Temporal Landscapes: A Graphical Temporal Logic for Reasoning

Brendan Fong* Alberto Speranzon David I. Spivak*

Abstract

We present an elementary introduction to a new logic for reasoning about behaviors that occur over time. This logic is based on *temporal type theory*. The syntax of the logic is similar to the usual first-order logic; what differs is the notion of *truth value*. Instead of reasoning about whether formulas are true or false, our logic reasons about *temporal landscapes*. A temporal landscape may be thought of as representing the set of durations over which a statement is true. To help understand the practical implications of this approach, we give a wide variety of examples where this logic is used to reason about autonomous systems.

1 Introduction

Logical formalization of temporal considerations has a long and rich history, though the first modern treatment is probably the tense logic of Prior [Pri67], which has yielded what is now known simply as temporal logic. In temporal logic, as in any logical system, one has a syntactic way of building new formulas from simpler ones, say using conjunction and negation ($\varphi \wedge \neg\psi$), as well as a notion of model whose purpose is to specify the truth value of each such formula.

The first innovation of temporal logic is that the truth value of a proposition should be a function of time, for some notion (called a "frame") T of time. For example, if we take time to be the linear order $T = (\mathbb{R}, <)$, then one writes $7.5 \models \varphi$ to mean that φ is true at time 7.5. The truth value for the conjunction or negation of formulas is simply computed pointwise, for each time t.

Temporal logic gets its expressive power from various operators, which collect information about other times into the current time. For example, in linear temporal logic (LTL), one considers the binary *until* operator U. At time t, one may ask whether φ will hold until ψ holds, denoted $t \models \varphi \,\mathsf{U}\, \psi$, which more precisely means that there exists $t' > t$ such that $t' \models \psi$ and $t'' \models \varphi$ for all $t < t'' < t'$. One can understand all such operators as given by some sort of quantification over $t : T$, replacing each formula, e.g. φ, by a predicate $\varphi(t)$ in one variable. Restricting first-order logic by requiring

*Fong and Spivak were supported by AFOSR grant FA9550–17–1–0058.

that all atomic predicate symbols take only one variable, one obtains what is known as *first-order monadic logic*, and adding the 2-ary predicate $t < t'$, one obtains what is known as the first order monadic logic *of order*, $FO(<)$. It was shown by Kamp [Kam68] that temporal logic with the until operator, together with its past-tense cousin *since*, is precisely as expressive as $FO(<)$. Various additions and restrictions have been proposed over the years, in attempts to co-optimize between expressivity and computability.

A completely different approach to temporal reasoning, known as *temporal type theory* (TTT), was given in [SS19]. Instead of defining new logical operators that collate past and future times into the present, temporal type theory alters the very notion of truth itself, to make truth inherently depend on time. The goal of this article is to describe, in elementary terms, how TTT makes this idea precise, as well as how it can be used in practice.

Temporal type theory begins by defining a topological space called the *interval domain* $\mathbb{IR}$, whose points are the closed intervals $[t_1, t_2] \subseteq \mathbb{R}$, which we call *time-intervals*, and whose open sets are generated by open intervals (a, b), each of which consists of all points $[t_1, t_2]$ with $a < t_2 \leqslant t_2 < b$. A *sheaf* B on $\mathbb{IR}$ is a type of behavior: it assigns to each basic open (a, b) a set $B(a, b) \in \mathsf{Set}$, and for every $a \leqslant a' \leqslant b' \leqslant b$, it assigns a restriction function $\rho\colon B(a, b) \to B(a', b')$ that clips a longer-lasting behavior $x \in B(a, b)$ to a shorter-lasting behavior $\rho(x) \in B(a', b')$. Behavior types include:

1. $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$, and $\mathbb{R}$, the behavior of natural numbers, integers, rationals, and reals (unchanging over any interval (a, b));
2. $\widetilde{\mathbb{R}}$, the behavior of "varying" real numbers (changing continuously over any (a, b));[1]
3. for any vector field V on a topological space, the behavior of integral curves through V (of duration $b - a$);
4. for any graph G, the behavior of stochastically-timed walks through G;
5. more generally, for any hybrid system [Hen00], the behavior of all legal trajectories;
6. `Prop`, the behavior of truth values, also known as propositions, which one can think of as audits or monitors of behavior. `Prop` will be the main character in this paper;
7. the empty behavior and the singleton behavior (`Time` itself), as well as products, unions, subobjects, quotients, and exponentials of all the above.

Discussing behavior types in detail is out of scope for this paper, as it includes definitions of sheaves and toposes; the interested reader is referred to [SS19] for a technical discussion, or to [FS19, Chapter 7] for a gentle introduction. We *do not* assume the reader has knowledge of category theory, sheaf theory, or topos theory. Our goal in this paper is to give the reader a relatively self-contained understanding of `Prop`—the behavior type of truth values—in terms of *temporal landscapes*.

[1] Note that the constant reals can be considered as a subtype $\mathbb{R} \subseteq \widetilde{\mathbb{R}}$ of the varying real numbers.

We will also not give a detailed comparison between the expressive power of various temporal logics, such as Metric Interval Temporal Logic (MITL) [AFH96] or Signal Temporal Logic (STL) [MN04], with that of temporal type theory. The main difference is simply that temporal type theory is a type theory, meaning that it can combine and reason about various types of behaviors, as exemplified above. Another is that temporal logic assumes a kind of omniscience about the future: the truth value of a proposition at time t_1 can contain information about what occurs over a whole interval $[t_1, t_2]$. TTT does not have this omniscience: a proposition whose truth value depends on more than one moment—such as "whenever A occurs at time t_1, B must occur before time t_2"—is only falsifiable on long-enough intervals, e.g. those containing $[t_1, t_2]$. The aggregated truth value of a proposition over all intervals is its *temporal landscape*; information about what occurs over an interval is "stored" over the interval itself, not at its left endpoint. However, LTL and MITL do embed as a fragment of TTT [SS19, Chapter 8.6], so proofs from these logics are valid in TTT.

The differences between TTT and variants of standard temporal logic are numerous; we concentrate our efforts here on explaining how to work with the former. In particular, our focus will be on the *descriptive power* of temporal type theory: through increasingly complex examples we shall demonstrate how TTT—and in particular temporal landscapes—can be used to accurately model the relevant time-varying phenomena. We hope that this will be enough to give the reader a basic understanding of these ideas, even if translating these ideas into reasoning power then holds few subtleties.

Luckily, the base language of TTT is standard higher-order logic, which is quite similar to first-order logic. Not only should reasoning in TTT thus be more familiar to users with a grounding in predicate logic, a wide variety of proof assistants, including HOL, Lean, and Coq, can hence be easily adapted to provide formal verification of reasoning in TTT [CH88; Mou+15; NPW02].

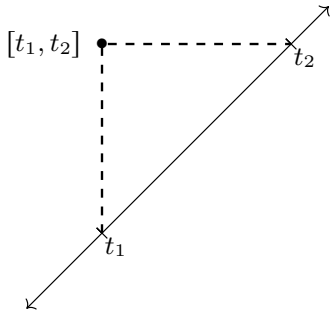
We will begin in Section 2 by defining temporal landscapes and the logical operations on them. In Section 3 we give several increasingly expressive examples of temporal landscapes in the context of autonomous agents. Finally in Sections 4 and 5, we briefly discuss how temporal landscapes may be used in practice, including how to reduce their computational complexity, and then conclude.

