The popular *AGM approach* to belief revision [1, 20] proposes a number of postulates to be satisfied by theories modeling changes in the propositional beliefs of a single agent. Recent work in Dynamic Epistemic Logic has labored to extend this approach to multi-agent systems having higher-order beliefs, all in a language compatible with other work in Dynamic Epistemic Logic. The interested reader is invited to consult [11, 12, 13, 40, 42, 51, 54] and the references therein for information on this work.

9.3 Updates and Probabilistic Reasoning

Logics for reasoning about static knowledge and probability have been around for some time [18, 23]. Some research in Dynamic Epistemic Logic has looked at dynamic knowledge with probabilities, where we can have not only change in knowledge or belief but also in agents' probability measures. Some of this work has been applied to (probabilistic) belief revision, while some of it has focused purely on dynamic probabilistic reasoning. For information on this work, the reader is invited to consult [3, 11, 12, 26, 27, 45] and the references therein.

References

[1] Carlos E. Alchourrón, Peter Gärdenfors, and David Makinson. On the logic of theory change: Partial meet contraction and revision functions. *Journal of Symbolic Logic*, 50:510–530, 1985.

[2] Hajnal Andréka, István Németi, and Johan van Benthem. Modal languages and bounded fragments of predicate logic. *Journal of Philosophical Logic*, 27:217–274, 1998.

[3] Guillaume Aucher. How our beliefs contribute to interpret actions. In L. Z. Varga, M. Pechoucek, and P. Petta, editors, *Proceedings of the 4th International Central and Eastern European Conference on Multi-Agent Systems (CEEMAS 2005)*, volume 3690 of *Lecture Notes in Artificial Intelligence*, pages 276–286. Springer, 2005.

[4] Guillaume Aucher and Andreas Herzig. From DEL to EDL: Exploring the power of converse events. In *Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, volume 4724 of *Lecture Notes in Computer Science*, pages 199–209. Springer Berlin/Heidelberg, 2007.

[5] Philippe Balbiani, Aleaxandru Baltag, Hans van Ditmarsch, Andreas Herzig, Tomohiro Hoshi, and Tiago de Lima. What can we achieve by arbitrary announcements? A dynamic take on Fitch's knowability. In *Proceedings of the 11th Conference on Theoretical Aspects of Rationality and Knowledge (TARK XI)*, pages 42–51, 2007.

[6] Alexandru Baltag. A logic of epistemic actions. Manuscript, 1999.

[7] Alexandru Baltag. A logic for suspicious players: Epistemic actions and belief-updates in games. *Bulltein of Economic Research*, 54(1):1–45, 2002.

[8] Alexandru Baltag and Lawrence S. Moss. Logics for epistemic programs. *Synthese*, 139(2):165–224, 2004.

[9] Alexandru Baltag, Lawrence S. Moss, and Sławomir Solecki. The logic of common knowledge, public announcements, and private suspicions. In Itzhak Gilboa, editor, *Proceedings of the 7th Conference on Theoretical Aspects of Rationality and Knowledge (TARK VII)*, pages 43–56, Evanston, IL, USA, 1998.

[10] Alexandru Baltag, Lawrence S. Moss, and Sławomir Solecki. Logics for epistemic actions: completeness, decidability, expressivity. Manuscript, 2005.

[11] Alexandru Baltag and Sonja Smets. From conditional probability to the logic of doxastic actions. In *Proceedings of the 11th Conference on Theoretical Aspects of Rationality and Knowledge (TARK XI)*, 2007.

[12] Alexandru Baltag and Sonja Smets. Probabilistic dynamic belief revision. In Johan van Benthem, Shier Ju, and Frank Veltman, editors, *A Meeting of the Minds: Proceedings of the Workshop on Logic, Rationality and Interaction, Beijing, 2007*, volume 8 of *Texts in Computer Science*, London, 2007. College Publications.

[13] Alexandru Baltag, Hans P. van Ditmarsch, and Lawrence S. Moss. Epistemic logic and information update. In Pieter Adriaans and Johan van Benthem, editors, *Handbook on the Philosophy of Information*, pages 369–463. Elsevier, 2008.

