QUANTICOL

A Quantitative Approach to Management and Design of Collective and Adaptive Behaviours

quantiCOL

http://www.quanticol.eu

TR-QC-06-2014

Specifying and Verifying Properties of Space

Extendend version

Revision: 1.0; May 16 2014

Author(s): Vincenzo Ciancia (CNR), Diego Latella (CNR), Michele Loreti (UNIFI), Mieke Massink (CNR)

Publication date: May 16 2014

Funding Scheme: Small or medium scale focused research project (STREP)
Topic: ICT-2011 9.10: FET-Proactive 'Fundamentals of Collective Adaptive Systems' (FOCAS)
Project number: 600708

Coordinator: Jane Hillston (UEDIN)
e-mail: Jane.Hillston@ed.ac.uk
Fax: +44 131 651 1426

Part. no.	Participant organisation name	Acronym	Country
1 (Coord.)	University of Edinburgh	UEDIN	UK
2	Consiglio Nazionale delle Ricerche – Istituto di Scienza e Tecnologie della Informazione "A. Faedo"	CNR	Italy
3	Ludwig-Maximilians-Universität München	LMU	Germany
4	Ecole Polytechnique Fédérale de Lausanne	EPFL	Switzerland
5	IMT Lucca	IMT	Italy
6	University of Southampton	SOTON	UK
7	Institut National de Recherche en Informatique et en Automatique	INRIA	France

COOPERATION

Contents

Abstract

The interplay between process behaviour and spatial aspects of computation has become more and more relevant in Computer Science, especially in the field of *collective adaptive systems*, but also, more generally, when dealing with systems distributed in physical space. Traditional verification techniques are well suited to analyse the temporal evolution of programs; however, properties of space are typically not explicitly taken into account. We propose a methodology to verify properties depending upon physical space. We define an appropriate logic, stemming from the tradition of topological interpretations of modal logics, dating back to earlier logicians such as Tarski, where modalities describe neighbourhood. We lift the topological definitions to the more general setting of *closure spaces*, also encompassing discrete, graph-based structures. We further extend the framework with a spatial *until* operator, and define an efficient model checking procedure, implemented in a proof-of-concept tool.

1 Introduction

Much attention has been devoted in Computer Science to formal verification of process behaviour. Several techniques, such as *run-time monitoring* and *model checking*, are based on a formal understanding of system requirements through *modal* logics. Such logics typically have a *temporal* flavour, describing the flow of events along time, and are interpreted in various kinds of transition structures.

In recent times, aspects of computation related to the distribution of systems in physical space have become more relevant. An example is provided by so called *collective adaptive systems*[1]. Such systems are typically composed of a large number of interacting objects. Their global behaviour critically depends on interactions which are often local in nature. The aspect of locality immediately poses issues of spatial distribution of objects. Abstraction from spatial distribution may sometimes provide insights in the system behaviour, but this is not always the case. For example, consider a bike (or car) sharing system having several parking stations, and featuring twice as many parking slots as there are vehicles in the system. Ignoring the spatial dimension, on average, the probability to find completely full or empty parking stations at an arbitrary station is very low; however, this kind of analysis may be misleading, as in practice some stations are much more popular than others, often depending on nearby points of interest. This leads to quite different probabilities to find stations completely full or empty, depending on the examined location. In such situations, it is important to be able to predicate over spatial aspects, and eventually find methods to certify that a given collective

[1] See e.g. the web site of the QUANTICOL project: `http://www.quanticol.eu`

adaptive system satisfies specific requirements in this respect. In Logics, there is quite an amount of literature focused on so called *spatial* logics, that is, a spatial interpretation of modal logics. Dating back to early logicians such as Tarski, modalities may be interpreted using the concept of *neighbourhood* in a topological space. The field of spatial logics is well developed in terms of descriptive languages and computability/complexity aspects. However, the frontier of current research does not yet address verification problems, and in particular, discrete models are still a relatively unexplored field.

In this paper, we extend the topological semantics of modal logics to *closure spaces*. A closure space (also called *Čech closure space* or *preclosure space* in the literature), is a generalisation of a standard topological space, where idempotence of closure is not required. By this, graphs and topological spaces are treated uniformly, letting the topological and graph-theoretical notions of neighbourhood coincide. We also provide a spatial interpretation of the *until* operator, which is fundamental in the classical temporal setting, arriving at the definition of a logic which is able to describe unbounded areas of space. Intuitively, the spatial until operator describes a situation in which it is not possible to "escape" an area of points satisfying a certain property, unless by passing through at least one point that satisfies another given formula. To formalise this intuition, we provide a characterising theorem that relates infinite paths in a closure space and until formulas. We introduce an efficient model-checking procedure. A prototype implementation of a spatial model checker has been made available; the tool is able to interpret spatial logics on digital images, providing graphical understanding of the meaning of formulas, and an immediate form of counterexample visualisation.

Related work. We use the terminology *spatial logics* in the "topological" sense; the reader should be warned that in Computer Science literature, spatial logics typically describe situations in which modal operators are interpreted syntactically, against the structure of agents in a process calculus (see [9, 7] for some classical examples). The object of discussion in this research line are operators that quantify e.g., over the parallel sub-components of a system, or the hidden resources of an agent. Furthermore, logics for graphs have been studied in the context of databases and process calculi (see [8, 14], and the references therein), even though the relationship with physical space is often not made explicit, if considered at all. The influence of space on agents interaction is also considered in the literature on process calculi using *named locations* [11]. Variants of spatial logics have also been proposed for the symbolic representation of the contents of images, and, combined with temporal logics, for sequences of images [4]. The approach is based on a discretisation of the space of the images in rectangular regions and the orthogonal projection of objects and regions onto Cartesian coordinate axes such that their possible intersections can be analysed from different perspectives. It involves two spatial until operators defined on such projections considering spatial shifts of regions along the positive, respectively negative, direction of the coordinate axes and it is very different from the topological spatial logic approach. There also exist logical frameworks, encompassing some spatial information, that have been applied to the analysis of medical images [17].

A successful attempt to bring topology and digital imaging together is represented by the field of *digital topology* [22, 18]. In spite of its name, this area studies digital images using models inspired by topological spaces, but neither generalising nor specialising these structures. Rather recently, closure spaces have been proposed as an alternative foundation of digital imaging by various authors, especially Smyth and Webster [23] and Galton [16]; we ideally continue that research line, enhancing it with a logical perspective. The idea of interpreting the until operator in a topological space is briefly discussed in the work by Aiello and van Benthem [1, 24]. We start from their definition, discuss its limitations, and provide a more fine-grained operator, which is interpreted in closure spaces, and has therefore also an interpretation in topological spaces. In the specific setting of complex and collective adaptive systems, techniques for efficient approximation have been developed in the form of mean-field / fluid-flow analysis (see [6] for a tutorial introduction). Recently (see e.g., [10]), the importance of spatial aspects has been recognised and studied in this context. In this work, we aim at paving the way for the inclusion of spatial logics, and their verification procedures, in the framework of mean-field

and fluid-flow analysis of collective adaptive systems.

Proofs. Formal proofs of lemmas, theorems, and other statements can be found in Appendix A.

2 Closure spaces

In this work, we use *closure spaces* to define basic concepts of *space*. Below, we recall several definitions, most of which are explained in [16].

Definition 2.1. A *closure space* is a pair $(X, \mathcal{C})$ where X is a set, and the *closure operator* $\mathcal{C} : 2^X \to 2^X$ assigns to each subset of X its *closure*, obeying to the following laws, for all $A, B \subseteq X$:

1. $\mathcal{C}(\emptyset) = \emptyset$;
2. $A \subseteq \mathcal{C}(A)$;
3. $\mathcal{C}(A \cup B) = \mathcal{C}(A) \cup \mathcal{C}(B)$.

As a matter of notation, in the following, for $(X, \mathcal{C})$ a closure space, and $A \subseteq X$, we let $\overline{A} = X \setminus A$ be the complement of A in X.

Definition 2.2. Let $(X, \mathcal{C})$ be a closure space, for each $A \subseteq X$:

1. the *interior* $\mathcal{I}(A)$ of A is the set $\overline{\mathcal{C}(\overline{A})}$;
2. A is a *neighbourhood* of $x \in X$ if and only if $x \in \mathcal{I}(A)$;
3. A is *closed* if $A = \mathcal{C}(A)$ while it is *open* if $A = \mathcal{I}(A)$.

Lemma 2.3. *Let $(X, \mathcal{C})$ be a closure space, the following properties hold:*

1. *$A \subseteq X$ is open if and only if $\overline{A}$ is closed;*
2. *closure and interior are monotone operators over the inclusion order, that is: $A \subseteq B \implies \mathcal{C}(A) \subseteq \mathcal{C}(B)$ and $\mathcal{I}(A) \subseteq \mathcal{I}(B)$*
3. *Finite intersections and arbitrary unions of open sets are open.*

Closure spaces are a generalisation of *topological spaces*. The axioms defining a closure space are also part of the definition of a *Kuratowski closure space*, which is one of the possible alternative definition of a topological space. The link between topological and closure spaces is deep, and also involves the so-called *Alexandrov* spaces; we refer the reader to, e.g., [16] for more information. We provide some further details below.

Definition 2.4. A *topological space* is a pair (X, O) of a set X and a collection $O \subseteq \mathcal{P}(X)$ of subsets of X called *open sets*, such that $\emptyset, X \in O$, and subject to closure under arbitrary unions and finite intersections.

Definition 2.5. In a topological space (X, O), $A \subseteq X$ is called *closed* if its complement is open.

Note that $\emptyset$ and X are always closed and open. An equivalent definition was given by Kuratowski; this exhibits topological spaces as closure spaces where closure is idempotent.

Definition 2.6. A *topological space* is a closure space where the closure operator is idempotent, that is, for all $A \subseteq X$, $\mathcal{C}(\mathcal{C}(A)) = \mathcal{C}(A)$.

The Kuratowski definition and the open sets definition of a topological space are equivalent. Each topological space is also an (idempotent) closure space, and each idempotent closure space is a topological space. The proof can be sketched as follows. To view a topological space defined in terms of open sets as a closure space, one defines $\mathcal{C}(A)$ as the smallest closed set containing A (this is the standard topological closure). The properties of Definition 2.1 hold. For the converse, starting from a closure space $(X, \mathcal{C})$ whose closure operator is idempotent, the open sets are given by Definition 2.2 (3). The topological definition of closure in the obtained space coincides with $\mathcal{C}$.

A closure space can be also obtained by restricting the domain of another space.

Definition 2.7. Given a closure space $(X, \mathcal{C}_X)$ and a subset $Y \subseteq X$, we call *subspace closure* the operation $\mathcal{C}_Y : \mathcal{P}(Y) \to \mathcal{P}(Y)$ defined as $\mathcal{C}_Y(A) = \mathcal{C}_X(A) \cap Y$. We call $(Y, \mathcal{C}_Y)$ the subspace of $(X, \mathcal{C}_X)$ *generated by* Y.

Lemma 2.8. *The subspace closure is a closure operator.*

In [15], a discrete variant of the topological definition of the boundary of a set A is given, for the case where a closure operator is derived from a reflexive and symmetric relation (see Definition 3.1 in the next section). Therein, in Lemma 5, it is proved that the definition coincides with the one we provide below. The latter is only given in terms of closure and interior, and coincides with the definition of boundary in a topological space. We also need to define two variants on this notion, namely the *interior* and *closure* boundary (the latter is sometimes called *frontier*).

Definition 2.9. In a closure space $(X, \mathcal{C})$, the *boundary* of $A \subseteq X$ is defined as $\mathcal{B}(A) = \mathcal{C}(A) \setminus \mathcal{I}(A)$. The *interior boundary* is $\mathcal{B}^-(A) = A \setminus \mathcal{I}(A)$, and the *closure boundary* is $\mathcal{B}^+(A) = \mathcal{C}(A) \setminus A$.

