The Epistemology of Nondeterminism

Adam Bjorndahl(✉)

Carnegie Mellon University, Pittsburgh, USA
abjorn@andrew.cmu.edu

Abstract. This paper proposes new semantics for propositional dynamic logic (PDL), replacing the standard relational semantics. Under these new semantics, program execution is represented as fundamentally deterministic (i.e., functional), while nondeterminism emerges as an epistemic relationship between the agent and the system: intuitively, the nondeterministic outcomes of a given process are precisely those that cannot be ruled out in advance. We formalize these notions using topology and the framework of dynamic topological logic (DTL) [1]. We show that DTL can be used to interpret the language of PDL in a manner that captures the intuition above, and moreover that continuous functions in this setting correspond exactly to deterministic processes. We also prove that certain axiomatizations of PDL remain sound and complete with respect to the corresponding classes of dynamic topological models. Finally, we extend the framework to incorporate knowledge using the machinery of subset space logic [2], and show that the topological interpretation of public announcements as given in [3] coincides exactly with a natural interpretation of test programs.

1 Introduction

Propositional dynamic logic (PDL) is a framework for reasoning about the effects of *nondeterministic programs* (or, more generally, *nondeterministic actions*).[1] The standard models for PDL are relational structures interpreted as state transition diagrams: each program π is associated with a binary relation R_π on the state space, and $xR_\pi y$ means that state y is one possible result of executing π in x.

What is the sense of "possibility" at play here? This paper explores an epistemic account. The standard models for PDL treat nondeterminism as a primitive, unanalyzed notion: effectively, for each state x, π is interpreted as nondeterministic at x just in case $|\{y : xR_\pi y\}| > 1$. But one might hope for a logic that provides some insight into the *nature* and *source* of nondeterminism, rather than simply stipulating its existence.

We investigate a richer class of models for nondeterministic program execution which differ from the standard models in two key respects: (1) states

[1] Mathematical treatments of nondeterminism have a long history in computer science (see, e.g., [4–7]), though this paper focuses specifically on the semantics of PDL. See [8] for an overview and brief history of this branch of modal logic.

© Springer-Verlag GmbH Germany, part of Springer Nature 2018

https://doi.org/10.1007/978-3-662-57669-4_8

completely determine the effects of actions, and (2) nondeterminism emerges, loosely speaking, as a kind of epistemic relationship between a given agent (or collection of agents) and the program (or action) in question. As we argue in the next section, to make this relationship precise we need structures rich enough to represent *potential observations*; for this we make use of *topology*. The resulting framework is very closely related to *dynamic topological logic* (DTL) as developed by Kremer and Mints [1]; roughly speaking, we show that DTL embeds a faithful interpretation of PDL. Furthermore, we demonstrate that *continuity* in this setting coincides exactly with the notion of determinism: that is, *determinism is continuity in the observation topology.*

The rest of the paper is organized as follows. In Sect. 2 we review the basics of PDL and present the intuitions that motivate the development of our new models and the importance of "potential observations" in the epistemic interpretation of nondeterminism. In Sect. 3 we motivate and review the use of topology for this purpose, and connect it to dynamic topological logic. This provides the tools we need to formalize our epistemic conception of nondeterminism and establish the correspondence between determinism and continuity mentioned above. In Sect. 4 we transform PDL models into DTL models in a manner that preserves the truth value of all PDL formulas, and use this to prove that certain standard PDL axiomatizations are also sound and complete with respect to corresponding classes of DTL models. In Sect. 5 we enrich our semantics using the machinery of *subset space logic* [2] in order to reason simultaneously about both knowledge and knowability in the context of nondeterministic program execution; furthermore, we show how *public announcements* [9], appropriately generalized to the topological setting [3], can be captured using *test programs*. Section 6 concludes with a brief discussion of ongoing work. Proofs and other details are collected in Appendix A.

2 Review and Motivation

Fix a countable set of *primitive propositions* PROP and a countable set of *programs* Π. The language of PDL, denoted $\mathcal{L}_{PDL}$, is given by

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \psi \mid \langle\pi\rangle\varphi,$$

where $p \in$ PROP, $\pi \in \Pi$, and $\langle\pi\rangle\varphi$ is read, "after some execution of π, φ is true". Often Π is constructed from a set of more "basic" programs by closing under certain operations, but for the moment we will take it for granted as a structureless set. A **(standard) PDL model** is a relational frame $(X, (R_\pi)_{\pi\in\Pi})$ together with a valuation $v : \text{PROP} \to 2^X$; Boolean formulas are interpreted in the usual way, while $\langle\pi\rangle$ is interpreted as existential quantification over the R_π-accessible states:

$$\begin{aligned} x \models p &\text{ iff } x \in v(p)\\ x \models \neg\varphi &\text{ iff } x \not\models \varphi\\ x \models \varphi\wedge\psi &\text{ iff } x \models \varphi \text{ and } x \models \psi\\ x \models \langle\pi\rangle\varphi &\text{ iff } (\exists y)(xR_\pi y \text{ and } y \models \varphi). \end{aligned}$$

Thus, $\langle\pi\rangle\varphi$ is true at a state x just in case some possible execution of π at x results in a φ-state.

Following standard conventions, we write $[\pi]$ as an abbreviation for $\neg\langle\pi\rangle\neg$, so we have

$$x \models [\pi]\varphi \text{ iff } (\forall y)(xR_\pi y \text{ implies } y \models \varphi).$$

We also treat R_π as a set-valued function when convenient, with $R_\pi(x) = \{y \in X : xR_\pi y\}$.

It is easy to adjust the standard models for PDL so that each state completely determines the outcome of each action: simply replace the relations R_π with functions $f_\pi : X \to X$. To emphasize this shift we introduce modalities $\bigcirc_\pi$ into the language, reading $\bigcirc_\pi\varphi$ as "after execution of π, φ holds"; these modalities are interpreted using the functions f_π in the natural way:

$$x \models \bigcirc_\pi\varphi \text{ iff } f_\pi(x) \models \varphi.$$

Perhaps the most direct attempt to formalize nondeterminism as an epistemic notion in this setting is to interpret the "nondeterministic outcomes" of π to be precisely those outcomes that the agent considers possible.

Somewhat more formally, supposing we have access to a knowledge modality K with corresponding dual $\hat{K}$ (so $\hat{K}\varphi$ is read "the agent considers φ possible"), we might define

$$\langle\pi\rangle\varphi \equiv \hat{K}\bigcirc_\pi\varphi.$$

Crucially, however, this seems to miss the essence of nondeterminism. For instance, according to this definition, when the agent in question happens to be very uncertain about π, we are forced to interpret π as having a great many possible nondeterministic outcomes. But there is a clear conceptual distinction between those outcomes of π that are possible as far as some agent knows—perhaps an agent with very poor information—as opposed to those outcomes that would remain possible *even with good information.* And it seems to be the latter concept that aligns more closely with our intuitions regarding nondeterminism.

