









Minimisation of Spatial Models Using Branching Bisimilarity

Vincenzo Ciancia¹ , Jan Friso Groote² , Diego Latella¹ ,
Mieke Massink¹  , and Erik P. de Vink² 

¹ CNR-ISTI, Pisa, Italy

{Vincenzo.Ciancia, Diego.Latella, Mieke.Massink}@cnr.it

² Eindhoven University of Technology, Eindhoven, The Netherlands
j.f.groote@TUE.nl, evink@win.tue.nl

Abstract. Spatial logic and spatial model checking have great potential for traditional computer science domains and beyond. Reasoning about space involves two different conditional reachability modalities: a forward reachability, similar to that used in temporal logic, and a backward modality representing that a point can be reached from another point, under certain conditions. Since spatial models can be huge, suitable model minimisation techniques are crucial for efficient model checking. An effective minimisation method for the recent notion of spatial Compatible Path (CoPa)-bisimilarity is proposed, and shown to be correct. The core of our method is the encoding of Closure Models as Labelled Transition Systems, enabling minimisation algorithms for branching bisimulation to compute CoPa equivalence classes. Initial validation via benchmark examples demonstrates a promising speed-up in model checking of spatial properties for models of realistic size.

Keywords: Spatial minimisation · Closure spaces · Spatial logics · Spatial bisimilarity · Branching bisimilarity · Spatial model checking

1 Introduction

Spatial and spatio-temporal model checking have recently been successfully employed in a variety of application areas, ranging from Collective Adaptive Systems [14, 20] to signals [30], images [5, 18, 25] and polyhedra [9], just to mention a few. These methods for spatial analysis are enjoying an increasing interest in computer science and beyond, also in unexpected domains such as medical imaging [6, 8]. Medical images are obtained from diagnostic instruments such as magnetic resonance images (MRI), computer tomography scans, positron emission tomography or dermoscopic images. Such images usually consist of millions of pixels, in 2D, or voxels (volumetric pixels) in 3D images.

Research partially funded by the Italian MUR Projects PRIN 2017FTXR7S, “IT- MaT-TerS”, PRIN 2020TL3X8X “T-LADIES”, and Next Generation EU - MUR Project PNRR PRI ECS00000017 “THE - Tuscany Health Ecosystem”. The authors are listed in alphabetical order; they contributed to this work equally.

Spatial model checking consists in the automatic verification of properties, expressed in a suitable spatial logic, on each point of a suitable spatial model. In [17] the Spatial Logic for Closure Spaces (SLCS) was introduced and further developed in [18]. Closure spaces, or Čech closure spaces [33], are a generalisation of topological spaces suitable to model many kinds of spatial objects, ranging from topological objects in continuous spaces, such as Euclidean spaces, to discrete spatial objects, such as general and regular graphs and adjacency spaces. The latter are particularly useful to represent images. Closure spaces (CS) and the sub-class of quasi-discrete closure spaces, QdCSs for short, form a convenient theoretical framework because of their generality and relative simplicity. A practical demonstration of this is the tool `VoxLogicA`, the recently developed spatial model checker [6–8] that can efficiently check SLCS properties of large images represented as *symmetric* quasi-discrete closure models—QdCMs, i.e. models with QdCSs as underlying spaces.

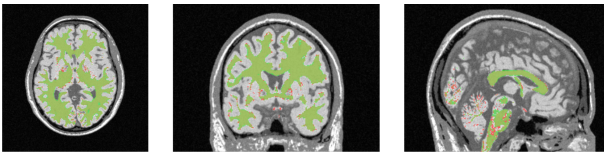


Fig. 1. Cross section of a dataset element of BrainWeb [4] pat04 MRI at slice $(x, y, z) = (129, 147, 78)$, (flTR: axial, coronal, sagittal view): `VoxLogicA` analysis of the segmentation of white matter, shown as a green overlay on top of a red overlay representing the ground truth.