Proposition 2.10. *The following equations hold in a closure space:*

$$\mathcal{B}(A) = \mathcal{B}^+(A) \cup \mathcal{B}^-(A) \tag{1}$$
$$\mathcal{B}^+(A) \cap \mathcal{B}^-(A) = \emptyset \tag{2}$$
$$\mathcal{B}(A) = \mathcal{B}(\overline{A}) \tag{3}$$
$$\mathcal{B}^+(A) = \mathcal{B}^-(\overline{A}) \tag{4}$$
$$\mathcal{B}^+(A) = \mathcal{B}(A) \cap \overline{A} \tag{5}$$
$$\mathcal{B}^-(A) = \mathcal{B}(A) \cap A \tag{6}$$
$$\mathcal{B}(A) = \mathcal{C}(A) \cap \mathcal{C}(\overline{A}) \tag{7}$$

3 Quasi-discrete closure spaces

In this section we see how a closure space may be derived starting from a *binary relation*, that is, a *graph*. The following comes from [16].

Definition 3.1. Consider a set X and a relation $R \subseteq X \times X$. A closure operator is obtained from R as $\mathcal{C}_R(A) = A \cup \{x \in X \mid \exists a \in A.(a, x) \in R\}$.

Remark 3.2. *One could also change Definition 3.1 so that* $\mathcal{C}_R(A) = A \cup \{x \in X \mid \exists a \in A.(x, a) \in R\}$, *which actually is the definition of [16]. This does not affect the theory presented in the paper. Indeed, one obtains the same results by replacing* R *with* R^{-1} *in statements of theorems that make explicit use of* R, *and are not invariant under such change.*

Proposition 3.3. *The pair* $(X, \mathcal{C}_R)$ *is a closure space.*

Closure operators obtained by Definition 3.1 are not necessarily idempotent. Lemma 11 in [16] provides a necessary and sufficient condition, that we rephrase below. We let $R^=$ denote the reflexive closure of R.

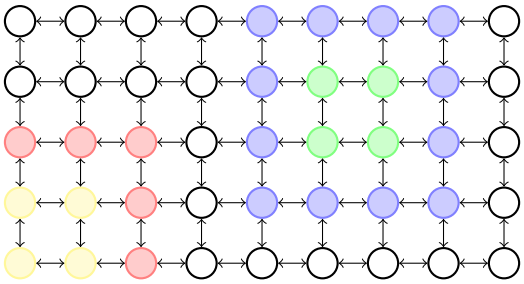

Figure 1: A graph inducing a *quasi-discrete* closure space

Lemma 3.4. *$\mathcal{C}_R$ is idempotent if and only if $R^=$ is transitive.*

Note that, when R is transitive, so is $R^=$, thus $\mathcal{C}_R$ is idempotent. The vice-versa is not true, e.g., when $(x, y) \in R$, $(y, x) \in R$, but $(x, x) \notin R$.

Remark 3.5. *In topology, open sets play a fundamental role. However, the situation is different in closure spaces derived from a relation R. For example, in the case of a closure space derived from a connected symmetric relation, the only open sets are the whole space, and the empty set.*

Proposition 3.6. *Given $R \subseteq X \times X$, in the space $(X, \mathcal{C}_R)$, we have:*

$$\mathcal{I}(A) = \{x \in A \mid \neg\exists a \in \overline{A}.(a, x) \in R\} \tag{8}$$

$$\mathcal{B}^-(A) = \{x \in A \mid \exists a \in \overline{A}.(a, x) \in R\} \tag{9}$$

$$\mathcal{B}^+(A) = \{x \in \overline{A} \mid \exists a \in A.(a, x) \in R\} \tag{10}$$

Closure spaces derived from a relation can be characterised as *quasi-discrete* spaces (see also Lemma 9 of [16] and the subsequent statements).

Definition 3.7. A closure space is *quasi-discrete* if and only if one of the following equivalent conditions holds: i) each $x \in X$ has a *minimal neighbourhood*[2] N_x; ii) for each $A \subseteq X$, $\mathcal{C}(A) = \bigcup_{a \in A} \mathcal{C}(\{a\})$.

The following is proved as Theorem 1 in [16].

Theorem 3.8. *A closure space $(X, \mathcal{C})$ is quasi-discrete if and only if there is a relation $R \subseteq X \times X$ such that $\mathcal{C} = \mathcal{C}_R$.*

Summing up, whenever one starts from an arbitrary relation $R \subseteq X \times X$, the obtained closure space $(X, \mathcal{C}_R)$ enjoys minimal neighbourhoods, and the closure of a set A is the union of the closure of the singletons composing A. Furthermore, such nice properties are only true in a closure space when there is some R such that the closure operator of the space is derived from R. In the reminder of this section, we exemplify some aspects of quasi-discreteness.

Example 3.9. Every graph induces a *quasi-discrete* closure space. For instance, we can consider the (undirected) graph depicted in Figure 1. Let R be the (symmetric) binary relation induced by the graph edges, and let Y and G denote the set of *yellow* and *green* nodes, respectively. The closure $\mathcal{C}_R(Y)$ consist of all *yellow* and *red* nodes, while the closure $\mathcal{C}_R(G)$ contains all *green* and *blue* nodes. The interior $\mathcal{I}(Y)$ of Y contains a single node, i.e. the one located at the bottom-left in Figure 1. On the contrary, the *interior* $\mathcal{I}(G)$ of G is empty. Indeed, we have that $\mathcal{B}(G) = \mathcal{C}(G)$, while $\mathcal{B}^-(G) = G$ and $\mathcal{B}^+(G)$ consists of the *blue* nodes.

[2] A *minimal neighbourhood* of x is a set that is a neighbourhood of x (Definition 2.2 (2)) and is included in all other neighbourhoods of x.

Example 3.10. Existence of minimal neighbourhoods does not depend on finiteness of the space, and not even do they depend on existence of a "closest element" for each point. To see this, consider the rational numbers $\mathbb{Q}$, equipped with the relation $\leq$. Such a relation is reflexive and transitive, thus the closure space $(\mathbb{Q}, \mathcal{C}_{\leq})$ is topological and quasi-discrete.

Example 3.11. Another example exhibiting minimal neighbourhoods in absence of closest elements is the following. Consider the rational numbers $\mathbb{Q}$ equipped with the relation $R = \{(x, y) \mid |x - y| \leq 1\}$, and let us look at the closure space $(\mathbb{Q}, \mathcal{C}_R)$. R is reflexive but not transitive, hence the obtained closure space is quasi-discrete but not topological. We have $\mathcal{C}_R(A) = \{x \in \mathbb{Q} \mid \exists a \in A. |a - x| \leq 1\}$. Consider a point x. For x to be included in $\mathcal{I}(A)$, a set A must include all the points whose distance from x is less than or equal to 1, in other words, it must be true that $[x - 1, x + 1] \subseteq A$. To see this, suppose that there is $z \notin A$ such that $|z - x| \leq 1$. Then $x \in \mathcal{C}_R(\{z\})$, thus since $\overline{A} = \overline{A} \cup \{z\}$, we have $x \in \mathcal{C}_R(\overline{A})$, and therefore $x \notin \mathcal{I}(A) = \overline{\mathcal{C}_R(\overline{A})}$. The minimal neighbourhood is thus $N_x = [x - 1, x + 1]$. It is easily verified that $\mathcal{I}(N_x) = \{x\}$. In other words, each point x has a minimal neighbourhood N_x, and there is no other point y such that N_x is a neighbourhood of y. However, there are infinitely many points belonging to N_x.

Example 3.12. An example of a topological closure space which is not quasi-discrete is the set of real numbers equipped with the Euclidean topology (the topology induced by arbitrary union and finite intersection of open intervals). To see that the space is not quasi-discrete, one applies Definition 3.7. Consider an open interval (x, y). We have $\mathcal{C}((x, y)) = [x, y]$, but for each point z, we also have $\mathcal{C}(z) = [z, z] = \{z\}$. Therefore $\bigcup_{z \in (x,y)} \mathcal{C}(z) = \bigcup_{z \in (x,y)} \{z\} = (x, y) \neq [x, y]$.

Example 3.13. The reader may think that quasi-discreteness is also related to the space having a smaller cardinality than that of the real numbers. This is not the case. To see this, just equip the real numbers with an arbitrary relation, e.g., the relation $\leq$, in a similar way to Example 3.10. The obtained closure space is quasi-discrete. This example illustrates why these spaces are not simply called "discrete".

4 The design of a spatial *until* operator

This paper is about spatial logics in closure spaces. Before introducing the complete framework, in this section, we discuss the design of a spatial *until operator*, that is, a binary logical connective of the form $\phi \mathcal{U} \psi$, with a natural intended meaning, that we state as a design principle.

Principle 4.1. We aim at defining a binary spatial logical operator $\mathcal{U}$, interpreted on points of a closure space. The basic idea is that point x satisfies $\phi \mathcal{U} \psi$ whenever it is included in an area A satisfying ϕ, and there is "no way out" from A unless passing trough an area B that satisfies ψ.

To turn this principle into a mathematical definition, one should clarify the meaning of the words *area*, *included*, *passing*, in the context of closure spaces. The variables are many. Consider the area A. Is it possible for A to be the singleton x, or do we require that x has a proper neighbourhood of points satisfying ϕ? This is more relevant in continuous spaces, where singletons are problematic (for example, in Euclidean spaces, they have measure 0). Definitions that are satisfactory in discrete spaces may have issues when switching to continuous domains, and vice-versa. We start from the topological definition of *until*, and investigate the situation with respect to Principle 4.1. For this, we need to define a *model*, providing a context of evaluation for the satisfaction relation, as in $\mathcal{M}, x \models \phi \mathcal{U} \psi$.

Definition 4.2. A *closure model* is a pair $\mathcal{M} = ((X, \mathcal{C}), \mathcal{V})$ consisting of a closure space $(X, \mathcal{C})$ and a valuation $\mathcal{V} : P \to 2^X$, assigning to each proposition letter the set of points where it holds.

When $(X, \mathcal{C})$ is the interpretation of a topological space as a closure space, we call $\mathcal{M}$ a *topological model*, in line with the definition used in [24], and [1]. Therein, the *topological until* operator is presented. We recall it below.

Definition 4.3. The *topological until* operator $\mathcal{U}_T$ is interpreted in a topological model $\mathcal{M}$ as $\mathcal{M}, x \models \phi \mathcal{U}_T \psi \iff \exists A$ open $.x \in A \wedge \forall y \in A. \mathcal{M}, y \models \phi \wedge \forall z \in \mathcal{B}(A). \mathcal{M}, z \models \psi$.

The intuition behind Definition 4.3 is that one seeks for an area A (which, topologically speaking, could sensibly be an open set) where ϕ holds, and that is completely surrounded by points where ψ holds. First, we report about a well-known problem of notions such as "inside" and "outside" in topology.

Issue 4.4. Definition 4.3 does not distinguish the inside from the outside of a boundary. Let $C = \{(x, y) \in \mathbb{R}^2 \mid x^2 + y^2 = 1\}$ be the circumference of radius 1 in $\mathbb{R}^2$, x a point inside of C (e.g., $(0, 0)$), y a point outside of C (e.g., $(2, 2)$). Consider a topological model based on $\mathbb{R}^2$ (with the Euclidean topology) such that $\mathcal{V}(p) = \mathbb{R}^2 \setminus C$ and $\mathcal{V}(q) = C$. Then we have $x \models p \mathcal{U}_T q$. This is in accordance to our stated requirements. But y satisfies $p \mathcal{U}_T q$ as well, as can be easily checked, by choosing as the set A *all* the points outside of C.