For a simple example, imagine running a random number generator. This seems like a canonical example of a nondeterministic process. Note that what is important here is not merely that you do not, in fact, know what number will be generated in advance, but also that you are unable *in principle* to determine this in advance.[2] By contrast, imagine running a program that queries a

[2] To be sure, if you had access to a more advanced set of tools than are standardly available, perhaps you *could* make such a determination. And in this case, thinking of the random number generator as a nondeterministic process loses much of its intuitive appeal. Indeed, *any* nondeterministic process whatsoever might be viewed as deterministic relative to a sufficiently powerful set of tools (e.g., from God's perspective). Thus, nondeterminism can be naturally construed as a *relative* notion that depends on a fixed background set of "feasible measurements". We make this precise below.

given database and prints the result; we would not want to call this program nondeterministic even if you happened to be ignorant about the contents of the database.

This is a distinction we want to respect. The relevant epistemic notion, then, is not what any given agent currently happens to know, but what they *could come to know*. This is where topology comes in: the notion of "potential knowledge" or "knowability" is naturally represented in topological spaces.

3 Topology, Dynamics, and Determinism

3.1 Topological Spaces and Models

A **topological space** is a set X together with a collection $\mathcal{T} \subseteq 2^X$ of subsets of X such that $\emptyset, X \in \mathcal{T}$ and $\mathcal{T}$ is closed under unions and finite intersections. Elements of $\mathcal{T}$ are called *open* and $\mathcal{T}$ is called the *topology*.

There are various intuitions that help to make sense of this definition, most of which tap into the notion of topology as the mathematics of physical space and proximity.[3] Here, though, we focus instead on epistemic intuitions, through which topology is naturally interpreted as a formalization of evidence and potential observations. In fact, these two intuitions overlap in cases where the relevant observations are measurements about locations in space.

Informally, if we think of X as a set of possible worlds encoding the potential uncertainties one may have, then we can think of open sets $U \in \mathcal{T}$ as the results of measurements or observations. More precisely, we can understand U to represent the observation that rules out precisely those worlds $x \notin U$. On this view, each $U \in \mathcal{T}$ corresponds to a possible state of knowledge, and the topology $\mathcal{T}$ itself can be conceptualized as the set of available observations.[4]

A core notion in topology is that of the *interior* of a set $A \subseteq X$, defined by:

$$int(A) = \{x \in A : (\exists U \in \mathcal{T})(x \in U \subseteq A)\}.$$

The interior of A therefore consists of those points x that are "robustly" inside A, in the sense that there is some "witness" $U \in \mathcal{T}$ to x's membership in A. When we interpret elements of $\mathcal{T}$ as the results of possible measurements, the notion of interior takes on a natural epistemic interpretation: x lies in the interior of A just in case there is *some* measurement one could potentially take that would

[3] For a standard introduction to topological notions, we refer the reader to [10].

[4] Suppose, for a simple example, that you measure your height and obtain a reading of 180 ± 2 cm. If we represent the space of your possible heights using the positive real numbers, $\mathbb{R}^+$, then it is natural to identify this measurement with the open interval $(178, 182)$. And with this measurement in hand, you can safely deduce that you are, for instance, less than 183 cm tall, while remaining uncertain about whether you are, say, taller than 179 cm.

entail A. In other words, the worlds in the interior of A are precisely the worlds where A *could come to be known.*[5]

The dual of the interior operator is called *closure*:

$$\begin{aligned} cl(A) &= X \setminus int(X \setminus A) \\ &= \{x \in X \;:\; (\forall U \in \mathcal{T})(x \in U \Rightarrow U \cap A \neq \emptyset)\}. \end{aligned}$$

Thus, epistemically speaking, worlds in the closure of A are precisely those worlds in which A is compatible with *every possible measurement.* The closure operator therefore offers a mathematical realization of our intuition about nondeterminism: namely, that a nondeterministic outcome of a program is one that remains possible no matter how good the agent's state of information.

A **topological model** M is a topological space $(X, \mathcal{T})$ together with a *valuation* $v : \textsc{prop} \to 2^X$. In such models we interpret the basic modal language $\mathcal{L}_\Box$ defined by

$$\varphi ::= p \,|\, \neg\varphi \,|\, \varphi \wedge \psi \,|\, \Box\varphi,$$

where $p \in \textsc{prop}$, via the usual recursive clauses for the Boolean connectives together with the following addition:

$$x \models \Box\varphi \text{ iff } x \in int([\![\varphi]\!]),$$

where $[\![\varphi]\!] = \{x \in X : x \models \varphi\}$. We also make use of the dual modality $\Diamond$, defined by

$$x \models \Diamond\varphi \text{ iff } x \in cl([\![\varphi]\!]).$$

Following the discussion above, we read $\Box\varphi$ as "φ is knowable" or "φ can be ascertained" and $\Diamond\varphi$ as "φ is unfalsifiable" or "φ cannot be ruled out".

3.2 Dynamic Topological Models

Kremer and Mints [1] introduce the notion of a *dynamic topological model*, which is simply a topological model equipped with a continuous function $f : X \to X$. Since we wish to capture the execution of a multitude of programs, we generalize this notion slightly to topological models equipped with a family of functions, one for each program $\pi \in \Pi$. Moreover, continuity is not something we will want to take for granted; we therefore drop this requirement as well.

[5] One might wonder about the closure conditions on topologies. Finite intersections can perhaps be accounted for by identifying them with sequences of measurements, but what about unions? One intuition comes by observing that for any set A, $int(A) = \bigcup\{U \in \mathcal{T} : U \subseteq A\}$, so $int(A)$ is the information state that arises from learning *that* A is true without learning *what particular measurement* was taken to ascertain this fact. This idea is formalized in [3] using public announcements; we direct the reader to this work for a more detailed discussion of this point.