2 Temporal landscapes

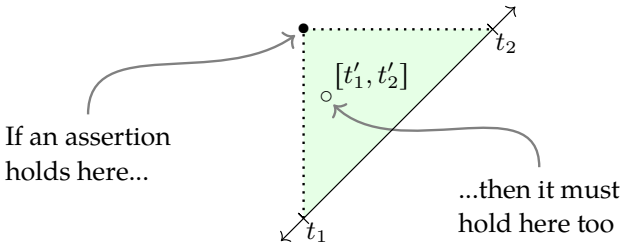
Temporal landscapes provide the *truth values* of a logical system, which we call *temporal landscape logic*. Truth values may be thought of as acceptable answers to "yes/no"-style questions. For example, in standard propositional logic, the truth values are simply `true` and `false`. In propositional logic then, the question "Is it raining?" may be answered with "yes" (`true`) or "no" (`false`). In temporal landscape logic, the answer to this question is a temporal landscape indicating precisely those time intervals during which it is raining. Let us be a bit more precise.

2.1 Definition of temporal landscape

Let $\mathbb{R}$ be the set of real numbers, thought of as representing points in time. Given two times $t_1, t_2 \in \mathbb{R}$ with $t_1 \leqslant t_2$, we write $[t_1, t_2]$ for the set of all times between t_1 and t_2; we call this a *time interval*. Graphically, we may represent a time interval by a point above the diagonal in the plane $\mathbb{R}^2$:

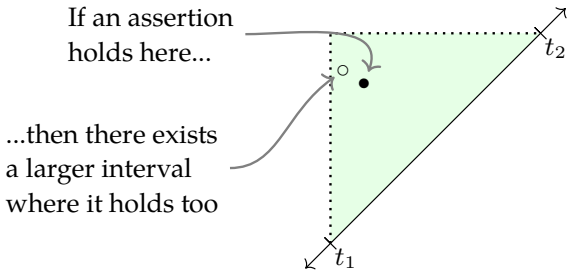


A *temporal landscape* is a set of time intervals with two special properties. The first is known as *down-closure*: if a time interval $[t_1, t_2]$ is in the temporal landscape, and $[t'_1, t'_2]$ is contained in $[t_1, t_2]$, then $[t'_1, t'_2]$ is in the temporal landscape too. This property makes the assumption that if an assertion holds throughout a time interval, then it holds on all subintervals. For example, if it is raining throughout the time interval from 9:00 to 13:00, then it is also raining throughout the time interval from 10:00 to 10:45. In pictures, this means that a temporal landscape must be closed under both moving right and moving downward:



(1)

The second property of temporal landscapes is an *openness* (sometimes called a *roundedness*) property: if an assertion holds on some $[t_1, t_2]$, then there exists some larger interval, $[t'_1, t'_2]$ with *both* $t'_1 < t_1$ and $t_2 < t'_2$. This larger interval may only be infinitesimally larger, but it must be strictly larger on both sides. In pictures, we illustrate this as follows:



(2)

The above is summarized in Definition 2.1.

Definition 2.1. *A* temporal landscape on $\mathbb{R}$ *is a set* L *of time intervals* $[t_1, t_2] \subseteq \mathbb{R}$*, where* $t_1 \leqslant t_2$*, such that*

(a) if $[t_1, t_2] \in L$*, and* $t_1 \leqslant t_1' \leqslant t_2' \leqslant t_2$*, then* $[t_1', t_2'] \in L$*.*

(b) if $[t_1, t_2] \in L$ *then there exists* $t_1' < t_1 \leqslant t_2 < t_2'$ *such that* $[t_1', t_2'] \in L$*.*

We write `Prop` *for the set of temporal landscapes.*

Together, requirements (a) and (b) state that temporal landscapes form the open sets of the Scott topology on the *interval domain* $\mathbb{IR}$, a well-studied topological space in domain theory [Gie+03]. While we will not need any topos or sheaf theory here, we remark that sheaves on $\mathbb{IR}$ form a topos, whose subobject classifier consists precisely of temporal landscapes; this topos is the subject of [SS19].

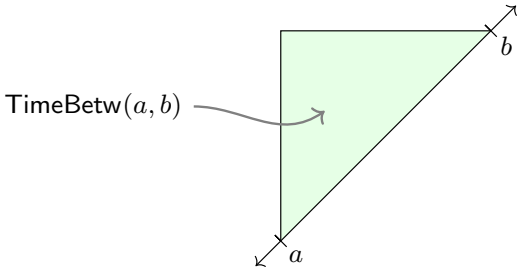
Remark 2.2. By definition—in particular Definition 2.1 (b)—a temporal landscape L does not include its boundary: it is an open set in $\mathbb{IR}$. Hence in our examples so far, Eqs. (1) and (2), we've drawn them using a dotted line. From now on, e.g. in Eq. (3), we use the visually simpler convention of drawing them with a solid line.

The simplest sort of temporal landscape is that of a *roof*; these form the basis for the topology on $\mathbb{IR}$.

Definition 2.3. *Given a pair* $a < b$ *in* $\mathbb{R}$*, the* roof over a, b *is the temporal landscape*

$$\mathsf{TimeBetw}(a, b) := \{[t_1, t_2] \mid a < t_1 \leqslant t_2 < b\}$$

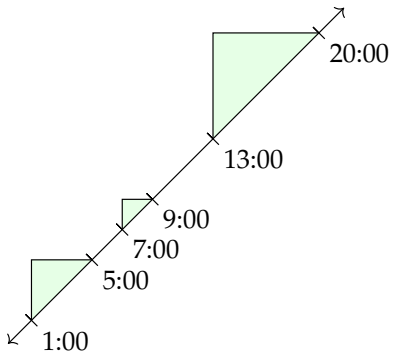
of all time intervals between a *and* b*. We draw this as follows:*



(3)

Given any pair of real numbers $a < b$*, a* temporal landscape on (a, b) *is a temporal landscape that is a subset of* $\mathsf{TimeBetw}(a, b)$*.*

Generically, a temporal landscape is a (possibly infinite) union of roofs. For example, yesterday's rainfall might be described by the temporal landscape

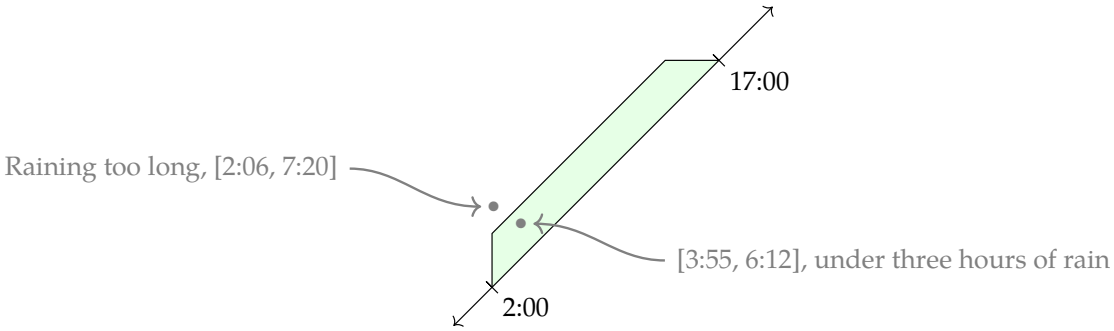


(4)

which says that it was raining from 1:00 to 5:00, 7:00 to 9:00, and 13:00 to 20:00. This is simply the union of the roofs $\mathsf{TimeBetw}(1{:}00, 5{:}00)$, $\mathsf{TimeBetw}(7{:}00, 9{:}00)$, and $\mathsf{TimeBetw}(13{:}00, 20{:}00)$.