Issue 4.4 is a well-known subject in topology, stemming from the statement of *Jordan's theorem*: the complement of a closed curve C in $\mathbb{R}^2$ consists of exactly two connected components A and A'. The statement of the theorem does not make a distinction between A and A'. In a general topological space, it is not actually possible to make such a distinction (think of a closed curve on the surface of a sphere). In Euclidean spaces, to address this, one may resort to *homotopy* [25]. Investigating an appropriate notion of homotopy in closure spaces is not a trivial task. For example, in graphs, it is always possible to rearrange nodes, so notions of "inside" and "outside" can not be simply translated from Euclidean spaces.

In this work, we will seek for a more pragmatic approach. We shall assume that, on a by-need basis, one may make use of predicates denoting "extremal points" that should not be reachable by any point satisfying an until formula. This is reasonable in finite models as they typically are submodels of some infinite model, where a boundary has been used to delineate the "end of the world". In this situation, let e be a proposition letter denoting the extremal points; then the formula $(\phi \wedge \neg e) \mathcal{U}_T \phi$ distinguishes points that are "inside" ψ (and satisfy ϕ), from points that are "outside" of it. Once this is established, we ought to seek for a definition which is apt to closure spaces.

Issue 4.5. Definition 4.3 can not be translated directly to closure spaces, even if all the used topological notions have a counterpart in the more general setting. Open sets in closure spaces are often too coarse (see Remark 3.5). Moreover, the usage of $\mathcal{B}$ in Definition 4.3 is not satisfactory either. By Proposition 2.10 we have $\mathcal{B}(A) = \mathcal{B}^+(A) \cup \mathcal{B}^-(A)$. Let us look at the closure space $(\mathbb{Z}^2, \mathcal{C})$ induced by the graph with nodes in $\mathbb{Z}^2$ and edges linking each point (x, y) to its four neighbours[3] $(x - 1, y), (x + 1, y), (x, y - 1), (x, y + 1)$. Consider a rectangle A such that all points in A satisfy ϕ and all points in $\mathcal{B}^-(A)$ satisfy ψ, whereas all points in $\overline{A}$ do not satisfy ϕ or ψ. Then each point in $\mathcal{I}(A)$ is enclosed in the region delimited by ψ, thus satisfying $\phi \mathcal{U} \psi$ according to Principle 4.1. However, not all points in $\mathcal{B}(A)$ satisfy ψ, as ψ does not hold in $\mathcal{B}^+(A)$.

By Issue 4.5, we could chose either $\mathcal{B}^-$ or $\mathcal{B}^+$ to play the same role as $\mathcal{B}$. In the first case, we shall seek for a set A such that all points in its *interior* satisfy ϕ, and all the points in $\mathcal{B}^-(A)$ satisfy ψ. In the second case, we shall instead require that ϕ is satisfied by *all* the points of A, and that in $\mathcal{B}^+(A)$, ψ holds. This is to ensure that there are no "gaps" between ϕ and ψ. The definition using $\mathcal{B}^-$ presents an additional problem.

Issue 4.6. Consider the closure space $(\mathbb{Z}^2, \mathcal{C})$ described in Issue 4.5. Consider the "cross" $H = \{(-1, 0), (0, 0), (1, 0), (0, -1), (0, 1)\}$ where point $x = (0, 0)$ satisfies ϕ and the remaining points satisfy ψ. Let $A = H$. Then $\mathcal{I}(A) = \{x\}$ satisfies ϕ and $\mathcal{B}^-(A)$ satisfies ψ, agreeing with our intuition. However, if we restrict the space to $(H, \mathcal{C})$ (see Definition 2.7), then A is the set of all points, thus

[3] In the literature, such space is sometimes referred to as the *digital plane*; however, the same term may also denote $\mathbb{Z}^2$ equipped with different notions of neighbourhood.

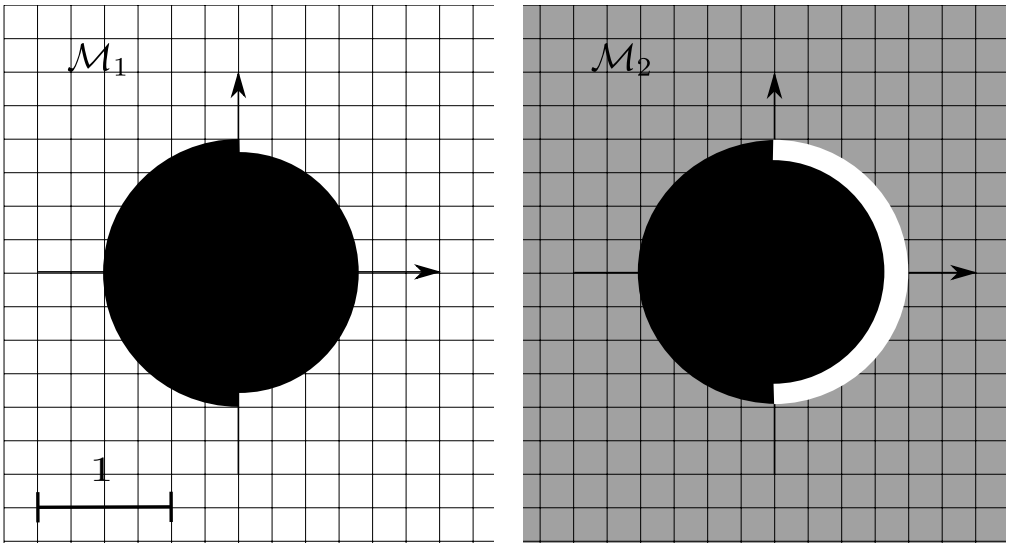

Figure 2: Three continuous closure models (boundaries are deliberately very tick, but the reader should think of them as infinitely thin).

$\mathcal{I}(A) = A$ and $\mathcal{B}^-(A) = \emptyset$. Therefore, not all points of $\mathcal{I}(A)$ satisfy ϕ. A quick analysis of all the possible choices for A shows that, in this case, it is impossible to find A such that ϕ holds in $\mathcal{I}(A)$ and ψ holds in $\mathcal{B}^-(A)$.

Furthermore, in closure spaces $(X, \mathcal{C}_R)$ derived from a non-symmetric relation R, $\mathcal{B}^-(A)$ is the set of points that have *incoming* edges from $\overline{A}$ (see Proposition 3.6). As in Principle 4.1 we mention the idea of having "no way *out*" of A, it intuitively makes sense to consider $\mathcal{B}^+(A)$ instead. In Section 5 and Section 7, we shall see how all these observations converge to a definition which behaves according to Principle 4.1 over discrete structures.

Until operators are especially difficult to study in continuous spaces (such as the Euclidean plane), as intuition may be easily misled. Let us consider an example.

Example 4.7. We shall define two models based on the Euclidean topology over $\mathbb{R}^2$, seen as a closure space $(\mathbb{R}^2, \mathcal{C})$. We use propositions b, w, g, depicted in Figure 2 as black, white and grey areas, respectively. Consider the sets $H = \{(x,y) | x^2 + y^2 < 1\}$, $H^< = \{(x,y) | x^2 + y^2 = 1 \wedge x < 0\}$, $H^\geq = \{(x,y) | x^2 + y^2 = 1 \wedge x \geq 0\}$. Let $\mathcal{M}_i = ((\mathbb{R}^2, \mathcal{C}), \mathcal{V}_i)$, for $i \in \{1,2\}$. Fix valuations as follows: $\mathcal{V}_1(b) = H \cup H^<$, $\mathcal{V}_1(w) = \mathbb{R}^2 \setminus \mathcal{V}_1(b)$, $\mathcal{V}_1(g) = \emptyset$, $\mathcal{V}_2(b) = \mathcal{V}_1(b)$, $\mathcal{V}_2(w) = H^\geq$, $\mathcal{V}_2(g) = \mathbb{R}_2 \setminus (H \cup H^< \cup H^\geq)$. Let $x \in H$. The expectation is that $b\mathcal{U}w$ holds at x in $\mathcal{M}_1$. Furthermore, $b\mathcal{U}w$ should not hold at x in $\mathcal{M}_2$. However, $\mathcal{M}_2, x \models b\mathcal{U}w$ by the choice $A = H \cup H^<$, because $\mathcal{B}^+(A) = H^\geq$.

We remark that, although it is not immediately useful for verification, being able to characterise also continuous space is certainly of theoretical interest, and it could be worth of some deeper analysis.

5 A Spatial Logic for Closure Spaces

In this section we present *SLCS*: a *Spatial Logic for Closure Spaces*. The logic features boolean operators, and two *spatial operators*: a "one step" modality, turning closure into a logical operator, and a binary *until* operator, which is interpreted spatially. As we shall see, the *SLCS* formula $\phi\mathcal{U}\psi$ requires ϕ to hold at least on one point. The operator is a *weak until* in temporal logics terminology, as there may be no point satisfying ψ, if ϕ holds everywhere; this is in line with the topological interpretation of until (Definition 4.3).

$$
\begin{array}{lcl@{\qquad}lcl}
\bot & \triangleq & \neg\top & \phi \vee \psi & \triangleq & \neg(\neg\phi \wedge \neg\psi) \\
\Box\phi & \triangleq & \neg(\Diamond\neg\phi) & \delta\phi & \triangleq & (\Diamond\phi) \wedge (\neg\Box\phi) \\
\delta^{-}\phi & \triangleq & \phi \wedge (\neg\Box\phi) & \delta^{+}\phi & \triangleq & (\Diamond\phi) \wedge (\neg\phi) \\
\phi\,\mathcal{R}\psi & \triangleq & \neg((\neg\psi)\,\mathcal{U}(\neg\phi)) & \mathcal{G}\phi & \triangleq & \phi\,\mathcal{U}\bot \\
\mathcal{F}\phi & \triangleq & \neg\mathcal{G}(\neg\phi) & & &
\end{array}
$$

Figure 3: $SLCS$ derivable operators

As we explained in Section 4, the spatial logical operator $\mathcal{U}$ is interpreted on points of a closure space. The basic idea is that point x satisfies $\phi\mathcal{U}\psi$ whenever it is included in an area A satisfying ϕ, and there is "no way out" from A unless passing trough an area B that satisfies ψ. For instance, if we consider the model of Figure 1, *yellow* nodes should satisfy *yellow* $\mathcal{U}$ *red* while *green* nodes should satisfy *green* $\mathcal{U}$ *blue*. In order to formally define $SLCS$, recall the definition of *model* (Definition 4.2). From now on, fix a (finite or countable) set P of *proposition letters.*

Definition 5.1. The syntax of $SLCS$ is defined by the following grammar, where p ranges over P:

$$\Phi ::= p \mid \top \mid \neg\Phi \mid \Phi \wedge \Phi \mid \Diamond\Phi \mid \Phi\,\mathcal{U}\Phi$$

Here, $\top$ denotes the truth value *true*, $\neg$ is negation, $\wedge$ is conjunction, $\Diamond$ is the *closure* operator, and $\mathcal{U}$ is the *until* operator. Closure (and interior) operators come from the tradition of topological spatial logics [24].

Definition 5.2. Satisfaction $\mathcal{M}, x \models \phi$ of formula ϕ at point x in model $\mathcal{M} = ((X, \mathcal{C}), \mathcal{V})$ is defined, by induction on terms, as follows:

$$
\begin{array}{lcll}
\mathcal{M}, x & \models & p & \iff x \in \mathcal{V}(p) \\
\mathcal{M}, x & \models & \top & \iff \mathit{true} \\
\mathcal{M}, x & \models & \neg\phi & \iff \mathcal{M}, x \not\models \phi \\
\mathcal{M}, x & \models & \phi \wedge \psi & \iff \mathcal{M}, x \models \phi \text{ and } \mathcal{M}, x \models \psi \\
\mathcal{M}, x & \models & \Diamond\phi & \iff x \in \mathcal{C}(\{y \in X | \mathcal{M}, y \models \phi\}) \\
\mathcal{M}, x & \models & \phi\mathcal{U}\psi & \iff \exists A \subseteq X. x \in A \wedge \forall y \in A. \mathcal{M}, y \models \phi \wedge \\
 & & & \qquad \wedge \forall z \in \mathcal{B}^{+}(A). \mathcal{M}, z \models \psi
\end{array}
$$

In Figure 3, we present some derived operators. Besides standard logical connectives, the logic can express the *interior* ($\Box\phi$), the *boundary* ($\delta\phi$), the *interior boundary* ($\delta^{-}\phi$) and the *closure boundary* ($\delta^{+}\phi$) of the set of points satisfying formula ϕ. Moreover, by appropriately using the *until* operator, operators concerning *reachability* ($\phi\,\mathcal{R}\psi$), *global satisfaction* ($\mathcal{G}$) and *possible satisfaction* ($\mathcal{F}\phi$) can be derived.