A **dynamic topological model** is a tuple $(X, \mathcal{T}, \{f_\pi\}_{\pi \in \Pi}, v)$ where $(X, \mathcal{T}, v)$ is a topological model and each f_π is a function (not necessarily continuous) from X to X. In such models we can interpret the language $\mathcal{L}_{\Box,\bigcirc}$ defined by

$$\varphi ::= p \,|\, \neg\varphi \,|\, \varphi \wedge \psi \,|\, \Box\varphi \,|\, \bigcirc_\pi \varphi,$$

where $p \in \textsc{prop}$ and $\pi \in \Pi$, via the additional semantic clause:

$$x \models \bigcirc_\pi \varphi \text{ iff } f_\pi(x) \models \varphi.$$

This provides the final tool we need to formalize the re-interpretation of nondeterministic program execution sketched in Sect. 2:

$$\langle \pi \rangle \varphi \equiv \Diamond \bigcirc_\pi \varphi.$$

Semantically:

$$\begin{aligned} x \models \langle \pi \rangle \varphi &\text{ iff } x \models \Diamond \bigcirc_\pi \varphi \\ &\text{ iff } x \in cl(f_\pi^{-1}(\llbracket \varphi \rrbracket)). \end{aligned}$$

So: φ is a nondeterministic outcome of π (at x) just in case it cannot be ruled out (at x) that φ will hold after π is executed. Topologically: every measurement at x (i.e., every open neighbourhood of x) is compatible with a state where φ results from executing π. Call this the *epistemic interpretation of nondeterminism.*

3.3 Determinism as Continuity

The epistemic interpretation of nondeterminism accords with our earlier intuitions about random number generators and database queries. Consider a process *rand* that randomly displays either a 0 or a 1, and an agent who (we presume) is unable to measure in advance the relevant quantities that determine the output of this process. This means that both $\bigcirc_{rand} 0$ and $\bigcirc_{rand} 1$ are compatible with every measurement the agent can take (in advance of running the process), so $\Diamond \bigcirc_{rand} 0$ and $\Diamond \bigcirc_{rand} 1$ both hold, i.e., $\langle rand \rangle 0 \wedge \langle rand \rangle 1$.[6] By contrast, consider a process *query* that outputs the next entry in a given database and an agent who can look up that entry in advance (which is, say, 0). This means there is a measurement that guarantees $\bigcirc_{query} 0$, so $\Box \bigcirc_{query} 0$ holds, which yields $[query]0$.

What exactly is (non)determinism in this setting? It is tempting to describe a (non)deterministic process as one in which the output state can(not) be determined in advance. But this is far too liberal: in principle, states may encode many details that are far beyond the ability of the agent to measure precisely,

[6] Of course, once the process *rand* has been called, presumably the agent *can* ascertain its output (e.g., by looking at the screen). In particular, if (say) the number displayed is 1, then this knowable, and therefore it is not the case that $\bigcirc_{rand} \Diamond 0$ holds. This shows that the order of the two modalities is crucial in establishing the proper correspondence with nondeterminism. Thanks to an anonymous reviewer for suggesting this clarification.

which would then trivially render every process nondeterministic. For instance, if the state description encodes the current temperature of some internal components of the system (e.g., as a seed value for the *rand* process), then even *after* executing a simple program like *query* the user will not be in a position to know the true state.

The correct notion is more subtle. Consider again the *rand* process: the crucial feature of this program is not that it produces a *state* that cannot be determined in advance, but that it produces a *measurable quantity*—namely, the number displayed—that cannot be determined in advance.

To make the same point slightly more formally: it may be that no measurement at x rules out *all* the other states (indeed, this will be the case whenever $\{x\}$ is not open). This is necessary but *not sufficient* for nondeterminism because it may still be possible to learn (in advance) everything there is to know about the effects of executing π (as describable in the language).

This suggests the following refined account of determinism: a deterministic process is one in which everything *one could learn* about the state of the system *after the program is executed* one can also determine in advance of the program execution.

This account aligns perfectly with the topological notion of *continuity*. Intuitively: a function is continuous if small changes in the input produce small changes in the output. Topologically: f is **continuous at** x if, for every open neighbourhood V of $f(x)$, there exists an open neighbourhood U of x such that $f(U) \subseteq V$. And, finally, epistemically: f is continuous at x if every measurement V compatible with the output state $f(x)$ can be guaranteed in advance by some measurement U compatible with the input state x. So the definition of continuity corresponds exactly to our refined account of determinism. In other words: *determinism is continuity in the observation topology.*

Continuity of f_π can be *defined* in the object language $\mathcal{L}_{\Box,\bigcirc}$ by the formula[7]

$$\bigcirc_\pi \Box \varphi \to \Box \bigcirc_\pi \varphi.$$

Unsurprisingly, this is precisely the scheme that Kremer and Mints call "the axiom of continuity" in their axiomatization of the class of (continuous) dynamic topological models [1]. It reads: "If, *after* executing π, φ is (not only true, but also) measurably true, then it is possible to take a measurement *before* executing π that guarantees that φ will be true after executing π." So this scheme expresses the idea that one need not actually execute π in order to determine whatever could be determined after its execution: all the measurable effects of π can be determined in advance. Again, this is determinism. Continuity is determinism.

4 Axiomatization and Model Transformation

We restrict our attention in this section to *serial* PDL models, in which each R_π is serial: $(\forall x)(\exists y)(x R_\pi y)$. Thus, we rule out the possibility of a program π producing no output at all in some states (intuitively, "crashing"), which corresponds

[7] This claim is made precise and proved in Appendix A.1.

to the fact that the functions in dynamic topological models are assumed to be total (i.e., everywhere defined). This allows for a cleaner translation between the two paradigms; in Sect. 5 we consider a framework that drops this assumption.

Table 1. Axioms and rules of inference for SPDL_0

(CPL)	All propositional tautologies	Classical propositional logic
(K_π)	$[\pi](\varphi \rightarrow \psi) \rightarrow ([\pi]\varphi \rightarrow [\pi]\psi)$	Distribution
(D_π)	$[\pi]\varphi \rightarrow \langle\pi\rangle\varphi$	Seriality
(MP)	From φ and $\varphi \rightarrow \psi$ deduce ψ	Modus ponens
(Nec_π)	From φ deduce $[\pi]\varphi$	Necessitation

The most basic version of serial PDL (without any operations on programs) is axiomatized by the axioms and rules of inference given in Table 1. Call this system SPDL_0.

Theorem 1. *SPDL_0 is a sound and complete axiomatization of the language $\mathcal{L}_{PDL}$ with respect to the class of all serial PDL models.*[8]

Using the epistemic interpretation of nondeterminism given in Sect. 3.2, we can also interpret the language $\mathcal{L}_{PDL}$ directly in dynamic topological models:

$$x \models \langle\pi\rangle\varphi \text{ iff } x \in cl(f_\pi^{-1}([\![\varphi]\!])).$$

And, dually:

$$x \models [\pi]\varphi \text{ iff } x \in int(f_\pi^{-1}([\![\varphi]\!])).$$

This puts us in a position to evaluate our re-interpretation in a precise way. Namely, we can ask: are all the properties of nondeterministic program execution that are captured by standard (serial) PDL models preserved under this new interpretation? And we can answer in the affirmative:

Theorem 2. *SPDL_0 is a sound and complete axiomatization of the language $\mathcal{L}_{PDL}$ with respect to the class of all dynamic topological models.*

Proof. Soundness of (CPL) and (MP) is immediate. Soundness of (Nec_π) follows from the fact that $f_\pi^{-1}(X) = X$ and $int(X) = X$, and soundness of (D_π) follows from the fact that, for all $A \subseteq X$, $int(A) \subseteq cl(A)$. Finally, to see that (K_π) is sound, observe that

$$\begin{aligned} int(f_\pi^{-1}([\![\varphi \rightarrow \psi]\!])) \cap int(f_\pi^{-1}([\![\varphi]\!])) &= int(f_\pi^{-1}([\![\varphi \rightarrow \psi]\!]) \cap f_\pi^{-1}([\![\varphi]\!])) \\ &\subseteq int(f_\pi^{-1}([\![\psi]\!])). \end{aligned}$$

[8] This can be proved using very standard techniques; see, e.g., [11].

