On Binding in the Spatial Logics for Closure Spaces*

Laura Bussi[1,2], Vincenzo Ciancia[1], Fabio Gadducci[2], Diego Latella[1], and Mieke Massink[1]

[1] CNR-ISTI, Pisa, Italy
{l.bussi, v.ciancia, d.latella, m.massink}@isti.cnr.it
[2] Dipartimento di Informatica, Università di Pisa, Pisa, Italy
fabio.gadducci@unipi.it

Abstract. We present two different extensions of the spatial logic for closure spaces (SLCS), and its spatio-temporal variant (τSLCS), with spatial quantification operators. The first concerns the existential quantification on *individual points* of a space. The second concerns the quantification on *sets* of points. The latter amounts to a form of quantification over atomic propositions, thus without the full power of second order logic. The spatial quantification operators are useful for reasoning about the existence of particular spatial objects in a space, their spatial relation with respect to other spatial objects, and, in the spatio-temporal setting, to reason about the dynamic evolution of such spatial objects in time and space, including reasoning about newly introduced items. In this preliminary study we illustrate the expressiveness of the operators by means of several small, but representative, examples.

Keywords: Closure spaces; Spatial logics; Spatio-temporal logics; Binding; Propositional Quantifiers;

1 Introduction

The notion of *space* plays a crucial role in the heterogeneous design, implementation and use of distributed (computer) systems. Debates on the nature of space date back to the ancient times of—and involve—Greek philosophers like Plato and Aristotle. In modern mathematics, spaces are typically defined as sets (of points) with some additional structure. This is the case, for instance, for *topological spaces*, where such additional structure captures a notion of "nearness".

* Research partially supported by the MIUR PRIN 2017FTXR7S IT-MaTTerS. The authors are listed in alphabetical order, as they equally contributed to this work. This is a post-print of the paper "On Binding in the Spatial Logics for Closure Spaces", by L. Bussi, V. Ciancia, F. Gadducci, D. Latella, and M. Massink. In Margaria, T., Steffen, B. (eds) *Leveraging Applications of Formal Methods, Verification and Validation. Verification Principles.* ISoLA 2022. Lecture Notes in Computer Science, vol 13701. Springer, Cham pp. 479-497, Springer, 2022, available at: https://link.springer.com/chapter/10.1007/978-3-031-19849-6_27.

Modal logics have been used since a long time as a means for reasoning about necessity and possibility, but also about (continuous) space. In fact, a topological interpretation of the $\Diamond$ modality was proposed already in the thirties by Tarski who later proved, together with McKinsey, that the simple modal logic **S4** is complete for interpreting $\Diamond$ as *topological closure* on Euclidean spaces, specifically the reals (see e.g. [6] for a detailed account).

Unfortunately, topological spaces turn out to be rather restrictive since there are structures that are useful to represent certain kinds of space but that are not topologies, like, for instance, general graphs or heterogeneous structures including both continuous and discrete notions of space. Consequently, in our work, we consider a larger class of models, namely that of *Čech closure spaces*, a generalisation of topological spaces [33, 20]. The relevant logics have been extended accordingly. In [15, 14] the *Spatial Logic for Closure Spaces* (SLCS) has been proposed that has the same operators as **S4**—where the closure operator is denoted by $\mathcal{N}$, standing for "near", instead of by $\Diamond$—plus a *surrounded* operator such that a point satisfies $\Phi_1 \,\mathcal{S}\, \Phi_2$ if it lays in a region that (i) consists of points all satisfying Φ_1 and (ii) is surrounded by points satisfying Φ_2.

SLCS has been extended with temporal modalities in [13, 12], giving rise to a spatio-temporal logic to reason also on heterogeneous properties concerning dynamic aspects of systems physically distributed in space. Spatial and spatio-temporal *model checking / monitoring* algorithms have been proposed in [15, 14, 13, 12, 28] and associated tools, among which VoxLogicA [5] and topochecker [16, 9], have been developed [5, 12, 22, 13, 21]. These, in turn, have been used in various applications, such as bike-sharing [16], Turing patterns [28] and medical image analysis [3, 9, 5, 4]—where a digital image is interpreted as a regular grid, i.e. a graph with an edge relation that models 2D pixel, or 3D voxel, *adjacency* (also called an *adjacency space*). Recently, the approach has also been extended to *polyhedral models* and polyhedral model checking [7, 27], leading to the polyhedral model checker PolyLogicA [7]. Notions of spatial bisimilarity have been proposed as well, and their potential for model minimisation plays an important role in the context of model-checking optimisation [17].

Despite their expressiveness, SLCS and its temporal extension are not suitable for the expression of properties involving dynamic entities that may *appear* or *disappear* over time. In order to reason about such entities, quantification has been introduced in well-known temporal logics such as LTL [25] and CTL [29]. Furthermore, in order to reason about graphs whose topology may change over time, combinations of temporal and graph logics have been proposed [19, 1].

A different perspective on quantification is given by *propositional quantification*, introduced in modal logics by Kripke [24] and thoroughly investigated in, for instance, [8, 23]. In the latter work, the object of quantification are propositions, and this allows one to reason about possible changes of properties holding in a certain region. In the case of spatial model checking for medical image analysis we mentioned before, this kind of quantification can be useful, for instance, to verify the occurrence of a new lesion in human organ tissue such as in the brain. Introducing quantification may introduce complexity issues [2] and may require the development of novel model checking tools and algorithms [30].

In the present paper, we present a preliminary investigation (ultimately aimed at model checking applications in medical image analysis, such as in [5]), by means of examples, on suitable extensions of `SLCS`, and its spatio-temporal variant, with quantification operators and with a notion of fresh point creation. We consider both *quantification on points* in the relevant space and *quantification on atomic propositions*. The latter allows for the characterisation of properties of *sets* of points, yet without involving the full power of second order logic.

The paper is organised as follows: Section 2 recalls some preliminary definitions, including the "kernel logic" `SLCS` and its temporal extension τ`SLCS`. Section 3 presents $\exists$x`SLCS`, the point quantification extension of `SLCS`, whereas the temporal variant thereof is presented in Section 4. The introduction of new points during the evolution of a system is briefly discussed in Section 5. Section 6 shows an extension of `SLCS` with atomic predicate quantification. Finally, Section 7 presents a brief discussion on this work and some lines of future work.

2 Preliminaries

Given a set X, we let $\mathcal{P}(X)$ denote the powerset of X. For function $f : A \to B$, $a \in A$, and $b \in B$, we let $f[a \mapsto b]$ be defined as follows: $f[a \mapsto b](x) \triangleq b$, if $x = a$, and $f[a \mapsto b](x) \triangleq f(x)$ otherwise.

The main notion which our framework for modelling space is based on is that of *Čech Closure Spaces* [33] that provide a convenient common framework for the study of several different kinds of spatial models, including models of discrete and continuous space [31]. More recently the notion of closure space has been used in the context of Artificial Intelligence (see for example [20]). We briefly recall several definitions and results on closure spaces, most of them from [20].

Definition 1 (Closure Space – CS). *A* closure space*, CS for short, is a pair $(X, \mathcal{C})$ where X is a non-empty set (of* points*) and $\mathcal{C} : \mathcal{P}(X) \to \mathcal{P}(X)$ is a function satisfying the following axioms: (i) $\mathcal{C}(\emptyset) = \emptyset$; (ii) $A \subseteq \mathcal{C}(A)$ for all $A \subseteq X$; and (iii) $\mathcal{C}(A_1 \cup A_2) = \mathcal{C}(A_1) \cup \mathcal{C}(A_2)$ for all $A_1, A_2 \subseteq X$.* •

In the remainder of the paper, we consider only closure spaces $(X, \mathcal{C})$ where X is equipped with equality and $x = x'$ is decidable, for all $x, x' \in X$. It is worth pointing out that topological spaces coincide with the sub-class of CSs where $\mathcal{C}$ satisfies the *idempotence* axiom $\mathcal{C}(\mathcal{C}(A)) = \mathcal{C}(A)$ (see [33, 20] for details).

Definition 2 (Quasi-discrete CS – QdCS). *A* quasi-discrete closure space *is a CS $(X, \mathcal{C})$ such that for each $A \subseteq X$ it holds that $\mathcal{C}(A) = \bigcup_{x \in A} \mathcal{C}(\{x\})$.* •

Given any relation $R \subseteq X \times X$, define the function $\mathcal{C}_R : \mathcal{P}(X) \to \mathcal{P}(X)$ as follows: for all $A \subseteq X$, $\mathcal{C}_R(A) \triangleq A \cup \{x \in X \mid a \in A \text{ exists s.t. } (a, x) \in R\}$. It is easy to see that, for any R, $\mathcal{C}_R$ satisfies the axioms of Definition 1 and so $(X, \mathcal{C}_R)$ is a CS. The following theorem is a standard result in the theory of CSs [20].

