How to Agree without Understanding Each Other

Gattinger, Malvin; Wang, Yanjing

Published in:
Electronic Proceedings in Theoretical Computer Science

DOI:
10.4204/EPTCS.297.14

How to Agree without Understanding Each Other: Public Announcement Logic with Boolean Definitions

Malvin Gattinger
Bernoulli Institute, University of Groningen
0000-0002-2498-5073
malvin@w4eg.eu

Yanjing Wang*
Department of Philosophy, Peking University
0000-0002-9499-416X
y.wang@pku.edu.cn

In standard epistemic logic, knowing that p is the same as knowing that p is true, but it does not say anything about understanding p or knowing its meaning. In this paper, we present a conservative extension of Public Announcement Logic (PAL) in which agents have knowledge or belief about both the truth values and the meanings of propositions. We give a complete axiomatization of *PAL with Boolean Definitions* and discuss various examples. An agent may understand a proposition without knowing its truth value or the other way round. Moreover, multiple agents can agree on something without agreeing on its meaning and vice versa.

1 Introduction

Konnyaku Mondo (jelly dialogue) is a story from the traditional Japanese comic storytelling in the *rakugo* form. Quoting [13], the story goes like the following:

> *There was a temple where no monks were living any longer. A devil's tongue jelly maker, named Rokubei, lived next door. He moved into the temple and started pretending to be a monk. One day, a traveling Zen Buddhist monk passed by and challenged Rokubei to a debate on Buddhism, Rokubei had no knowledge on Buddhism and was not able to have a debate. He tried to refuse, but he could not escape and finally agreed. The Buddhist dialogue started but Rokubei didn't know how to perform and therefore, he kept silent. The Buddhist monk tried to communicate to Rokubei in many ways. After some time, Rokubei started responding with gestures to the body movements the monk made. The monk took this as a style of dialogue and tried to answer in gestures, too. They exchanged gestures, and after some time, the monk told Rokubei, "your thoughts are profound and mine are of no comparison. I am very sorry to have bothered you", After saying this, he left the temple.*

In fact, the monk thought "the master" (Rokubei) had expressed deep Buddhist thoughts by his gestures, but Rokubei had never learned any Buddhist thoughts. Rather, from some stage on, he thought the monk was talking badly about his jelly with those gestures, thus he gave the monk a lesson by some angry moves, and apparently defeated the monk.

The intriguing nature of the story is that, as remarked in [13], it seems the proposition *Rokubei has defeated the monk* is common knowledge between the two, but it is based on mutual misunderstanding. The two actually have completely different understanding for the commonly agreed "defeat of the monk". Such mutual misunderstanding also happens a lot in everyday life communications, even in academic exchanges when people "agree" to the same thing due to different interpretations or definitions of the

*Corresponding author

L.S. Moss (Ed.): TARK 2019
EPTCS 297, 2019, pp. 206–220, doi:10.4204/EPTCS.297.14

same concept. See [13] for excellent (and entertaining) interactive discussions about such Konnyaku Mondo phenomena in Game theory.

Mutual misunderstanding is not always harmful. To postpone immediate conflicts and achieve some consensus, it is sometimes even intended to allow respective interpretations of the same proposition, which happens in diplomatic scenarios. For example, two brothers may disagree about who represents officially their father, but they may reach the temporary consensus that there is one and only one successor in order to avoid immediate conflicts.

Philosophically, if we require that knowledge should be at least properly justified as Gettier's examples suggest [11], we can hardly say both Rokubei and the monk "know" that Rokubei has defeated the monk, since the same proposition is justified by two different reasons by different parties. The tricky thing here is that perhaps there is no single "real" justification for the defeat. More crucially, it is debatable in this particular case, whether there is a fixed "real" meaning of the proposition that Rokubei has defeated the monk.

As logicians, the story makes us think about how to represent such situations in the framework of epistemic logic, where $K_i p$ expresses that agent i knows that p. According to the standard Kripke semantics, $K_i p$ merely means that agent i is certain that p is true, and there is nothing about the meaning of p in the semantics. For example, suppose you do not understand Chinese, but someone said a Chinese sentence p and guaranteed you its truth, then it seems you indeed know the *truth value* of p, but without knowing its *meaning*. Now suppose the speaker then tells you that by p, he actually meant $r \wedge q$ in your own language, then you know both the meaning of p and its truth value (and the truth values of q and r). Note that, even when p is uttered in a language that you know, it still can have a different meaning than the surface one, e.g, when a Chinese says "we will think about it later" when asked a yes-no question, it often means "no". It is crucial to see that knowing that *p means φ* is different from knowing that $p \leftrightarrow \varphi$, where the latter is again simply about the truth value. For example, knowing that $p \leftrightarrow \top$ does not imply that *p means $\top$*. We need new techniques to handle knowledge of meanings in epistemic logic.

More generally, knowing the value of something does not imply understanding what information it carries. For example, knowing the ID number of a Chinese person (e.g., 110105198002290022) does not tell you much, unless you know that the first 6 digits encode the residence (Chaoyang District in Beijing), the next 8 digits encode the birth date (29th of February 1980) and so on. Clearly, the interpretation of the structure of the ID is important to your understanding. You may also only know part of the meaning of the message and there are different "layers" of your understanding. You may or may not know that the gender of the person is given by the parity of the second to last digit (in this case even for female). Merely knowing that the first 6 digits code the residence does not tell you the exact city where this person is registered, you may need to know further that the fist three digits code the city and 110 is the code for Beijing. Therefore, by limited knowledge of the structure, you may only get part of the meaning.

Coming back to the propositional setting in this paper, for a sentence φ, you might also just understand some part of it, e.g., knowing what q means in $p \vee q$ but only understand p as $\neg r$ for some incomprehensible r. Now, if others elaborate that r means $q \wedge o$, then you have a deeper understanding of $p \vee q$. We may also enhance our understanding by matching the structure of the proposition — if someone utters $(p \wedge p') \vee p''$ and later explains that it means $q \vee (r \wedge r')$, then we know q means $p \wedge p'$, and p'' means $(r \wedge r')$. The technical goal of this paper is to formally flesh out such reasoning patterns with a minimal extension of public announcement logic [17, 18]. The basic idea is to treat incomprehensible propositions as atomic propositions with *boolean definitions* based on the basic atomic propositions.

Concerning related work, the distinction between knowing the value and knowing the meaning is crucial in cryptographic message passing. The goal of encryption schemes is to hide the meaning of a message, even if the transmitted ciphertext is known [19, 4]. Another related recent attempt [6], based

on a logic of knowing value [22, 9, 1], introduces a functional dependency operator to express that the agent knows a function which can explain the dependency between variables c and d. However, as the author of [6] also remarked, it is not enough to capture the meaning of variables. In [16], the meaning of an utterance by an agent depends on the *type* of the agent, which is a function mapping the uttered proposition to its actual meaning. Similarly, protocols can also give meaning to communicative actions, as demonstrated in [2, 20, 7], where the meaning of an action is defined by the corresponding precondition of the protocol regarding this. We discuss other related work in the conclusion.

In the rest of this paper, we first layout the language and semantics of our logic in Section 2, and then axiomatize it in Section 3, before concluding with ideas for future work.