To clarify the expressive power of $\mathcal{U}$ and operators derived from it we provide Theorem 5.5 and Theorem 5.7, giving a formal meaning to the idea of "way out" of ϕ, and providing an interpretation of $\mathcal{U}$ in terms of *paths*.

Definition 5.3. A *continuous function* $f : (X_1, \mathcal{C}_1) \to (X_2, \mathcal{C}_2)$ is a function $f : X_1 \to X_2$ such that, for all $A \subseteq X_1$, $f(\mathcal{C}_1(A)) \subseteq \mathcal{C}_2(f(A))$.

Definition 5.4. Consider a closure space $(X, \mathcal{C})$, and the quasi-discrete space $(\mathbb{N}, \mathcal{C}_{Succ})$, where $(n, m) \in Succ \iff m = n + 1$. A (countable) *path* in $(X, \mathcal{C})$ is a continuous function $p : (\mathbb{N}, \mathcal{C}_{Succ}) \to (X, \mathcal{C})$. We call p a path *from* x, and write $p : x \rightsquigarrow \infty$, when $p(0) = x$. We write $y \in p$ whenever there is $l \in \mathbb{N}$ such that $p(l) = y$. Finally, we write $p : x \overset{A}{\underset{y}{\rightsquigarrow}} \infty$ when p is a path from x, and there is l with $p(l) = y$ and for all $l' \leq l . p(l') \in A$.

Theorem 5.5. *If $\mathcal{M}, x \models \phi \mathcal{U} \psi$, then for each $p : x \rightsquigarrow \infty$ and l, if $\mathcal{M}, p(l) \models \neg\phi$, there is $k \in \{1, \ldots, l\}$ such that $\mathcal{M}, p(k) \models \psi$.*

Theorem 5.5 can be strengthened to a necessary and sufficient condition in the case of models based on quasi-discrete spaces. First, we establish that paths in a quasi-discrete space are also paths in its underlying graph.

Lemma 5.6. *Given path p in a quasi-discrete space $(X, \mathcal{C}_R)$, for all $i \in \mathbb{N}$ with $p(i) \neq p(i+1)$, we have $(p(i), p(i+1) \in R)$, i.e., the image of p is a (graph theoretical, infinite) path in the graph of R. Conversely, each path in the graph of R uniquely determines a path in the sense of Definition 5.4.*

Theorem 5.7. *In a quasi-discrete closure model $\mathcal{M}$, $\mathcal{M}, x \models \phi \mathcal{U} \psi$ if and only if $\mathcal{M}, x \models \phi$, and for each path $p : x \rightsquigarrow \infty$ and $l \in \mathbb{N}$, if $\mathcal{M}, p(l) \models \neg\phi$, there is $k \in \{1, \ldots, l\}$ such that $\mathcal{M}, p(k) \models \psi$.*

Remark 5.8. *Directly from Theorem 5.7 and from definitions in Figure 3 we have also that in a quasi-discrete closure model $\mathcal{M}$:*

1. *$\mathcal{M}, x \models \phi \mathcal{R} \psi$ iff there is $p : x \rightsquigarrow \infty$ and $k \in \mathbb{N}$ such that $\mathcal{M}, p(k) \models \psi$ and for each $j \in \{1, \ldots, k\}$ $\mathcal{M}, p(j) \models \phi$;*
2. *$\mathcal{M}, x \models \mathcal{G}\phi$ iff for each $p : x \rightsquigarrow \infty$ and $i \in \mathbb{N}$, $\mathcal{M}, p(i) \models \phi$;*
3. *$\mathcal{M}, x \models \mathcal{F}\phi$ iff there is $p : x \rightsquigarrow \infty$ and $i \in \mathbb{N}$ such that $\mathcal{M}, p(i) \models \phi$.*

Note that point x satisfies $\phi \mathcal{R} \psi$ if and only if either ψ is satisfied by x or there exists a sequence of points after x, all satisfying ϕ, leading to a point satisfying both ψ and ϕ. In the second case, it is not required that x satisfies ϕ.

6 Model checking $SLCS$ formulas

In this section we present a model checking algorithm for $SLCS$, which is a variant of standard CTL model checking [3].

Function `Sat`, presented in Algorithm 1, takes as input a quasi-discrete model $\mathcal{M} = ((X, \mathcal{C}_R), \mathcal{V})$ and a $SLCS$ formula ϕ, and returns the set of all points in X satisfying ϕ. The function is inductively defined on the structure of ϕ and, following a bottom-up approach, computes the resulting set via an appropriate combination of the recursive invocations of `Sat` on the sub formulas of ϕ. When ϕ is $\top$, p, $\neg\psi$ or $\psi \wedge \xi$, definition of $\texttt{Sat}(\mathcal{M}, \phi)$ is as expected. To compute the set of points satisfying $\Diamond\psi$, the closure operator $\mathcal{C}$ of the space is applied to the set of points satisfying ψ.

When ϕ is of the form $\psi \mathcal{U} \xi$, function `Sat` relies on the function `CheckUntil` defined in Algorithm 2. This function takes as parameters a model $\mathcal{M}$ and two $SLCS$ formulas ψ and ξ and computes the set of points in $\mathcal{M}$ satisfying $\psi \mathcal{U} \xi$ by removing from $V = \texttt{Sat}(\mathcal{M}, \psi)$ all the *bad* points. A point is *bad* if it can *reach* a point satisfying $\neg\psi$ without passing through a point satisfying ξ. Let $Q = \texttt{Sat}(\mathcal{M}, \xi)$ be the set of points in $\mathcal{M}$ satisfying ξ. To identify the *bad* points in V the function `CheckUntil` performs a *backward search* from $T = \mathcal{B}^+(V \cup Q)$. Note that any *path* exiting from $V \cup Q$ has to pass through points in T. Moreover, the latter only contains points that satisfy neither ψ nor ξ. Until T is empty, function `CheckUntil` first picks an element x in T and then removes from V the set of (bad) points N that can reach x in *one step*. To compute the set N we use the function $pre(x) = \{y \in X \mid (y, x) \in R\}$. At the end of each iteration the set T is updated by considering the set of new discovered *bad points*.

Finally, we define the notion of *size* of a formula, in order to address termination, complexity and correctness of the model checking algorithm.

Definition 6.1. For Φ a $SLCS$ formula, let $size(\Phi)$ be inductively defined as follow:

Function $\texttt{Sat}(\mathcal{M}, \phi)$
  Input: Quasi-discrete closure model
    $\mathcal{M} = ((X, \mathcal{C}), \mathcal{V})$, formula ϕ
  Output: Set of points
    $\{x \in X \mid \mathcal{M}, x \models \phi\}$
  Match ϕ
    case $\top$: **return** X
    case p : **return** $\mathcal{V}(p)$
    case $\neg\psi$:
      let $P = \texttt{Sat}(\mathcal{M}, \psi)$ **in**
      return $X \setminus P$
    case $\psi \wedge \xi$:
      let $P = \texttt{Sat}(\mathcal{M}, \psi)$ **in**
      let $Q = \texttt{Sat}(\mathcal{M}, \xi)$ **in**
      return $P \cap Q$
    case $\Diamond\psi$:
      let $P = \texttt{Sat}(\mathcal{M}, \psi)$ **in**
      return $\mathcal{C}(P)$
    case $\psi \mathcal{U} \xi$:
      return `CheckUntil` $(\mathcal{M}, \psi, \xi)$

Algorithm 1: Decision procedure for the model checking problem.

Function `CheckUntil` $(\mathcal{M}, \psi, \xi)$
  let $V = \texttt{Sat}(\mathcal{M}, \psi)$ **in**
  let $Q = \texttt{Sat}(\mathcal{M}, \xi)$ **in**
  var $T := \mathcal{B}^+(V \cup Q)$
  while $T \neq \emptyset$ **do**
    $T' := \emptyset$
    for $x \in T$ **do**
      $N := pre(x) \cap V$
      $V := V \setminus N$
      $T' := T' \cup (N \setminus Q)$
    $T := T'$;
  return V

Algorithm 2: Checking until formulas in a quasi-discrete closure space.

- $size(\top) = size(p) = 1$
- $size(\neg\Phi) = size(\Diamond\Phi) = 1 + size(\Phi)$
- $size(\Phi \wedge \Psi) = size(\Phi \mathcal{U} \Psi) = 1 + size(\Phi) + size(\Psi)$

Lemma 6.2. *For any finite quasi-discrete model $\mathcal{M} = ((X, \mathcal{C}_R), \mathcal{V})$ and SLCS formula ϕ of size k,* `Sat` *terminates in $\mathcal{O}(k \cdot (|X| + |R|))$ steps.*

Theorem 6.3. *For any finite quasi-discrete closure model $\mathcal{M} = ((X, \mathcal{C}), \mathcal{V})$ and SLCS formula ϕ, $x \in \texttt{Sat}(\mathcal{M}, \phi)$ if and only if $\mathcal{M}, x \models \phi$.*

7 A model checker for spatial logics

The algorithm described in Section 6 is available as a proof-of-concept tool[4]. The tool, implemented using the functional language OCaml, contains a generic implementation of a global model checker using closure spaces, parametrised by the type of models.

An example of the tool usage is to approximately identify regions of interest on a digital picture (e.g., a map, or a medical image), using spatial formulas. In this case, digital pictures are treated as quasi-discrete models in the plane $\mathbb{Z} \times \mathbb{Z}$. The language of propositions is extended to formulas dealing with colour ranges, in order to cope with images where there are different shades of certain colours.

In Figure 4 we show a digital picture of a maze. The green area is the exit. The blue areas are start points. The input of the tool is shown in Figure 6, where the `Paint` command is used to invoke the global model checker and colour points satisfying a given formula. The concrete syntax of the model checker uses symbols `!`, `&`, `|`, `U`, `C` to denote the logical operators $\neg$, $\wedge$, $\vee$, $\mathcal{U}$, $\mathcal{C}$, respectively. Three formulas, making use of the until operator, are used to identify interesting areas. The output of the tool is in Figure 5. The colour red denotes start points from which the exit can be reached. Orange and yellow indicate the two regions through which the exit can be reached, including and excluding a start point, respectively. The properties of interest for this example are expressed in terms of the derived

[4]Web site: `http://www.github.com/vincenzoml/slcs`.

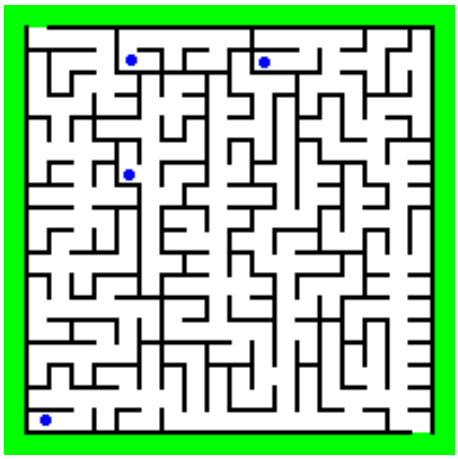

Figure 4: A maze.