Theorem 1. *A CS* $(X, \mathcal{C})$ *is quasi-discrete if and only if there is a relation* $R \subseteq X \times X$ *such that* $\mathcal{C} = \mathcal{C}_R$. $\square$

A notable example of quasi-discrete closure spaces that are not necessarily topological spaces is that of general graphs, i.e. graphs where no restriction is imposed on the edge relation. There are also closure spaces that are neither quasi-discrete nor topological. An example of such spaces is a heterogeneous space like the disjoint union of an Euclidean space—a topological space which is clearly not quasi-discrete—with a quasi-discrete, but not topological, closure space[3]. In the sequel, whenever a CS $(X, \mathcal{C})$ is quasi-discrete, we use $\vec{\mathcal{C}}$ to denote $\mathcal{C}_R$, and, consequently, $(X, \vec{\mathcal{C}})$ to denote the closure space, abstracting from the specification of R, when the latter is not necessary. We let $\overleftarrow{\mathcal{C}}$ denote $\mathcal{C}_{R^{-1}}$.

We use *paths* over QdCSs; we follow the tradition of topology and define them based on the notion of *continuous function*.

Definition 3 (Continuous function). *A* continuous *function from* $(X_1, \mathcal{C}_1)$ *to* $(X_2, \mathcal{C}_2)$ *is a function* $f : X_1 \to X_2$ *such that* $f^*(\mathcal{C}_1(A)) \subseteq \mathcal{C}_2(f^*(A))$ *for all sets* $A \subseteq X_1$*, where we let* $f^*(B) = \bigcup_{x \in B} f(x)$. $\bullet$

Should $(X_1, \mathcal{C}_1)$ be a QdCS, continuity coincides with just requiring that $f^*(\mathcal{C}_1(\{x\})) \subseteq \mathcal{C}_2(\{f(x)\})$ for all $x \in X_1$. So, let $(\mathbb{N}, \mathcal{C}_{\mathtt{succ}})$ be the QdCS of natural numbers, where `succ` is the *successor* relation, i.e. $\mathtt{succ} \triangleq \{(m, n) \mid n = m + 1\}$.

Definition 4 (Quasi-discrete path). *A* quasi-discrete path $(X, \vec{\mathcal{C}})$ *is a continuous function from* $(\mathbb{N}, \mathcal{C}_{\mathtt{succ}})$ *to* $(X, \vec{\mathcal{C}})$. $\bullet$

In the sequel we will consider only quasi-discrete paths. Below, we introduce the notion of closure *model*. To that purpose, we assume that a set `AP` of *atomic propositions* is given. In addition, the points of the model are often enriched with suitable *attributes*, that can facilitate the definition of appropriate propositions, as briefly described below. Obvious examples of attributes are the *red, green* and *blue* components of pixels, seen as RGB vectors—recording the colour intensities—in digital images, when the latter are seen as adjacency spaces.

Definition 5 (Closure model – CM). *Given a set of* atomic propositions `AP`*, a set of* attribute names A*, and a set of* attribute values AV*, a* closure model*, CM for short, is a tuple* $(X, \mathcal{C}, \mathcal{A}, \mathcal{V})$ *consisting of a closure space* $(X, \mathcal{C})$*, a valuation* $\mathcal{A} : X \times A \to AV$*, assigning to each point and attribute the value of the attribute at that point, and a valuation* $\mathcal{V} : \mathtt{AP} \to \mathcal{P}(X)$ *assigning to each atomic predicate the set of points where it holds.* $\bullet$

All the definitions for CSs apply to CMs as well; thus, a *quasi-discrete closure model* (QdCM for short) is a CM $\mathcal{M} = (X, \vec{\mathcal{C}}, \mathcal{A}, \mathcal{V})$ where $(X, \vec{\mathcal{C}})$ is a QdCS.

[3] The disjoint union $(X_1, \mathcal{C}_1) + (X_2, \mathcal{C}_2)$ of closure spaces $(X_1, \mathcal{C}_1)$ and $(X_2, \mathcal{C}_2)$ is the closure space $(X, \mathcal{C})$ whose set of points X is the disjoint union $X_1 + X_2 \triangleq \{(x, 1) \mid x \in X_1\} \cup \{(x, 2) \mid x \in X_2\}$ while, for $A \subseteq X_1 + X_2$ we define $\mathcal{C}(A) \triangleq \{(x, 1) \mid x \in A_1\} \cup \{(x, 2) \mid x \in A_2\}$ with $A_j \triangleq \{x \mid (x, j) \in A\}$ for $j = 1, 2$.

2.1 SLCS: The Spatial Logic for Closure Spaces

We recall the kernel logic SLCS; in the present paper, we interpret it on QdCMs.

Definition 6 (Spatial Logic for Closure Spaces – SLCS). *For $p \in \mathtt{AP}$ the syntax of the logic is the following*

$$\Phi ::= p \mid \neg\,\Phi \mid \Phi \,\wedge\, \Phi \mid \vec{\rho}\,\Phi[\Phi] \mid \overleftarrow{\rho}\,\Phi[\Phi]$$

Satisfaction $\mathcal{M}, x \models \Phi$ of an SLCS *formula Φ at point $x \in X$ in QdCM $\mathcal{M} = (X, \vec{\mathcal{C}}, \mathcal{A}, \mathcal{V})$ is defined by induction on the structure of formulas*

$$\begin{array}{lll}
\mathcal{M}, x \models p & \Leftrightarrow & x \in \mathcal{V}(p) \\
\mathcal{M}, x \models \neg\,\Phi & \Leftrightarrow & \mathcal{M}, x \models \Phi \textit{ does not hold} \\
\mathcal{M}, x \models \Phi_1 \,\wedge\, \Phi_2 & \Leftrightarrow & \mathcal{M}, x \models \Phi_1 \textit{ and } \mathcal{M}, x \models \Phi_2 \\
\mathcal{M}, x \models \vec{\rho}\,\Phi_1[\Phi_2] & \Leftrightarrow & \textit{path } \pi \textit{ and index } \ell \textit{ exist s.t. } \pi(0) = x \textit{ and } \mathcal{M}, \pi(\ell) \models \Phi_1 \\
& & \textit{and for all indexes } j : 0 < j < \ell \textit{ implies } \mathcal{M}, \pi(j) \models \Phi_2 \\
\mathcal{M}, x \models \overleftarrow{\rho}\,\Phi_1[\Phi_2] & \Leftrightarrow & \textit{path } \pi \textit{ and index } \ell \textit{ exist s.t. } \pi(\ell) = x \textit{ and } \mathcal{M}, \pi(0) \models \Phi_1 \\
& & \textit{and for all indexes } j : 0 < j < \ell \textit{ implies } \mathcal{M}, \pi(j) \models \Phi_2 \quad \bullet
\end{array}$$

A point x satisfies formula $\vec{\rho}\,\Phi_1[\Phi_2]$ if a point that satisfies Φ_1 can be reached from x, via a path whose internal points, if any, all satisfy Φ_2. Conversely, x satisfies formula $\overleftarrow{\rho}\,\Phi_1[\Phi_2]$ if it can be reached from a point that satisfies Φ_1, via a path whose internal points, if any, all satisfy Φ_2.

Note that, in the context of space, and in particular when dealing with notions of directionality (e.g. one-way roads, public area gates), it is essential to be able to distinguish between the concept of "reaching" and that of "being reached". A formula like $\vec{\rho}(\mathtt{rescue\text{-}area} \wedge \neg(\overleftarrow{\rho}\,\mathtt{danger\text{-}area})[\mathtt{true}])[\mathtt{safe\text{-}corridor}]$, given a suitable interpretation of the atomic propositions, expresses the fact that, via a safe corridor, a rescue area can be reached that cannot be reached from a dangerous area. Such situations have no obvious counterpart in the temporal domain, where there can be more than one future, like in the case of branching time logics, but there is typically only *one*, fixed, past, i.e. the one that occurred[4].

It is also worth noting that it is not always possible to define the reversed path in CSs: indeed, while this is immediate in the case, for instance, of graphs, the same idea is not applicable in continuous spaces, as, e.g., used in [7, 27]. Thus, for the sake of generality, the semantics for the "reach" and "being reached" operators is given explicitly.

The standard derived operators $\vee$ and $\implies$ will be used in the sequel: $\Phi_1 \vee \Phi_2 \equiv \neg(\neg\Phi_1 \wedge \neg\Phi_2)$ and $\Phi_1 \implies \Phi_2 \equiv \neg\Phi_1 \vee \Phi_2$. We recall here that the proximity operator $\vec{\mathcal{N}}$ defined as $\mathcal{M}, x \models \vec{\mathcal{N}}\,\Phi \Leftrightarrow x \in \vec{\mathcal{C}}(\{x' \in X \mid \mathcal{M}, x' \models \Phi\})$, for QdCMs can be derived from the *reachability* one, namely $\vec{\mathcal{N}}\,\Phi \equiv \overleftarrow{\rho}\,\Phi[\mathtt{false}]$; similarly $\overleftarrow{\mathcal{N}}\,\Phi \equiv \vec{\rho}\,\Phi[\mathtt{false}]$, with $\mathcal{M}, x \models \overleftarrow{\mathcal{N}}\,\Phi \Leftrightarrow x \in \overleftarrow{\mathcal{C}}(\{x' \in X \mid \mathcal{M}, x' \models \Phi\})$. In particular, $\overleftarrow{\mathcal{N}}$ coincides with the classical $\Diamond$ closure modality.