2 PAL with Boolean Definitions

2.1 Language, Models, Semantics

Throughout the paper our languages and models are parameterized by a set of proposition letters Prop and a finite set of agents I. The language we study consists of two parts: a purely boolean layer for which our models will also provide definitions, and on top of that a version of Public Announcement Logic (PAL) as in [17, 8].

Definition 1 (Language)**.** *The boolean language $\mathscr{L}_B$ is defined by*

$$P ::= p \mid \neg P \mid (P \wedge P)$$

where $p \in \mathsf{Prop}$, a countable set of propositional letters. We already note that the parentheses are essential for pattern matching as we will see later.

The full language $\mathscr{L}$ is given by

$$\varphi ::= P \mid P \equiv P \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid \Box_i\varphi \mid [\varphi]\varphi$$

where $P \in \mathscr{L}_B$ and $i \in I$. We write $Q \not\equiv P$ for $\neg(Q \equiv P)$ and use the standard abbreviations for other boolean operators $\varphi \vee \psi := \neg(\neg\varphi \wedge \neg\psi)$, $\varphi \to \psi := \neg\varphi \vee \psi$ and $\varphi \leftrightarrow \psi := (\varphi \to \psi) \wedge (\psi \to \varphi)$.

For modalities we use the notation $\Box_i$ and avoid the symbol K_i which would suggest knowledge. This is because we will not assume additional modal axioms besides K. Our framework can easily be adapted to multi-agent $KD45$, $S5$ and other logics.

We read $P_1 \equiv P_2$ as "P_1 and P_2 have the same meaning" or "P_1 and P_2 are equivalent by definition". For example, if p means $r \vee r'$ and q means $\neg o$, then our semantics below will also imply that $p \wedge \neg o$ has the same meaning as $((r \vee r') \wedge q)$.

We avoid reading the $\equiv$ operator as "…is defined as …" which would suggest a directedness and uniqueness on the right side. In fact, while our models will contain unique definitions for which one might write $:=$, we do not refer to *the* definition. The formal language can only express that certain propositions are *equivalent by definition*.

A model in our framework is a Kripke model with a local *definition function* DEF_w on each world w, which assigns to each $p \in \mathsf{Prop}$ a definition as its (most thorough) meaning. To stay as general as possible, we *do not* assume any frame property in this paper. Intuitively, if a proposition letter p is assigned itself as the definition, this means that p is *self-evident* or truly basic. Based on those definitions, we can then "unravel" each boolean formula in $\mathscr{L}_B$ to obtain its meaning by recursively applying DEF_w. Further constraints are imposed to avoid circularity and make sure each non-self-evident proposition is assigned a definition using self-evident propositions.

Definition 2 (Models)**.** *A* premodel $\mathscr{M}$ *is a tuple* (W, R, V, DEF) *where* W *is a non-empty set of worlds,* $R_i \subseteq W \times W$ *is a relation for each agent* i*,* $V : W \to \mathsf{Prop} \to \{0,1\}$ *is a valuation function and* DEF*:* $W \to \mathsf{Prop} \to \mathscr{L}_B$ *is a definition function.*

We write V_w *for the valuation at* w *and lift it to* $\mathscr{L}_B$ *as usual by the standard boolean semantics included in Definition 3 below. Similarly, we lift the definition function* DEF_w *at* w *from* Prop *to* $\mathscr{L}_B$ *using definitions from the premodel for atoms and recursing over* $\neg$ *and* $\wedge$*. Formally, let* $def_w : \mathscr{L}_B \to \mathscr{L}_B$ *be defined by:*

$$\begin{array}{rcl} def_w(p) & := & DEF_w(p) \\ def_w(\neg P) & := & \neg def_w(P) \\ def_w((P_1 \wedge P_2)) & := & (def_w(P_1) \wedge def_w(P_2)) \end{array}$$

Intuitively, $def_w(P)$ *is obtained by replacing all the propositions letters in* P *by their definitions.*

A premodel is a model *iff we have for all worlds* w*:*

- *For all* $P, Q \in \mathscr{L}_B$*: If* $def_w(P) = def_w(Q)$*, then* $V_w(P) = V_w(Q)$*.*
- *For all* $p, q \in \mathsf{Prop}$*: If* p *is in* $DEF_w(q)$*, then* $DEF_w(p) = p$*.*

To connect definitions and truthvalues the first model constraint demands that whatever is definitionally equivalent is also assigned the same truth value. We note that only demanding this condition for atomic propositions does not suffice for our purposes.

The second model constraint ensures well-foundedness: Models never contain circular definitions like $p := (p \wedge q)$. Moreover, they also do not contain chains of definitions that would imply such a definition if they were unraveled, for example $p := r$ and $r := (p \wedge q)$. While some of these might actually make sense as fixpoints or infinite conjunctions, for now we do not allow them in our framework.

Definition 3 (Semantics)**.** *We interpret* $\mathscr{L}$ *on models as follows.*

$$\begin{array}{lcl} \mathscr{M}, w \vDash p & \iff & V_w(p) = 1 \\ \mathscr{M}, w \vDash P_1 \equiv P_2 & \iff & def_w(P_1) = def_w(P_2) \\ \mathscr{M}, w \vDash \neg\varphi & \iff & \textit{not } \mathscr{M}, w \vDash \varphi \\ \mathscr{M}, w \vDash (\varphi \wedge \psi) & \iff & \mathscr{M}, w \vDash \varphi \textit{ and } \mathscr{M}, w \vDash \psi \\ \mathscr{M}, w \vDash \Box_i\varphi & \iff & \textit{for all } v : wR_iv \textit{ implies } \mathscr{M}, v \vDash \varphi \\ \mathscr{M}, w \vDash [\varphi]\psi & \iff & \mathscr{M}, w \vDash \varphi \textit{ implies } \mathscr{M}^\varphi, w \vDash \psi \end{array}$$

where $\mathscr{M}^\varphi$ *is the restriction of* $\mathscr{M}$ *to the new set of worlds* $\{v \in W \mid \mathscr{M}, v \vDash \varphi\}$*.*

In the second condition, the $=$ symbol on the right side is *syntactic* equality within $\mathscr{L}_B$. As we mentioned, parentheses in $\equiv$ formulas matter, e.g., $(p \wedge (q \wedge r)) \not\equiv ((p \wedge q) \wedge r)$ is valid. In contrast $(p \wedge (q \wedge r)) \leftrightarrow ((p \wedge q) \wedge r)$ is clearly valid, where we can omit the parentheses. Note that all clauses besides the one for $\equiv$ are standard PAL as in [17, 8]. In particular, the result of announcements is defined as usual, preserving the valuation and now also the local definitions at each world.

2.2 Examples

To illustrate our semantics and to show that it can describe various different scenarios, we now give some examples.

Example 1 (Knowing without Understanding)**.** *As mentioned in the introduction, you can know that something is true without knowing what it means and without knowing that its meaning is true. Figure 1 shows such a model. We use undirected edges to indicate an equivalence relation and we omit the*

self-evident definitions for q and r in all three worlds. At the actual world in the middle we have $\Box_i p \wedge (p \equiv q) \wedge \neg\Box_i(p \equiv q) \wedge \neg\Box_i q$. Moreover, even if $p \leftrightarrow q$ were announced, only the right world would be removed. The agent would then know that $p \leftrightarrow q$ but still not know the stronger $p \equiv q$.