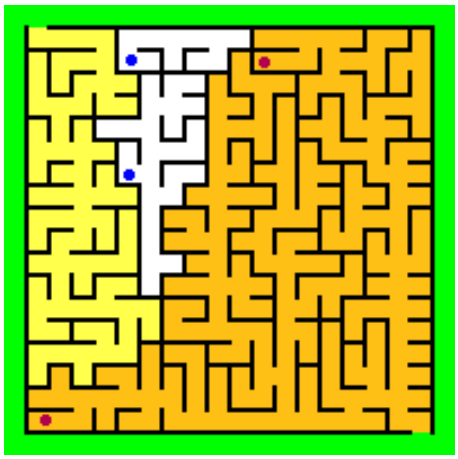

Figure 5: Model checker output.

```
Let reach(a,b) = !( (!b) U (!a) );
Let reachThrough(a,b) = a & reach((a|b),b);

Let toExit = reachThrough(["white"],["green"]);
Let fromStartToExit = toExit & reachThrough(["white"],["blue"]);
Let startCanExit = reachThrough(["blue"],fromStartToExit);

Paint "yellow" toExit;
Paint "orange" fromStartToExit;
Paint "red" startCanExit;
```

Figure 6: Input to the model checker.

reachability operator $\mathcal{R}$ (see Figure 3), introduced in the model checking session by the definition of `reach(a,b)`. The definition of `reach(a,b)` by itself is not yet expressing exactly what we need; thus, the definition of `reachThrough(a,b)` is introduced. The definition of `reachThrough(a,b)` requires that the initial state of the paths satisfying the reachability property also satisfy `a`, and moreover, that along each path `a` or `b` holds until `b` is reached. The latter is needed since `reach(a,b)` requires that each path leads to a point in which both `a` and `b` hold. Since in the maze each point is labelled by exactly one proposition (colour) this implies that we are looking for paths that lead to points that satisfy `(a|b) R b`.

In Figure 7 we show a digital image[5] depicting a portion of the map of Pisa, featuring a red circle which denotes a train station. Streets of different importance are painted with different colours in the map. The model checker is used to identify the area surrounding the station which is delimited by main streets, and the delimiting main streets. The output of the tool is shown in Figure 8, where the station area is coloured in orange, the surrounding main streets are red, and other main streets are in green. The source code of the model checking session is shown in Figure 9. The most relevant formula is `Let stationArea = !((!station) U street);`. The formula `(!station) U street` captures those points that live in an area not containing the station, and completely surrounded by main streets of a certain importance; this also includes those points included in the main streets. The complement of these points is the area containing the station. The operator `C^n`, for `n` a positive integer, is a shorthand for n nested applications of `C`, which corresponds to the "dilation" operation of digital imaging and is used to reconnect parts of a picture that would be disconnected due to noise in the image (such as street names printed on the map). In the formula `streetsAroundStation`, dilation

[5] © *OpenStreetMap contributors* – `http://www.openstreetmap.org/copyright`.

Figure 7: Input: the map of a town.

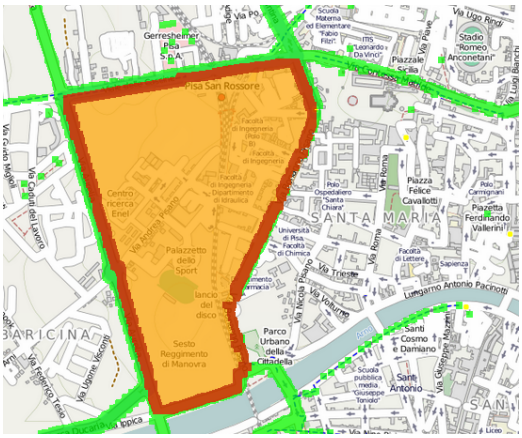

Figure 8: Output of the tool.

```
Let bigStreet = [RED > 160] & [RED < 180] &
    [GREEN > 150] & [GREEN < 180] & [BLUE > 200] & [BLUE < 255];
Let midStreet1 = [RED > 230] & [RED < 240] &
    [GREEN > 233] & [GREEN < 240] & [BLUE > 190] & [BLUE < 210];
Let midStreet2 = [RED > 235] & [RED < 245] &
    [GREEN > 220] & [GREEN < 230] & [BLUE > 220] & [BLUE < 230];

Let street = C^3 (bigStreet | midStreet1 | midStreet2);

Let station = [RED > 200] & [RED < 225] &
    [GREEN > 30] & [GREEN < 60] & [BLUE > 50] & [BLUE < 80];
Let stationArea = !((!station) U street);
Let streetsAroundStation = street & (C^10 stationArea);

Paint "orange" stationArea;
Paint "green" street;
Paint "red" streetsAroundStation;
```

Figure 9: Source code of the model checking session.

is used to capture main streets that are near the station area[6]. The `Paint` command invokes the model checking function and superimposes an user-specified colour over the points that satisfy a given formula, using some transparency as a visual aid. As a mere hint on how practical is to use a model checker for image analysis, the execution time on our test image, consisting of about 250000 pixels, is in the order of ten seconds on a standard laptop equipped with a 2Ghz processor.

8 Conclusions and Future Work

In this paper, we have presented a methodology to verify properties that depend upon physical space. We have defined the spatial logic *SLCS*, stemming from the tradition of topological interpretations of modal logics, dating back to earlier logicians such as Tarski, where modalities describe neighbourhood. The topological definitions have been lifted to a more general setting, also encompassing discrete,

[6]The value of `n` (in this case 10) is an estimate of the width of a street, expressed in number of pixels; this is typically a constant in maps taken from the same source.

graph-based structures. The proposed framework has been extended with a spatial variant of the *until* operator, and we have also defined an efficient model checking procedure, which is implemented in a proof-of-concept tool.

As future work, we first of all plan to merge the results presented in this paper with temporal reasoning. This integration can be done in more than one way. It is not difficult to consider "snapshot" models consisting of a temporal model (e.g., a Kripke frame) where each state is in turn a closure model, and atomic formulas of the temporal fragment are replaced by spatial formulas. The various possible combinations of temporal and spatial operators, in linear and branching time, are examined for the case of topological models and basic modal formulas in [19]. Snapshot models may be susceptible to state-space explosion problems as spatial formulas could need to be recomputed at every state. On the other hand, one might be able to exploit the fact that changes of space over time are incremental and local in nature. Promising ideas are presented both in [16], where principles of "continuous change" are proposed in the setting of closure spaces, and in [20] where spatio-temporal models are generated by locally-scoped update functions, in order to describe dynamic systems. In the setting of collective adaptive systems, it will be certainly needed to extend the basic framework we presented with metric aspects (e.g., distance-bounded variants of the until operator), and probabilistic aspects, using atomic formulas that are probability distributions. A thorough investigation of these issues will be the object of future research.

A further challenge in spatial and spatio-temporal reasoning is posed by recursive spatial formulas, *a la* μ-calculus, especially on infinite structures with relatively straightforward generating functions (think of fractals, or fluid flow analysis of continuous structures). Such infinite structures could be described by topologically enhanced variants of ω-automata. Classes of automata exist living in specific topological structures; an example is given by *nominal automata* (see e.g., [5, 13, 21]), that can be defined using presheaf toposes [12]. This standpoint could be enhanced with notions of neighbourhood coming from closure spaces, with the aim of developing a unifying theory of languages and automata describing physical spaces, graphs, and process calculi with resources.

References

[1] Marco Aiello. *Spatial Reasoning: Theory and Practice*. PhD thesis, Institute of Logic, Language and Computation, University of Amsterdam, 2002.

[2] Marco Aiello, Ian Pratt-Hartmann, and Johan van Benthem, editors. *Handbook of Spatial Logics*. Springer, 2007.

[3] Christel Baier and Joost-Pieter Katoen. *Principles of model checking*. MIT Press, 2008.

[4] Alberto Del Bimbo, Enrico Vicario, and Daniele Zingoni. Symbolic description and visual querying of image sequences using spatio-temporal logic. *IEEE Trans. Knowl. Data Eng.*, 7(4):609–622, 1995.

[5] Mikolaj Bojanczyk, Bartek Klin, and Slawomir Lasota. Automata with group actions. In *LICS*, pages 355–364, 2011.

[6] Luca Bortolussi, Jane Hillston, Diego Latella, and Mieke Massink. Continuous approximation of collective system behaviour: A tutorial. *Performance Evaluation*, 70(5):317 – 349, 2013.

[7] L. Caires and L. Cardelli. A spatial logic for concurrency (part I). *Information and Computation*, 186(2):194–235, 2003.

[8] Luca Cardelli, Philippa Gardner, and Giorgio Ghelli. A spatial logic for querying graphs. In *ICALP*, pages 597–610, 2002.

[9] Luca Cardelli and Andrew D. Gordon. Anytime, anywhere: Modal logics for mobile ambients. In *Proceedings of the 30th SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL'00)*, pages 365–377, 2000.

[10] Augustin Chaintreau, Jean-Yves Le Boudec, and Nikodin Ristanovic. The age of gossip: Spatial mean field regime. In *Proceedings of the Eleventh International Joint Conference on Measurement and Modeling of Computer Systems*, SIGMETRICS '09, pages 109–120, New York, NY, USA, 2009. ACM.

[11] Rocco De Nicola, Gian Luigi Ferrari, and Rosario Pugliese. Klaim: A kernel language for agents interaction and mobility. *IEEE Trans. Software Eng.*, 24(5):315–330, 1998.

[12] Marcelo P. Fiore and Sam Staton. Comparing operational models of name-passing process calculi. *Inf. Comput.*, 204(4):524–560, 2006.

[13] Murdoch James Gabbay and Vincenzo Ciancia. Freshness and name-restriction in sets of traces with names. In *FOSSACS*, pages 365–380, 2011.

[14] Fabio Gadducci and Alberto Lluch-Lafuente. Graphical encoding of a spatial logic for the *pi* -calculus. In *CALCO*, pages 209–225, 2007.

[15] Antony Galton. The mereotopology of discrete space. In Christian Freksa and DavidM. Mark, editors, *Spatial Information Theory. Cognitive and Computational Foundations of Geographic Information Science*, volume 1661 of *Lecture Notes in Computer Science*, pages 251–266. Springer Berlin Heidelberg, 1999.

[16] Antony Galton. A generalized topological view of motion in discrete space. *Theoretical Computer Science*, 305(1–3):111 – 134, 2003.

[17] Radu Grosu, Scott A. Smolka, Flavio Corradini, Anita Wasilewska, Emilia Entcheva, and Ezio Bartocci. Learning and detecting emergent behavior in networks of cardiac myocytes. *Commun. ACM*, 52(3):97–105, March 2009.

[18] T. Yung Kong and Azriel Rosenfeld. Digital topology: Introduction and survey. *Computer Vision, Graphics, and Image Processing*, 48(3):357–393, 1989.

[19] Roman Kontchakov, Agi Kurucz, Frank Wolter, and Michael Zakharyaschev. Spatial logic + temporal logic = ? In Aiello et al. [2], pages 497–564.

[20] Philip Kremer and Grigori Mints. Dynamic topological logic. In Aiello et al. [2], pages 565–606.

[21] Alexander Kurz, Tomoyuki Suzuki, and Emilio Tuosto. On nominal regular languages with binders. In *FoSSaCS*, pages 255–269, 2012.

[22] Azriel Rosenfeld. Digital topology. *The American Mathematical Monthly*, 86(8):621–630, 1979.

[23] Michael B. Smyth and Julian Webster. Discrete spatial models. In Springer [2], pages 713–798.

[24] Johan van Benthem and Guram Bezhanishvili. Modal logics of space. In *Handbook of Spatial Logics*, pages 217–298. 2007.

[25] George William Whitehead. *Elements of Homotopy Theory*. Springer-Verlag, 1978.