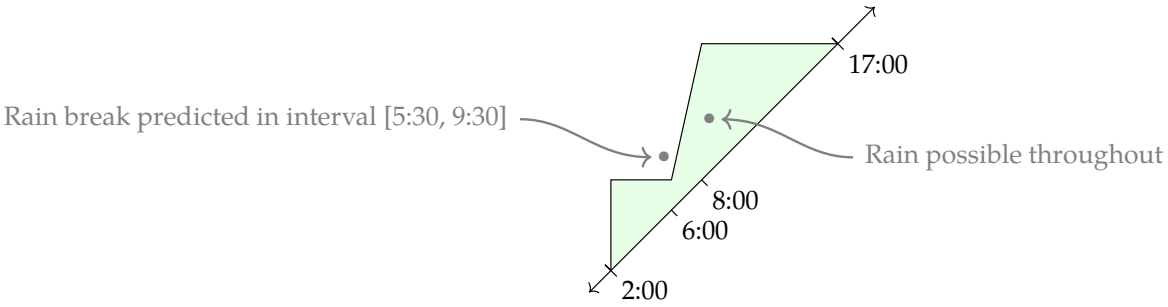
Note, however, that this temporal landscape can be generated by describing the rainfall pointwise: it suffices to simply answer the question whether it was raining at each point in time. This sort of semantics is describable with the usual temporal logic and its variants.

Temporal landscapes go beyond this. A key advantage of temporal landscapes is that they can describe temporal properties that *cannot* be established by considering only instantaneous points. Such descriptions are especially useful in specification or prediction of temporal behaviors. For example, tomorrow's rainfall forecast might predict rain between 2:00 and 17:00, but also that the rain will never last more than three hours at a time. This would be represented by the landscape



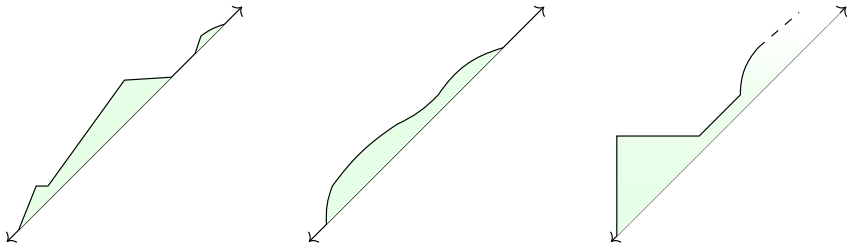
Note that this is an infinite union of the roofs $\mathsf{TimeBetw}(t, t+3)$ for all t between 2:00 and 17:00. Each point in this landscape represents a time interval over which continuous rainfall is forecast to be possible. So the fact that [3:55, 6:12] is in the green region means that it is possible that it will rain continuously from 3:55 to 6:12, while the fact that [2:06, 7:20] is outside the green region means that it will not rain continuously from 2:06 to 7:20.

Similarly, a storm that might hit for a short duration earlier in the day and/or a longer duration later in the day, but will definitely break at some point between 5:30 and 9:30, might be depicted by the temporal landscape:



Observe in particular that the point [5:30, 9:30] does not lie within this temporal landscape; this formalizes the forecasted break in the rain.

Here are a few generic examples of temporal landscapes:



The condition here is that the curves remain above the diagonal line and that their slope is piecewise continuous and remains in the interval $[0, \infty]$. Note that rotating by 45°, the slope condition becomes precisely the statement that the curve must be what is known as 1-Lipschitz.

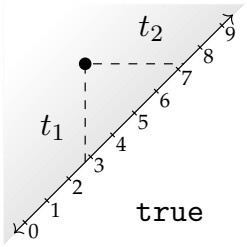
2.2 Temporal landscape logic

Temporal landscapes form the truth values of a logical system. More precisely, temporal landscapes form the elements of what is known as a Heyting algebra. This means that standard logical constants and operations, such as `true`, `false`, AND ($\wedge$), OR ($\vee$) and implication ($\Rightarrow$) have interpretations as temporal landscapes and operations on them.

To begin, we introduce the temporal landscapes `true` and `false`. The temporal landscape `true` contains all time intervals:

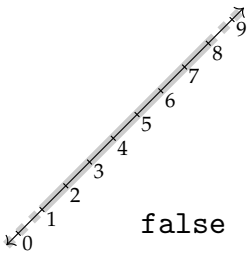
$$\texttt{true} \coloneqq \{[t_1, t_2] \mid t_1 < t_2 \in \mathbb{R}\}\,.$$

This landscape is the maximal one and is depicted as follows:



(5)

On the other hand, the temporal landscape `false` contains no time intervals at all: `false` $\coloneqq \varnothing$. It is the minimal landscape is depicted as empty:



(6)

Given temporal landscapes φ and ψ, their conjunction $\varphi \wedge \psi$ is given by their intersection, and their disjunction $\varphi \vee \psi$ is given by their union. For an explicit

example of conjunction, see Eq. (10) on page 13. In that example, the temporal landscape $\mathsf{Free}(\mathsf{Nbr}(v))$ on the right is the conjunction—that is, the intersection—of the temporal landscapes $\mathsf{Free}(w_r)$ and $\mathsf{Free}(w_u)$ on the left and center.

It is straightforward to check that the results of these operations are again temporal landscapes, and that $\wedge, \vee$ obey the usual properties of (constructive) first-order logic; for example, for any temporal landscape φ, we have $\varphi \wedge \texttt{true} = \varphi$.

Defining an implication that obeys the usual properties is a bit more subtle. Given temporal landscapes φ and ψ, we define the temporal landscape

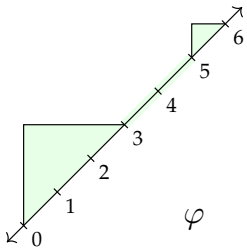
$$(\varphi \Rightarrow \psi) := \{[a, b] \mid \mathsf{TimeBetw}(a, b) \cap \varphi \subseteq \psi\}.$$

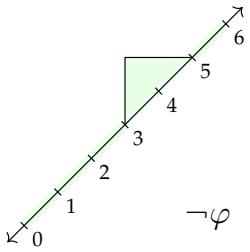
To become acquainted with implication in general, we start with a special case, namely that of negation.