Finally, we recall that in [15, 14] a *surrounded* operator $\mathcal{S}$ was introduced such that a point x satisfies $\Phi_1\,\mathcal{S}\,\Phi_2$ if and only if it belongs to a set of mutually

[4] There are a few exceptions to this view of past-tense operators, e.g. [26, 32].

connected points all satisfying Φ_1 and this set is directly surrounded by points all satisfying Φ_2. In other words, no path rooted in x can leave the Φ_1 area without passing by a point satisfying Φ_2. It is worth noting that the surrounded operator can be expressed using $\vec{\rho}$ as follows: $\Phi_1 \mathcal{S} \Phi_2 \equiv \Phi_1 \land \neg\vec{\rho}(\neg(\Phi_1 \lor \Phi_2))[\neg\Phi_2]$.

2.2 τSLCS: The Temporal Extension of SLCS

We recall here a temporal extension of SLCS, similar to the one presented in [13, 12], that provides a formal, unified framework for reasoning about both spatial and temporal features of systems and their behaviour. The version of SLCS defined in [13, 12] is based on the proximity and surrounded operators, both expressible in terms of reachability operators, as we have seen in Section 2.1.

For what concerns the satisfaction relation, now a *spatio-temporal* model $\mathcal{M}$ (more simply, just a *temporal* model) is composed of a Kripke structure and a family of closure models, one for each world of the Kripke structure.

Definition 7 (Temporal closure model - TCM). *Given a set of* atomic propositions AP*, a* temporal *closure model, TCM for short, is composed of a Kripke structure $\mathcal{K} = (S, T)$—with set of states (worlds) S and transition (accessibility) relation $T \subseteq S \times S$—and of a family $\{(X, \mathcal{C})_s, \mathcal{V}_s, \mathcal{A}_s\}_{s \in S}$ of closure models. We say that TCM $\mathcal{M} = ((S, T), \{(X, \mathcal{C})_s, \mathcal{V}_s, \mathcal{A}_s\}_{s \in S})$ is* conservative *if, for all $s, s' \in S$, it holds $X_s \subseteq X_{s'}$ whenever $(s, s') \in T$.* •

All definitions for CMs extend also to TCMs; so, for instance, a quasi-discrete temporal closure model, QdTCM for short, is a temporal closure model where all closure models (i.e., one for each state $s \in S$) are quasi-discrete. Here, we consider only conservative QdTCM[5]. This means that space can only *grow* when time advances: i.e., the points of space *can never be lost.* This condition simplifies the definition of spatio-temporal logics; we leave the investigation of non-conservative models to future work. We recall the definition of temporal paths.

Definition 8. *Given Kripke structure $\mathcal{K} = (S, T)$ and $s \in S$, a temporal path rooted in s is a function $\tau : \mathbb{N} \to S$ such that $\tau(0) = s$ and $(\tau(i), \tau(i+1)) \in T$ for all $i \in \mathbb{N}$. For $s \in S$, $\mathcal{T}_s$ denotes the set of all temporal paths τ rooted in s.* •

Note that the above definition implies that the relation T is total, which is readily obtained by adding self-loops (s, s) whenever there is no s' s.t. $(s, s') \in T$.

Definition 9 (Spatio-Temporal Logic for Closure Spaces – τSLCS). *For $p \in$ AP the syntax of the logic is the following*

$$\Phi ::= p \mid \neg\Phi \mid \Phi \land \Phi \mid \vec{\rho}\,\Phi[\Phi] \mid \overleftarrow{\rho}\,\Phi[\Phi] \mid \mathtt{A}\,\varphi \mid \mathtt{E}\,\varphi$$

$$\varphi ::= \mathcal{X}\Phi \mid \Phi\,\mathcal{U}\,\Phi$$

[5] In [13, 12], the stronger condition $X_s = X_{s'}$ for all $s, s' \in S$ was required.

Given QdTCM $\mathcal{M} = (\mathcal{K}, \{(X, \vec{\mathcal{C}})_s, \mathcal{V}_s, \mathcal{A}_s\}_{s \in S})$, *with* $\mathcal{K} = (S, T)$, satisfaction $\mathcal{M}, s, x \models \Phi$ *of a formula* Φ *in state* $s \in S$ *and at point* $x \in X_s$ *is defined by induction on the structure of formulas as given below*

$$\begin{array}{lll}
\mathcal{M}, s, x \models p & \Leftrightarrow & x \in \mathcal{V}_s(p) \\
\mathcal{M}, s, x \models \neg \Phi & \Leftrightarrow & \mathcal{M}, s, x \models \Phi \textit{ does not hold} \\
\mathcal{M}, s, x \models \Phi_1 \wedge \Phi_2 & \Leftrightarrow & \mathcal{M}, s, x \models \Phi_1 \textit{ and } \mathcal{M}, s, x \models \Phi_2 \\
\mathcal{M}, s, x \models \vec{\rho}\, \Phi_1[\Phi_2] & \Leftrightarrow & \textit{path } \pi \textit{ in } (X, \vec{\mathcal{C}})_s \textit{ and index } \ell \textit{ exist s.t.} \\
 & & \pi(0) = x \textit{ and } \mathcal{M}, s, \pi(\ell) \models \Phi_1 \\
 & & \textit{and for all indexes } j : 0 < j < \ell \textit{ implies } \mathcal{M}, s, \pi(j) \models \Phi_2 \\
\mathcal{M}, s, x \models \overleftarrow{\rho}\, \Phi_1[\Phi_2] & \Leftrightarrow & \textit{path } \pi \textit{ in } (X, \vec{\mathcal{C}})_s \textit{ and index } \ell \textit{ exist s.t.} \\
 & & \pi(\ell) = x \textit{ and } \mathcal{M}, s, \pi(0) \models \Phi_1 \\
 & & \textit{and for all indexes } j : 0 < j < \ell \textit{ implies } \mathcal{M}, s, \pi(j) \models \Phi_2 \\
\mathcal{M}, s, x \models \mathtt{A}\, \varphi & \Leftrightarrow & \textit{for all } \tau \in \mathcal{T}_s \textit{ it holds } \mathcal{M}, \tau, x \models \varphi \\
\mathcal{M}, s, x \models \mathtt{E}\, \varphi & \Leftrightarrow & \tau \textit{ exists s.t. } \tau \in \mathcal{T}_s \textit{ and } \mathcal{M}, \tau, x \models \varphi \\
 & & \\
\mathcal{M}, \tau, x \models \mathcal{X}\, \Phi & \Leftrightarrow & \mathcal{M}, \tau(1), x \models \Phi \\
\mathcal{M}, \tau, x \models \Phi_1 \mathcal{U}\, \Phi_2 & \Leftrightarrow & n \textit{ exists s.t. } n \in \mathbb{N} \textit{ and } \mathcal{M}, \tau(n), x \models \Phi_2 \\
 & & \textit{and for all } n' \in \mathbb{N} \textit{ s.t. } 0 \le n' < n \textit{ it holds} \\
 & & \mathcal{M}, \tau(n'), x \models \Phi_1
\end{array}$$

•

In the sequel we will often use the *eventually* operator F, that is derived from $\mathcal{U}$ in the standard way: $\mathtt{F}\,\Phi \equiv \mathtt{true}\,\mathcal{U}\,\Phi$. In addition, we define the following operator ag as derived from E and F as follows: $\mathtt{ag}\,\Phi \equiv \neg\,\mathtt{E}\,\mathtt{F}\neg\,\Phi$. Similarly, we define eg as follows: $\mathtt{eg}\,\Phi \equiv \neg\,\mathtt{A}\,\mathtt{F}\neg\,\Phi$.

3 ∃xSLCS: Point Existential Extension of SLCS

We extend the logic presented in Section 2.1 with a *point existential quantification* operator $\exists\gamma.\_$. To that purpose, we assume a denumerable set Γ of *point variables*, ranged over by $\gamma, \gamma', \gamma_1 \ldots$ Furthermore, we use the standard notion of free occurrence of a (point) variable in a formula, with respect to quantifiers. We also refine the syntax of the logic, by introducing a syntactic category E of *expressions* on the attributes of points, and point equality, as specified below.

For set A of attribute names, set AC of attribute constants—which is assumed to include boolean values, true and false, for which the usual boolean operators are assumed defined— and set Γ of point variables, with a distinguished item $\mathtt{this} \notin \Gamma$, the abstract syntax for expressions E follows

$$E ::= c \mid a \mid \gamma = \gamma \mid \gamma = \mathtt{this} \mid \gamma.a \mid \mathtt{this}.a \mid f(E, \ldots, E) \tag{1}$$

where $c \in AC$, $a \in A$, $\gamma \in \Gamma$ and f is the name of an n-ary function; f can denote any standard function of attribute values, e.g., equality, for which a standard semantics $\mathcal{I}_F$ is assumed given. Similarly, we assume an interpretation function $\mathcal{I}_C$ for attribute constants that maps every attribute constant c to a value $\mathcal{I}_C(c) \in AV$. The special item this will be used for referring to the point on which the formula at hand—which the expression where `this` occurs is a component of—is interpreted, as shown in Example 1 below.