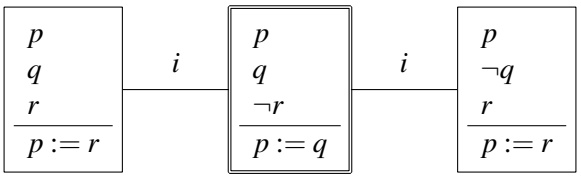

Figure 1: Knowing without Understanding

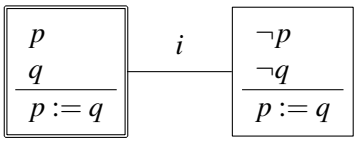

Figure 2: Understanding without Knowing

Example 2 (Understanding without Knowing)**.** *Conversely, an agent can know the meaning of a proposition but still not know whether it is true. Both worlds in the model shown in Figure 2 satisfy $\Box_i(p \equiv q) \wedge \Box_i(p \leftrightarrow q) \wedge \neg\Box_i p$.*

Example 3 (Understanding different parts)**.** *Two agents can also have different partial knowledge of the meaning of some proposition. The middle world of the model in Figure 3 satisfies these three formulas:*

$$\Box_a(p \equiv (q \wedge r)) \wedge \Box_b(p \equiv (q \wedge r))$$

$$\Box_b(p \equiv (\neg q_1 \wedge r)) \wedge \neg\Box_a(p \equiv (\neg q_1 \wedge r))$$

$$\Box_a(p \equiv (q \wedge \neg r_1)) \wedge \neg\Box_b(p \equiv (q \wedge \neg r_1))$$

Note that if the agents would combine their knowledge by announcing both partial meanings, they would arrive at the most thorough meaning, which is more informative than what either of them knows at the moment. Formally, we have $[r \equiv \neg r_1][q \equiv \neg q_1](\wedge_{i=a,b}\Box_i(p \equiv (\neg q_1 \wedge (\neg r_1))))$.

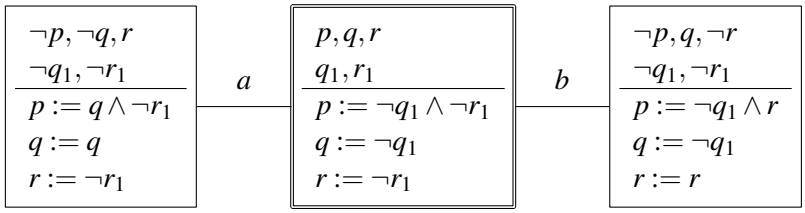

Figure 3: Understanding different parts

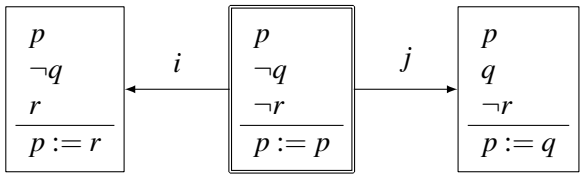

Figure 4: Consensus with misunderstanding

Example 4 (Consensus with misunderstanding)**.** *As in the Konnyaku Mondo example, two agents can agree on something but actually have different beliefs about what it means. In the model from Figure 4, at the middle world, we have $p \wedge \Box_i p \wedge \Box_j p$ but also $\Box_i((p \equiv r) \wedge r \wedge \neg q)$ and $\Box_j((p \equiv q) \wedge q \wedge \neg r)$.*

3 Axiomatization

Note that the addition of $\equiv$ does not invalidate the standard reduction axioms of PAL. We can rather think of $\varphi \equiv \psi$ as a new atomic proposition which is evaluated purely locally.

Definition 4. *A formula $P \equiv Q$ is a* circular *formula iff there is an atomic p such that either (i) $P = p$ and $Q \neq p$ and p occurs in Q or (ii) vice versa.*

For example the formulas $p \equiv (p \wedge q)$ and $\neg q \equiv q$ are both circular, but $p \equiv p$ is not circular.

Definition 5 (Axioms and Rules)**.** *We define the following proof system $\mathbb{SPALD}$ for $\mathscr{L}$. We call the system without the PAL reduction axioms $\mathbb{SMLD}$.*

Axiom Schemes

- *All propositional tautologies.*
- *The K axiom:* $\Box_i(\varphi \to \psi) \to (\Box_i\varphi \to \Box_i\psi)$
- *PAL reduction axioms:*
  - $[\varphi]p \leftrightarrow (\varphi \to p)$
  - $[\varphi](P \equiv Q) \leftrightarrow (\varphi \to (P \equiv Q))$
  - $[\varphi]\neg\psi \leftrightarrow (\varphi \to \neg[\varphi]\psi)$
  - $[\varphi](\psi \wedge \theta) \leftrightarrow ([\varphi]\psi \wedge [\varphi]\theta)$
  - $[\varphi]\Box_i\psi \leftrightarrow (\varphi \to \Box_i(\varphi \to [\varphi]\psi))$
  - $[\varphi][\psi]\xi \leftrightarrow [\varphi \wedge [\varphi]\psi]\xi$
- *Definition axioms*
  - *Reflexivity:* $(P \equiv P)$
  - *Symmetry:* $(P \equiv Q) \to (Q \equiv P)$
  - *Transitivity:* $((P \equiv Q) \wedge (Q \equiv R)) \to (P \equiv R)$
  - *Equivalence:* $(P \equiv Q) \to (P \leftrightarrow Q)$
  - *Atomic* occurrence *substitution:* $((p \equiv Q) \wedge (R \equiv S)) \to (R \equiv [k{:}p \mapsto Q]S)$ *where $k{:}p$ denotes the kth occurrence of p in S.*
  - *Pattern* $\neg$*:* $(\neg P \equiv \neg Q) \leftrightarrow (P \equiv Q)$
  - *Pattern* $\wedge$*:* $((P \wedge Q) \equiv (R \wedge S)) \leftrightarrow ((P \equiv R) \wedge (Q \equiv S))$
  - *Pattern mismatch:* $\neg P \not\equiv (Q \wedge R)$
  - *Non-circularity:* $p \not\equiv P$ *where p occurs in P but $P \neq p$*

Rules

- *Modus Ponens: from φ and $\varphi \to \psi$ infer $\vdash \psi$.*
- *Necessitation for $\Box_i$: from $\vdash \varphi$ infer $\Box_i\varphi$.*

Somewhat non-standard in our axiom system is the usage of occurrence substitutions, in contrast to standard substitutions which usually replace all occurrences of a given atom. We also use the notation $[k{:}p \mapsto Q]$ in proofs in the next section and it will become more clear there.

We note that our logic does not allow replacement of equivalents in $\equiv$ formulas, hence this is not an admissible proof rule. For example, $p \leftrightarrow (p \wedge p)$ is valid and so is $p \equiv p$, but $p \equiv (p \wedge p)$ is not valid and in fact a contradiction, by the non-circularity axiom. Finally, we note that necessitation for $[\varphi]$ is an admissible rule [21].

Theorem 1. *The axiom system $\mathbb{SPALD}$ from Definition 5 is sound for the semantics from Definition 3.*

4 Completeness

To show the completeness of our axiomatization, we can first show the completeness of $\mathbb{SMLD}$ for announcement-free formulas. Then by using the reduction axioms we can obtain the completeness of $\mathbb{SPALD}$ in the usual way [17]. In the following, we write $\vdash \varphi$ if φ is provable in $\mathbb{SMLD}$.