The negation operator $\neg\varphi$ is given by $\neg\varphi := (\varphi \Rightarrow \texttt{false})$. Equivalently, since `false` is the empty set, we may write

$$\neg\varphi = \{[a, b] \mid \mathsf{TimeBetw}(a, b) \cap \varphi = \varnothing\}.$$

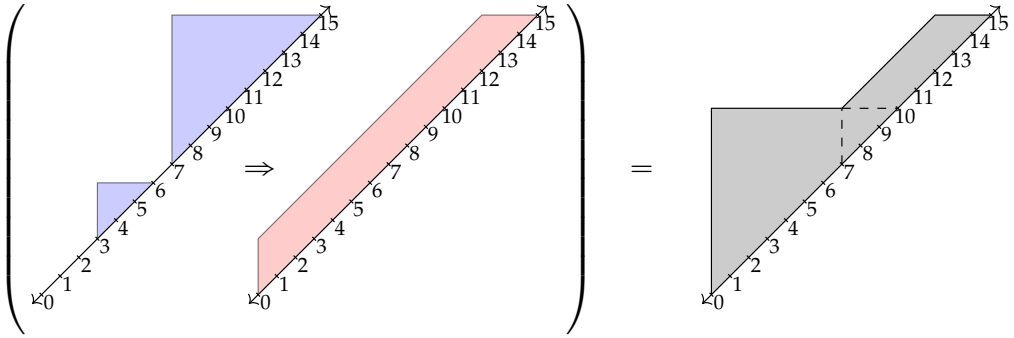
Thus the negation of a temporal landscape φ consists of all time intervals I that have empty intersection with any time interval in φ. In diagrams, this operation takes a landscape φ and draws a roof wherever the landscape is false (i.e. flat against the diagonal). For example, given the temporal landscapes on the left, its negation is shown on the right:





where we have restricted the landscapes over the time window $(0, 6)$.[2]

The visual intuition of the implication $\varphi \Rightarrow \psi$ generalizes that of negation, replacing the empty landscape `false` with ψ. The temporal landscape of $\varphi \Rightarrow \psi$ contains a roof over all time intervals within which φ is contained in ψ:



[2] In the following examples, we will often restrict ourselves to temporal landscapes on some arbitrary bounded interval, typically starting at 0, just for typographical convenience.

Now that we have defined the logical connectives, we move on to the quantifiers $\exists(x : X).\, P(x)$ and $\forall(x : X).\, P(x)$. The simplest case is when these quantifiers range over a constant type A, which we may think of simply as a set.[3] Given a set A, a function $P\colon A \to \mathtt{Prop}$ is a collection of $|A|$-many temporal landscapes. Taking their union defines the temporal landscape $\exists(a : A).\, P(a)$, which will be a temporal landscape. Taking their intersection may not satisfy condition (b) of Definition 2.1, so we define $\forall(a : A).\, P(a)$ to be the largest temporal landscape contained in this intersection.

Throughout this document, the reader will see the connectives $\wedge, \vee, \Rightarrow, \neg$, and the quantifiers $\exists$ and $\forall$. In each case, they refer to the operations on landscapes defined above.

3 Predicates over grid worlds

In this section we give increasingly expressive examples of how to use temporal landscapes to describe the behavior of an agent moving in various types of environments. We start with a fairly standard model, used in the Artificial Intelligence (AI) literature, to describe motions of an agent over a discretized space or environment.

In a non-temporal situation, it is typical to represent an environment as a two- (or higher-)dimensional regular grid where each region of the space is a cell and cells overlap only on specified boundaries. Mathematically it is convenient to model this with an undirected graph $G = (V, E)$, where $V = \{0, \ldots, N\}$ is a set of vertices, associated to subdivisions (or cells) of the environment, and $E \subseteq V \times V$ is the set of edges representing the fact that it is possible to move from one subdivision to another.

As we are working temporally, we replace sets with *behavior types*, which may be thought of as time-varying sets. More precisely, a behavior type specifies, for every temporal landscape, a set of behaviors that could take place over those durations. These sets of behaviors are required to obey a certain compatibility condition, so that for example behaviors over long time intervals restrict to behaviors on shorter subintervals.

Modelling the environment: constant and non-constant behavior types

Static environments. To simplify matters, let's first consider the case in which the environment does not vary in time. For this, we use constant behavior types: given a set X, the *constant behavior type* on X, by abuse of notation written again simply as X, is the behavior type that for every temporal landscape simply specifies X as its set of possible behaviors.

Suppose we want to say that our environment is modelled by the graph (V, E), and that it does not change over time. To do this, we simply take V and construct its constant behavior type V, and take E as the constant subtype of $V \times V$ consisting

[3] In temporal type theory we can also quantify over non-constant behavior types, e.g. $\forall(x : X).\, P(x)$, but this is a bit more technical. Since such quantification appears only in a single subsection (e.g. in Eq. (14)), we simply refer the reader to [SS19] for a definition.

precisely of the pairs (v_1, v_2) in the set E. The constancy of the subtype E says that the adjacency relation does not change: v_1 and v_2 either are adjacent or are not adjacent, independent of time.

To describe this fact logically in temporal type theory (TTT), we may write the formula

$$\forall(v_1, v_2 : V).\, (v_1, v_2) \in E \vee (v_1, v_2) \notin E.$$

That said, in higher order logics like TTT, one typically exchanges subobjects for predicates, e.g. replacing $E \subseteq V \times V$ with $E\colon V \times V \to \mathtt{Prop}$.[4] Then the statement would read

$$\forall(v_1, v_2 : V).\, E(v_1, v_2) \vee \neg E(v_1, v_2). \tag{7}$$

If we impose axiom Eq. (7) then for any v_1, v_2, the landscape for $E(v_1, v_2)$ is either the *always-true landscape* `true` (see Eq. (5)), or the *always-false landscape* `false` (see Eq. (6)), depending on whether we want an edge (v_1, v_2) or not. In this case we would say that (V, E) forms a *constant graph*.

Dynamic environments. It is also interesting, however, to decline to require that our environment obey axiom Eq. (7), and thus model environments in which the adjacency of cells changes over time. This makes sense in the autonomous setting if we imagine that sometimes a door is blocked or a secret passage is opened.

That said, for most situations, including all that follows, it is good enough to use a model of the environment where adjacency is *symmetric*: if v_1 is connected to v_2, then v_2 is connected to v_1. Using the language of TTT, this means our environment obeys the axiom

$$\forall(v_1, v_2 : V).\, (v_1, v_2) \in E \Leftrightarrow (v_2, v_1) \in E\,.$$

It will also be convenient to work with the function $V \to (V \to \mathtt{Prop})$ given by *currying* $E\colon V \times V \to \mathtt{Prop}$. It sends a cell $v : V$ to the (time-varying) set $\{v' : V \mid E(v, v')\}$ of cells adjacent to v. For our example, we want to consider the notion of *neighbor*, by which we mean an adjacent cell, not including the cell itself. For each $v : V$, neighbor is the subtype of V defined by the formula

$$\mathsf{Nbr}(v) := \{v' : V \mid v' \neq v \wedge E(v, v')\}\,.$$

It is worth noticing, once more, that we can interpret Nbr in terms of temporal landscapes, by considering $\mathsf{Nbr} : V \to (V \to \mathtt{Prop})$. This means that $\mathsf{Nbr}(v)(v')$ is a truth value—i.e. a temporal landscape—that describes when a given $v' : V$ is a neighbor of $v : V$. Of course, if we assume that (V, E) is a constant graph, again $\mathsf{Nbr}(v)(v')$ will either be the `true` landscape or the `false` landscape, depending on whether v and v' are neighbors.

In any case, notice that working with TTT feels much like working with the usual predicate logic and set theoretic constructions. In contrast with temporal logic then, where one must get used to working with new logical operators such as 'until' and 'since', for those who are trained in these languages, TTT provides an intuitive and easy to read language for reasoning about time.