The proof of completeness proceeds by way of a model-transformation construction we provide in Appendix A.2: specifically, we show that every serial PDL model can be transformed into a dynamic topological model in a manner that preserves the truth of all formulas in $\mathcal{L}_{PDL}$ (Proposition 2). By Theorem 1, every non-theorem of $\mathsf{SPDL_0}$ is refuted on some serial PDL model, so our transformation produces a dynamic topological model that refutes the same formula, thereby establishing completeness. □

Typically one works with richer versions of PDL in which the set of programs Π is equipped with one or more operations corresponding, intuitively, to ways of building new programs from old programs. Standard examples include:

– *Sequencing*: $\pi_1; \pi_2$ executes π_1 followed immediately by π_2.
– *Nondeterministic union*: $\pi_1 \cup \pi_2$ nondeterministically chooses to execute either π_1 or π_2.
– *Iteration*: π^* repeatedly executes π some nondeterministic finite number of times.

Can we make sense of these operations in our enriched epistemic setting? The latter two transform deterministic programs into nondeterministic programs, and for this reason they are difficult to interpret in a setting where program execution is fundamentally deterministic (i.e., interpreted by functions). We return to discuss this point in Sect. 6. Sequencing, on the other hand, does not have this issue; one might guess that it is straightforwardly captured by the condition

$$f_{\pi_1;\pi_2} = f_{\pi_2} \circ f_{\pi_1}.$$

While function composition certainly seems like the natural way to interpret sequential program execution, there is a wrinkle in the axiomatization. PDL with sequencing is axiomatized by including the following axiom scheme:

$$\text{(Seq)} \qquad \langle \pi_1; \pi_2 \rangle \varphi \leftrightarrow \langle \pi_1 \rangle \langle \pi_2 \rangle \varphi.$$

Interestingly, this scheme is *not* valid in arbitrary dynamic topological models. This is because

$$[\![\langle \pi_1; \pi_2 \rangle \varphi]\!] = cl(f^{-1}_{\pi_1;\pi_2}([\![\varphi]\!])) = cl(f^{-1}_{\pi_1}(f^{-1}_{\pi_2}([\![\varphi]\!]))),$$

whereas

$$[\![\langle \pi_1 \rangle \langle \pi_2 \rangle \varphi]\!] = cl(f^{-1}_{\pi_1}(cl(f^{-1}_{\pi_2}([\![\varphi]\!]))));$$

the extra closure operator means we have

$$[\![\langle \pi_1; \pi_2 \rangle \varphi]\!] \subseteq [\![\langle \pi_1 \rangle \langle \pi_2 \rangle \varphi]\!]$$

but not, in general, equality.[9]

[9] For instance, consider the set $X = \{a, b\}$ equipped with the topology $\mathcal{T} = \{\emptyset, \{b\}, X\}$, and the function f_π defined by $f_\pi(a) = b$ and $f_\pi(b) = a$. Let $v(p) = \{a\}$. Then, since $f_\pi \circ f_\pi = id$, we have $cl(f^{-1}_{\pi;\pi}([\![p]\!])) = cl(\{a\}) = \{a\}$, whereas $cl(f^{-1}_\pi(cl(f^{-1}_\pi([\![p]\!])))) = cl(f^{-1}_\pi(cl(\{b\}))) = cl(f^{-1}_\pi(X)) = X$.

A function $f : X \to Y$ is called **open** if for every open subset $U \subseteq X$, the set $f(U)$ is open in Y. It turns out that when the function f_{π_1} is open, the mismatch above vanishes (all of the following claims are proved in Appendix A.3):

Lemma 1. *Let $(X, \mathcal{T}, \{f_\pi\}_{\pi \in \Pi}, v)$ be a dynamic topological model. If f_{π_1} is open, then*

$$[\![\langle \pi_1; \pi_2 \rangle \varphi]\!] = [\![\langle \pi_1 \rangle \langle \pi_2 \rangle \varphi]\!].$$

Say that a dynamic topological model is *open* if each f_π is open.

Theorem 3. *SPDL_0 + (Seq) is a sound and complete axiomatization of the language $\mathcal{L}_{PDL}$ with respect to the class of all open dynamic topological models.*

Like continuity, openness of the function f_π can be defined in the object language; in fact, it is defined by the converse of the scheme defining continuity:

$$\Box \bigcirc_\pi \varphi \to \bigcirc_\pi \Box \varphi.$$

Roughly speaking, this says that whatever you can (in principle) predict about executing π beforehand you could also come to know afterward. This has a "perfect recall" type flavour, except the relevant epistemic notion is not what is actually known but what *could come to be known*. Besides serving to validate the standard sequencing axiom, this principle also plays a crucial role in the next section, where we extend the present framework to incorporate knowledge.

5 Knowledge and Learning

To study the epistemology of nondeterministic program execution, we want to be able to reason not only about what *can be* known, but also about what *is* known. To do so we need a richer semantic setting, for which we turn to *topological subset models* [2]; essentially, these use an additional parameter to keep track of the current state of information, through which a standard knowledge modality can be interpreted.

Topological subset models have experienced renewed interest in recent years [3,12–14], beginning with the work in [3] studying public announcements in the topological setting. Standard semantics for public announcement logic take the *precondition* of an announcement of φ to be the truth of φ; in the topological setting, this precondition is strengthened to the *knowability* of φ. As we will see, this interpretation of public announcements is recapitulated in the present framework via a natural interpretation of *test programs*.