Before we go on, note that our axioms enforce non-circularity but not well-foundedness. That is, the set $\{p_i \equiv p_{i+1} \wedge p_{i+2} \mid i \in \mathbb{N}\}$ is consistent according to the proof system above, but it does not have a model, since you cannot give definitions to p_i using some self-evident atoms. However, any finite subset of this set has a model, hence our logic is not compact and we cannot show strong completeness.

Fortunately, we can still show completeness because any finite set of formulas only uses finitely many atomic propositions.

For the rest of this section, we fix a finite vocabulary Prop.

Example 5. *Consider the following three formulas and an infinite sequence of consequences:*

$$\frac{\begin{array}{l} p \equiv q \wedge r \\ q \equiv p \wedge r \\ s \equiv p \end{array}}{\begin{array}{l} s \equiv q \wedge r \\ s \equiv (p \wedge r) \wedge r \\ s \equiv ((q \wedge r) \wedge r) \wedge r \\ \vdots \end{array}}$$

Note that none of these formulas alone is circular. However, it is easy to spot another consequence $p \equiv (p \wedge r) \wedge r$ which is circular. Hence the original set of three formulas is inconsistent.

The main idea for our completeness proof is that Example 5 is not an exception: Whenever a set of equivalent definitions is infinite we can systematically derive a circular formula. Before stating our central Lemma 1 we need a few more definitions.

Definition 6. *For each boolean formula $P \in \mathscr{L}_B$ we define its* length $l(P)$ *as follows:*

$$\begin{array}{lcl} l(p) & := & 1 \\ l(\neg P) & := & l(P)+1 \\ l((P \wedge Q)) & := & l(P)+l(Q)+3 \end{array}$$

Additionally, we define its vocabulary $v(P)$ *as follows:*

$$\begin{array}{lcl} v(p) & := & \{p\} \\ v(\neg P) & := & v(P) \\ v((P \wedge Q)) & := & v(P) \cup v(Q) \end{array}$$

As an example, $l(\neg p) = 2$ and $l(p \wedge (p \wedge q)) = 9$. Note that the parentheses also count.

Definition 7. *Given a maximally consistent set $\Gamma \subseteq \mathscr{L}(\mathsf{Prop})$, we define a relation over $\mathscr{L}_B(\mathsf{Prop})$ by $P \equiv_\Gamma Q :\iff (P \equiv Q) \in \Gamma$. Per relevant axioms in Definition 5 this is an equivalence relation. For each $P \in \mathscr{L}_B(\mathsf{Prop})$ we denote its $\equiv_\Gamma$-equivalence class by*

$$[P]_\Gamma = \{Q \in \mathscr{L}_B(\mathsf{Prop}) \mid (P \equiv Q) \in \Gamma\}$$

and call it the set of Γ-definitions *of P.*

Lemma 1. *For each maximally consistent set $\Gamma \subseteq \mathscr{L}(\mathsf{Prop})$ and each atomic proposition $p \in \mathsf{Prop}$, the set $[p]_\Gamma$ is finite.*

To prove Lemma 1, we first need some definitions and notation. We fix an enumeration of Prop, say alphabetically. This induces a lexicographic ordering $<$ over $\mathscr{L}_B$. For example, we have $p < q$, therefore also $\neg p < \neg q$ and similarly for conjunctions.

Definition 8. *Let* merge$\colon \mathscr{L}_B \times \mathscr{L}_B \to \mathscr{L}_B$ *be defined as follows:*

$$\begin{array}{lcl}
\mathsf{merge}(p,q) & := & \textit{if } p < q \textit{ then } p \textit{ else } q \\
\mathsf{merge}(p,Q) \ \textit{when } Q \neq q & := & Q \\
\mathsf{merge}(P,q) \ \textit{when } P \neq p & := & P \\
\mathsf{merge}((P \wedge Q),(R \wedge S)) & := & (\mathsf{merge}(P,R) \wedge \mathsf{merge}(Q,S)) \\
\mathsf{merge}(\neg P, \neg R) & := & \neg \mathsf{merge}(P,R) \\
\mathsf{merge}(\neg P,(Q \wedge R)) & := & \textit{undefined} \\
\mathsf{merge}((Q \wedge R), \neg P) & := & \textit{undefined}
\end{array}$$

By definition, merge is symmetric: $\mathsf{merge}(P,Q) = \mathsf{merge}(Q,P)$.

Example 6. *We have* $\mathsf{merge}((p \wedge (q \wedge r)),(\neg s \wedge t)) = (\neg s \wedge (q \wedge r))$.

It is also easy to see that, viewed as term-rewriting rules, merge always terminates, since the recursive clauses reduce the complexity of the formulas.

Definition 9. *Let $[k{:}p \mapsto P]Q$ denote the result of replacing the k-th occurrence of p in Q with P. For multiple such* occurrence substitutions*, let $[k{:}p \mapsto P \oplus k'{:}p' \mapsto R]Q$ denote their simultaneous application to Q.*

Example 7. *We have* $[2{:}p \mapsto (q \wedge r)](p \wedge p) \ = (\ p \wedge (q \wedge r))$. *As an example of two simultaneous occurrence substitutions, we have* $[2{:}p \mapsto (q \wedge r) \ \oplus \ 1{:}q \mapsto \neg r]((p \wedge p) \wedge q) \ = \ ((p \wedge (q \wedge r)) \wedge \neg r)$.

Lemma 2. *For all P,Q such that $P \equiv Q \in \Gamma$ we have:*

(i) $\mathsf{merge}(P,Q) \equiv P \in \Gamma$

(ii) There are indices $k_1,\ldots,k_n \in \mathbb{N}$, atoms $p_1,\ldots,p_n \in \mathsf{Prop}$ and formulas $R_1,\ldots,R_n \in \mathscr{L}_B$ for some fixed $n \geq 0$ such that we have $\mathsf{merge}(P,Q) = [k_n{:}p_n \mapsto R_n \ \oplus \ldots \oplus \ k_1{:}p_1 \mapsto R_1]P$ *and $p_i \equiv R_i \in \Gamma$ for all $i \leq n$. (Note that $n = 0$ iff* $\mathsf{merge}(P,Q) = P$.*)*

Proof. Since Γ is consistent, if $P \equiv Q \in \Gamma$ then $\mathsf{merge}(P,Q)$ is always defined, based on the axioms of pattern mismatch.

For (i) we do induction on the structure of P.

Suppose $P = p$, then $\mathsf{merge}(P,Q)$ is p or Q. Then it is straightforward that $\mathsf{merge}(P,Q) \equiv P \in \Gamma$, since $P \equiv Q \in \Gamma$.

Suppose $P = \neg P'$. Since $P \equiv Q \in \Gamma$, the shape of Q is either q or $\neg Q'$ due to the pattern mismatch axiom and the fact that Γ is consistent. The first case reduces to the above case due to the axiom of symmetry and the fact that merge is symmetric. For the second case, by pattern inference we have $P' \equiv Q' \in \Gamma$. Now by induction hypothesis, $\mathsf{merge}(P',Q') \equiv P' \in \Gamma$. Therefore $\neg\mathsf{merge}(P',Q') \equiv \neg P' \in \Gamma$ by pattern inference axiom, namely $\mathsf{merge}(P,Q) \equiv P \in \Gamma$.