[4] Recall that `Prop` is the set of temporal landscapes; see Definition 2.1.

Free/occupied cells: the negation operator

We now expand our example by adding a predicate:

$$\mathsf{Occ}(v) : V \to \texttt{Prop}\,.$$

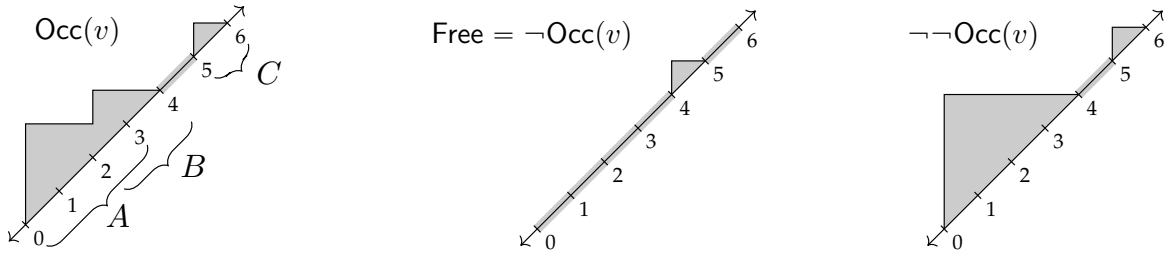
This predicate will be assumed to model the idea of a cell being *occupied*: for each cell v, it specifies the set of time intervals over which v is occupied. We will see that this predicate can capture situations that are more interesting than "mere occupancy", and that temporal landscapes provides a formal language to express such scenarios.

If a cell is not occupied, we will say that it is *free*. We further define $\mathsf{Free} := \neg\mathsf{Occ}$; for each $v : V$, the landscape $\mathsf{Free}(v)$ is the set of time intervals over which v is free. As mentioned in Section 2.2, double negation is not a trivial operation (the logic is constructive rather than Boolean). The predicate Occ gives a good example of why this might be useful, i.e. why it makes sense that $\mathsf{Occ} \stackrel{?}{=} \neg\neg\mathsf{Occ}$ need not hold.

Suppose we have agents A, B, and C, and predicates Occ_A, Occ_B, and Occ_C, which map a cell v to the temporal landscape of intervals over which the respective agent is in v. Suppose that we wish to define Occ to be the predicate describing the intervals over which at least one of the agents A, B, and C is in v. Note that this is slightly ambiguous in English, but we will see that the negation operator in TTT allows us to easily distinguish between the two readings of this sentence as Occ and $\neg\neg\mathsf{Occ}$.

To do this, define Occ to be the disjunction of these three predicates. For each v, $\mathsf{Occ}(v)$ thus specifies the time intervals over which a single agent, whether it be A, B, or C, remains in the cell throughout. Then $\neg\neg\mathsf{Occ}(v)$ specifies the time intervals over there is always at least one agent in v, but agents are allowed to come and go.

More concretely, fix some cell v and suppose that an agent A is in v throughout the interval $[0, 3]$, an agent B is in v throughout $[2, 4]$, and another agent C is in v throughout $[5, 6]$. Then the temporal landscapes for $\mathsf{Occ}(v)$, for $\mathsf{Free}(v) := \neg\mathsf{Occ}(v)$, and for $\neg\mathsf{Free} = \neg\neg\mathsf{Occ}$ are shown on the left, middle, and right, respectively:



While the middle diagram and right-hand diagram fit the usual interpretations of "when" the room is free / occupied, they are derived from the left-hand diagram, which is more expressive.

In particular, note that on the left-hand side diagram we have that $\mathsf{Occ}(v)$ does not contain the time interval $[1.5, 3.5]$, because this point falls in between the two roofs corresponding to A and B occupying the cell. This might appear strange, since there is at least one agent in the cell throughout $[1.5, 3.5]$, and thus we might expect $\mathsf{Occ}(v)$ to contain this interval. However, Occ expresses the more refined idea of those

intervals over which there exists any specific agent occupying v: agent A occupies v on the interval $[1, 3]$ and B occupies v on the interval $[2, 4]$; $\mathsf{Occ}(v)$ is their union.

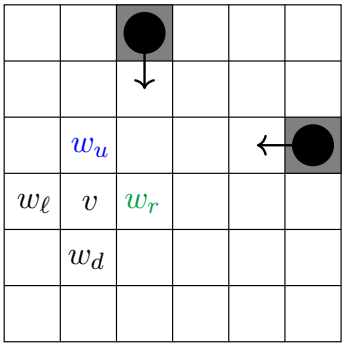
As an example of where the extra expressivity of Occ might be important, one might recall a scene from the movie *The Matrix*, where the main character Neo has déja vu, noticing the same cat appear twice in a similar spot, a signal of impending danger. If the déja vu occurs over any interval in Occ, either A, B, or C, will notice it, but there are intervals in $\neg\neg\mathsf{Occ}$, such as $[1.5, 3.5]$, for which it cannot be noticed. That is, if the cat appeared at 1.5 and then again at 3.5, then none of A, B, C will witness both occurrences.

While the *Matrix* example is fictional and thus might appear unrealistic, there are analogous realistic examples where the added expressiveness is useful in practice. Consider a simple situation where an emergency light is placed within the cell v and where two consecutive "blinks" of the light would represent a dangerous situation. In the case the light is ON at 1.5 and again at 3.5, the temporal landscape for $\mathsf{Occ}(v)$ would correctly capture the fact that such an alarm would be missed as there is not a single agent in the cell at those instances. Thus, unless A and B communicate about the status of the light, the notification of danger would be completely missed. Such communication—and memory / recall in general—amounts to a strategy for persistently encoding intervallic facts into the present state.

Objects in a room: quantifiers

In this subsection we use TTT to describe when neighbors of a cell are free (unoccupied), despite possibly moving obstacles. This might be important, for example, in the case of autonomous vehicles, where you might want to know when it is immediately adjacent to an obstacle, and hence should be wary of a collsion.

We will consider two scenarios, both depicted by the following diagram.



(8)

In these scenarios the large black dots each represent an obstacle. In the first scenario the objects are stationary; in the second, they move in the direction shown by the arrow.

To begin, note that we can extend a predicate Free over any subtype $N : V \to \mathtt{Prop}$ as follows:[5]

$$\mathsf{Free}(N) := \forall(v : V).\, N(v) \Rightarrow \mathsf{Free}(v)\,. \tag{9}$$

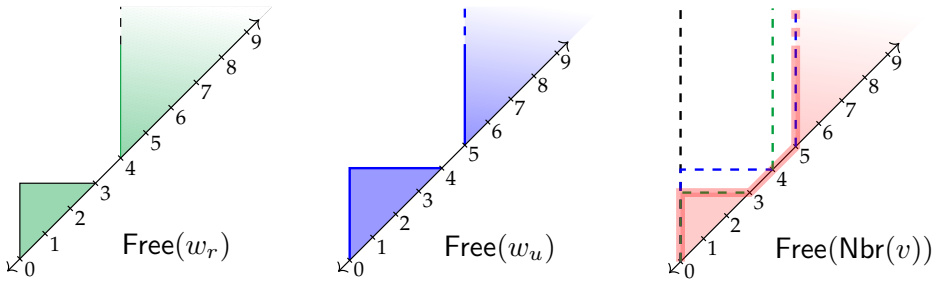
[5]Note that $\mathsf{Free}(N)$ is (constructively) equivalent to $\forall(v : V).\, \mathsf{Occ}(v) \Rightarrow \neg N(v)$.