For example, the 3D MRI image of a healthy brain shown in Fig. 1 consists of circa 12 M voxels (i.e. $256 \times 256 \times 181$) requiring approximately 10 s to analyse using `VoxLogicA` on a desktop computer [7].¹ Note that `VoxLogicA` checks such logical specifications for *every* point in the model exploiting parallel execution, memoization, and state-of-the-art imaging libraries [8].

A way to increase the time efficiency of spatial model checking is to exploit suitable model minimisation algorithms based on *spatial* bisimilarity. To that purpose several spatial bisimilarities have been proposed in [13]. In particular, CoPa-bisimilarity, based on a notion of “path-compatibility” is promising. The notion of path compatibility essentially requires that two paths, in order to be compatible, have to be both composed of a (non-empty) sequence of an equal number of non-empty adjacent “zones”, such that each point in one zone of one path must be related, by the bisimulation relation, to every point in the corresponding zone of the other path (see the illustration in Fig. 3b).

In [13], a logical characterisation of CoPa-bisimilarity has been given. More precisely, Infinitary Compatible Reachability Logic (ICRL) has been defined that is a modal logic with infinitary conjunction and two modalities, $\vec{\zeta}$ and $\overleftarrow{\zeta}$, expressing conditional forward and backward reachability, respectively. Given two ICRL

¹ Intel Core i9 9900K processor (with 8 cores) and 32 GB of RAM.

formulas Φ_1 and Φ_2 , a point x satisfies $\vec{\zeta}\Phi_1[\Phi_2]$ if it satisfies Φ_1 or there is a path from x to a point y along the path satisfying Φ_1 and all points from x (included) to y satisfy Φ_2 . Similarly for $\overleftarrow{\zeta}\Phi_1[\Phi_2]$, which is satisfied by x if it satisfies Φ_1 or there is a path *from* a point y satisfying Φ_1 to x and all points on the path *from* y to x (included) satisfy Φ_2 .²

This paper includes two original contributions, one of more theoretical nature and another more practical one.

Theoretical Contribution. Definition and correctness proof of an encoding of finite Closure Models (CM) in Labelled Transition Systems (LTS), that preserves CoPa-bisimilarity. More precisely, two points in the input CM are CoPa-bisimilar if and only if the states they are mapped onto by the encoding are branching bisimilar [23,24,26]. Thus, given a finite CM, the encoding makes it possible to effectively compute the minimal model with respect to CoPa-bisimilarity via the composition of the encoding and a very efficient minimisation algorithm for branching bisimulation, proposed in [24,26].

Practical Contribution. For a feasibility study and validation of the approach, we developed a prototype implementation of the encoding, and assembled a toolchain involving mCRL2 [10], VoxLogicA and GraphLogicA, a prototype spatial model checker. The latter is a variant of VoxLogicA handling general graphs. We applied our toolchain to a set of images at various resolutions, in order to gather insight on the potential gain in computational efficiency of spatial model checking. We observed a considerable speed-up, especially at larger resolutions, which suggests interesting directions for future research and applications.

Related Work. Qualitative reasoning about spatial entities [21] has been, and still is, a very active area of research in which the theory of topology and closure spaces play a important role. Prominent examples of that area are the region connection calculi, such as RCC8D. An embedding of the latter in the collective variant of SLCS was presented recently in [19]. Our work is also inspired by spatial logics (see [3] for an extensive overview), with seminal work dating back by Tarski and McKinsey in the forties of the previous century. The work on *spatial model checking* for logics with reachability originated in [18], which includes a comparison to the work of Aiello on spatial *until* operators (see e.g. [1]). In [2], Aiello envisaged practical applications of topological logics with *until* to minimisation of images. The present paper builds on and extends that vision. Bisimilarity for spatial logics with reachability is a relatively new subject. In [27], a bisimulation relation that is correct with respect to SLCS has been presented. Such definition has not yet been proved complete, and is aimed at characterising

² Note that, different from the context of classical temporal logics, in the context of space, and in particular when dealing with notions of directionality (e.g. one way roads, public area gates), it is important to be able to distinguish between the concept of “reaching” and that of “being reached”. The interested reader is referred to [13] for a discussion on the issue.

the logic including the *near* operator, therefore, not quotienting up-to reachability, as done in the present paper. The papers [9] and [28] introduce bisimulation relations that characterise spatial logics with reachability in polyhedral models and in simplicial complexes, respectively. It will be interesting future work to apply the minimisation techniques we present to such relevant classes of models.