5.1 Incorporating Knowledge

A **topological subset model** just *is* a topological model $(X, \mathcal{T}, v)$; the difference lies in the semantic clauses for truth, which are defined with respect to *pairs* of the form (x, U), where $x \in U \in \mathcal{T}$; such pairs are called *epistemic scenarios*. Intuitively, x represents the actual world, while U captures the agent's current information and thus what they know. Formally, we interpret the language $\mathcal{L}_{K,\Box}$ given by

$$\varphi ::= p \,|\, \neg\varphi \,|\, \varphi \wedge \psi \,|\, K\varphi \,|\, \Box\varphi,$$

where $p \in \textsc{prop}$, as follows:

$$\begin{aligned}(x,U) &\models p \text{ iff } x \in v(p)\\ (x,U) &\models \neg\varphi \text{ iff } (x,U) \not\models \varphi\\ (x,U) &\models \varphi \wedge \psi \text{ iff } (x,U) \models \varphi \text{and} (x,U) \models \psi\\ (x,U) &\models K\varphi \text{ iff } U \subseteq [\![\varphi]\!]^U\\ (x,U) &\models \Box\varphi \text{ iff } x \in \mathit{int}([\![\varphi]\!]^U),\end{aligned}$$

where $[\![\varphi]\!]^U = \{x \in U : (x,U) \models \varphi\}$. So the agent knows φ in the epistemic scenario (x,U) just in case it is guaranteed by their current information U.

We next define *dynamic topological subset models* by incorporating functions f_π as above. Of course, we need subset-style semantics for the dynamic modalities. Perhaps the most natural way to define the updated epistemic scenario is as follows:

$$(x,U) \models \bigcirc_\pi\varphi \text{ iff } (f_\pi(x), f_\pi(U)) \models \varphi.$$

This definition raises two issues, one technical and the other conceptual. First, as a technical matter, the definition only makes sense if $f_\pi(U)$ is open—otherwise $(f_\pi(x), f_\pi(U))$ is not an epistemic scenario. So we have here another reason to restrict our attention to *open* functions f_π.

Second, conceptually, in a sense this framework does not permit *learning*. True, an agent's state of knowledge changes in accordance with program execution, but every "live" possibility $y \in U$ is preserved as the corresponding state $f_\pi(y)$ in the updated information set $f_\pi(U)$. Intuitively, then, the agent can never truly eliminate possibilities.

Dynamic epistemic logic [15] is a modern and vibrant area of research concerned exactly with this issue of how to capture the dynamics of information update. But rather than explicitly importing machinery from this paradigm (e.g., announcements) to represent learning in the present context, we can take advantage of a mechanic that PDL already has available: crashing. In Sect. 4 we restricted attention to standard PDL models that were serial, corresponding in our framework to total functions. We now drop this assumption to allow *partial* functions $f_\pi : X \rightharpoonup X$ that are undefined at some points in X. This allows the corresponding updates to effectively delete states and thus capture information update in much the same way that, e.g., public announcements do.

A **dynamic topological subset model (over Π)** is a topological subset model together with a family of partial, open functions[10] $f_\pi : X \rightharpoonup X$, $\pi \in \Pi$.

[10] Typically the concept of openness is applied to total functions, but the definition makes sense for partial functions as well: f is open provided, for all open U, $f(U) = \{y \in X : (\exists x \in U)(f(x) = y)\}$ is open.

Formulas of the language $\mathcal{L}_{K,\Box,\bigcirc}$ given by

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \psi \mid K\varphi \mid \Box\varphi \mid \bigcirc_\pi \varphi,$$

are interpreted at epistemic scenarios via the semantic clauses introduced above, taking care to note that the righthand side of the clause defining $\bigcirc_\pi$ now carries the implicit assumption that $f_\pi(x)$ is actually defined:

$$(x, U) \models \bigcirc_\pi \varphi \text{ iff } f_\pi(x) \text{ is defined and } (f_\pi(x), f_\pi(U)) \models \varphi.$$

We provide a sound and complete axiomatization of this logic in Appendix A.4.

5.2 Test Programs and Public Announcements

A standard enrichment of PDL expands the set of programs Π to include *test programs*. Unlike other program constructions, test programs are not built from existing programs but instead from formulas in the language: given a formula φ, the program $\varphi?$ is introduced to be interpreted by the relation $R_{\varphi?}$ defined by

$$xR_{\varphi?}y \text{ iff } x = y \text{ and } x \models \varphi.$$

Intuitively, the program $\varphi?$ crashes at states where φ is false, and otherwise does nothing.

Test programs are deterministic, so can be represented just as easily by functions:

$$f_{\varphi?}(x) = \begin{cases} x & \text{if } x \models \varphi \\ \text{undefined} & \text{otherwise.} \end{cases}$$

But to make sense of this definition in dynamic topological subset models, two issues must be addressed. First, the relation $x \models \varphi$ is not actually defined in subset models—formulas are evaluated with respect to epistemic scenarios, not individual states. However, it is easy to see that when φ belongs to the fragment $\mathcal{L}_{\Box,\bigcirc}$, its truth in an epistemic scenario (x, U) is independent of U; in this case, we can simply declare that $x \models \varphi$ iff $(x, U) \models \varphi$ for some (equivalently, all) open sets U containing x. We therefore restrict the formation of test programs to $\varphi \in \mathcal{L}_{\Box,\bigcirc}$.

Second, $f_{\varphi?}$ may not be open. Indeed, $f_{\varphi?}$ is open just in case $[\![\varphi]\!] = \{x : x \models \varphi\}$ is open. If $[\![\varphi]\!]$ is not open then it contains at least one state $x \in [\![\varphi]\!] \setminus \mathit{int}([\![\varphi]\!])$, that is, a state at which φ is true but not *measurably* true. Intuitively, at such states the test program $\varphi?$ should crash, since it must fail to determine that φ is true. This motivates the following revised definition of $f_{\varphi?}$:

$$f_{\varphi?}(x) = \begin{cases} x & \text{if } x \in \mathit{int}([\![\varphi]\!]) \\ \text{undefined} & \text{otherwise.} \end{cases}$$

These functions *are* open. Moreover, under these revised semantics, we have:

$$\begin{aligned} (x, U) \models \bigcirc_{\varphi?}\psi &\text{ iff } f_{\varphi?}(x) \text{ is defined and } (f_{\varphi?}(x), f_{\varphi?}(U)) \models \psi \\ &\text{ iff } x \in \mathit{int}([\![\varphi]\!]) \text{ and } (x, U \cap \mathit{int}([\![\varphi]\!])) \models \psi, \end{aligned}$$

which coincides exactly with the topological definition of a public announcement of φ as given in [3].[11]

6 Future Work

We have formalized a relatively simple idea: namely, that the nondeterministic outcomes of a process are precisely those that the agent cannot rule out in advance. Using the tools of topology to represent potential observations, we have demonstrated a striking connection between deterministic processes and continuous functions, proved that certain axiomatizations of PDL remain sound and complete when reinterpreted in this enriched setting, and established a natural identity between test programs and public announcements.

Many questions remain, both conceptual and technical. What is the relationship between the (probabilistic) notion of *chance* and the topological construal of nondeterminism presented here? Is there a way to import "nondeterministic" operations on programs, such as nondeterministic union or iteration, into this setting? Or is it perhaps better to focus on deterministic analogues of these program constructions, such as, "If φ do π_1, else do π_2", or, "Do π until φ"? How much of dynamic epistemic logic can be recovered as program execution in dynamic topological subset models? For instance, can we make sense of test programs based on epistemic formulas (i.e., formulas that include the K modality), as we can with public announcements? And how can we extend the axiomatization of dynamic topological subset models given in Appendix A.4 to include test programs? These questions and more are the subject of ongoing research.