[14] Patrick Blackburn, Maarten de Rijke, and Yde Venema. *Modal Logic*. Cambridge University Press, 2001.

[15] Berit Brogaard and Joe Salerno. Fitch's paradox of knowability. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Summer 2004.

[16] Balder ten Cate. Internalizing epistemic actions. In Maricarmen Martinez, editor, *Proceedings of the Student Session of the First North American Summer School in Logic, Language, and Information (NASSLLI-2002)*, pages 109–123, Stanford, CA, USA, 2002.

[17] Gerard R. Renardel de Lavalette. Changing modalities. *Journal of Logic and Computation*, 14(2):253–278, 2004.

[18] Ronald Fagin and Joseph Y. Halpern. Reasoning about knowledge and probability. *Journal of the ACM*, 41(2):340–367, 1993.

[19] Ronald Fagin, Joseph Y. Halpern, Yoram Moses, and Moshe Y. Vardi. *Reasoning about Knowledge*. The MIT Press, 1995.

[20] Peter Gärdenfors. *Knowledge in Flux: Modeling the Dynamics of Epistemic States*. MIT Press, 1988.

[21] Jelle Gerbrandy. *Bisimulations on Planet Kripke*. PhD thesis, University of Amsterdam, 1999. ILLC Dissertation Series DS-1999-01.

[22] Jelle Gerbrandy. The surprise examination in Dynamic Epistemic Logic. *Synthese*, 155:21–33, 2007.

[23] Joseph Y. Halpern and Mark R. Tuttle. Knowledge, probability, and adversaries. *Journal of the ACM*, 40(4):917–962, 1993.

[24] David Harel, Dexter Kozen, and Jerzy Tiuryn. *Dynamic Logic*. The MIT Press, 2000.

[25] Jaakko Hintikka. *Knowledge and Belief*. Cornell University Press, 1962.

[26] Barteld Kooi. *Knowledge, Chance, and Change*. PhD thesis, University of Amsterdam, 2003. ILLC Dissertation Series DS-2003-01.

[27] Barteld Kooi. Probabilistic dynamic epistemic logic. *Journal of Logic, Language, and Information*, 12:381–408, 2003.

[28] Barteld Kooi. Expressivity and completeness for public update logics via reduction axioms. *Journal of Applied Non-Classical Logics*, 17(2):231–253, 2007.

[29] Saul A. Kripke. A completeness theorem in modal logic. *The Journal of Symbolic Logic*, 24(1):1–14, 1959.

[30] Carsten Lutz. Complexity and succinctness of public announcement logic. In Peter Stone and Gerhard Weiss, editors, *Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'06)*, pages 137–144. Association for Computing Machinery (ACM), 2006.

[31] Joseph S. Miller and Lawrence S. Moss. The undecidability of iterated modal relativization. *Studia Logica*, 79:373–407, 2005.

[32] Rohit Parikh and Ramaswamy Ramanujam. Distributed processes and the logic of knowledge. In *Logics of Programs*, volume 193 of *Lecture Notes in Computer Science*, pages 256–268. Springer-Verlag, 1985.

[33] Rohit Parikh and Ramaswamy Ramanujam. A knowledge based semantics of messages. *Journal of Logic, Language, and Information*, 12:453–467, 2003.

[34] Jan Plaza. Logics of public communications. In M. L. Emrich, M. S. Pfeifer, M. Hadzikadic, and Z. W. Ras, editors, *Proceedings of the Fourth International Symposium on Methodologies for Intelligent Systems (ISMIS 1989): Poster Session Program*, pages 201–216. Oak Ridge National Laboratory ORNL/DSRD-24, 1989.

[35] Bryan Renne. Public and private communication are different: Results on relative expressivity. Technical Report TR-2008001, CUNY Ph.D. Program in Computer Science, 2008.

[36] Ji Ruan. Exploring the update universe. Master's thesis, University of Amsterdam, 2004. ILLC Master of Logic Thesis Series MoL-2004-08.

[37] Joshua Sack. Logic for update products and steps into the past. Manuscript, 2007.