In the Computer Science literature, other kinds of spatial logics have been proposed that typically describe situations in which modal operators are interpreted *syntactically* against the structure of agents in a process calculus. We refer to [11, 12] for some classical examples. Along the same lines, a recent example is given in [32], concerning model checking of security aspects in cyber-physical systems, in a spatial context based on the idea of bigraphical reactive systems introduced by Milner [29]. A bigraph consists of two graphs: A place graph, i.e. a forest defined over a set of nodes which is intended to represent entities and their locality in terms of a containment structure, and a link graph, a hypergraph composed over the same set of nodes representing arbitrary linking among those entities. The QdCS models that are the topic of the present paper, instead, address space from a topological point of view rather than as a containment structure for spatial entities.

The structure of the paper is as follows. Section 2 recalls relevant concepts and introduces notation. Section 3 recalls CoPa-bisimilarity for QdCMs. In Sect. 4 the encoding of finite QdCMs into LTSs is presented, together with the correctness results. Section 5 briefly describes a feasibility study of the application of the encoding and related toolchain to a series of examples. All detailed proofs can be found in [15].

2 Preliminaries

We first introduce some relevant concepts and notation, in particular recalling an LTS, branching bisimilarity [23, 24, 26], (quasi-discrete) closure spaces and closure models and paths therein.

Given a set X , $\mathcal{P}(X)$ denotes the powerset of X . For a function $f : X \rightarrow Y$, $A \subseteq X$ and $B \subseteq Y$, we let $f(A)$ and $f^{-1}(B)$ be defined as $\{f(a) \mid a \in A\}$ and $\{a \mid f(a) \in B\}$, respectively. For binary relation $R \subseteq X \times X$, we let R^{-1} denote the *converse* of R and $R^=$ denote the *reflexive* closure of R . The set of natural numbers is denoted by \mathbb{N} . For $n, m \in \mathbb{N}$ we often use the interval notation $[m, n]$ denoting the set $\{\iota \in \mathbb{N} \mid m \leq \iota \leq n\}$, $[m, n)$ denoting the set $\{\iota \in \mathbb{N} \mid m \leq \iota < n\}$, and similarly for $(m, n]$ and (m, n) .

In the sequel, branching bisimilarity [23, 24, 26] of states of LTSs plays a central role. Below we recall the relevant definitions.

Definition 1 (Labelled Transition System - LTS). A Labelled Transition System, *LTS for short*, is a tuple $(S, \text{Act}, \rightarrow)$ where S and Act are non-empty sets of states, and action labels respectively and relation $\rightarrow \subseteq S \times \text{Act} \times S$ is the transition relation.

As usual, we distinguish an action $\tau \in \mathbf{Act}$ that models a “silent move” of the LTS. Moreover, we call the elements of \rightarrow *transitions*, and we write $s \xrightarrow{\alpha} s'$ whenever $(s, \alpha, s') \in \rightarrow$. A (non-empty, finite) *trace* in the LTS is a sequence $s_0 \alpha_1 s_1 \dots s_{n-1} \alpha_n s_n$ of states and actions such that $n > 0$ and $s_{i-1} \xrightarrow{\alpha_i} s_i$ for $i = 1, \dots, n$. For such traces, we use the notation $s_0 \xrightarrow{\alpha_1} s_1 \dots s_{n-1} \xrightarrow{\alpha_n} s_n$. In such a situation, if $s = s_0$, $w = \alpha_1 \dots \alpha_n$, and $s' = s_n$, we have occasion to write $s \xrightarrow{w} s'$, as in the definition of branching bisimilarity which follows.