A Proofs and Details

A.1 Characterizing Continuity

A **dynamic topological frame (over Π)** is a tuple $F = (X, \mathfrak{T}, \{f_\pi\}_{\pi \in \Pi})$ where $(X, \mathfrak{T})$ is a topological space and each $f_\pi : X \to X$. In other words, a frame is simply a dynamic topological model without a valuation function. A frame F is said to *validate* a formula φ just in case φ is true at every point of every model of the form (F, v).

Proposition 1. *The formula scheme $\bigcirc_\pi \Box\varphi \to \Box\bigcirc_\pi\varphi$ defines the class of dynamic topological frames in which f_π is continuous: that is, for every dynamic topological frame F, F validates every instance of $\bigcirc_\pi \Box\varphi \to \Box\bigcirc_\pi\varphi$ iff f_π is continuous.*

Proof. First suppose that M is a dynamic topological model in which f_π is continuous, and let x be a point in this model satisfying $\bigcirc_\pi\Box\varphi$. Then $f_\pi(x) \in$

[11] Or, rather, it coincides with the dual of the definition given in [3], but this is not an important difference.

$int([\![\varphi]\!])$. By continuity, the set $U = f_\pi^{-1}(int([\![\varphi]\!]))$ is open. Moreover, it is easy to see that $x \in U$ and $U \subseteq [\![\bigcirc_\pi \varphi]\!]$, from which it follows that $x \models \Box \bigcirc_\pi \varphi$.

Conversely, suppose that F is a dynamic topological frame in which f_π is not continuous. Let U be an open subset of X such that $A = f_\pi^{-1}(U)$ is not open, and let $x \in A \setminus int(A)$; consider a valuation v such that $v(p) = U$. In the resulting model, since $f_\pi(x) \in U = int(U)$, we have $x \models \bigcirc_\pi \Box p$. On the other hand, since by definition $x \notin int(f_\pi^{-1}(U))$, we have $x \not\models \Box \bigcirc_\pi p$. □

A.2 Model Transformation

Our task in this section is to transform an arbitrary serial PDL model into a dynamic topological model in a truth-preserving manner. The intuition for this transformation is fairly straightforward: in a serial PDL model, each state may be nondeterministically compatible with many possible execution paths corresponding to all the possible ways of successively traversing R_π-edges. In a dynamic topological model, by contrast, all such execution paths must be differentiated by state—roughly speaking, this means we need to create a new state for each possible execution path in the standard model. Then, to preserve the original notion of nondeterminism, we overlay a topological structure that "remembers" which new states originated from the same state in the standard model by rendering them topologically indistinguishable.

Let $M = (X, (R_\pi)_{\pi \in \Pi}, v)$ be a serial PDL model. Let Π^* denote the set of all finite sequences from Π. A map $\alpha : \Pi^* \to X$ is called a **network (through** M**)** provided $(\forall \boldsymbol{\pi} \in \Pi^*)(\forall \pi \in \Pi)(\alpha(\boldsymbol{\pi}) R_\pi \alpha(\boldsymbol{\pi}, \pi))$. In other words, a network α must respect R_π-edges in the sense that it associates with each sequence $(\pi_1, \ldots, \pi_n)$ a path $(x_1, \ldots, x_{n+1})$ through X such that, for each i, $x_i R_{\pi_i} x_{i+1}$.[12] Networks through M constitute the points of the dynamic topological model we are building:

$$\tilde{X} = \{\alpha : \alpha \text{ is a network through } M.\}.$$

The topology we equip $\tilde{X}$ with is particularly simple: for each $x \in X$, let $U_x = \{\alpha \in \tilde{X} : \alpha(\emptyset) = x\}$. Clearly the sets U_x partition X and so form a topological basis; let $\mathcal{T}$ be the topology they generate.

Next we define the functions $f_\pi : \tilde{X} \to \tilde{X}$. Intuitively, $\alpha \in \tilde{X}$ is a complete record of what paths will be traversed in the original state space X for every sequence of program executions. Therefore, after executing π, the updated record $f_\pi(\alpha)$ should simply consist in those paths determined by α that start with an execution of π:

$$f_\pi(\alpha)(\boldsymbol{\pi}) = \alpha(\pi, \boldsymbol{\pi}).$$

Finally, define $\tilde{v} : \textsc{prop} \to 2^{\tilde{X}}$ by

$$\tilde{v}(p) = \{\alpha \in \tilde{X} : \alpha(\emptyset) \in v(p)\}.$$

Let $\tilde{M} = (\tilde{X}, \mathcal{T}, (f_\pi)_{\pi \in \Pi}, \tilde{v})$.

[12] Specifically, $x_1 = \alpha(\emptyset)$ and $(\forall i \geq 2)(x_i = \alpha(x_1, \ldots, x_{i-1}))$.

Proposition 2. *For every $\varphi \in \mathcal{L}_{PDL}$, for every $\alpha \in \tilde{X}$, $(\tilde{M}, \alpha) \models \varphi$ iff $(M, \alpha(\emptyset)) \models \varphi$.*

Proof. We proceed by induction on the structure of φ. The base case when $\varphi = p \in \textsc{prop}$ follows directly from the definition of $\tilde{v}$:

$$\begin{aligned}(\tilde{M}, \alpha) \models p &\text{ iff } \alpha \in \tilde{v}(p)\\ &\text{ iff } \alpha(\emptyset) \in v(p)\\ &\text{ iff } (M, \alpha(\emptyset)) \models p.\end{aligned}$$

The inductive steps for the Boolean connectives are straightforward. So suppose inductively that the result holds for φ; we wish to show it holds for $\langle \pi \rangle \varphi$.