A Proofs

Proof. (of Lemma 2.3)

Proof of item 1

$$
\begin{array}{ll}
 & A = \mathcal{I}(A) \\
\iff & \overline{A} = \overline{\mathcal{I}(A)} \\
\iff & \overline{A} = \mathcal{C}(\overline{A})
\end{array}
$$

Proof of item 2

$$
\begin{array}{ll}
 & A \subseteq B \\
\iff & A \cup B = B \\
\implies & [\,\text{def. closure}\,] \\
 & \mathcal{C}(A) \cup \mathcal{C}(B) = \mathcal{C}(B) \\
\iff & \mathcal{C}(A) \subseteq \mathcal{C}(B)
\end{array}
$$

$$
\begin{array}{ll}
 & A \subseteq B \\
\implies & \overline{B} \subseteq \overline{A} \\
\implies & [\,\text{previous part of the proof}\,] \\
 & \mathcal{C}(\overline{B}) \subseteq \mathcal{C}(\overline{A}) \\
\iff & \overline{\mathcal{I}(B)} \subseteq \overline{\mathcal{I}(A)} \\
\iff & \mathcal{I}(A) \subseteq \mathcal{I}(B)
\end{array}
$$

Proof of item 3

$$
\begin{array}{ll}
 & \mathcal{I}(A \cap B) \\
= & \overline{\mathcal{C}(\overline{A \cap B})} \\
= & \overline{\mathcal{C}(\overline{A} \cup \overline{B})} \\
= & [\,\text{definition of closure}\,] \\
 & \overline{\mathcal{C}(\overline{A}) \cup \mathcal{C}(\overline{B})} \\
= & \overline{\mathcal{C}(\overline{A})} \cap \overline{\mathcal{C}(\overline{B})} \\
= & \mathcal{I}(A) \cap \mathcal{I}(B) \\
= & [\,A \text{ and } B \text{ are open}\,] \\
 & A \cap B
\end{array}
$$

Finally, we prove that, whenever all sets in a collection $A_{i \in I}$ are open, we have $\mathcal{I}(\bigcup_{i \in I} A_i) = \bigcup_{i \in I} A_i$, that is, the union of open sets is open. The left-to-right inclusion is true since $\forall A.\mathcal{I}(A) \subseteq A$, which is the property $\forall A.A \subseteq \mathcal{C}(A)$ (Definition 2.1), dualised by the definition of interior. For the right-to-left inclusion we have:

$$
\begin{array}{ll}
 & \mathit{true} \\
\implies & [\,\text{definition of } \bigcup\,] \\
 & \forall i \in I.A_i \subseteq \bigcup_{i \in I} A_i \\
\implies & [\,\mathcal{I} \text{ is monotone by Lemma 2.3, item 2}\,]
\end{array}
$$

$$
\begin{array}{ll}
& \forall i \in I.\mathcal{I}(A_i) \subseteq \mathcal{I}(\bigcup_{i\in I} A_i) \\
\Longrightarrow & [\,\forall i \in I.A_i \text{ is open}\,] \\
& \forall i \in I.A_i \subseteq \mathcal{I}(\bigcup_{i\in I} A_i) \\
\Longrightarrow & \bigcup_{i\in I} A_i \subseteq \mathcal{I}(\bigcup_{i\in I} A_i)
\end{array}
$$

□

Proof. (of Lemma 2.8) We have

$$
\begin{array}{ll}
& \mathcal{C}_Y(\emptyset) \\
= & \mathcal{C}_X(\emptyset \cap Y) \\
= & \mathcal{C}_X(\emptyset) \\
= & \emptyset
\end{array}
$$

$$
\begin{array}{ll}
& \mathit{true} \\
\Longrightarrow & [\,\mathcal{C}_X \text{ is a closure operator}\,] \\
& A \subseteq \mathcal{C}_X(A) \\
\Longrightarrow & [\,A \subseteq Y\,] \\
& A \subseteq \mathcal{C}_X(A) \cap Y \\
\Longleftrightarrow & A \subseteq \mathcal{C}_Y(A)
\end{array}
$$

$$
\begin{array}{ll}
& \mathcal{C}_Y(A \cup B) \\
= & \mathcal{C}_X(A \cup B) \cap Y \\
= & (\mathcal{C}_X(A) \cup \mathcal{C}_X(B)) \cap Y \\
= & (\mathcal{C}_X(A) \cap Y) \cup (\mathcal{C}_X(B) \cap Y) \\
= & \mathcal{C}_Y(A) \cup \mathcal{C}_Y(B)
\end{array}
$$

□

Proof. (of Proposition 2.10)

Equation 1:

$$
\begin{array}{ll}
& \mathcal{B}(A) \\
= & \mathcal{C}(A) \setminus \mathcal{I}(A) \\
= & [\,\mathcal{I}(A) \subseteq A, \forall A, B, C.B \subseteq C \implies A \setminus B = (A \setminus C) \cup (C \setminus B)\,] \\
& (\mathcal{C}(A) \setminus A) \cup (A \setminus \mathcal{I}(A)) \\
= & \mathcal{B}^+(A) \cup \mathcal{B}^-(A)
\end{array}
$$

Equation 2:

$$
\begin{array}{ll}
& \mathcal{B}^+(A) \cap \mathcal{B}^-(A) \\
= & (\mathcal{C}(A) \setminus A) \cap (A \setminus \mathcal{I}(A)) \\
= & [\,\mathcal{C}(A) \setminus A \subseteq \overline{A}, A \setminus \mathcal{I}(A) \subseteq A\,] \\
& \emptyset
\end{array}
$$

Equation 3:

$$
\begin{aligned}
& \mathcal{B}(A) \\
= \quad & \mathcal{C}(A) \setminus \mathcal{I}(A) \\
= \quad & \overline{\mathcal{I}(\overline{A})} \setminus \overline{\mathcal{C}(\overline{A})} \\
= \quad & \mathcal{C}(\overline{A}) \setminus \mathcal{I}(\overline{A}) \\
= \quad & \mathcal{B}(\overline{A})
\end{aligned}
$$

Equation 4:

$$
\begin{aligned}
& \mathcal{B}^-(\overline{A}) \\
= \quad & \overline{A} \setminus \mathcal{I}(\overline{A}) \\
= \quad & \overline{A} \setminus \overline{\mathcal{C}(A)} \\
= \quad & \mathcal{C}(A) \setminus A \\
= \quad & \mathcal{B}^+(A)
\end{aligned}
$$

Equation 5:

$$
\begin{aligned}
& \mathcal{B}^+(A) \\
= \quad & \mathcal{C}(A) \setminus A \\
= \quad & [\, \mathcal{I}(A) \subseteq A \,] \\
& (\mathcal{C}(A) \setminus \mathcal{I}(A)) \setminus A \\
= \quad & \mathcal{B}(A) \setminus A \\
= \quad & \mathcal{B}(A) \cap \overline{A}
\end{aligned}
$$

Equation 6:

$$
\begin{aligned}
& \mathcal{B}^-(A) \\
= \quad & [\, \text{Statement 4} \,] \\
& \mathcal{B}^+(\overline{A}) \\
= \quad & [\, \text{Statement 5} \,] \\
& \mathcal{B}(\overline{A}) \cap A \\
= \quad & [\, \text{Statement 3} \,] \\
& \mathcal{B}(A) \cap A
\end{aligned}
$$

Equation 7:

$$
\begin{aligned}
& \mathcal{B}(A) \\
= \quad & \mathcal{C}(A) \setminus \mathcal{I}(A) \\
= \quad & \mathcal{C}(A) \cap \overline{\mathcal{I}(A)} \\
= \quad & \mathcal{C}(A) \cap \mathcal{C}(\overline{A})
\end{aligned}
$$

□

Proof. (of Proposition 3.3)

Axiom 1:

$$\mathcal{C}_R(\emptyset) = \emptyset \cup \{x \in X \mid \exists a \in \emptyset.(a, x) \in R\} = \emptyset$$

Axiom 2:

$$
\begin{aligned}
& A \\
\subseteq \quad & [\, A \subseteq A \cup B \,] \\
& \mathcal{C}_R(A)
\end{aligned}
$$

Axiom 3:

$$
\begin{aligned}
& \mathcal{C}_R(A \cup B) \\
= \quad & A \cup B \cup \{x \in X \mid \exists c \in A \cup B.(c,x) \in R\} \\
= \quad & [\, c \in A \cup B \iff c \in A \vee c \in B \,] \\
& A \cup B \cup \{x \in X \mid \exists c \in A.(c,x) \in R\} \cup \{x \in X \mid \exists c \in B.(c,x) \in R\} \\
= \quad & \mathcal{C}_R(A) \cup \mathcal{C}_R(B)
\end{aligned}
$$

□

Proof. (of Proposition 3.6)

Equation 8:

$$
\begin{aligned}
& \mathcal{I}(A) \\
= \quad & \overline{\mathcal{C}_R(\overline{A})} \\
= \quad & \overline{\overline{A} \cup \{x \in X \mid \exists a \in \overline{A}.(a,x) \in R\}} \\
= \quad & A \cap \{x \in X \mid \neg\exists a \in \overline{A}.(a,x) \in R\} \\
= \quad & \{x \in A \mid \neg\exists a \in \overline{A}.(a,x) \in R\}
\end{aligned}
$$

Equation 9:

$$
\begin{aligned}
& \mathcal{B}^-(A) \\
= \quad & A \setminus \mathcal{I}(A) \\
= \quad & A \setminus \{x \in A \mid \neg\exists a \in \overline{A}.(a,x) \in R\} \\
= \quad & A \cap \{x \in A \mid \exists a \in \overline{A}.(a,x) \in R\} \\
= \quad & \{x \in A \mid \exists a \in \overline{A}.(a,x) \in R\}
\end{aligned}
$$

Equation 10:

$$
\begin{aligned}
& \mathcal{B}^+(A) \\
= \quad & \mathcal{C}(A) \setminus A \\
= \quad & (A \cup \{x \in X \mid \exists a \in A.(a,x) \in R\}) \setminus A \\
= \quad & (A \cup \{x \in X \mid \exists a \in A.(a,x) \in R\}) \cap \overline{A} \\
= \quad & (A \cap \overline{A}) \cup (\{x \in X \mid \exists a \in A.(a,x) \in R\} \cap \overline{A}) \\
= \quad & \{x \in \overline{A} \mid \exists a \in A.(a,x) \in R\}
\end{aligned}
$$

□

Proof. (of Theorem 5.5) Let $\mathcal{M} = ((X, \mathcal{C}), \mathcal{V})$. Since $\mathcal{M}, x \models \phi \mathcal{U} \psi$, let A be the set from Definition 5.2. Let $p : x \rightsquigarrow \infty$, and l be such that $\mathcal{M}, p(l) \models \neg\phi$. Consider the set $K^- = \{k \mid \forall h \in \{0, \ldots, k\}.p(h) \in A\}$. Since $0 \in K^-$, we have $K^- \neq \emptyset$. Consider the complement of K^-, namely $K^+ = \mathbb{N} \setminus K^-$. Since all points in A satisfy ϕ, and $p(l) \models \neg\phi$, we have $l \in K^+$, thus $K^+ \neq \emptyset$. By finiteness of both sets, consider $k^- = \max K^-$ and $k^+ = \min K^+$. Noting that if $k \in K^-$ and $h \in [0, k)$, then $h \in K^-$, we have $k^- + 1 = k^+$, thus $(k^-, k^+) \in Succ$. Let $S = \{p(k) | k \in K^-\} \subseteq A$. By monotonicity of closure, we have $\mathcal{C}(S) \subseteq \mathcal{C}(A)$. By definition of $\mathcal{C}_{Succ}$, we have $k^+ \in \mathcal{C}_{Succ}(K^-)$, thus by continuity $p(k^+) \in \mathcal{C}(S)$ and therefore $p(k^+) \in \mathcal{C}(A)$. But it is also true that $p(k^+) \notin A$; if $p(k^+) \in A$, then we would have $k^+ \in K^-$, by definition of K^-. Thus, $p(k^+) \in \mathcal{B}^+(A)$, therefore $p(k^+) \models \psi$. Note that in particular $k^+ \neq 0$ as $p(0) = x \in A$, and $k^+ \leq l$ as $l \in K^+$ and $k^+ = \min K^+$. □