[38] Joshua Sack. Temporal languages for epistemic programs. *Journal of Logic, Language, and Information*, 17(2):183–216, 2007.

[39] Roy Sorensen. Epistemic paradoxes. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Fall 2006.

[40] David Steiner. A system for consistency preserving belief change. In Sergei Artemov and Rohit Parikh, editors, *Proceedings of the Workshop on Rationality and Knowledge, 18th European Summer School in Logic, Language, and Information (ESSLLI)*, pages 133–144, Málaga, Spain, 2006.

[41] David Steiner and Thomas Studer. Total public announcements. In Sergei Artemov and Anil Nerode, editors, *Proceedings of Logical Foundations of Computer Science*, volume 4514 of *Lecture Notes in Computer Science*, pages 498–511. Springer, 2007.

[42] Johan van Benthem. Dynamic logic for belief revision. *Journal of Applied Non-Classical Logics*, 14(2):129–155, 2004.

[43] Johan van Benthem. What one may come to know. *Analysis*, 64(2):95–105, April 2004.

[44] Johan van Benthem. One is a lonely number: Logic and communication. In Z. Chatzidakis, P. Koepke, and W. Pohlers, editors, *Logic Colloquium '02*, volume 27 of *Lecture Notes in Logic*. Association for Symbolic Logic, 2006.

[45] Johan van Benthem, Jelle Gerbrandy, and Barteld Kooi. Dynamic update with probabilities. Technical Report PP-2006-21, University of Amsterdam, 2006.

[46] Johan van Benthem, Jelle Gerbrandy, and Eric Pacuit. Merging frameworks for interaction: DEL and ETL. In *Proceedings of the 11th Conference on Theoretical Aspects of Rationality and Knowledge (TARK XI)*, pages 72–81, 2007.

[47] Johan van Benthem and Eric Pacuit. The tree of knowledge in action: Towards a common perspective. In Guido Governatori, Ian Hodkinson, and Yde Venema, editors, *Advances in Modal Logic*, volume 6, pages 87–106. College Publications, 2006.

[48] Johan van Benthem, Jan van Eijck, and Barteld Kooi. Logics of communication and change. *Information and Computation*, 204(11):1620–1662, 2006.

[49] Hans van Ditmarsch. *Knowledge Games*. PhD thesis, University of Groningen, 2000. ILLC Dissertation Series DS-2000-06.

[50] Hans van Ditmarsch. The Russian cards problem. *Studia Logica*, 75:31–62, 2003.

[51] Hans van Ditmarsch. Prolegomena to dynamic logic for belief revision. *Synthese (Knowledge, Rationality, and Action)*, 147:41–87, 2005.

[52] Hans van Ditmarsch and Barteld Kooi. The secret of my success. *Synthese*, 151:201–232, 2006.

[53] Hans van Ditmarsch, Ji Ruan, and Rineke Verbrugge. Model checking sum and product. In T. Dillon and P. Compton, editors, *Proceeding of the Eighth Australian Joint Conference on Artificial Intelligence (AI'05)*, volume 3809 of *Lecture Notes in Computer Science*, pages 183–188. Springer Verlag, 2005.

[54] Hans van Ditmarsch, Wiebe van der Hoek, and Barteld Kooi. Public announcements and belief expansion. In Renate Schmidt, Ian Pratt-Hartmann, Mark Reynolds, and Heinrich Wansing, editors, *Advances in Modal Logic*, volume 5, pages 335–346. College Publications, 2005.

[55] Hans van Ditmarsch, Wiebe van der Hoek, and Barteld Kooi. *Dynamic Epistemic Logic*. Springer, 2007.

[56] Hans van Ditmarsch, Wiebe van der Hoek, Ron van der Meyden, and Ji Ruan. Model checking Russian cards. *Electronic Notes in Theoretical Computer Science*, 149(2):105–123, 2006.

[57] Jan van Eijck and Ji Ruan. Action emulation. Manuscript, December 2005.

[58] Audrey Yap. Product update and looking backward. Technical Report PP-2006-39, Institute for Logic, Language, Information, and Computation, 2006.