Assertions are the subclass of expressions that evaluate to boolean values[6]. In order to compute assertions we define below the valuation function $\mathcal{E}$ of expressions E that extends function $\mathcal{A}$ in the expected way. Function $\mathcal{E}$ is defined using an auxiliary variable assignment (partial) function $\mu : (\Gamma \cup \{\texttt{this}\}) \to X$. We let μ_0 denote the assignment that is undefined for all elements of $\Gamma \cup \{\texttt{this}\}$. For all $x \in X$, $c \in AC$, $a \in A$, $\gamma, \gamma_1, \gamma_2 \in \Gamma$ and $\mu : (\Gamma \cup \{\texttt{this}\}) \to X$

$$\begin{array}{ll}
\mathcal{E}_\mu(x, c) & \triangleq \mathcal{I}_C(c) \\
\mathcal{E}_\mu(x, a) & \triangleq \mathcal{A}(x, a) \\
\mathcal{E}_\mu(x, \gamma_1 = \gamma_2) & \triangleq \mu(\gamma_1) = \mu(\gamma_2) \\
\mathcal{E}_\mu(x, \gamma = \texttt{this}) & \triangleq \mu(\gamma) = \mu(\texttt{this}) \\
\mathcal{E}_\mu(x, \gamma.a) & \triangleq \mathcal{A}(\mu(\gamma), a) \\
\mathcal{E}_\mu(x, \texttt{this}.a) & \triangleq \mathcal{A}(\mu(\texttt{this}), a) \\
\mathcal{E}_\mu(x, f(E_1, \ldots, E_n)) & \triangleq \mathcal{I}_F(f)(\mathcal{E}_\mu(x, E_1), \ldots, \mathcal{E}_\mu(x, E_n)).
\end{array}$$

Definition 10 ($\exists$xSLCS). *The syntax of* $\exists x$SLCS *is the same as in Definition 6 with the addition of the following productions*

$$\Phi ::= E \mid \exists\gamma.\Phi$$

The scope of γ in $\exists\gamma.\Phi$ is Φ. For every $\exists x$SLCS formula Φ, where no point variable occurs free, satisfaction $\mathcal{M}, x \models \Phi$ *of Φ at point $x \in X$ in QdCM $\mathcal{M} = (X, \vec{\mathcal{C}}, \mathcal{A}, \mathcal{V})$ is defined as follows*

$$\mathcal{M}, x \models \Phi \Leftrightarrow \mathcal{M}, \mu_0[\texttt{this} \mapsto x], x \Vdash \Phi$$

where relation $\Vdash$ is defined below, by induction on the structure of the formulas

$$\begin{array}{ll}
\mathcal{M}, \mu, x \Vdash p & \Leftrightarrow x \in \mathcal{V}(p) \\
\mathcal{M}, \mu, x \Vdash \neg\,\Phi & \Leftrightarrow \mathcal{M}, \mu, x \Vdash \Phi \textit{ does not hold} \\
\mathcal{M}, \mu, x \Vdash \Phi_1 \wedge \Phi_2 & \Leftrightarrow \mathcal{M}, \mu, x \Vdash \Phi_1 \textit{ and } \mathcal{M}, \mu, x \Vdash \Phi_2 \\
\mathcal{M}, \mu, x \Vdash \vec{\rho}\,\Phi_1[\Phi_2] & \Leftrightarrow \textit{path } \pi \textit{ and index } \ell \textit{ exist s.t. } \pi(0) = x \textit{ and } \mathcal{M}, \mu, \pi(\ell) \Vdash \Phi_1 \\
 & \quad \textit{and for all indexes } j : 0 < j < \ell \textit{ implies } \mathcal{M}, \mu, \pi(j) \Vdash \Phi_2 \\
\mathcal{M}, \mu, x \Vdash \overleftarrow{\rho}\,\Phi_1[\Phi_2] & \Leftrightarrow \textit{path } \pi \textit{ and index } \ell \textit{ exist s.t. } \pi(\ell) = x \textit{ and } \mathcal{M}, \mu, \pi(0) \Vdash \Phi_1 \\
 & \quad \textit{and for all indexes } j : 0 < j < \ell \textit{ implies } \mathcal{M}, \mu, \pi(j) \Vdash \Phi_2 \\
\mathcal{M}, \mu, x \Vdash E & \Leftrightarrow \mathcal{E}_\mu(x, E) \textit{ is true} \\
\mathcal{M}, \mu, x \Vdash \exists\gamma.\Phi & \Leftrightarrow \textit{there is } x' \in X \textit{ s.t. } \mathcal{M}, \mu[\gamma \mapsto x'], x \Vdash \Phi.
\end{array}$$

•

The universal quantifier operator is derived from the existential one, in the usual way: $\forall\gamma.\Phi \equiv \neg\exists\gamma.\neg\Phi$.

[6] For the sake of notational simplicity, we refrain from giving an explicit syntactic characterisation of assertions here.

Example 1. Suppose we are interested in the safety condition of a building where (adjacent) rooms, corridors and stairs—collectively called “premises”—are connected by means of doors that can be locked or unlocked. In particular, we consider a critical safety hazard the fact that there are premises with a concentration of a certain substance in the air higher than a given threshold `threshold` that are reachable from other premises, via unlocked doors, i.e. doors that can accidentally be opened by unauthorized people. We can model the building as a QdCM $\mathcal{M} = ((X, \vec{\mathcal{C}}), \mathcal{V}, \mathcal{A})$ where each element $x \in X$ has an attribute, `sort`, that can take as values those represented by the following constants: $\texttt{room}, \texttt{corridor}, \texttt{stair}, \texttt{door} \in AC$. The topological requirement that premises are connected one another via doors can be enforced by a requirement on $\vec{\mathcal{C}}$ and $\mathcal{A}$, namely, that for all $x, y \in X$

- if $\mathcal{A}(x, \texttt{sort}) \neq \mathcal{I}_C(\texttt{door})$ and $y \in \vec{\mathcal{C}}(\{x\}) \setminus \{x\}$, then $\mathcal{A}(y, \texttt{sort}) = \mathcal{I}_C(\texttt{door})$;
- if $\mathcal{A}(y, \texttt{sort}) = \mathcal{I}_C(\texttt{door})$ and $x \in \vec{\mathcal{C}}(\{y\}) \setminus \{y\}$, then $\mathcal{A}(x, \texttt{sort}) \neq \mathcal{I}_C(\texttt{door})$.

Moreover, every $x \in X$ has an attribute `concentration` that can take natural numbers as values and is relevant only for elements representing rooms, corridors and stairs. Finally, attribute `status` takes as values those represented by $\texttt{locked}, \texttt{unlocked} \in AC$ and is relevant only for doors. The hazardous situation can then be formalised by formula ϕ_1, defined below, that states that there is a location (i.e., room, corridor or stairs) from which premises can be reached, via locations and unlocked doors, in which the substance concentration is higher than the threshold

$$\phi_1 \equiv \exists \gamma. (\neg(\gamma.\texttt{sort} = \texttt{door}) \wedge \vec{\rho}\, \phi_2\, [\phi_3])$$

where

$$\phi_2 \equiv \neg(\texttt{sort} = \texttt{door}) \wedge \texttt{concentration} > \texttt{threshold}$$

$$\phi_3 \equiv (\texttt{sort} = \texttt{door}) \Rightarrow (\texttt{status} = \texttt{unlocked}).$$

Note the *global* nature of the formula: either all points in X satisfy it or none does. In contrast, if we replace ϕ_2 with ϕ_2' defined below, we get a formula that is satisfied by $x \in X$ only if the unsafe location has the same sort as x

$$\phi_2' \equiv \texttt{sort} = \texttt{this.sort} \wedge \texttt{concentration} > \texttt{threshold}$$

•

Example 2. We consider the example proposed in [19] that falls in the category of circle elimination games and is a sort of distributed leader survivor game. The choice of the example is motivated by the large use of leader election protocols in the area of distributed systems design and applications[7].

[7] In line with [19], we are not interested in the algorithm(s) used for deciding the winner of the game. We are only interested in providing a representation for the configurations of the game and investigating properties of any such configuration, as well as the whole game, that can be expressed using the extensions of `SLCS` we discuss in the present paper.

A finite set of entities is given; the entities are connected through communication channels in such a way that a ring topology is formed. The game evolves performing a series of elimination rounds. After each round the loser is eliminated from the game: his neighbours should be connected in such a way that the ring is closed and no longer includes the loser. The game ends when there is only one entity left: the leader, that is, the winner of the game.