Proof. (of Lemma 5.6) For one direction of the proof, assume p is a continuous function. Importing definitions from Definition 5.4 and the statement of Lemma 5.6, we have

$$
\begin{aligned}
& (i, i+1) \in \mathit{Succ} \\
\implies\ & i+1 \in \mathcal{C}_{\mathit{Succ}}(\{i\}) \\
\implies\ & [\, p \text{ continuous} \,] \\
& p(i+1) \in \mathcal{C}_R(p(\{i\})) \\
\iff\ & p(i+1) \in \mathcal{C}_R(\{p(i)\}) \\
\iff\ & p(i+1) \in \{p(i)\} \cup \{x \mid (p(i), x) \in R\} \\
\iff\ & p(i+1) = p(i) \vee (p(i), p(i+1)) \in R
\end{aligned}
$$

For the other direction, given a path x_i of length l in R, define $p(i) = x_i$. Continuity of p is straightforward. □

Proof. (of Theorem 5.7) One direction is given by Theorem 5.5. For the other direction, assume $\mathcal{M} = ((X, \mathcal{C}_R), \mathcal{V})$ where $\mathcal{C}_R$ is the closure operator derived by a relation R. Consider point x with $\mathcal{M}, x \models \phi$, and assume that for each $p : x \rightsquigarrow \infty$ and l such that $\mathcal{M}, p(l) \models \neg\phi$ there is $k \in \{1, \ldots, l\}$ such that $\mathcal{M}, p(k) \models \psi$. Define the following set:

$$A_x = \{x\} \cup \{y \in X \mid \exists p : x \rightsquigarrow \infty . \exists l > 0 . p(l) = y \wedge \forall k \in \{1, \ldots, l\} . \mathcal{M}, p(k) \models \phi \wedge \neg\psi\}$$

We will use A_x as a witness of the existence of a set A in Definition 5.2, in order to prove that $\mathcal{M}, x \models \phi \mathcal{U} \psi$. Note that by definition of A_x, $x \in A_x$ and $\forall y \in A_x . \mathcal{M}, p(y) \models \phi$. We need to show that $\forall z \in \mathcal{B}^+(A_x) . \mathcal{M}, z \models \psi$. Consider $z \in \mathcal{B}^+(A_x)$. Since $\mathcal{M}$ is based on a quasi-discrete closure space, by Equation 10 in Proposition 3.6, we have $z \in \overline{A_x}$ and there is $y \in A_x$ such that $(y, z) \in R$. Suppose $y = x$. Let p be the path defined by $p(0) = x$, $p(i \neq 0) = z$. If $\mathcal{M}, z \models \phi$, suppose $\mathcal{M}, z \not\models \psi$; then $z \in A_x$, witnessed by the path p, with $l = 1$; therefore, since $z \in \overline{A_x}$ we have $\mathcal{M}, z \models \psi$. If $\mathcal{M}, z \not\models \phi$, then noting $p(1) = z$, by hypothesis, there is $k \in \{1, \ldots, 1\}$ with $\mathcal{M}, p(k) \models \psi$, that is $\mathcal{M}, z \models \psi$. Suppose $y \neq x$. Then there are $p : x \rightsquigarrow \infty$ and $l > 0$ such that $p(l) = y \wedge \forall k \in \{1, \ldots, l\} . \mathcal{M}, p(k) \models \phi \wedge \neg\psi$. Define p' by $p'(l') = p(l')$ if $l' \leq l$, and $p'(l') = z$ otherwise. The rest of the proof mimics the case $y = x$. If $\mathcal{M}, z \models \phi$, then $\mathcal{M}, z \not\models \psi$ implies $z \in A_x$, witnessed by p' and $l' = l + 1$, therefore $\mathcal{M}, z \models \psi$. If $\mathcal{M}, z \models \neg\phi$, then by hypothesis there must be $k \in \{1, \ldots, l+1\}$ such that $\mathcal{M}, p'(k) \models \psi$. By definition of p', it is not possible that $k \in \{1, \ldots, l\}$, thus $k = l + 1$ and $\mathcal{M}, z \models \psi$. By this argument, we have $\mathcal{M}, x \models \phi \mathcal{U} \psi$ using the set A_x to verify the definition of satisfaction. □

Proof. (of Remark 5.8)

1. $\mathcal{M}, x \models \phi \mathcal{R} \psi$

$$
\begin{aligned}
\iff\ & [\text{Definition of } \mathcal{R}] \\
& \mathcal{M}, x \models \neg(\neg\psi \mathcal{U} \neg\phi) \\
\iff\ & \mathcal{M}, x \not\models \neg\psi \mathcal{U} \neg\phi \\
\iff\ & [\text{Theorem 5.7}] \\
& \neg(\mathcal{M}, x \models \neg\psi \text{ and } \forall p : x \rightsquigarrow \infty \forall l \in \mathbb{N} : \mathcal{M}, p(l) \models \neg\neg\psi \Rightarrow \exists k \in \{1, \ldots, l\} : \mathcal{M}, p(k) \models \neg\phi) \\
\iff\ & \neg(\mathcal{M}, x \models \neg\psi \text{ and } \forall p : x \rightsquigarrow \infty \forall l \in \mathbb{N} :\ \neg(\mathcal{M}, p(l) \models \psi) \vee (\exists k \in \{1, \ldots, l\} : \mathcal{M}, p(k) \models \neg\phi)) \\
\iff\ & \mathcal{M}, x \models \psi \text{ or } \exists p : x \rightsquigarrow \infty \exists l \in \mathbb{N} :\ \mathcal{M}, p(l) \models \psi \wedge \neg(\exists k \in \{1, \ldots, l\} : \mathcal{M}, p(k) \models \neg\phi) \\
\iff\ & \exists p : x \rightsquigarrow \infty \exists l \in \mathbb{N} :\ \mathcal{M}, p(l) \models \psi \wedge \forall k \in \{1, \ldots, l\} : \mathcal{M}, p(k) \models \phi
\end{aligned}
$$

2. $\mathcal{M}, x \models \mathcal{G}\phi$

   $\iff$ [Definition of $\mathcal{G}$]

   $\mathcal{M}, x \models \phi \mathcal{U} \bot$

   $\iff$ [Theorem 5.7]

   $\forall p : x \rightsquigarrow \infty \forall l \in \mathbb{N} : \mathcal{M}, p(l) \models \neg\phi \Rightarrow \exists k \in \{1, \ldots, l\} : \mathcal{M}, p(k) \models \bot$

   $\iff \forall p : x \rightsquigarrow \infty \forall l \in \mathbb{N} : \mathcal{M}, p(l) \models \phi$

3. $\mathcal{M}, x \models \mathcal{F}\phi$

   $\iff$ [Definition of $\mathcal{F}$]

   $\mathcal{M}, x \models \neg\mathcal{G}\neg\phi$

   $\iff$ [Remark 5.8 (2)]

   $\neg(\forall p : x \rightsquigarrow \infty \forall l \in \mathbb{N} : \mathcal{M}, p(l) \models \neg\phi)$

   $\iff \exists p : x \rightsquigarrow \infty \exists l \in \mathbb{N} : \mathcal{M}, p(l) \models \phi$

□

Proof. Lemma 6.2 We prove by induction on the syntax of $SLCS$ formulae that for any quasi-discrete closure model $\mathcal{M} = ((X, \mathcal{C}_R), \mathcal{V})$, and for any formula Φ function `Sat` terminates in at most $\mathcal{O}(size(\Phi) \cdot (|X| + |R|))$ steps.

Base of Induction. If $\Phi = \top$ or $\Phi = p$ the statement follows directly from the definition of `Sat`. Indeed, in both these cases function `Sat` computes the final result in just 1 step.

Inductive Hypothesis. Let Φ_1 and Φ_2 be such that for any quasi-discrete closure model $\mathcal{M} = ((X, \mathcal{C}_R), \mathcal{V})$, function $\texttt{Sat}(\mathcal{M}, \Phi_i)$, $i = 1, 2$, terminate in at most $\mathcal{O}(size(\Phi_i) \cdot (|X| + |R|))$ steps.

Inductive Step.

$\Phi = \neg\Phi_1$**:** In this case function `Sat` first recursively computes the set $P = \texttt{Sat}(\mathcal{M}, \Phi_1)$, then returns $X - P$. By inductive hypothesis, the calculation of P terminates in at most $\mathcal{O}(size(\Phi_1) \cdot (|X| + |R|))$ steps, while to compute $X - P$ we need $\mathcal{O}(|X|)$ steps. Hence, $\texttt{Sat}(\mathcal{M}, \neg\Phi_1)$ terminates in at most $\mathcal{O}(size(\Phi_1) \cdot (|X| + |R|)) + \mathcal{O}(|X|)$. However:

$$\begin{array}{rl} & \mathcal{O}(size(\Phi_1) \cdot (|X| + |R|)) + \mathcal{O}(|X|) \\ \leq & \mathcal{O}(size(\Phi_1) \cdot (|X| + |R|)) + \mathcal{O}(|X| + |R|) \\ = & \mathcal{O}((1 + size(\Phi_1)) \cdot (|X| + |R|)) \\ = & \mathcal{O}(size(\neg\Phi_1) \cdot (|X| + |R|)) \end{array}$$

$\Phi = \Phi_1 \wedge \Phi_2$**:** To compute $P = \texttt{Sat}(\mathcal{M}, \Phi_1 \wedge \Phi_2)$ function `Sat` first computes $P = \texttt{Sat}(\mathcal{M}, \Phi_1)$ and $Q = \texttt{Sat}(\mathcal{M}, \Phi_2)$. Then the final result is obtained as $P \cap Q$. Like for the previous case, we have that the statement follows from inductive hypothesis and by using the fact that $P \cap Q$ can be computed in at most $\mathcal{O}(|X|)$.

$\Phi = \Diamond\Phi_1$**:** In this case function `Sat` first computes, in at most $\mathcal{O}(size(\Phi_1) \cdot (|X| + |R|))$ steps, the set $P = \texttt{Sat}(\mathcal{M}, \Phi_1)$. Then the final result is obtained as $\mathcal{C}_R(P)$. Note that, to compute $\mathcal{C}_R(P)$ one needs $\mathcal{O}(|X| + |R|)$ steps. According to Definition 3.1, $\mathcal{C}_R(P)$ is obtained as the union, computable in $\mathcal{O}(|X|)$ steps, of P with $\{x \in X | \exists a \in P.(a, x) \in R\}$. The latter can be computed

in $\mathcal{O}(|R|)$ steps. Indeed, we need to consider all the *edges* exiting from P. Hence, $\texttt{Sat}(\mathcal{M}, \Diamond\Phi_1)$ terminates in a number of steps that is:

$$\begin{array}{rl} & \mathcal{O}(size(\Phi_1)\cdot(|X|+|R|)) + \mathcal{O}(|X|) + \mathcal{O}(|R|) \\ = & \mathcal{O}(size(\Phi_1)\cdot(|X|+|R|)) + \mathcal{O}(|X|+|R|) \\ = & \mathcal{O}((1+size(\Phi_1))\cdot(|X|+|R|)) \\ = & \mathcal{O}(size(\Diamond\Phi_1)\cdot(|X|+|R|)) \end{array}$$

$\Phi = \Phi_1 \mathcal{U} \Phi_2$**:** When $\Phi = \Phi_1 \mathcal{U} \Phi_2$ function `Sat` recursively invokes function `CheckUntil` that first computes the sets $P = \texttt{Sat}(\mathcal{M}, \Phi_1)$, $Q = \texttt{Sat}(\mathcal{M}, \Phi_2)$ and $T = \mathcal{B}^+(P \cup Q)$. By inductive hypothesis, the computations of P and Q terminate in at most $\mathcal{O}(size(\Phi_1)\cdot(|X|+|R|))$ and $\mathcal{O}(size(\Phi_2)\cdot(|X|+|R|))$ steps, respectively, while T can be computed in $\mathcal{O}(|X|+|R|)$. After that, the loop at the end of function `CheckUntil` is executed. We can observe that:

- a point x is added to T only one time (i.e. if an element is removed from T, it is never reinserted in T);
- all the points in T are eventually removed from T;
- each *edge* in $\mathcal{M}$ is traversed at most one time.