Definition 2 (Branching bisimilarity – \leftrightarrow_b). *Given an LTS $\mathcal{S} = (S, \mathbf{Act}, \rightarrow)$, a symmetric relation $B \subseteq S \times S$ is a branching bisimulation for \mathcal{S} iff, for $s, t, s' \in S$ and $\alpha \in \mathbf{Act}$, whenever $s B t$ and $s \xrightarrow{\alpha} s'$, it holds that: (i) $s' B t$ and $\alpha = \tau$, or (ii) $s B \bar{t}$, $s' B t'$ and $t \xrightarrow{\tau^*} \bar{t}$, $\bar{t} \xrightarrow{\alpha} t'$ for some $\bar{t}, t' \in S$.*

Two states $s, t \in S$ are called branching bisimilar in \mathcal{S} if $s B t$ for some branching bisimulation B for \mathcal{S} . Notation, $s \leftrightarrow_b^{\mathcal{S}} t$.

From now on, for readability, we omit the superscript \mathcal{S} in $\leftrightarrow_b^{\mathcal{S}}$, when this does not cause confusion. Our framework for modelling space is based on the notion of *Čech closure space* [33], CS for short, that provides a convenient common framework for the study of several different kinds of spatial models, including models of both discrete and continuous space [31]. We briefly recall definitions and results on CSs, that are relevant for this paper — most of which are borrowed from [22] (see also [13, 18]).

Definition 3 (Closure Space – CS). *A closure space is a pair (X, \mathcal{C}) where X is a set (of points) and $\mathcal{C} : \mathcal{P}(X) \rightarrow \mathcal{P}(X)$ is the closure operator, i.e. 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 $A_1, A_2 \subseteq X$.*

It is worth pointing out that CSs are a generalisation of topological spaces. In fact, the latter coincide with CSs that satisfy the *idempotence* axiom, i.e. $\mathcal{C}(\mathcal{C}(A)) = \mathcal{C}(A)$ for all $A \subseteq X$.

Definition 4 (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 a relation $R \subseteq X \times X$, define the function $\mathcal{C}_R : \mathcal{P}(X) \rightarrow \mathcal{P}(X)$ as follows: for all $A \subseteq X$, $\mathcal{C}_R(A) = A \cup \{x \in X \mid \exists a \in A \text{ s.t. } a R x\}$. It is easy to see that, for any R , \mathcal{C}_R satisfies all the axioms of Definition 3 and so (X, \mathcal{C}_R) is a CS. The following theorem is a standard result in the theory of CSs [22].

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 $\mathcal{C} = \mathcal{C}_R$.*

The above theorem implies that graphs coincide with QdCSs. We prefer to treat graphs as QdCSs since in this way we can formulate key definitions at the level of closure spaces leading to a uniform treatment for graphs and other kinds of models for space (e.g. topological spaces) [31]. Furthermore, if X is finite,

any closure space (X, \mathcal{C}) is quasi-discrete. In the sequel, we consider only *finite* CSs and often refrain from explicitly writing the subscript R in \mathcal{C}_R , when this does not cause confusion. Finally, we say that (X, \mathcal{C}_R) is *symmetric* iff R is a symmetric relation. An example of the result of applying the closure operator \mathcal{C} induced by a relation R to a set A is shown in Fig. 2.

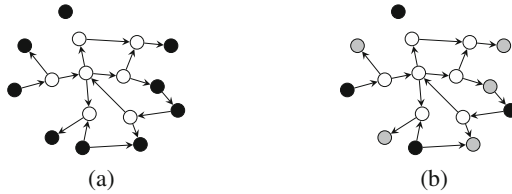


Fig. 2. a: a finite QdCS (X, \mathcal{C}) ; the arrows represent the relation underlying \mathcal{C} . The points of the set $A \subseteq X$ are shown in white, remaining points are shown in black. b: additional points in $\mathcal{C}(A)$ are shown in grey.

In the context of the present paper, *paths* over CSs play an important role. Following the tradition in topology, in the theory of CSs paths are defined as continuous functions from an appropriate index space to the CS at hand. For finite CSs, it is sufficient to consider bounded, finite, paths.