A session of the game can be represented as a sequence of graphs, each graph modelling the situation at a given step of the game. For instance, the graph $\mathbf{G_0}$ of Figure 1a represents a configuration in which there are three agents—the nodes n_0, n_1, n_2—and three channels—the edges e_0, e_1, e_2, with source and target functions, s and t, defined as follows: $s^{\mathbf{G_0}} = \{e_0 \mapsto n_0, e_1 \mapsto n_2, e_2 \mapsto n_1\}$ and $t^{\mathbf{G_0}} = \{e_0 \mapsto n_1, e_1 \mapsto n_0, e_2 \mapsto n_2\}$. A final (correct) configuration is represented by a graph with a single node, say n_1—the (unique) winner—and a single edge, say e, the source and target of which coincide with this node—i.e. $s(e) = t(e) = n_1$, namely a self-loop in n_1 (Figure 1c).

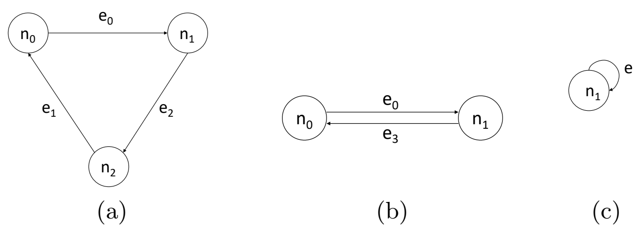


Fig. 1: (a) A configuration with three agents and three communication channels; (b) an intermediate configuration in which n_2 has been eliminated; (c) a configuration where the leader (n_1, in this case) has emerged.

We can represent a configuration in the game using a QdCM $(X, \vec{\mathcal{C}}, \mathcal{A}, \mathcal{V})$, the underlying binary relation of which is the empty relation (so that $\vec{\mathcal{C}}(A) = A$ for all $A \subseteq X$) and where each $x \in X$ can represent either an agent of the game or a communication channel. To that purpose, we use again a `sort` attribute, that takes as values those represented by `node`, `edge` $\in AC$. Finally, attributes `source` and `target`—relevant only for elements of sort `edge`—yield the (unique identifier of the) source and target node of an edge. Under the above assumptions, a leader can be specified as follows

$$\mathbf{leader}(\gamma) \equiv (\gamma.\texttt{sort} = \texttt{edge}) \wedge (\gamma.\texttt{source} = \gamma.\texttt{target}).$$

The game terminates correctly only if there is a *unique* leader. Unicity can be specified as follows

$$\mathbf{correctNLeader}(\gamma_1, \gamma_2) \equiv (\mathbf{leader}(\gamma_1) \wedge \mathbf{leader}(\gamma_2)) \Rightarrow (\gamma_1 = \gamma_2).$$

Thus a correct final configuration is one in which there is a unique leader, as specified by the formula $(\exists\gamma.\mathbf{leader}(\gamma)) \wedge (\forall\gamma_1.\forall\gamma_2.\mathbf{correctNLeader}(\gamma_1, \gamma_2))$.

The two requirements on the leader can obviously be expressed in a combined way by the following formula: $\exists\gamma_1.(\mathbf{leader}(\gamma_1) \wedge (\forall\gamma_2.\mathbf{leader}(\gamma_2) \implies \gamma_1 = \gamma_2))$.

We close the example noting that the same technique based on unique identifiers can be used for expressing properties about the size of a system (see page 191 of [19]). For instance a bound of two is expressed as follows

$$\textbf{at-most-two} \equiv \forall\gamma_1.\forall\gamma_2.\forall\gamma_3.\{\gamma_1 = \gamma_2 \vee \gamma_2 = \gamma_3 \vee \gamma_3 = \gamma_1\}.$$

4 $\exists x\tau$SLCS: Point Existential Temporal Extension of SLCS

We extend the spatio-temporal logic presented in Section 2.2 with the *point existential quantification* operator $\exists\gamma.\_$ introduced in Section 3. To that purpose, we need to extend the valuation function $\mathcal{E}$ with an additional parameter for the state where the evaluation has to be performed

$$\begin{array}{ll}
\mathcal{E}_\mu(x, c, s) & \triangleq \mathcal{I}_C(c) \\
\mathcal{E}_\mu(x, a, s) & \triangleq \mathcal{A}_s(x, a) \\
\mathcal{E}_\mu(x, \gamma_1 = \gamma_2, s) & \triangleq \mu(\gamma_1) = \mu(\gamma_2) \\
\mathcal{E}_\mu(x, \gamma = \texttt{this}, s) & \triangleq \mu(\gamma) = \mu(\texttt{this}) \\
\mathcal{E}_\mu(x, \gamma.a, s) & \triangleq \mathcal{A}_s(\mu(\gamma), a) \\
\mathcal{E}_\mu(x, \texttt{this}.a, s) & \triangleq \mathcal{A}_s(\mu(\texttt{this}), a) \\
\mathcal{E}_\mu(x, f(E_1, \ldots, E_n), s) & \triangleq \mathcal{I}_F(f)(\mathcal{E}_\mu(x, E_1, s), \ldots, \mathcal{E}_\mu(x, E_n, s)).
\end{array}$$

Definition 11 ($\exists x\tau$SLCS)**.** *The syntax of $\exists x\tau$SLCS is the same as in Definition 9 with the addition of the following productions*

$$\Phi ::= E \mid \exists\gamma.\Phi$$

The scope of γ in $\exists\gamma.\Phi$ is Φ. For every $\exists x\tau$SLCS formula Φ, where no point variable occurs free, satisfaction *$\mathcal{M}, s, x \models \Phi$ of Φ in state $s \in S$ and at point $x \in X_s$ in QdTCM $\mathcal{M} = (\mathcal{K}, \{(X, \vec{\mathcal{C}})_s, \mathcal{V}_s, \mathcal{A}_s\}_{s\in S})$, with $\mathcal{K} = (S, T)$, is defined as follows*

$$\mathcal{M}, s, x \models \Phi \Leftrightarrow \mathcal{M}, \mu_0[\texttt{this} \mapsto x], s, x \,\|\!\!= \Phi$$

where relation $\|\!\!\models$ is defined below, by induction on the structure of the formulas

$$\begin{array}{lcl}
\mathcal{M}, \mu, s, x \|\!\!\models p & \Leftrightarrow & x \in \mathcal{V}_s(p) \\
\mathcal{M}, \mu, s, x \|\!\!\models \neg \Phi & \Leftrightarrow & \mathcal{M}, \mu, s, x \|\!\!\models \Phi \textit{ does not hold} \\
\mathcal{M}, \mu, s, x \|\!\!\models \Phi_1 \wedge \Phi_2 & \Leftrightarrow & \mathcal{M}, \mu, s, x \|\!\!\models \Phi_1 \textit{ and } \mathcal{M}, \mu, s, x \|\!\!\models \Phi_2 \\
\mathcal{M}, \mu, s, x \|\!\!\models \vec{\rho}\, \Phi_1[\Phi_2] & \Leftrightarrow & \textit{path } \pi \textit{ in } (X, \vec{\mathcal{C}})_s \textit{ and index } \ell \textit{ exist s.t.} \\
 & & \pi(0) = x \textit{ and } \mathcal{M}, \mu, s, \pi(\ell) \|\!\!\models \Phi_1 \\
 & & \textit{and for all indexes } j : \\
 & & 0 < j < \ell \textit{ implies } \mathcal{M}, \mu, s, \pi(j) \|\!\!\models \Phi_2 \\
\mathcal{M}, \mu, s, x \|\!\!\models \overleftarrow{\rho}\, \Phi_1[\Phi_2] & \Leftrightarrow & \textit{path } \pi \textit{ in } (X, \vec{\mathcal{C}})_s \textit{ and index } \ell \textit{ exist s.t.} \\
 & & \pi(\ell) = x \textit{ and } \mathcal{M}, \mu, s, \pi(0) \|\!\!\models \Phi_1 \\
 & & \textit{and for all indexes } j : \\
 & & 0 < j < \ell \textit{ implies } \mathcal{M}, \mu, s, \pi(j) \|\!\!\models \Phi_2 \\
\mathcal{M}, \mu, s, x \|\!\!\models E & \Leftrightarrow & \mathcal{E}_\mu(x, E, s) \textit{ is true} \\
\mathcal{M}, \mu, s, x \|\!\!\models \exists \gamma . \Phi & \Leftrightarrow & \textit{there is } x' \in X_s \textit{ s.t. } \mathcal{M}, \mu[\gamma \mapsto x'], s, x \|\!\!\models \Phi. \\
\mathcal{M}, \mu, s, x \|\!\!\models \mathtt{A}\, \varphi & \Leftrightarrow & \textit{for all } \tau \in \mathcal{T}_s \textit{ it holds } \mathcal{M}, \mu, \tau, x \|\!\!\models \varphi \\
\mathcal{M}, \mu, s, x \|\!\!\models \mathtt{E}\, \varphi & \Leftrightarrow & \tau \textit{ exists s.t. } \tau \in \mathcal{T}_s \textit{ and } \mathcal{M}, \mu, \tau, x \|\!\!\models \varphi \\
 & & \\
\mathcal{M}, \mu, \tau, x \|\!\!\models \mathcal{X}\, \Phi & \Leftrightarrow & \mathcal{M}, \mu, \tau(1), x \|\!\!\models \Phi \\
\mathcal{M}, \mu, \tau, x \|\!\!\models \Phi_1 \,\mathcal{U}\, \Phi_2 & \Leftrightarrow & n \textit{ exists s.t. } n \in \mathbb{N} \textit{ and } \mathcal{M}, \mu, \tau(n), x \|\!\!\models \Phi_2 \\
 & & \textit{and for all } n' \in \mathbb{N} \textit{ s.t. } 0 \leq n' < n \textit{ it holds} \\
 & & \mathcal{M}, \mu, \tau(n'), x \|\!\!\models \Phi_1
\end{array}$$