The case of $P = (P_1 \wedge P_2)$ is similar.

For (ii), we also do induction on the structure of P.

Suppose $P = p$, then $\mathsf{merge}(P,Q)$ is p or Q. In the first case we just need a trivial substitution $[1{:}p \mapsto p]$ since $p \equiv p \in \Gamma$. In the second case we take $[1{:}p \mapsto Q]$ since $p \equiv Q \in \Gamma$.

Suppose $P = \neg P'$. As in the proof of (i) we can show that Q is in the shape of either q or $\neg Q'$. For the first case, $\mathsf{merge}(P,q) = P = \neg P'$ by definition of merge. Then we can take the trivial substitution to prove the claim. In the second case, $\mathsf{merge}(P,Q) = \neg\mathsf{merge}(P',Q')$. Since $P \equiv Q \in \Gamma$, $P' \equiv Q' \in \Gamma$ by pattern inference. Now by induction hypothesis, there are substitutions to turn P' into $\mathsf{merge}(P',Q')$. The same substitutions can also turn $P = \neg P'$ into $\mathsf{merge}(P,Q) = \neg\mathsf{merge}(P',Q')$.

Again the case of $P = (P_1 \wedge P_2)$ is similar. □

Lemma 3. *For all $p \in \mathsf{Prop}$ and all $P,Q \in [p]_\Gamma$ we have:*

(i) $\mathsf{merge}(P,Q) \in [p]_\Gamma$

(ii) $l(\mathsf{merge}(P,Q)) \geq \max(l(P),l(Q))$

(iii) There are indices $k_1,\ldots,k_n \in \mathbb{N}$, atoms $p_1,\ldots,p_n \in \mathsf{Prop}$ and formulas $R_1,\ldots,R_n \in \mathscr{L}_B$ for some fixed $n \geq 0$ such that we have $\mathsf{merge}(P,Q) = [k_n{:}p_n \mapsto R_n \ \oplus \ldots \oplus \ k_1{:}p_1 \mapsto R_1]P$ and $p_i \equiv R_i \in \Gamma$ for all $i \leq n$.

Proof. For (i): Suppose $P,Q \in [p]_\Gamma$. Then $P \equiv Q \in \Gamma$ and according to part (i) of Lemma 2 we have $\mathsf{merge}(P,Q) \equiv P \in \Gamma$. Since $P \in [p]_\Gamma$, thus by transitivity axiom, $\mathsf{merge}(P,Q) \in [p]_\Gamma$.

For (ii): a simple induction on the clauses of merge suffices.

Finally, (iii) is a special case of part (ii) in Lemma 2 since $P,Q \in [p]_\Gamma$ implies $P \equiv Q \in \Gamma$. □

Intuitively, Lemma 3 says that the result of merging two Γ-definitions of p (i) is also a Γ-definition of p, (ii) is at least as long as the longest given formula, and (iii) can be reached by replacing atom occurrences step by step.

In fact, the occurrence substitutions given by (iii) are unique up to enumeration and trivial substitutions of the form $[k{:}p \mapsto p]$. Moreover, the pattern matching axioms imply that $R_i \in [p_i]_\Gamma$.

Proof of Lemma 1. Suppose $[p]_\Gamma$ is infinite.

Then in particular the length of formulas in $[p]_\Gamma$ is unbounded, because Prop is finite. Hence there must be an infinite chain $P_0 = p, P_1, P_2, \ldots$ in $[p]_\Gamma$ which is unbounded in length. Note that there might be big "jumps" in length and in general the formulas will not be related systematically.

To deduce a circular formula and thus a contradiction, we now define a second chain $Q_0, Q_1, \ldots$ using merge. Let $Q_0 := P_0 = p$ and for all $k \geq q$ let $Q_k := \mathsf{merge}(Q_{k-1}, P_k)$. From Lemma 3 we now get that (i) the chain of Q_is is also a chain in $[p]_\Gamma$, (ii) $l(Q_i) \geq l(P_i)$ and thus this chain is also unbounded in length, and (iii) for each step from Q_i to Q_{i+1} there are finitely many atoms that are replaced with longer formulas.

Let us list such a sequence of substitutions as:

$$Q_0 \xrightarrow{\overline{k{:}p\mapsto R}} Q_1 \xrightarrow{\overline{m{:}q\mapsto T}} Q_2 \ \cdots$$

where $\overline{k{:}p \mapsto R}$ denotes the simultaneous substitution $[k_n{:}p_n \mapsto R_n \ \oplus \ldots \oplus \ k_1{:}p_1 \mapsto R_1]$ for some n. We can denote each single substitution as $[k : p \mapsto R]^i$ where the superscript i means the substitution happens at Q_i.

Now let $(N, \rightsquigarrow)$ be the graph where N is the set of all occurrence substitutions in the Q chain (indexed with superscripts) and there is an edge $[m{:}p \mapsto R]^i \rightsquigarrow [n{:}q \mapsto S]^j$ iff $i < j$ and $n{:}q$ is an occurrence within R of Q_{i+1}. Then $(N, \rightsquigarrow)$ is a tree with the first substitution $[1{:}p \mapsto P_1]^0$ as its root. Intuitively, we have an edge from one substitution to another iff the second "happens within" the result of the first. This tree of substitutions is illustrated in Figures 5 and 6 below as examples.

The Q_i chain of formulas is infinite, hence there are infinitely many substitutions and N is infinite. However, each occurrence of an atomic proposition can be replaced with a longer formula only once. Hence the number of children of each node in $(N, \rightsquigarrow)$ is bounded by the length of the formula. Formally, each node $[k{:}p \mapsto R]^i$ can only have as many children as there are occurrences of atoms in R.

Together, $(N, \rightsquigarrow)$ is an infinite but also finitely branching tree. By Kőnig's Lemma [15, 10] there must be an infinite branch. In particular, there must be a branch longer than $|\mathsf{Prop}|$ and there must be two substitutions of the same atom along this branch. We now use this branch to derive a circular formula in Γ.

Let R_i be the sequence of formulas starting with $R_0 := p$ and then applying the substitutions from the branch. For each i, let $[k_i{:}q_i \mapsto S_i]^{\iota_i}$ be the substitution at node i happening at Q_{ι_i}. Note that the branch need not have a node at every level, hence $\iota_i \geq i$ but not necessarily $\iota_i = i$.

Because $|\mathsf{Prop}|$ is finite there must be $j < k$ such that $q_j = q_k =: q$. Then we have $q \in R_j$ and $q \in R_k$ and $S_j, S_k \in [q]_\Gamma$. Now consider the sequence of substitutions Rs:

$$\begin{array}{rcl} q \text{ occurs in } R_j & & \\ q \text{ does not occur in } R_{j+1} & = & [k_{j+1}{:}q \mapsto S_j]^{\iota_j} R_j \\ \vdots & & q \text{ must be reintroduced by some step } [k_m{:}q_m \mapsto S_m]^{\iota_m} \\ q \text{ occurs in } R_k & & \\ q \text{ does not occur in } R_{k+1} & = & [k_{k+1}{:}q \mapsto S_k]^{\iota_k} R_k \end{array}$$

In particular, because these substitutions happen along the same branch, $k_m{:}q_m$ is an occurrence in Q_j. Hence, reasoning inside Γ we have $q \equiv Q_j$ and $q \equiv [k_m{:}q_m \mapsto Q_m]^{\iota_m} Q_j$. But because q occurs in Q_m, the latter is a circular formula in Γ. Contradiction! □

To illustrate our proof method, we give two examples.