The first two items, together with the fact that $\mathcal{M}$ is finite, guarantee that the loop terminates. The last item guarantees that the loop terminates in at most $\mathcal{O}(|R|)$ steps[7]. Summing up, the computation of $\texttt{Sat}(\mathcal{M}, \Phi_1 \mathcal{U} \Phi_2)$ terminates in at most

$$\begin{array}{rl} & \mathcal{O}(size(\Phi_1)\cdot(|X|+|R|)) + \mathcal{O}(size(\Phi_2)\cdot(|X|+|R|)) \\ & +\mathcal{O}(|X|+|R|) + \mathcal{O}(|R|) \\ = & \mathcal{O}((size(\Phi_1)+size(\Phi_2))\cdot(|X|+|R|)) + \mathcal{O}(|X|+|R|) \\ = & \mathcal{O}((1+size(\Phi_1)+size(\Phi_2))\cdot(|X|+|R|)) \\ = & \mathcal{O}(size(\Phi_1 \mathcal{U} \Phi_2)\cdot(|X|+|R|)) \end{array}$$

□

Proof. Theorem 6.3 The proof proceeds by induction on the syntax of $SLCS$ formulae.

Base of Induction. If $\Phi = \top$ or $\Phi = p$ the statement follows directly from the definition of function `Sat` and from Definition 5.2.

Inductive Hypothesis. Let Φ_1 and Φ_2 be such that for any finite quasi-discrete closure model $\mathcal{M} = ((X, \mathcal{C}_R), \mathcal{V})$, function $x \in \texttt{Sat}(\mathcal{M}, \Phi_i)$ if and only if $\mathcal{M}, x \models \Phi_i$, for $i = 1, 2$.

Inductive Step.

$\Phi = \neg\Phi_1$**:** $x \in \texttt{Sat}(\mathcal{M}, \neg\Phi_1)$

$\iff$ [Definition of `Sat`]

$x \notin \texttt{Sat}(\mathcal{M}, \Phi_1)$

$\iff$ [Inductive Hypothesis]

$\mathcal{M}, x \not\models \Phi_1$

$\iff$ [Definition 5.2]

$\mathcal{M}, x \models \neg\Phi_1$

[7]Note that this is the complexity for a DFS in a graph

$\Phi = \Phi_1 \wedge \Phi_2$**:** $x \in \texttt{Sat}(\mathcal{M}, \Phi_1 \wedge \Phi_2)$

$\iff$ [Definition of `Sat`]

$x \in \texttt{Sat}(\mathcal{M}, \Phi_1) \cap \texttt{Sat}(\mathcal{M}, \Phi_2)$

$\iff$ $x \in \texttt{Sat}(\mathcal{M}, \Phi_1)$ and $x \in \texttt{Sat}(\mathcal{M}, \Phi_2)$

$\iff$ [Inductive Hypothesis]

$\mathcal{M}, x \models \Phi_1$ and $\mathcal{M}, x \models \Phi_2$

$\iff$ [Definition 5.2]

$\mathcal{M}, x \models \Phi_1 \wedge \Phi_2$

$\Phi = \Diamond\Phi_1$**:** $x \in \texttt{Sat}(\Diamond\Phi_1)$

$\iff$ [Definition of `Sat`]

$x \in \mathcal{C}_R(\texttt{Sat}(\mathcal{M}, \Phi_1))$

$\iff$ [Definition of $\mathcal{C}_R$]

$\exists A \subseteq \texttt{Sat}(\mathcal{M}, \Phi_1) : x \in \mathcal{C}_R(A)$

$\iff$ [Inductive Hypothesis]

$\exists A \subseteq X. \forall y \in A. \mathcal{M}, y, \models \Phi_i$ and $x \in \mathcal{C}_R(A)$

$\iff$ [Definition 5.2]

$\mathcal{M}, x \models \Diamond\Phi_1$

$\Phi = \Phi_1 \mathcal{U} \Phi_2$**:** We prove that $x \in \texttt{CheckUntil}(\mathcal{M}, \Phi_1, \Phi_2)$ if and only if $\mathcal{M}, x \models \Phi_1 \mathcal{U} \Phi_2$. Function `CheckUntil` takes as parameters a model $\mathcal{M}$ and two $SLCS$ formulas Φ_1 and Φ_2 and computes the set of points in $\mathcal{M}$ satisfying $\Phi_1 \mathcal{U} \Phi_2$ by removing from $V = \texttt{Sat}(\mathcal{M}, \Phi_1)$ all the *bad* points.

A point is *bad* if it can *reach* a point satisfying $\neg\Phi_1$ without passing through a point satisfying Φ_2. Let $Q = \texttt{Sat}(\mathcal{M}, \Phi_2)$ be the set of points in $\mathcal{M}$ satisfying Φ_2. To identify the *bad* points in V the function `CheckUntil` performs a *backward search* from $T = \mathcal{B}^+(V \cup Q)$. Note that any *path exiting* from $V \cup Q$ has to pass through points in T. Moreover, the latter only contains points that satisfy neither Φ_1 nor Φ_2, by definition. Until T is empty, function `CheckUntil` first picks all the elements x in T and then removes from V the set of (bad) points N that are in $V - Q$ and that can reach x in *one step*. At the end of each iteration the set T contains the set of *bad* points discovered in the last iteration. The proof proceeds in two steps. The first step guarantees that if x does not satisfy $\Phi_1 \mathcal{U} \Phi_2$, then x is eventually removed from V. The second step shows that if x is removed from V then x does not satisfy $\Phi_1 \mathcal{U} \Phi_2$.

Note that, by Inductive Hypothesis, we have that:

$$x \in V = \texttt{Sat}(\mathcal{M}, \Phi_1) \Leftrightarrow \mathcal{M}, x \models \Phi_1 \tag{11}$$

$$x \in Q = \texttt{Sat}(\mathcal{M}, \Phi_2) \Leftrightarrow \mathcal{M}, x \models \Phi_2 \tag{12}$$

For each $x \in X$ we let:

$$\mathcal{I}_x = \{i \in \mathbb{N} | \exists p : x \rightsquigarrow \infty . \mathcal{M}, p[i] \models \neg\Phi_1 \wedge \forall j \in \{1, \ldots, i\} . \mathcal{M}, p[j] \models \neg\Phi_2\}$$

Note that, directly from Theorem 5.7, we have that $\mathcal{M}, x \models \Phi_1 \mathcal{U} \Phi_2$ if and only if $\mathcal{M}, x \models \Phi_1$ and $\mathcal{I}_x = \emptyset$.

First we prove that if $\mathcal{I}_x \neq \emptyset$ and $\mathcal{M}, x \models \Phi_1$, then x is removed from V at iteration $i = \min \mathcal{I}_x$. This guarantees that if x does not satisfy $\Phi_1 \mathcal{U} \Phi_2$, then x is eventually removed from V. The proof of this result proceeds by induction on i:

Base of Induction: Let $x \in X$ such that $\mathcal{M}, x \models \Phi_1$, $\mathcal{I}_x \neq \emptyset$ and $\min \mathcal{I}_x = 1$. Since $\min \mathcal{I}_x = 1$, we have that there exists $p : x \rightsquigarrow \infty$ such that $\mathcal{M}, p[1] \models \neg\Phi_1$ and $\mathcal{M}, p[1] \models \neg\Phi_2$. By definition of paths, we also have that $x = p[0]$ and $(x, p[1]) \in R$. This implies that $p[1] \in \mathcal{B}^+(V \cup Q)$ and $x \in pre(p[1])$. By definition of function `CheckUntil` we have that $p[1]$ is in T and x is removed from V during the first iteration. Note that x will be added to T only if it does not satisfy Φ_2 (i.e. if $x \notin Q$).

Inductive Hypothesis: For each $x \in X$ be such that $\mathcal{M}, x \models \Phi_1$, $\mathcal{I}_x \neq \emptyset$ and $\min \mathcal{I}_x = k$, x is removed from V at iteration k.

Inductive Step: Let $x \in X$ be such that $\mathcal{M}, x \models \Phi_1$, $\mathcal{I}_x \neq \emptyset$ and $\min \mathcal{I}_x = k + 1$. If $\min \mathcal{I}_x = k+1$ then there exists $p : x \rightsquigarrow \infty$ such that $\mathcal{M}, p[k+1] \models \neg\Phi_1$ and for each $j \in \{1, \ldots, k+1\}$ $\mathcal{M}, p[j] \models \neg\Phi_2$. We have also that $\mathcal{M}, p[1] \models \Phi_1$ (otherwise $\min \mathcal{I}_x = 1$) and $\min \mathcal{I}_{p[1]} = k$ (otherwise $\min \mathcal{I}_x \neq k + 1$). By inductive hypothesis we have that $p[1]$ is removed from V at iteration k. However, since $\mathcal{M}, p[1] \models \neg\Phi_2$ we have that $p[1] \notin Q$ and $p[1]$ is in the set T at the beginning of iteration $k + 1$. This implies that $x = p[0]$ is removed from V at iteration $k + 1$, since $x \in pre(p[1])$.

We now prove that if x is removed from V at iteration i, then $\mathcal{I}_x \neq \emptyset$ and $i = \min \mathcal{I}_x$. This ensures that if x is removed from V then x does not satisfy $\Phi_1 \mathcal{U} \Phi_2$. We proceed by induction on the number of iterations i:

Base of Induction: If $x \in V$ is removed in the first iteration we have that there exists a point $y \in \mathcal{B}^+(V \cup Q)$ such that $(x, y) \in R$. From Equation 11 and Equation 12 we have that $\mathcal{M}, x \models \Phi_1$ while $\mathcal{M}, y \models \neg\Phi_1 \wedge \neg\Phi_2$. This implies that there exists a path $p : x \rightsquigarrow \infty$ such that $p[1] = y$ and $1 = \min \mathcal{I}_x$.

Inductive Hypothesis: For each point $x \in V$, if x is removed from V at iteration $i \leq k$, then $\mathcal{I}_x \neq \emptyset$ and $i = \min \mathcal{I}_x$.

Inductive Step: Let $x \in V$ be removed at iteration $k + 1$. This implies that after k iterations, there exists a point y in T such that $(x, y) \in R$. This implies that y has been removed from V at iteration k and, by inductive hypothesis, $\mathcal{I}_y \neq \emptyset$ and $k = \min \mathcal{I}_y$. Hence, there exists a path $p : y \rightsquigarrow \infty$ such that $\mathcal{M}, p[k] \models \neg\Phi_1$ and for each $j \in \{1, \ldots, k\}$ $\mathcal{M}, p[j] \models \neg\Phi_2$. Moreover, since $y \in T$, we have also that $y \notin Q$ and, from Equation 12, $\mathcal{M}, y \models \neg\Phi_2$. We can consider the path $p' : x \rightsquigarrow \infty$ such that, for each j, $p'[0] = x$ and $p'[j + 1] = p[j]$. We have that $\mathcal{M}, p'[k + 1] \models \neg\Phi_1$ and for each $j \in \{1, \ldots, k + 1\}$, $\mathcal{M}, p'[j] \models \neg\Phi_2$. Hence $\mathcal{I}_x \neq \emptyset$ and $k+1 = \min \mathcal{I}_x$ (otherwise x should be removed from V in a previous iteration).

□