Let $\alpha \in \tilde{X}$ and $x = \alpha(\emptyset)$. By definition, $(\tilde{M}, \alpha) \models \langle \pi \rangle \varphi$ iff $\alpha \in cl(f_\pi^{-1}(\llbracket \varphi \rrbracket_{\tilde{M}}))$. Since the topology is generated by a partition, we know that U_x is a minimal neighbourhood of α, and therefore the preceding condition is equivalent to:

$$U_x \cap f_\pi^{-1}(\llbracket \varphi \rrbracket_{\tilde{M}}) \neq \emptyset.$$

This intersection is nonempty just in case there exists an $\alpha' \in \tilde{X}$ such that $\alpha'(\emptyset) = x$ and $f_\pi(\alpha') \in \llbracket \varphi \rrbracket_{\tilde{M}}$. By the induction hypothesis,

$$\begin{aligned}f_\pi(\alpha') \in \llbracket \varphi \rrbracket_{\tilde{M}} &\text{ iff } (\tilde{M}, f_\pi(\alpha')) \models \varphi\\ &\text{ iff } (M, f_\pi(\alpha')(\emptyset)) \models \varphi\\ &\text{ iff } (M, \alpha'(\pi)) \models \varphi.\end{aligned}$$

So to summarize, we have shown that $(\tilde{M}, \alpha) \models \langle \pi \rangle \varphi$ iff there exists an $\alpha' \in \tilde{X}$ such that $\alpha'(\emptyset) = x$ and $(M, \alpha'(\pi)) \models \varphi$. Since we know that $\alpha'(\emptyset) R_\pi \alpha'(\pi)$, this is in turn equivalent to $(M, x) \models \langle \pi \rangle \varphi$, which completes the induction. □

A.3 Sequencing

Lemma 1. *Let $(X, \mathcal{T}, \{f_\pi\}_{\pi \in \Pi}, v)$ be a dynamic topological model. If f_{π_1} is open, then*

$$\langle \pi_1; \pi_2 \rangle \varphi = \llbracket \langle \pi_1 \rangle \langle \pi_2 \rangle \varphi \rrbracket.$$

Proof. It suffices to show that $\llbracket \langle \pi_1; \pi_2 \rangle \varphi \rrbracket \supseteq \llbracket \langle \pi_1 \rangle \langle \pi_2 \rangle \varphi \rrbracket$. So let

$$x \in \llbracket \langle \pi_1 \rangle \langle \pi_2 \rangle \varphi \rrbracket = cl(f_{\pi_1}^{-1}(cl(f_{\pi_2}^{-1}(\llbracket \varphi \rrbracket))));$$

then for every open neighbourhood U containing x, we know that $U \cap f_{\pi_1}^{-1}(cl(f_{\pi_2}^{-1}(\llbracket \varphi \rrbracket))) \neq \emptyset$. This implies that $f_{\pi_1}(U) \cap cl(f_{\pi_2}^{-1}(\llbracket \varphi \rrbracket)) \neq \emptyset$; since $f_{\pi_1}(U)$ is open, it follows that $f_{\pi_1}(U) \cap f_{\pi_2}^{-1}(\llbracket \varphi \rrbracket) \neq \emptyset$ as well. This then implies that $U \cap f_{\pi_1}^{-1}(f_{\pi_2}^{-1}(\llbracket \varphi \rrbracket)) \neq \emptyset$, and therefore

$$x \in cl(f_{\pi_1}^{-1}(f_{\pi_2}^{-1}(\llbracket \varphi \rrbracket))) = \llbracket \langle \pi_1; \pi_2 \rangle \varphi \rrbracket,$$

as desired. □

Say that a dynamic topological model is *open* if each f_π is open.

Theorem 3. *SPDL_0 + (Seq) is a sound and complete axiomatization of the language $\mathcal{L}_{PDL}$ with respect to the class of all open dynamic topological models.*

Proof. Lemma 1 shows that (Seq) is valid in the class of all open dynamic topological models. For completeness, it suffices to observe that the dynamic topological model $\tilde{M}$ constructed in Appendix A.2 is itself open: indeed, for each basic open U_x, we have

$$\begin{aligned} f_\pi(U_x) &= \{\alpha \in \tilde{X} : xR_\pi\alpha(\emptyset)\} \\ &= \bigcup_{y \in R(x)} U_y, \end{aligned}$$

which of course is open. □

Proposition 3. *The formula scheme $\Box\bigcirc_\pi\varphi \rightarrow \bigcirc_\pi\Box\varphi$ defines the class of dynamic topological frames in which f_π is open: that is, for every dynamic topological frame F, F validates every instance of $\Box\bigcirc_\pi\varphi \rightarrow \bigcirc_\pi\Box\varphi$ iff f_π is open.*

Proof. First suppose that M is a dynamic topological model in which f_π is open, and let x be a point in this model satisfying $\Box\bigcirc_\pi\varphi$. Then $x \in int(f_\pi^{-1}(\llbracket\varphi\rrbracket))$. By openness, the set $V = f(int(f_\pi^{-1}(\llbracket\varphi\rrbracket)))$ is open. Moreover, it is easy to see that $f_\pi(x) \in V$ and $V \subseteq \llbracket\varphi\rrbracket$, from which it follows that $x \models \bigcirc_\pi\Box\varphi$.

Conversely, suppose that F is a dynamic topological frame in which f_π is not open. Let U be an open subset of X such that $A = f_\pi(U)$ is not open, and let $x \in U$ be such that $f_\pi(x) \in A \setminus int(A)$; consider a valuation v such that $v(p) = A$. In the resulting model, since $x \in U$ and $f_\pi(U) \subseteq \llbracket p\rrbracket$, we have $x \models \Box\bigcirc_\pi p$. On the other hand, since by definition $f_\pi(x) \notin int(\llbracket p\rrbracket)$, we have $x \not\models \bigcirc_\pi\Box p$. □

A.4 Dynamic Topological Epistemic Logic

We provide a sound and complete axiomatization of the language $\mathcal{L}_{K,\Box,\bigcirc}$ with respect to the class of all dynamic topological subset models.

Let CPL denote the axioms and rules of classical propositional logic, let $\mathsf{S5}_K$ denote the $\mathsf{S5}$ axioms and rules for the K modality, and let $\mathsf{S4}_\Box$ denote the $\mathsf{S4}$ axioms and rules for the $\Box$ modality (see, e.g., [11]). Let (KI) denote the axiom scheme $K\varphi \rightarrow \Box\varphi$, and set

$$\mathsf{EL}_{K,\Box} := \mathsf{CPL} + \mathsf{S5}_K + \mathsf{S4}_\Box + \text{(KI)}.$$

Theorem 4 *([3, Theorem 1]). $\mathsf{EL}_{K,\Box}$ is a sound and complete axiomatization of $\mathcal{L}_{K,\Box}$ with respect to the class of all dynamic topological subset models.*

Let DTEL denote the axiom system $\mathsf{EL}_{K,\Box}$ together with the axiom schemes and rules of inference given in Table 2.