Example 8. *The chain of Q_i might for example be all right parts of consequences in Example 5. That is, in the set $[s]_\Gamma$ we have $Q_0 = p$, $Q_1 = (q \wedge r)$, $Q_2 = ((p \wedge r) \wedge r)$, $Q_3 = (((q \wedge r) \wedge r) \wedge r)$, and so on. The sequence of occurrence substitutions is then $[1{:}p \mapsto (q \wedge r)]^0$, $[1{:}q \mapsto (p \wedge r)]^1$, $[1{:}p \mapsto (q \wedge r)]^2$, and so on. This is an easy case: each step replaces an occurrence within the previous substituens. Hence the tree of substitutions shown in Figure 5 only has one branch.*

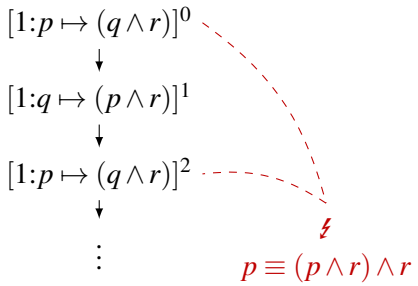

Figure 5: A simple substitution tree and the resulting circular formula.

Example 9. *An example which yields a proper tree is shown in Figure 6. On the left side we first show the P_i chain. Note that its formulas are strictly increasing in complexity, but for example P_2 and P_3 are not directly related via substitutions. The second chain Q_i is obtained using the* merge *function from Definition 8 and restores this property. For example, we can go from Q_2 to Q_3 via the substitution*

$$P_0 = p \qquad Q_0 = p$$
$$P_1 = q \wedge r \qquad Q_1 = q \wedge r$$
$$P_2 = (u \vee r) \wedge r \qquad Q_2 = (u \vee r) \wedge r$$
$$P_3 = q \wedge \neg\neg s \qquad Q_3 = (u \vee r) \wedge \neg\neg s$$
$$P_4 = (u \vee \neg\neg s) \wedge \neg\neg(r \to p) \qquad Q_4 = (u \vee \neg\neg s) \wedge \neg\neg(r \to p)$$
$$P_5 = q \wedge \neg\neg(\neg\neg s \to p) \qquad Q_5 = (u \vee \neg\neg s) \wedge \neg\neg(\neg\neg s \to p)$$
$$\vdots \qquad\qquad \vdots$$

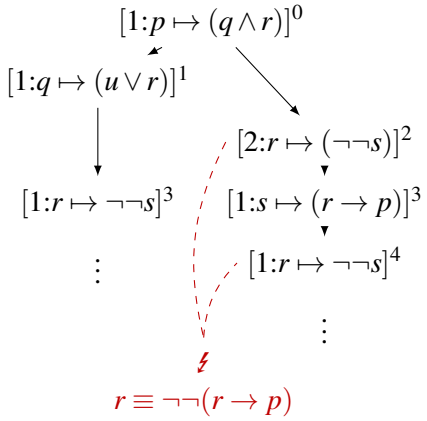

Figure 6: A definition chain and the resulting occurrence substitution tree. The right branch yields a circular formula.

$[2{:}r \mapsto \neg\neg s]^2$. *All those substitutions are then arranged in the* $(N, \rightsquigarrow)$ *tree. Finally, we show how two substitutions of the same proposition* (r) *in the same branch lead to a circular formula* $(r \equiv \neg\neg(r \to p))$.

Given that all $[p]_\Gamma$ sets are finite, we can now choose a definition for each $p \in \mathsf{Prop}$, namely the longest and among those the lexicographically least formula in $[p]_\Gamma$. In fact, this is the same as applying all possible substitutions until reaching the leaves in the substitution trees from the proof of Lemma 1.

Definition 10. *For any finite set of boolean formulas* $X \subseteq \mathscr{L}_B$*, we define* $\mathsf{pick}(X)$ *as the longest element of* X *and among those the lexicographically first.*

Formally, let $\mathsf{pick}(X) := \min_<\{P \in X \mid \forall Q \in X : l(P) \geq l(Q)\}$

Definition 11 (Canonical Model)**.** *For a finite vocabulary* Prop *the corresponding canonical model* $\mathscr{M} = (W, R, V, DEF)$ *is defined as follows:*

- $W := \{\Gamma \subseteq \mathscr{L}(\mathsf{Prop}) \mid \Gamma$ *is maximally consistent*$\}$
- $\Gamma_1 R_i \Gamma_2 :\iff \{\varphi \mid \Box_i \varphi \in \Gamma_1\} \subseteq \Gamma_2$
- $V(\Gamma) := \Gamma \cap \mathsf{Prop}$
- *For each* Γ *and any* $p \in \mathsf{Prop}$*, let* $DEF_\Gamma(p) := \mathsf{pick}([p]_\Gamma)$

Lemma 4. *The canonical model is indeed a model, and not just a premodel.*

Proof. We need to show two properties.

- For all $P, Q \in \mathscr{L}_B$: If $\mathsf{def}_w(P) = \mathsf{def}_w(Q)$, then we also have $V_w(P) = V_w(Q)$. This follows from the equivalence axiom $(P \equiv Q) \to (P \leftrightarrow Q)$ and the boolean part of the Truth Lemma 6 below which can be shown independently.
- For all $p, q \in \mathsf{Prop}$, we need to show that if p occurs in $\mathsf{DEF}_w(q)$, then $\mathsf{DEF}_w(p) = p$. Now, take any p in $\mathsf{DEF}_w(q)$. Because $\mathsf{DEF}_w(q)$ is among the longest formulas in $[q]_\Gamma$, there cannot be any $P \in [p]_\Gamma$ with $l(P) > 1$. Namely, $[p]_\Gamma$ only consists of propositional letters.

  Because $\mathsf{DEF}_w(q)$ is the lexicographically first among the longest formulas in $[q]_\Gamma$, also p must be lexicographically first in $[p]_\Gamma$, for otherwise we can replace it in $\mathsf{DEF}_w(q)$ by another propositional letter which is lexicographically smaller than p to obtain a lexicographically smaller formula in $[q]_\Gamma$. Together, we have $\mathsf{DEF}_w(p) = p$. □

Lemma 5. *In the canonical model we have for all $P \in \mathscr{L}_B(\mathsf{Prop})$ that $\mathsf{def}_\Gamma(P) = \mathsf{pick}([P]_\Gamma)$.*

Proof. By induction on the structure of P. The base case $P = p$ follows from Definition 11.