This predicate describes the intervals over which all $v \in N$ are free. In particular, we will be interested in $\mathsf{Free}(\mathsf{Nbr}(v))$ for a cell v, which tells us when every neighbor of v is free, or equivalently, when none of v's neighbors are occupied.

Static objects. Assume that the objects, represented by the two large black dots, are static (i.e. forget the arrows in (8) for now). Consider the cell v indicated in (8). In the case that the black objects are static, one sees that the predicate $\mathsf{Free}(\mathsf{Nbr}(v))$ is the always-true landscape `true`, since the configuration of "free" cells does not change over time. In particular, if we were to draw the temporal landscapes $\mathsf{Free}(w)$ for each $w : \mathsf{Nbr}(v)$, each one would be the always-true landscape, and so their conjunction would be too.

Dynamic objects. Next we consider a situation in which the black dots represent moving objects. In this scenario, the two objects move in the indicated directions (one downwards and the other leftwards) at a rate of one cell per unit time. When they reach a cell adjacent to boundary of the domain, they remain there forever after.

In this case there is an equality of predicates $\mathsf{Free}(w_\ell) = \mathsf{Free}(w_d)$; both correspond to the always-true landscape, because these two cells are never occupied by either of the moving obstacles. For the predicate $\mathsf{Free}(w_r)$, however we note that the cell will be occupied for one time unit, between 3 and 4. Thus the cell w_r is free for the interval $[0, 3]$, and of course for any subinterval of it, such as $[1, 1.46]$. The cell w_r is also free for any interval $[4, b]$, as long as $4 < b$; the temporal landscape $\mathsf{Free}(w_r)$ is shown on the left below. Similar reasoning applied to $\mathsf{Free}(w_u)$ yields the temporal landscape shown in the center below:



(10)

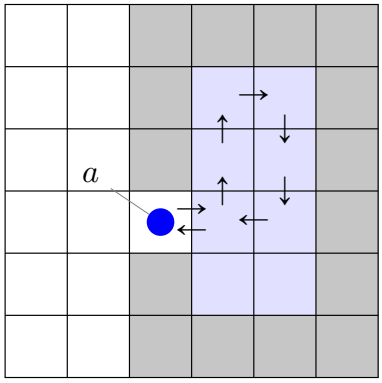
The temporal landscape for $\mathsf{Free}(\mathsf{Nbr}(v))$ is the conjunction of these temporal landscapes, i.e. all of v's neighbors must be free. As described in Section 2.2 the resulting temporal landscape is going to be the "minimum" landscape—shown in red boldface on the right in (10)—of the four landscapes shown in dashed lines $\mathsf{Free}(w_\ell)$ and $\mathsf{Free}(w_d)$ in black, $\mathsf{Free}(w_r)$ in green, and $\mathsf{Free}(w_u)$ in blue.

Looking at the red landscape on the right of (10) we immediately see that the neighborhood of v, namely $\mathsf{Nbr}(v)$, is not free in the interval $[3, 5]$.

Max dwell time in a room: implication

Let A denote the type of agents' IDs and let Pos: $A \to (V \to \mathtt{Prop})$ denote the predicate that an agent $a : A$ is at a vertex $v : V$. We also define a room R to be a subset of

vertices, $R\colon V \to \texttt{Prop}$.[6] Then for an agent $a : A$, one may write $\mathsf{Pos}(a) \subseteq R$ to denote the proposition $\forall(v : V).\, \mathsf{Pos}(a)(v) \Rightarrow R(v)$. The situation is shown below, where we indicate an agent by a blue dot and shade in lighter blue the cells forming a room R.



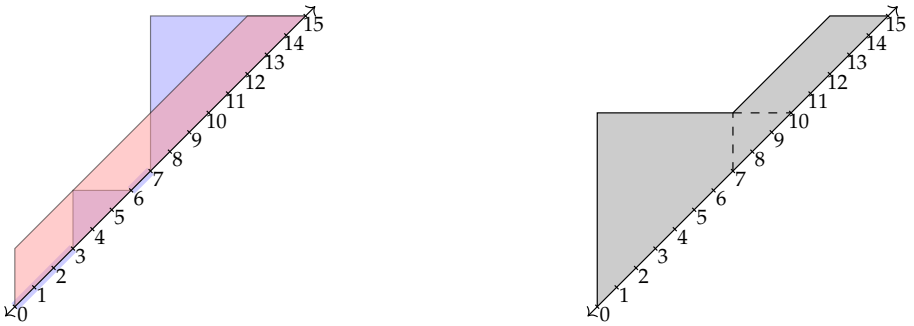
The wall—cells that are always occupied—are depicted in gray. Arrows depict possible trajectories that agent a can take to move within the room R and then exit.

Suppose we want to express the proposition that an agent stays in a room R for at most τ units of time before it must exit the room. To model this, for some $\tau : \mathbb{R}_{\geqslant 0}$, we can use the predicate

$$\forall(a : A).\, (\mathsf{Pos}(a) \subseteq R) \Rightarrow \exists(s : \mathbb{R}).\, \mathsf{TimeBetw}(s, s + \tau)\,, \tag{11}$$

which says that given an agent a, as long as a's position remains in room R, there is some start time s such that the clock remains between s and $s + \tau$.

Let us consider an example. Let $\tau = 3$ and suppose the agent is not in the room during the intervals $[0, 3]$ and $[6, 7]$, but is in the room during $[3, 6]$ and $[7, 15]$. The landscapes for the left and right hand side of (11), namely $\mathsf{Pos} \subseteq R$ and $\exists(s : \mathbb{R}).\, s < t < s + \tau$ respectively, are shown in blue and red on the left hand side below:



(12)

Note the temporal landscape of the right hand side of (11) (red), it is an "always true" capped at 3 time units. This is because given a time t there always exists a real value s with $t \in [s, s + \tau]$, and such predicate is true for intervals $[t_1, t_2]$ of length $t_2 - t_1 \leqslant 3$.

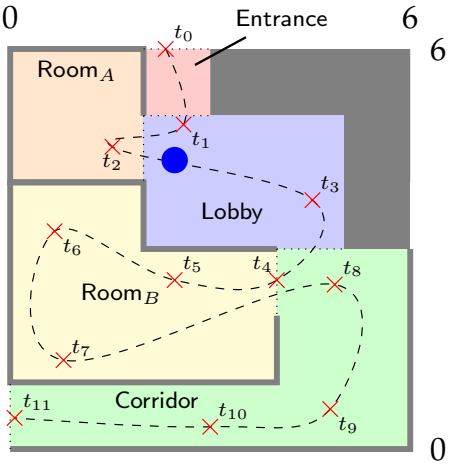
The temporal landscape for the entire predicate (11), is shown on the right hand side in (12). Note that in the interval [0,10] it is always true that the agent is within the room for at most 3 time units, however from [7,15] it is never true that the agent is

[6] Note that we have not said A and R are constant over time: agents might come into service or be decommissioned, and rooms might be constructed, demolished, or expanded over time.

within the room for at most 3 time units. Indeed, all we can say, is that on intervals of length at most 3, the agent is clearly is in the room for at most 3 time units, however this is not true for longer time intervals.