Definition 5 (Finite path). A finite path in a finite CS (X, \mathcal{C}) is a continuous function $\pi : [0, \ell] \rightarrow X$, for some $\ell \in \mathbb{N}$, such that $\pi(i + 1) \in \mathcal{C}(\{\pi(i)\})$ for $i = 0, \dots, \ell - 1$. We call ℓ the length of π and we denote it by $\mathbf{len}(\pi)$.

For $x \in X$, $\mathbf{FPaths}^F(x)$ denotes the set of all finite paths π in (X, \mathcal{C}) such that $\pi(0) = x$ (paths From x). Similarly, $\mathbf{FPaths}^T(x)$ denotes the set of all finite paths π in (X, \mathcal{C}) such that $\pi(\mathbf{len}(\pi)) = x$ (paths To x). In the sequel, whenever we write “path” we mean “finite path”.

Remark 1. It is worth pointing out that the notion of path in a QdCS is similar to that of a path in a graph or of a trace in an LTS, but it is *not* the same. In particular, due to axiom (ii) of closure operator \mathcal{C} and the requirement $\pi(i + 1) \in \mathcal{C}(\pi(i))$, paths in CSs allow *stuttering*; in other words, for QdCS (X, \mathcal{C}) , $x \in X$, and path π , it may happen that $\pi(i) = \pi(i + 1) = x$, for $i < \mathbf{len}(\pi)$ even when (x, x) is *not* an element of the relation $R \subseteq X \times X$ underlying \mathcal{C} . This is different for a path $\dots n_1 n_2 \dots$ in a graph (N, E) , where in order for nodes n_1 and n_2 in N to be adjacent, it is required that (n_1, n_2) is an element of the edge relation E . A similar issue arises when comparing paths in QdCSs with traces in LTSs. In fact, for LTS $(S, \mathbf{Act}, \rightarrow)$, two states s_1 and s_2 can be adjacent in a trace $\dots s_1 \xrightarrow{\alpha} s_2 \dots$ only if $(s_1, \alpha, s_2) \in \rightarrow$, and this holds also if $s_2 = s_1$.

We assume a set \mathbf{AP} of *atomic proposition letters* is given and introduce the notion of closure *model* (CM for short).

Definition 6 (Closure model – CM). A closure model is a tuple $\mathcal{M} = (X, \mathcal{C}, \mathcal{V})$, with (X, \mathcal{C}) a CS, and $\mathcal{V} : \text{AP} \rightarrow \mathcal{P}(X)$ the valuation function, assigning to each $p \in \text{AP}$ the set of points where p holds.

All definitions for CSs also apply to CMs; thus, a *quasi-discrete closure model* (QdCM for short) is a CM $\mathcal{M} = (X, \mathcal{C}, \mathcal{V})$ where (X, \mathcal{C}) is a QdCS. For a closure model $\mathcal{M} = (X, \mathcal{C}, \mathcal{V})$ we often write $x \in \mathcal{M}$ when $x \in X$. Similarly, we speak of paths in \mathcal{M} meaning paths in (X, \mathcal{C}) .

In the sequel, for a logic \mathcal{L} , a formula $\Phi \in \mathcal{L}$, and a model $\mathcal{M} = (X, \mathcal{C}, \mathcal{V})$ we let $\llbracket \Phi \rrbracket_{\mathcal{C}}^{\mathcal{M}}$ denote the set $\{x \in X \mid \mathcal{M}, x \models_{\mathcal{L}} \Phi\}$ of all the points in \mathcal{M} that satisfy Φ , where $\models_{\mathcal{L}}$ is the satisfaction relation for \mathcal{L} . For the sake of readability, we refrain from writing the subscript \mathcal{L} when this does not cause confusion.