For the induction step, consider the case $P = \neg Q$. Then we have the following chain of identities:

$$\mathsf{def}_\Gamma(\neg Q) \overset{\text{Def. 11}}{=} \neg\mathsf{def}_\Gamma(Q) \overset{\text{IH}}{=} \neg\mathsf{pick}([Q]_\Gamma) \overset{*}{=} \mathsf{pick}([\neg Q]_\Gamma)$$

where the step IH is by induction hypothesis and the step $*$ is shown as follows. By definition of pick we have that $\mathsf{pick}([\neg Q]_\Gamma)$ is

$$\min_<\{P \in [\neg Q]_\Gamma \mid \forall R \in [\neg Q]_\Gamma : l(P) \geq l(R)\}$$

which by pattern matching is the same as

$$\min_<\{\neg P \mid P \in [Q]_\Gamma \text{ and } \forall R \in [Q]_\Gamma : l(\neg P) \geq l(\neg R)\}.$$

Because $<$ and $\geq$ with respect to $l(\cdot)$ is preserved under negation, this is the same as

$$\neg\min_<\{P \in [Q]_\Gamma \mid \forall R \in [Q]_\Gamma : l(P) \geq l(R)\}$$

which is $\neg\mathsf{pick}([Q]_\Gamma)$.

A similar chain covers the case $P = (Q_1 \wedge Q_2)$. □

Lemma 6 (Truth Lemma)**.** *Consider the canonical model $\mathscr{M}$. For all worlds Γ and all formulas φ without announcement operators we have $\mathscr{M}, \Gamma \vDash \varphi$ iff $\varphi \in \Gamma$.*

Proof. By induction on the complexity of φ. The only non-standard case is the $\equiv$ operator. We want to show

$$P \equiv Q \in \Gamma \iff \mathscr{M}, \Gamma \vDash P \equiv Q$$

for which it suffices to show

$$[P]_\Gamma = [Q]_\Gamma \iff \mathsf{def}_\Gamma(P) = \mathsf{def}_\Gamma(Q).$$

For left to right, suppose $[P]_\Gamma = [Q]_\Gamma$. This implies $\mathsf{pick}([P]_\Gamma) = \mathsf{pick}([Q]_\Gamma)$. Hence we have $\mathsf{def}_\Gamma(P) = \mathsf{def}_\Gamma(Q)$ by Lemma 5.

For right to left, suppose we have $\mathsf{def}_\Gamma(P) = \mathsf{def}_\Gamma(Q)$. Then by Lemma 5 we have a single formula $R := \mathsf{pick}([P]_\Gamma) = \mathsf{pick}([Q]_\Gamma)$. In particular we have $R \in [P]_\Gamma$ and $R \in [Q]_\Gamma$. Hence $P \equiv R \in \Gamma$ and $R \equiv Q \in \Gamma$. Now by transitivity from Definition 5 we have $P \equiv Q \in \Gamma$. Therefore $[P]_\Gamma = [Q]_\Gamma$. □

Finally, we can state and prove completeness.

Theorem 2 (Completeness)**.** $\mathbb{SMLD}$ *is weakly complete for announcement-free fragment of $\mathscr{L}$, and* $\mathbb{SPALD}$ *is weakly complete for the full $\mathscr{L}$.*

Proof. Suppose φ is not provable. By the PAL reduction axioms there is a formula φ' without announcements such that $\vdash \varphi \leftrightarrow \varphi'$. Hence φ' is not provable and therefore $\neg\varphi'$ is consistent.

Let Prop be the vocabulary of φ'. Let $\mathscr{M}$ be the canonical model for Prop per Definition 11. Let Γ be any maximally consistent set containing $\neg\varphi'$. Such a set always exists and can be defined using a standard Lindenbaum Lemma [3, p. 197]. In particular, Γ is an element of W in $\mathscr{M}$. Then by the Truth Lemma we have $\mathscr{M}, \Gamma \vDash \neg\varphi'$.

The reduction axioms are also semantically valid, so we have $\mathscr{M}, \Gamma \vDash \neg\varphi$. Hence φ is not semantically valid. □

5 Knowing *the* Definition

Our language does not allow us to express that an agent knows *the* definition of a proposition. We can add this using an operator similar to Kv from [22]. Formally, let Kd_iP where $P \in \mathscr{L}_B$ have the following semantics:

$$\mathscr{M}, w \vDash Kd_i(P) \iff \forall w' : wR_iw' \text{ implies } \mathsf{def}_w(P) = \mathsf{def}_{w'}(P)$$

As we have shown in Examples 1 and 2 knowing whether something is true and knowing its definition are contingent, neither implies the other.

We can then also define the notion of *explicitly knowing*, which is the combination of knowing that and knowing the definition:

$$Kx_i(P) := \Box_i P \wedge Kd_i(P)$$

However, in our framework it is only possible to know propositional formulas explicitly. For example, the formula $Kx_i(\Box_j(p \wedge q))$ is not in the language.

Also adding *the* definition itself to the language, with the following operator $:=$ could be helpful.

$$\mathscr{M}, w \vDash p := P \iff \mathsf{DEF}_w(p) = P$$

For example we then have the following validities:

- $p := P \to p \equiv P$ (definition implies equivalence)
- $p := P \to \neg(p := Q)$ for all $P \neq Q$ (uniqueness)
- $(p := P \wedge Kd_i(p)) \to \Box_i(p := P)$
- $Kd_iP \wedge \Box_i(P \equiv Q) \to Kd_iQ$

In fact, the $:=$ operator could also simplify our original completeness proof. Choosing definitions for the canonical model is trivial for this extended language, because each maximally consistent set Γ will contain exactly one formula of the form $p := P$ for each $p \in \mathsf{Prop}$.[1]

We leave it as future work to axiomatize the extension of our logic with Kd.

6 Conclusion and Future Work

We presented an extension of Public Announcement Logic (PAL) to model the knowledge of meanings with boolean definitions. In our logic agents can understand a proposition without knowing its truth value or the other way round. Moreover, multiple agents can agree on something without agreeing on its meaning and vice versa.

We also presented a sound and complete axiomatization with intuitive axioms to characterize the equivalence operator $\equiv$. The completeness proof is based on a standard canonical model construction, extended with two new ideas for boolean definitions. We use merge to combine different possible definitions and then use a tree of *occurrence substitutions* to ensure that there are only finitely many definitions to choose from.

Our formal contributions are thus more about pattern matching and less the epistemic and dynamic operators. Nevertheless, we think that our logic showcases an interesting interaction between boolean definitions and the standard operators K and $[\varphi]$. On the other hand, as a subsystem of our proof system, the pattern matching logic regarding $\equiv$ seems interesting on its own. It may be applied in computer

[1] We thank Alexandru Baltag for pointing out how $:=$ may simplify our logic.

science or combined with other philosophical logics. In fact, our pattern matching and mismatching axioms are reminiscent of term rewriting rules. We therefore conjecture that our central Lemma 1 can also be shown using Kruskal's Theorem applied to term rewriting, as discussed in [5].[2]

Our work can be extended in several ways.

The framework is compatible with the range of multi-agent epistemic logics from K_n to $S5_n$. The usual axioms for frame properties can be added, for example one might add transitivity for positive introspection, or reflexivity for truthfulness of knowledge.

We already mentioned in the last section that our logic relates to the Kv operator from [22]. We think that "knowing value" models could also be equipped with definitions for their general variables instead of propositions. This can then be combined with "public inspection" from [9] and the resulting framework might give a new perspective on knowledge of terms.