Regions and occupancy

Let us now consider a modified grid world, where instead of constructing a uniform spatial partitioning of the environment we leverage what we as humans would argue is a reasonable "semantic" subdivision of the space. Take for example a building, such a partitioning would be based on more abstract concepts than cells, such as "rooms", "corridors", "foyers" etc. To ground the discussion, consider the following diagram:



(13)

This picture depicts an agent (shown with a red dot) traversing a continuous environment, following a trajectory shown by a dashed line. The environment has been semantically subdivided into different regions (rooms).

In order to model the agent moving through the building, we begin by saying what a trajectory is. We normalize the building to be the square $S := [0, 6]^2 \subseteq \mathbb{R}^2$. Then define

$$\mathcal{X} := \left\{(x_1, x_2) : \widetilde{\mathbb{R}} \times \widetilde{\mathbb{R}} \mid 0 \leqslant x_1 \leqslant 6 \text{ and } 0 \leqslant x_2 \leqslant 6\right\}$$

to be the set of all possible time-parametrized trajectories through the building, i.e., the 6-unit square domain S. For example, in Eq. (13), we depict a time-parametrized trajectory over an interval (t_0, t_{11}), where say $t_0 = 0$, $t_1 = 1$, $t_2 = 2$, etc. making the distance traveled per unit time non-uniform along the trajectory.

It is worth noticing that in earlier examples, we considered a discretized space, whereas now we are considering a continuous space S, and behaviors defined over such space.

As in the discrete case, suppose we are given a predicate $\mathrm{Occ} : S \to \texttt{Prop}$ that models the subset of S (possibly changing in time) in which the agent cannot be. For example in Figure 13 we show some gray regions, which are intended to be walls, and hence always occupied/non-traversable. Thus if $s : S$ is in a wall, we

put $\mathsf{Occ}(s) := \mathtt{true}$. Again as in the discrete case, we use this to define a predicate $\mathsf{Free}\colon \mathcal{X} \to \mathtt{Prop}$, by $\mathsf{Free}(x) := \forall(s : S).\, \mathsf{Occ}(s) \Rightarrow \neg(x = s)$.

The agent is moving through the building, but to be more realistic we could imagine that the agent occupies space larger than a point, and that different parts of the agent move at slightly different speeds. Hence the agent consists of several different trajectories, all of which are close to one another, say within a distance of $\gamma : \mathbb{R}$. We define $\mathsf{close}\colon \mathcal{X} \times \mathcal{X} \to \mathtt{Prop}$ using the Euclidean norm $\mathsf{close}(x_1, x_2) := \|x_1 - x_2\|_2 \leqslant \gamma$. We also put an upper bound on the speed of the agent, say $\mathsf{v}_{\mathsf{max}} : \mathbb{R}$, and define the agent possible positions as the following behavior type:

$$\mathsf{AgentPos} := \left\{ \mathsf{p} : \mathcal{X} \to \mathtt{Prop} \;\middle|\; \begin{array}{l} \forall(x_1, x_2 : \mathcal{X}).\, \big((\mathsf{p}(x_1) \wedge \mathsf{p}(x_2)) \Rightarrow \mathsf{close}(x_1, x_2)\big) \wedge \\ \forall(x : \mathcal{X}).\, \mathsf{p}(x) \Rightarrow \big(\mathsf{Free}(x) \wedge -\mathsf{v}_{\mathsf{max}} \leqslant \dot{x} \leqslant \mathsf{v}_{\mathsf{max}}\big) \end{array} \right\}. \tag{14}$$

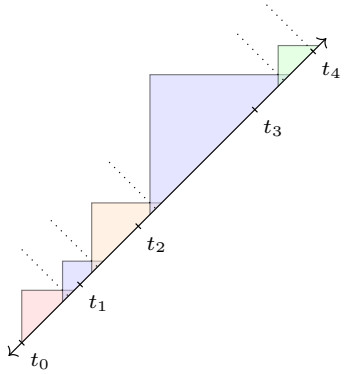
Here the bound $-\mathsf{v}_{\mathsf{max}} \leqslant \dot{x} \leqslant \mathsf{v}_{\mathsf{max}}$ is the temporal landscape consisting of those intervals $[t_1, t_2]$ over which $(t_1' - t_2')\mathsf{v}_{\mathsf{max}} < \|x(t_1') - x(t_2')\|_2 < (t_2' - t_1')\mathsf{v}_{\mathsf{max}}$ holds for all $t_1 < t_1' < t_2' < t_2$. For more on derivatives in temporal type theory, see [SS19, Section 7.3].

Let us consider the constant type $R := \{\mathsf{Room}_\mathsf{A}, \mathsf{Room}_\mathsf{B}, \mathsf{Entrance}, \mathsf{Lobby}, \mathsf{Corridor}\}$ representing the rooms as shown in Figure 13. Suppose we have a predicate $\mathsf{Room}\colon R \to (\mathcal{X} \to \mathtt{Prop})$, indicating the landscape on which a trajectory stays within a room. As before, for any moving agent $a : A$ with trajectories $\mathsf{Pos}(a) : \mathsf{AgentPos}$, let us denote with $\mathsf{Pos}(a) \subseteq r$ the predicate $\forall(x : \mathcal{X}).\, \mathsf{Pos}(a)(x) \Rightarrow \mathsf{Room}(r)(x)$, which says that agent a is in a room r if all of the trajectories that make up a are in r.

The temporal landscape for the proposition $\mathsf{Pos}(a) \subseteq r$ for when a single agent a is in a room r is a roof. Thus the temporal landscape of

$$\mathsf{AgentInARoom} := \exists(r : R).\, \mathsf{Pos}(a) \subseteq r\,.$$

is obtained by taking the union—namely the max—of these roofs:



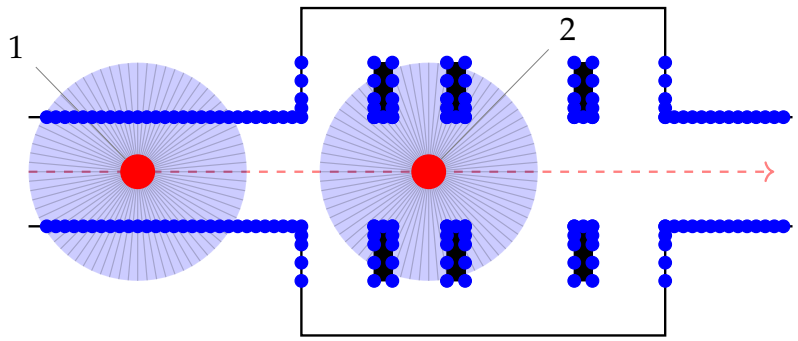
Note that because of the agent footprint there are intervals where the agent can be in two rooms.

Finally note that the agent is always in some room and thus $\neg\neg\mathsf{AgentInARoom}$ is the always-true landscape.

Landmarks and maps: "slanted" temporal landscapes

So far, for most of the examples—except for that in Eq. (12)—all the temporal landscapes have consisted of a finite union of roofs. One thus wonders when a "slanted" temporal landscape would be relevant in an application and what it would represent.