•

Example 3. Let us consider again the building of Example 1. We want to formalise the fact that whenever the building is in a hazardous situation, as specified by formula ϕ_1, it will recover from it. This is the same to say that there is no evolution in the behaviour of the building where a situation is reached in which ϕ_1—expressing the hazardous situation—holds and in no future configuration $\neg\phi_1$ holds. This is formalised by ϕ_4 as follows

$$\phi_4 \equiv \neg \mathtt{EF}(\phi_1 \wedge \neg \mathtt{AF} \neg \phi_1)$$

or equivalently, using the `eg` derived operator

$$\phi_4 : \neg \mathtt{E}\,\mathtt{F}\,\mathtt{eg}\,\Phi_1$$

A stronger property is expressed by ϕ_5 below where we also require that eventually nowhere the concentration is above the threshold

$$\phi_5 \equiv \neg \mathtt{EF}(\phi_1 \wedge \neg \mathtt{AF}(\neg \phi_1 \wedge \mathtt{AF}\, \phi_6))$$

where $\phi_6 \equiv \forall \gamma . \mathbf{safe}(\gamma)$ and predicate **safe** is defined as follows

$$\mathbf{safe}(\gamma) \equiv (\neg(\gamma.\mathtt{sort} = \mathtt{door})) \Rightarrow \gamma.\mathtt{concentration} \leq \mathtt{threshold}.$$

•

Example 4. With reference to Example 2, now we consider a QdTCM $\mathcal{M} = ((S, T), \{(X, \vec{\mathcal{C}})_s, \mathcal{V}_s, \mathcal{A}_s\}_{s \in S})$. In [19] the following properties are introduced

ψ_1: *for all departing paths, eventually there will be a leader* (Example 4.2 on page 185, property ψ_1);
ψ_2: *there is an entity that, for all departing paths, will eventually become the leader* (Example 4.2 on page 185, property ψ_2);
ψ_3: *for any evolution of the game, eventually there will be a state containing an entity that will become the leader* (Example 4.2 on page 185, property ψ_3);
ψ_4: *any evolution, starting from any state, will lead to a state with (at least) a leader* (property **p1** on pages 181 and 193).

In $\exists x\tau$SLCS property ψ_1 can be expressed as $\psi_1 \equiv \mathtt{AF}\,\exists\gamma.\mathbf{leader}(\gamma)$. Property ψ_2 is expressed by formula $\psi_2 \equiv \exists\gamma.\mathtt{AF}\,\mathbf{leader}(\gamma)$, while property ψ_3 is expressed by $\psi_3 \equiv \mathtt{AF}\,\psi_2$. Property ψ_4 is expressed as

$$\psi_4 \equiv \mathtt{ag}\,\mathtt{A}\,\mathtt{F}(\exists\gamma.\mathbf{leader}(\gamma)).$$

In Example 2 we have shown a formula expressing existence and uniqueness of the leader. The following formula enforces the property at the global level

$$\mathtt{AF}(\exists\gamma_1.(\mathbf{leader}(\gamma_1) \wedge (\forall\gamma_2.\mathbf{leader}(\gamma_2) \implies \gamma_1 = \gamma_2))).$$

•

5 $\exists\tau\nu$xSLCS: Dealing with Fresh Point Names

In situations where the space can change (e.g., grow) during the computation, it can be useful to quantify *only* over those paths where such a change takes place. For instance, with reference to the leader election example (Example 2), one might want to express that existence and unicity of the leader are preserved whenever new entities enter the game (see Example 5 below). Similarly, it can be useful to quantify *only* over those paths where no such changes take place. In addition, in the first case, one might be interested requiring that *all* newly introduced points satisfy a certain property or that such a property is indeed satisfied by *some* of such new points. Therefore, we extend the spatio-temporal logics presented in Section 2.2 with modalities $\mathtt{A}_\nu$ and $\mathtt{E}_\nu$, quantifying over paths where new points are introduced; in contrast, modalities $\mathtt{A}_{\bar\nu}$ and $\mathtt{E}_{\bar\nu}$ quantify over paths where *no* new points are introduced.

In addition, we introduce the *point quantified* next unary operators $\mathcal{X}_{\forall\gamma\text{-}}$ and $\mathcal{X}_{\exists\gamma\text{-}}$; the former universally quantifies over all *new* points whereas the latter quantifies existentially over *new* points.

Definition 12 ($\exists\tau\nu$xSLCS)**.** *The syntax of* $\exists\tau\nu$xSLCS *is the same as in Definition 11 with the addition of the following productions*

$$\Phi ::= \mathtt{A}_\nu\,\varphi \mid \mathtt{A}_{\bar\nu}\,\varphi \mid \mathtt{E}_\nu\,\varphi \mid \mathtt{E}_{\bar\nu}\,\varphi$$

$$\varphi ::= \mathcal{X}_{\forall\gamma}\,\Phi \mid \mathcal{X}_{\exists\gamma}\,\Phi$$

The scope of γ in $\mathcal{X}_{\forall\gamma}\Phi$ and in $\mathcal{X}_{\exists\gamma}\Phi$ is Φ. For every $\exists\tau\nu$x$\mathtt{SLCS}$ formula Φ, where no point variable occurs free, satisfaction $\mathcal{M}, s, x \models \Phi$ *of Φ in state $s \in S$ and at point $x \in X_s$ in QdTCM $\mathcal{M} = (\mathcal{K}, \{(X, \vec{\mathcal{C}})_s, \mathcal{V}_s, \mathcal{A}_s\}_{s\in S})$, with $\mathcal{K} = (S, T)$, is defined as follows*

$$\mathcal{M}, s, x \models \Phi \Leftrightarrow \mathcal{M}, \mu_0[\mathtt{this} \mapsto x], s, x \mid\!\models \Phi$$

where relation $\mid\!\models$ is defined by induction on the structure of the formulas as in Definition 11, with the addition of the following equalities

$$\begin{array}{lll}
\mathcal{M}, \mu, s, x \mid\!\models \mathtt{A}_\nu\, \varphi & \Leftrightarrow \textit{for all } \tau \in \mathcal{T}_s \textit{ s.t. } X_{\tau(0)} \subset X_{\tau(1)} \textit{ it holds } \mathcal{M}, \mu, \tau, x \mid\!\models \varphi \\
\mathcal{M}, \mu, s, x \mid\!\models \mathtt{A}_{\bar\nu}\, \varphi & \Leftrightarrow \textit{for all } \tau \in \mathcal{T}_s \textit{ s.t. } X_{\tau(0)} = X_{\tau(1)} \textit{ it holds } \mathcal{M}, \mu, \tau, x \mid\!\models \varphi \\
\mathcal{M}, \mu, s, x \mid\!\models \mathtt{E}_\nu\, \varphi & \Leftrightarrow \textit{there is } \tau \in \mathcal{T}_s \textit{ s.t. } X_{\tau(0)} \subset X_{\tau(1)} \textit{ and } \mathcal{M}, \mu, \tau, x \mid\!\models \varphi \\
\mathcal{M}, \mu, s, x \mid\!\models \mathtt{E}_{\bar\nu}\, \varphi & \Leftrightarrow \textit{there is } \tau \in \mathcal{T}_s \textit{ s.t. } X_{\tau(0)} = X_{\tau(1)} \textit{ and } \mathcal{M}, \mu, \tau, x \mid\!\models \varphi
\end{array}$$

$$\begin{array}{ll}
\mathcal{M}, \mu, \tau, x \mid\!\models \mathcal{X}_{\forall\gamma}\, \Phi \Leftrightarrow \textit{for all } x' \in X_{\tau(1)} \setminus X_{\tau(0)} \textit{ it holds } \mathcal{M}, \mu[\gamma \mapsto x'], \tau(1), x' \mid\!\models \Phi \\
\mathcal{M}, \mu, \tau, x \mid\!\models \mathcal{X}_{\exists\gamma}\, \Phi \Leftrightarrow \textit{there is } x' \in X_{\tau(1)} \setminus X_{\tau(0)} \textit{ s.t. } \mathcal{M}, \mu[\gamma \mapsto x'], \tau(1), x' \mid\!\models \Phi
\end{array}$$

•

With reference to the above definition, recall that for temporal path τ, $X_{\tau(0)}$ is the current space and $X_{\tau(1)}$ is the space at the next step of the computation.