Theorem 5. *DTEL is a sound and complete axiomatization of $\mathcal{L}_{K,\Box,\bigcirc}$ with respect to the class of all dynamic topological subset models.*

Table 2. Additional axioms and rules of inference for DTEL

$(\neg\text{-PC}_\pi)$	$\bigcirc_\pi\neg\varphi \leftrightarrow (\neg\bigcirc_\pi\varphi \wedge \bigcirc_\pi\top)$	Partial commutativity of $\neg$
$(\wedge\text{-C}_\pi)$	$\bigcirc_\pi(\varphi\wedge\psi) \leftrightarrow (\bigcirc_\pi\varphi \wedge \bigcirc_\pi\psi)$	Commutativity of $\wedge$
$(K\text{-PC}_\pi)$	$\bigcirc_\pi\top \to (\bigcirc_\pi K\varphi \leftrightarrow K(\bigcirc_\pi\top \to \bigcirc_\pi\varphi))$	Partial commutativity of K
(O_π)	$(\Box\neg\bigcirc_\pi\varphi \wedge \bigcirc_\pi\top) \to \bigcirc_\pi\Box\neg\varphi$	Openness
(Mon_π)	From $\varphi\to\psi$ deduce $\bigcirc_\pi\varphi \to \bigcirc_\pi\psi$	Monotonicity

Proof. Soundness of $\mathsf{EL}_{K,\Box}$ follows as in the proof given in [3, Theorem 1], while soundness of the additions presented in Table 2 is easy to check. The presence of $\bigcirc_\pi\top$ in $(\neg\text{-PC}_\pi)$ accounts for the fact that f_π can be partial (since both $\neg\bigcirc_\pi\varphi$ and $\neg\bigcirc_\pi\neg\varphi$ are true at states where f_π is undefined), and plays an analogous role in $(K\text{-PC}_\pi)$ and (O_π). Similarly, the usual "necessitation" rule for $\bigcirc_\pi$ is not valid, since even if φ is a theorem, $\bigcirc_\pi\varphi$ still fails at states where f_π is undefined.

Completeness is proved by a canonical model construction. Let X denote the set of all maximal DTEL-consistent subsets of $\mathcal{L}_{K,\Box,\bigcirc}$. Define a binary relation $\sim$ on X by

$$x \sim y \;\Leftrightarrow\; (\forall\varphi\in\mathcal{L}_{K,\Box,\bigcirc})(K\varphi\in x \Leftrightarrow K\varphi\in y).$$

Clearly $\sim$ is an equivalence relation; let $[x]$ denote the equivalence class of x under $\sim$. For each $x\in X$, define

$$R(x) = \{y\in X : (\forall\varphi\in\mathcal{L}_{K,\Box,\bigcirc})(\Box\varphi\in x \Rightarrow \varphi\in y)\}.$$

Let $\mathcal{B} = \{R(x) : x\in X\}$, and let $\mathcal{T}$ be the topology generated by $\mathcal{B}$. It is easy to check that $\mathcal{B}$ is a basis for $\mathcal{T}$, and each $R(x)$ is a minimal neighbourhood about x (see, e.g., [16]). Given $x\in X$, define

$$f_\pi(x) = \begin{cases} \{\varphi : \bigcirc_\pi\varphi\in x\} & \text{if } \bigcirc_\pi\top\in x \\ \text{undefined} & \text{otherwise.}\end{cases}$$

Lemma 2. *Each f_π is a partial, open function $X \rightharpoonup X$.*

For each $p\in\textsc{prop}$, set $v(p) := \{x\in X : p\in x\}$. Let $\mathcal{X} = (X,\mathcal{T},\{f_\pi\}_{\pi\in\Pi},v)$. Clearly $\mathcal{X}$ is a dynamic topological subset model.

Lemma 3 (Truth Lemma). *For every $\varphi\in\mathcal{L}_{K,\Box,\bigcirc}$, for all $x\in X$, $\varphi\in x$ iff $(\mathcal{X},x,[x])\models\varphi$.*

Completeness is an easy consequence: if φ is not a theorem of DTEL, then $\{\neg\varphi\}$ is consistent and so can be extended by Lindenbaum's lemma to some $x\in X$; by Lemma 3, we have $(\mathcal{X},x,[x])\not\models\varphi$. $\Box$

References

1. Kremer, P., Mints, G.: Dynamic topological logic. Ann. Pure Appl. Logic **131**, 133–158 (2005)
2. Dabrowski, A., Moss, L., Parikh, R.: Topological reasoning and the logic of knowledge. Ann. Pure Appl. Logic **78**, 73–110 (1996)
3. Bjorndahl, A.: Topological subset space models for public announcements. In: van Ditmarsch, H., Sandu, G. (eds.) Jaakko Hintikka on Knowledge and Game-Theoretical Semantics. OCL, vol. 12, pp. 165–186. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-62864-6_6
4. Rabin, M.O., Scott, D.: Finite automata and their decision problems. IBM J. Res. **3**(2), 115–125 (1959)
5. Dijkstra, E.W.: A Discipline of Programming. Prentice-Hall, Englewood Cliffs (1976)
6. Francez, N., Hoare, C.A.R., Lehmann, D.J., de Roever, W.P.: Semantics of non-determinism, concurrency, and communication. J. Comput. Syst. Sci. **19**, 290–308 (1979)
7. Søndergaard, H., Sestoft, P.: Non-determinism in functional langauges. Comput. J. **35**(5), 514–523 (1992)
8. Troquard, N., Balbiani, P.: Propositional dynamic logic. The Stanford Encyclopedia of Philosophy ((Spring 2015 Edition)) Zalta, E.N. (ed.). https://plato.stanford.edu/archives/spr2015/entries/logic-dynamic/
9. Plaza, J.: Logics of public communications. Synthese **158**, 165–179 (2007)
10. Munkres, J.: Topology, 2nd edn. Prentice-Hall, Englewood Cliffs (2000)
11. Blackburn, P., de Rijke, M., Venema, Y.: Modal Logic. Cambridge Tracts in Theoretical Computer Science, vol. 53. Cambridge University Press, Cambridge (2001)
12. van Ditmarsch, H., Knight, S., Özgün, A.: Announcement as effort on topological spaces. In: Proceedings of the 15th conference on Theoretical Aspects of Rationality and Knowledge (TARK), pp. 95–102 (2015)
13. Baltag, A., Özgün, A., Vargas Sandoval, A.L.: Topo-logic as a dynamic-epistemic logic. In: Baltag, A., Seligman, J., Yamada, T. (eds.) LORI 2017. LNCS, vol. 10455, pp. 330–346. Springer, Heidelberg (2017). https://doi.org/10.1007/978-3-662-55665-8_23
14. Bjorndahl, A., Özgün, A.: Logic and topology for knowledge, knowability, and belief. In: Lang, J. (ed.) Proceedings of the 16th conference on Theoretical Aspects of Rationality and Knowledge (TARK) (2017)
15. van Ditmarsch, H., van der Hoek, W., Kooi, B.: Dynamic Epistemic Logic. Springer, Heidelberg (2008). https://doi.org/10.1007/978-1-4020-5839-4
16. Aiello, M., van Benthem, J., Bezhanishvili, G.: Reasoning about space: the modal way. J. Logic Comput. **13**(6), 889–920 (2003)