Consider the following picture:



(15)

An agent (red dot) travels, at constant velocity, within an indoor environment along the red dashed path. The agent is equipped with a range limited sensor, such as a LIDAR (blue disk) which emits a set of discrete laser beams. For each laser beam the LIDAR gets a return whenever a laser beam hits a surface. The measurement is the (possibly noisy) location of the surface along each beam (blue dots). We have denoted with (1) and (2) two specific locations along the path. We depict with small blue dots a possible set of sensor measurements—samples—along walls and columns (black rectangles).

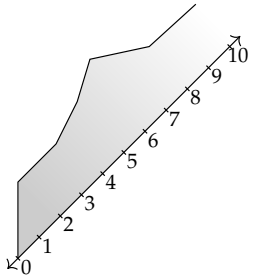
Further suppose that the agent is equipped with a fixed amount of onboard memory, so that not all the samples can be stored. When the buffer used to store samples is full, past samples will be deleted to make space for new samples. If we identify each sample with an integer $i : \mathbb{N}$, we can define the predicate $\mathsf{SampleInMem}(i)$ that will be true over an interval $[t_1, t_2]$ as long as the sample i is in the memory of the agent.

The temporal landscape for $\mathsf{SampleInMem}(i)$ is clearly a roof over the interval $[t_1, t_2]$ where t_1 is the instance when the sample i was first stored in memory and t_2 is the instance of time when it was overwritten by a new sample (of course $t_2 = +\infty$ when there is enough memory so that no overwriting occurs).

The following predicate will have a "slanted" temporal landscape:

$$\mathsf{SamplesInMem} = \bigvee_i \mathsf{SampleInMem}(i)\,,$$

For example, it might look like the following:

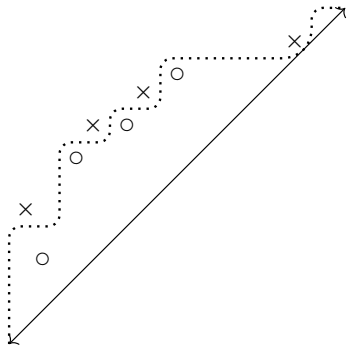


Initially, in the corridor, the number of samples is high and the memory will be fully allocated. As new samples are obtained, old ones will be overwritten. Assuming a constant velocity and number of samples per unit of time, we have a constant overwriting so that a sample is in memory only over a constant size interval: thus the landscape will be parallel to the time line. As the agent enters a part of the environment that has fewer surfaces, the number of samples per unit time decreases, and thus samples will persist in memory over longer and longer periods of time, especially given that the environment becomes sparser as the agent moves left to right. Once the agent start sensing the beginning of the right-most corridor, the number of samples starts to quickly increase and thus the persistence of a sample in memory decreases and returns to the maximum amount possible given by the onboard memory, again shown as a landscape that is parallel to the time line.

4 Finite approximations and monitoring

We conclude with a short section discussing how temporal landscapes may appear in practice. For example, autonomous agents may communicate using temporal landscapes as a way of being predictable, e.g. for coordination or collision avoidance. Temporal landscapes could also serve as a standard protocol within an agent. For example, its predictive unit could issue a temporal landscape specifying the expected occupancy of a given room, for use by a path-planning unit.

However temporal landscapes as we have discussed them involve an infinite amount of data (any curve with slope in $[0, \infty]$). For representation and computation, it may be important to use finite approximations. In other words, rather than express in perfect detail the set of intervals on which some statement will hold, it might be convenient to express a finite set of intervals on which it will *surely* hold and another finite set of intervals on which it will *surely not* hold. Writing $\circ$ for the included intervals and $\times$ for the excluded intervals, the diagram would look something like this:



The dotted line represents any temporal landscape that includes all the $\circ$'s and excludes all the $\times$'s.

This sort of finite information could be supplied by a monitor, which returns $\circ$'s wherever the proposition was checked to hold and $\times$'s wherever it was seen to fail. Such finite approximations (together with a special landscape for `false`) again form a logical system: one can easily define connectives, as well as quantifiers over finite

samples. This provides the beginnings of a computationally tractable approach to the logic of temporal landscapes.

5 Conclusion

In this paper, we have attempted to give an intuitive introduction to the logic of temporal type theory in terms of temporal landscapes. On the one hand, these are just collections of time intervals over which a proposition may be true. On the other, they can be drawn as Lipschitz functions and hence visualized. They form a logical system, where all of the connectives and quantifiers are defined by operations on these Lipschitz functions.

After introducing these landscapes, we discussed a series of examples from the domain of autonomous agents. These became fairly complex, e.g. considering an agent's position not just as a point but as a collection of points, each moving with bounded speed, avoiding possibly moving obstacles, and storing recent LIDAR measurements in a small-capacity memory that is constantly being overwritten. These examples point to the great expressivity of temporal type theory.

In practice, TTT can serve as a sort of big tent, where calculations from model checkers or ODE solvers can be embedded. While infinite in nature, we explained how temporal landscapes can be finitely approximated. We thus hope to have shown how TTT can be used to specify and guide algorithmic developments in autonomous systems and any modeling environment in which time is an issue.

References

[AFH96] Rajeev Alur, Tomás Feder, and Thomas A. Henzinger. "The benefits of relaxing punctuality". In: *Journal of the ACM (JACM)* 43.1 (1996), pp. 116–146.

[CH88] Thierry Coquand and Gérard Huet. "The calculus of constructions". In: *Information and Computation* 76.2-3 (1988), pp. 95–120.

[FS19] Brendan Fong and David I. Spivak. *An Invitation to Applied Category Theory: Seven Sketches in Compositionality*. Cambridge University Press, 2019.

[Gie+03] G. Gierz, K. H. Hofmann, K. Keimel, J. D. Lawson, M. Mislove, and D. S. Scott. *Continuous lattices and domains*. Vol. 93. Encyclopedia of Mathematics and its Applications. Cambridge University Press, Cambridge, 2003, pp. xxxvi+591.

[Hen00] Thomas A Henzinger. "The theory of hybrid automata". In: *Verification of Digital and Hybrid Systems*. Springer, 2000, pp. 265–292.

[Kam68] Johan Anthony Wilem Kamp. "Tense logic and the theory of linear order". PhD thesis. University of California, Los Angeles, 1968.

[MN04] Odet Maler and Dejan Nickovic. "Monitoring temporal properties of continuous signals." In: *FORMATS*. 2004, pp. 152–166.

[Mou+15] Leonardo de Moura, Soonho Kong, Jeremy Avigad, Floris Van Doorn, and Jakob von Raumer. "The Lean theorem prover (system description)". In: *International Conference on Automated Deduction*. Springer. 2015, pp. 378–388.

[NPW02] Tobias Nipkow, Lawrence C Paulson, and Markus Wenzel. *Isabelle/HOL: a proof assistant for higher-order logic*. Vol. 2283. Springer Science & Business Media, 2002.

[Pri67] Arthur N. Prior. *Past, present and future*. Vol. 154. Clarendon Press Oxford, 1967.

[SS19] Patrick Schultz and David I. Spivak. *Temporal Type Theory: A topos-theoretic approach to systems and behavior*. Springer, Birkhäuser, 2019.