Example 5. With reference to Example 2, the fact that the existence and unicity of the leader is preserved in the next time instant if a new element is added to the game can be expressed now by the following formula

$$\xi \equiv \mathtt{A}_\nu \mathcal{X}((\exists\gamma.\mathbf{leader}(\gamma)) \wedge (\forall\gamma_1.\forall\gamma_2.\mathbf{correctNLeader}(\gamma_1, \gamma_2))).$$

The requirement can be turned into an invariant in the expected way: $\mathtt{ag}\,\xi$. The following formula is true only if the number of entities keeps growing forever during the evolution of the game: $\neg\mathtt{E}\,\mathtt{F}\,\mathtt{E}_{\bar\nu}\,\mathtt{F}\,\mathtt{true}$. •

Example 6. Again with reference to Example 1, suppose that, due to new safety & security rules, additional premises have to be added eventually. Obviously, authorities will require that these new rooms, corridors or stairs have a concentration of the dangerous material that is less than the given threshold, as formalised by the following formula: $\mathtt{AF}((\mathtt{E}_\nu\mathcal{X}_{\exists\gamma}\mathbf{safe}(\gamma)) \wedge (\mathtt{A}_\nu\mathcal{X}_{\forall\gamma}\mathbf{safe}(\gamma)))$, with $\mathbf{safe}(\gamma)$ defined as in Example 3. •

6 ∃pSLCS: Predicate Existential Extension of SLCS

In this section, we extend the logic presented in Section 2.1 with an *atomic proposition existential quantification* operator $\exists v.\_$. The operator we introduce quantifies over atomic propositions in a rather syntactic way, since two different atomic proposition symbols are not considered equal (not even if they denote the same subset of points of the space). In this variant of SLCS, atomic propositions play the role of *labels* or *identities* that can be associated to sets of points.

Although quantification operators have a wide applicability, we are particularly interested in applications in the field of medical imaging, and especially embedding known *lesion tracking* methods in the logical framework of `VoxLogicA`. Lesion tracking is the labelling of different lesions of a patient along the temporal axis (e.g., in *longitudinal studies*). In that context, it is required that the same label is assigned to the same lesion in different temporal snapshots. Several lesion tracking algorithms exist, aimed at different kinds of lesions. Since the number of lesions found in a patient is unknown *a priori*, in order to model the situation in our logical language, we intend to use atomic propositions to denote such lesions by internal labels, so that one can express formulas such as "there is a lesion x that outgrows all the other ones after one month", and then use x in other formulas, to better qualify the lesion x. We note in passing that the identification of the motion of discrete regions alongside the temporal axis is also the main topic of [20], where, notably, closure spaces are used as models.

Further examples include the analysis of video streams [11], in which different entities may be labelled by non-logical primitives, ranging from simple connected components labelling operations to machine-learning based methods. Similarly to the lesion tracking example, it is not known at formula design time how many entities exist in the analysed stream, and the identity of each entity is meant to be denoted by (internal) atomic propositions.

To that purpose, we assume a denumerable set *Var* of *proposition variables*, ranged over by $v, v', v_1 \ldots$ For formula Φ, $p \in \mathtt{AP}$ and variable v we let $\Phi[p/v]$ denote, as usual, the formula obtained by substituting all *free occurrences* of v in Φ with p, where the notion of free occurrence is the standard one, with respect to quantifiers. The substitution is performed syntactically: a variable occurrence is thus replaced by a *proposition name*. This kind of syntactic quantification is inspired by the work of Demri et al. on LTL with the freeze quantifier [18].

We refine the syntax of the logic by introducing a syntactic category of *proposition expressions* P and an equality operator on proposition letters.

Definition 13 ($\exists$p`SLCS`). *For $p \in \mathtt{AP}$ and $v \in \mathrm{Var}$ the syntax of $\exists$p`SLCS` is the same as in Definition 6 with the addition of the following productions*

$$\Phi ::= v \mid \exists v.\Phi \mid P = P$$

$$P ::= p \mid v.$$

The scope of v in $\exists v.\Phi$ is Φ. For every $\exists$p`SLCS` formula Φ, where no proposition variable occurs free, satisfaction *$\mathcal{M}, x \models \Phi$ of Φ at point $x \in X$ in QdCM $\mathcal{M} = (X, \vec{\mathcal{C}}, \mathcal{A}, \mathcal{V})$ is defined by induction on the structure of formulas as in Definition 6, with the addition of the following equalities*

$$\begin{array}{ll} \mathcal{M}, x \models \exists v.\Phi & \Leftrightarrow \textit{there is atomic proposition } p \in \mathtt{AP} \textit{ s.t. } \mathcal{M}, x \models \Phi[p/v] \\ \mathcal{M}, x \models P_1 = P_2 & \Leftrightarrow P_1 \textit{ and } P_2 \textit{ are the same atomic proposition.} \end{array}$$

●

Example 7. Consider model $\mathcal{M} = (X, \vec{\mathcal{C}}, \mathcal{A}, \mathcal{V})$, where $X = \{x_1, x_2, x_3, x_4, x_5\}$, $\mathcal{V}(p) = \{x_1, x_2, x_3\}$ and $\mathcal{V}(q) = \{x_3, x_4, x_5\}$, for $p \neq q$. We can use the following formula for overlap detection

$$\eta_1 \equiv \exists v_1 . \exists v_2 . v_1 \wedge v_2 \wedge \neg(v_1 = v_2).$$

Clearly, the formula η_1 holds only in point x_3, as $\mathcal{V}(q) \cap \mathcal{V}(p) = \{x_3\}$, assigning p and q to v_1 and v_2, respectively. Thus we have that $\mathcal{M}, x_3 \models \eta_1$ whereas $\mathcal{M}, x_j \not\models \eta_1$ for $j \in \{1, 2, 4, 5\}$.

Example 8. In this example we show how $\exists$**p**`SLCS` formulas can be used to distinguish regions in a digital image based on their identity (which is not possible in "classic" `SLCS`). Let us consider the image of Figure 2a and let us assume that all the points (pixels) belonging to a *connected region* of the same colour have been labelled with the same label—this can easily be achieved by running a standard algorithm for computing the connected components on the image. In terms of logics, we can assume that such labels are elements of `AP`; without loss of generality, let us call them $1, 2, 3, 4, 5, 6, 7$, identifying each of the yellow, red and blue areas in the image. In addition, we assume that all yellow pixels satisfy atomic proposition `yellow`, and similarly for red and blue pixels and atomic propositions `red` and `blue`, respectively. So, we assume $\{1, 2, 3, 4, 5, 6, 7, \texttt{yellow}, \texttt{red}, \texttt{blue}\} \subseteq \texttt{AP}$; we also assume that atomic propositions $1, 2, 3, 4, 5, 6$ and 7 are unknown to users—for instance they are the result of the algorithm for computing connected components mentioned above—so that they cannot use them explicitly in the formulas. Under these assumptions, formula η_2 below can be used to distinguish the top-left yellow region from the bottom-left one

$$\eta_2 \equiv \texttt{yellow} \wedge \vec{\rho}\, \eta_3[\texttt{yellow}]$$

where

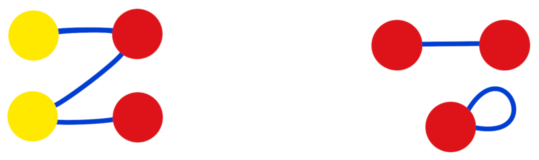
$$\eta_3 \equiv \texttt{yellow} \wedge \forall v_1 . \forall v_2 . ((\vec{\rho}\, (v_1 \wedge \neg\texttt{blue})[\texttt{blue}] \wedge \vec{\rho}\, (v_2 \wedge \neg\texttt{blue})[\texttt{blue}]) \implies v_1 = v_2)$$

In fact, any point of the top-left yellow region satisfies η_2, having v_1 and v_2 assigned to the label 3, which identifies the top red area, whereas no point of the bottom-left yellow one satisfies it. Note also that there is no need to explicitly mention `red` in the formula, as it is the only possible area satisfying $\neg$`blue` which is reachable via a blue path. Finally, we can find the unique label identifying the yellow top circle by mean of the formula

$$\exists v . v \wedge \eta_2$$

Example 9. With reference to Figure 2b, let us assume a similar labelling schema as for Example 8, with $\{1, 2, 3, 4, 5, 6, \texttt{red}, \texttt{blue}\} \subseteq \texttt{AP}$. It is easy to see that formula η_4 below distinguishes the points of any of the two red regions in the top from those of the red one in the bottom of the image

$$\eta_4 \equiv \texttt{red} \wedge \vec{\rho}\, \eta_5[\texttt{red}]$$



(a) (b)

Fig. 2: Two example images: interesting areas can be identified by means of atomic propositions and the existential quantifier over propositions. Note that areas are not treated as nodes and edges, rather as regions of adjacent pixels having colors red, yellow and blue.

where

$$\eta_5 \equiv \forall v_1.\forall v_2.v_1 \wedge \vec{\rho}\, v_2[\texttt{blue}] \wedge \neg(v_1 = \texttt{blue} \vee v_2 = \texttt{blue} \vee v_1 = v_2)$$

Any point of any of the two red regions in the top satisfies η_4, whereas no point of the red region in the bottom satisfies it.