The distinction between value and meaning is best illustrated in the setting of cryptographic protocols, where you may know the value of the message but only part of the meaning. To illustrate this, suppose you have your own private key k. If you receive an encrypted message $\{a, \{b\}_{k'}\}_k$, then you can decode the outer level and learn the value of a. But you cannot learn b if it is encrypted by another key k' that you do not have. In fact, you might not even know that the message you received is of this form and only understand $\{a,x\}_k$. Such phenomena also happen in everyday communication. You may only get part of the meaning of a sentence, but by asking "what do you mean by ...?" you can learn more.

After seeing the details of our framework one might wonder how it relates to other logics of ambiguity. We only mention two related works and leave more detailed comparisons for the future.

A semantic approach is [12] where agents are given different valuation functions to encode their disagreement about (ambiguous) atomic propositions. In our models with definitions there is no need for additional valuation functions and our approach is more syntactic. In our logic, knowing the meaning of a proposition is not reducible to knowing the valuation, e.g., two tautologies can still be very different definitionally. Moreover, we also handle uncertainty about the actual meaning of propositions and provide agents with a way to learn it by update and pattern matching.

Another related approach to model syntactic ambiguity is [14] where the meaning of connectives is not fixed. For example, an agent might wonder whether $p * q$ means $p \vee q$ or $p \wedge q$. A similar example could be modelled in our logic with $r \equiv (p \vee q)$ vs. $r \equiv (p \wedge q)$.

Finally, there are two obvious limitations of our logic. First, all definitions are boolean but in principle also modal definitions like $p := \Box_j q$ are interesting. Second, as mentioned above, logical equivalence and equivalence by definition are not connected. For example, $\neg(p \wedge q)$ and $\neg p \vee \neg q$ are logically equivalent, but due to the pattern mismatch axiom we can never have $\neg(p \wedge q) \equiv (\neg p \vee \neg q)$. Depending on the application, users of our logic might consider this a problem or a feature. We think that our ideas can be extended in both directions, but leave this as future work.

References

[1] Alexandru Baltag (2016): *To Know is to Know the Value of a Variable*. In: *Advances in Modal Logic*, 11, College Publications, London, pp. 135–155. Available at `http://www.aiml.net/volumes/volume11/Baltag.pdf`.

[2] Jon Barwise & Jerry Seligman (1997): *Information flow: the logic of distributed systems*. Cambridge University Press, New York, NY, USA, doi:10.1017/CBO9780511895968.

[2] We thank one of the anonymous reviewers for suggesting this connection.

[3] Patrick Blackburn, Maarten de Rijke & Yde Venema (2001): *Modal Logic*. Cambridge Tracts in Theoretical Computer Science 53, Cambridge University Press, Cambridge, doi:10.1017/CBO9781107050884.

[4] Mika Cohen & Mads Dam (2007): *A Complete Axiomatization of Knowledge and Cryptography*. In: 22nd IEEE Symposium on Logic in Computer Science (LICS 2007), 10-12 July 2007, Wroclaw, Poland, Proceedings, IEEE, California, pp. 77–88, doi:10.1109/LICS.2007.4.

[5] Nachum Dershowitz (1982): *Orderings for term-rewriting systems*. Theoretical Computer Science 17(3), pp. 279–301, doi:10.1016/0304-3975(82)90026-3.

[6] Yifeng Ding (2016): *Epistemic Logic with Functional Dependency Operator*. Studies in Logic 9(4), pp. 55–84. Available at `https://arxiv.org/abs/1706.02048`.

[7] Hans van Ditmarsch, Sujata Ghosh, Rineke Verbrugge & Yanjing Wang (2014): *Hidden protocols: Modifying our expectations in an evolving world*. Artificial Intelligence 208, pp. 18–40, doi:10.1016/j.artint.2013.12.001.

[8] Hans van Ditmarsch, Wiebe van der Hoek & Barteld Kooi (2007): *Dynamic epistemic logic*. 1, Springer Heidelberg, Dordrecht, doi:10.1007/978-1-4020-5839-4.

[9] Jan van Eijck, Malvin Gattinger & Yanjing Wang (2017): *Knowing Values and Public Inspection*. In Sujata Ghosh & Sanjiva Prasad, editors: Logic and Its Applications: 7th Indian Conference, ICLA 2017, Kanpur, India, January 5-7, 2017, Proceedings, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 77–90, doi:10.1007/978-3-662-54069-5_7.

[10] Miriam Franchella (1997): *On the origins of Dénes König's infinity lemma*. Archive for History of Exact Sciences 51, pp. 3–27, doi:10.1007/BF00376449.

[11] Edmund L. Gettier (1963): *Is Justified True Belief Knowledge?* Analysis 23(6), pp. 121–123, doi:10.1093/analys/23.6.121.

[12] Joseph Y. Halpern & Willemien Kets (2014): *A logic for reasoning about ambiguity*. Artificial Intelligence 209, pp. 1–10, doi:10.1016/j.artint.2013.12.003.

[13] Mamoru Kaneko (2004): *Game Theory and Mutual Misunderstanding*. Springer, Berlin, Heidelberg, doi:10.1007/b138120.

[14] Louwe B. Kuijer (2013): *Sequent Systems for Nondeterministic Propositional Logics without Reflexivity*. In Davide Grossi, Olivier Roy & Huaxin Huang, editors: Logic, Rationality, and Interaction: 4th International Workshop, LORI 2013, pp. 190–203, doi:10.1007/978-3-642-40948-6_15.

[15] Dénes Kőnig (1927): *Über eine Schlußweise aus dem Endlichen ins Unendliche*. Acta Litterarum ac Scientiarum, Szeged 3, pp. 121–130.

[16] Fenrong Liu & Yanjing Wang (2013): *Reasoning About Agent Types and the Hardest Logic Puzzle Ever*. Minds and Machines 23(1), pp. 123–161, doi:10.1007/s11023-012-9287-x.

[17] Jan Plaza (1989): *Logics of public communications*. In: Proceedings of the 4th International Symposium on Methodologies for Intelligent Systems, North-Holland, New York, pp. 201–216. Republished in [18].

[18] Jan Plaza (2007): *Logics of public communications*. Synthese 158(2), pp. 165–179, doi:10.1007/s11229-007-9168-7.

[19] R. Ramanujam & S. P. Suresh (2005): *Decidability of context-explicit security protocols*. Journal of Computer Security 13(1), pp. 135–165, doi:10.3233/JCS-2005-13106.

[20] Yanjing Wang (2011): *Reasoning about Protocol Change and Knowledge*. In: Logic and Its Applications - 4th Indian Conference, ICLA 2011, Delhi, India, January 5-11, 2011. Proceedings, Springer, Berlin, Heidelberg, pp. 189–203, doi:10.1007/978-3-642-18026-2_16.

[21] Yanjing Wang & Qinxiang Cao (2013): *On axiomatizations of public announcement logic*. Synthese 190, pp. 103–134, doi:10.1007/s11229-012-0233-5.

[22] Yanjing Wang & Jie Fan (2013): *Knowing That, Knowing What, and Public Communication: Public Announcement Logic with Kv Operators*. In: Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence (IJCAI '13), International Joint Conferences on Artificial Intelligence Organization, California, pp. 1147–1154. Available at `https://www.ijcai.org/Proceedings/13/Papers/173.pdf`.