7 Conclusions and Future Work

In this work we presented a preliminary investigation on binding in the setting of spatial and spatio-temporal logic. Our aim was to provide an intuition of how we can deal with identity of (groups of) points in a space by means of quantifiers. We provided illustrative examples of how these can be used to express interesting properties in different settings, i.e., when dealing with images and graphs, and introduced a point quantified spatio-temporal logic for closure spaces. The extension of τ`SLCS` with predicate binding is as expected, and is not introduced here for reasons of space. Furthermore, while this preliminary investigation focuses on conservative spatio-temporal models, in future work we plan to consider also non-conservative ones, thus widening the set of possible application domains. We note in passing that it is very likely that some of the presented logical constructs can be expressed in more general formalisms such as first or second order logic, or the modal μ-calculus. Although this may be the subject of future investigation, we remark that our main aim is to investigate model checking algorithms for the newly introduced extensions. Therefore, the study of fragments that admit an efficient model checking algorithm is more immediate to our research line. In particular, a major objective is to extend the tool `VoxLogicA` [5] with a suitable implementation of the binding operator, in order to apply spatial model checking to a larger set of case studies. In particular, we plan to exploit the computational power of GPUs in order to achieve high efficiency via parallelisation, in a similar way as done for standard operators in its variant `VoxLogicA-GPU` [10].

References

1. Baldan, P., Corradini, A., König, B., Lluch Lafuente, A.: A temporal graph logic for verification of graph transformation systems. In: Fiadeiro, J.L., Schobbens, P.Y. (eds.) WADT 2006. LNCS, vol. 4409, pp. 1–20. Springer (2007)
2. Bednarczyk, B., Demri, S.: Why propositional quantification makes modal logics on trees robustly hard? In: LICS 2019. pp. 1–13. IEEE (2019)
3. Belmonte, G., Broccia, G., Ciancia, V., Latella, D., Massink, M.: Feasibility of spatial model checking for nevus segmentation. In: Bliudze, S., Gnesi, S., Plat, N., Semini, L. (eds.) FormaliSE@ICSE 2021. pp. 1–12. IEEE (2021)
4. Belmonte, G., Ciancia, V., Latella, D., Massink, M.: Innovating medical image analysis via spatial logics. In: ter Beek, M.H., Fantechi, A., Semini, L. (eds.) From Software Engineering to Formal Methods and Tools, and Back. LNCS, vol. 11865, pp. 85–109. Springer (2019)
5. Belmonte, G., Ciancia, V., Latella, D., Massink, M.: Voxlogica: A spatial model checker for declarative image analysis. In: Vojnar, T., Zhang, L. (eds.) TACAS 2019. LNCS, vol. 11427, pp. 281–298. Springer (2019)
6. Benthem, J.v., Bezhanishvili, G.: Modal logics of space. In: Aiello, M., Pratt-Hartmann, I., Benthem, J.v. (eds.) Handbook of Spatial Logics, pp. 217–298. Springer (2007)
7. Bezhanishvili, N., Ciancia, V., Gabelaia, D., Grilletti, G., Latella, D., Massink, M.: Geometric model checking of continuous space. CoRR **abs/2105.06194** (2021), https://arxiv.org/abs/2105.06194
8. Bull, R.A.: On modal logic with propositional quantifiers. The Journal of Symbolic Logic **34**(2), 257–263 (1969)
9. Buonamici, F.B., Belmonte, G., Ciancia, V., Latella, D., Massink, M.: Spatial logics and model checking for medical imaging. International Journal on Software Tools and Technology Transfer **22**(2), 195–217 (2020)
10. Bussi, L., Ciancia, V., Gadducci, F.: Towards a spatial model checker on GPU. In: Peters, K., Willemse, T.A.C. (eds.) FORTE 2021. LNCS, vol. 12719, pp. 188–196. Springer (2021)
11. Bussi, L., Ciancia, V., Gadducci, F., Latella, D., Massink, M.: Towards model checking video streams using VoxLogicA on GPU's. In: DataMod 2021. LNCS, Springer (2022), to appear
12. Ciancia, V., Gilmore, S., Grilletti, G., Latella, D., Loreti, M., Massink, M.: Spatio-temporal model checking of vehicular movement in public transport systems. International Journal on Software Tools and Technology Transfer **20**(3), 289–311 (2018)
13. Ciancia, V., Grilletti, G., Latella, D., Loreti, M., Massink, M.: An experimental spatio-temporal model checker. In: Bianculli, D., Calinescu, R., Rumpe, B. (eds.) SEFM Workshops 2015. LNCS, vol. 9509, pp. 297–311. Springer (2015)
14. Ciancia, V., Latella, D., Loreti, M., Massink, M.: Specifying and verifying properties of space. In: Díaz, J., Lanese, I., Sangiorgi, D. (eds.) TCS 2014. LNCS, vol. 8705, pp. 222–235. Springer (2014)
15. Ciancia, V., Latella, D., Loreti, M., Massink, M.: Model checking spatial logics for closure spaces. Journal of Logical Methods in Computer Science **12**(4) (2016)
16. Ciancia, V., Latella, D., Massink, M., Paskauskas, R., Vandin, A.: A tool-chain for statistical spatio-temporal model checking of bike sharing systems. In: Margaria, T., Steffen, B. (eds.) ISoLA 2016, Part I. LNCS, vol. 9952, pp. 657–673. Springer (2016)

17. Ciancia, V., Latella, D., Massink, M., de Vink, E.P.: On bisimilarities for closure spaces - preliminary version. CoRR **abs/2105.06690** (2021), https://arxiv.org/abs/2105.06690
18. Demri, S., Lazić, R.: LTL with the freeze quantifier and register automata. ACM Transactions in Computional Logic **10**(3) (2009)
19. Gadducci, F., Lluch-Lafuente, A., Vandin, A.: Counterpart semantics for a second-order μ-calculus. Fundamenta Informaticae **118**(1-2), 177–205 (2012)
20. Galton, A.: A generalized topological view of motion in discrete space. Theoretical Computer Science **305**(1-3), 111–134 (2003)
21. Grosu, R., Smolka, S.A., Corradini, F., Wasilewska, A., Entcheva, E., Bartocci, E.: Learning and detecting emergent behavior in networks of cardiac myocytes. Communications of the ACM **52**(3), 97–105 (2009)
22. Haghighi, I., Jones, A., Kong, Z., Bartocci, E., Grosu, R., Belta, C.: Spatel: a novel spatial-temporal logic and its applications to networked systems. In: Girard, A., Sankaranarayanan, S. (eds.) HSCC 2015. pp. 189–198. ACM (2015)
23. Holliday, W.H.: A note on algebraic semantics for S5 with propositional quantifiers. Notre Dame Journal of Formal Logic **60**(2), 321–332 (2017)
24. Kripke, S.A.: A completeness theorem in modal logic. Journal of Symbolic Logic **24**(1), 1–14 (1959)
25. Kröger, F., Merz, S.: First-order linear temporal logic. In: Temporal Logic and State Systems, pp. 153–179. Springer (2008)
26. Kurtonina, N., de Rijke, M.: Bisimulations for temporal logic. Journal of Logic, Language and Information **6**(4), 403–425 (1997)
27. Loreti, M., Quadrini, M.: A spatial logic for a simplicial complex model. CoRR **abs/2105.08708** (2021), https://arxiv.org/abs/2105.08708
28. Nenzi, L., Bortolussi, L., Ciancia, V., Loreti, M., Massink, M.: Qualitative and quantitative monitoring of spatio-temporal properties with SSTL. Journal of Logical Methods in Computer Science **14**(4) (2018)
29. Patthak, A., Bhattacharya, I., Dasgupta, A., Dasgupta, P., Chakrabarti, P.: Quantified computation tree logic. Information Processing Letters **82**(3), 123–129 (2002)
30. Rensink, A.: Model checking quantified computation tree logic. In: Baier, C., Hermanns, H. (eds.) CONCUR 2006. pp. 110–125. Springer (2006)
31. Smyth, M.B., Webster, J.: Discrete spatial models. In: Aiello, M., Pratt-Hartmann, I., van Benthem, J. (eds.) Handbook of Spatial Logics, pp. 713–798. Springer (2007)
32. Stirling, C.: Modal and temporal logics. In: Abramsky, S., Gabbay, D., Maibaum, T. (eds.) Handbook of Logic in Computer Science, pp. 477–563. Oxford University Press (1993)
33. Čech, E.: Topological Spaces. In: Pták, V. (ed.) Topological Spaces, chap. III, pp. 233–394. Publishing House of the Czechoslovak Academy of Sciences/Interscience Publishers, John Wiley & Sons, Prague/London-New York-Sydney (1966), Revised edition by Zdeněk Frolíc and Miroslav Katětov. Scientific editor, Vlastimil Pták. Editor of the English translation, Charles O. Junge. MR0211373