3 CoPa-Bisimilarity for QdCM

In [13] several notions of spatial bisimilarity for closure models have been investigated. In particular, CM-bisimilarity, and its refinement for QdCMs CMC-bisimilarity, are a fundamental starting point for the study of spatial bisimilarity due to their strong links to topo-bisimilarity. On the other hand, they are rather fine-grained relations for reasoning about general properties of space, since they are based directly on the closure operator.³ For instance, with reference to the model of Fig. 3a, where all black points satisfy only atomic proposition b while the grey ones satisfy only g , the point at the center of the model is *not* CMC-bisimilar to any other black point. This is because CMC-bisimilarity is based on the fact that points reachable “in one step”—i.e. contained in the closure—are taken into consideration. This, in turn, gives bisimilarity a sort of “counting” power, that goes against the idea that, for instance, all black points in the model could be considered spatially equivalent. In fact, they are all black and all can reach black or grey points. Furthermore, they could be considered equivalent to the black point of a smaller model consisting of just one black and one grey point mutually connected—that would, in fact, be a minimal representation of the closure model.

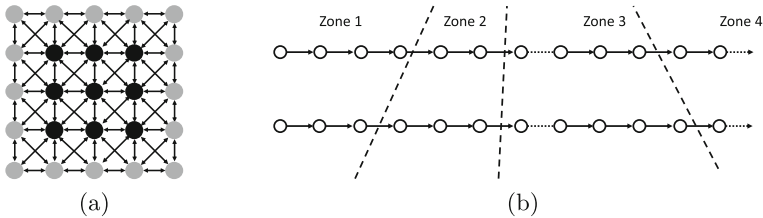


Fig. 3. A model (a); zones in paths (b).

In order to relax “counting” capability of bisimilarity as mentioned, a weaker notion of bisimulation has been introduced in [13] that is based on paths,

³ Or its dual operator called ‘interior’.

instead of single closure steps, and on a notion of “compatibility” between relevant paths that essentially requires each of them be composed of a non-empty sequence of non-empty, adjacent “zones”. More precisely, both paths under consideration in a transfer condition should share the same structure, as follows (see Fig. 3b):

- both paths are composed by a sequence of (non-empty) “zones”;
- the number of zones should be the same in both paths, *but*
- the length of “corresponding” zones can be different, *as well as* the length of the two paths;
- *each* point in one zone of a path should be related by the bisimulation to *every* point in the corresponding zone of the other path.

This notion of compatibility gives rise to *Compatible Path bisimulation*, CoPa-bisimulation, recalled below for QdCMs.

Definition 7 (CoPa-bisimilarity - $\equiv_{\text{CoPa}}^{\mathcal{M}}$). *Given QdCS $\mathcal{M} = (X, \mathcal{C}, \mathcal{V})$, a symmetric relation $B \subseteq X \times X$ is a CoPa-bisimulation for \mathcal{M} if, whenever $x_1 B x_2$, the following holds:*

1. *for all $p \in \text{AP}$ we have $x_1 \in \mathcal{V}(p)$ if and only if $x_2 \in \mathcal{V}(p)$;*
2. *for all $\pi_1 \in \text{FPaths}^{\text{F}}(x_1)$ such that $\pi_1(i_1) B x_2$ for all $i_1 \in [0, \text{len}(\pi_1))$ there is $\pi_2 \in \text{FPaths}^{\text{F}}(x_2)$ such that the following holds: $x_1 B \pi_2(i_2)$ for all $i_2 \in [0, \text{len}(\pi_2))$, and $\pi_1(\text{len}(\pi_1)) B \pi_2(\text{len}(\pi_2))$;*
3. *for all $\pi_1 \in \text{FPaths}^{\text{T}}(x_1)$ such that $\pi_1(i_1) B x_2$ for all $i_1 \in (0, \text{len}(\pi_1)]$ there is $\pi_2 \in \text{FPaths}^{\text{T}}(x_2)$ such that the following holds: $x_1 B \pi_2(i_2)$ for all $i_2 \in (0, \text{len}(\pi_2)]$, and $\pi_1(0) B \pi_2(0)$.*

Two points $x_1, x_2 \in X$ are called CoPa-bisimilar in \mathcal{M} if $x_1 B x_2$ for some CoPa-bisimulation B for \mathcal{M} . Notation, $x_1 \equiv_{\text{CoPa}}^{\mathcal{M}} x_2$.

It is easy to see that \equiv_{CMC} is strictly stronger than \equiv_{CoPa} ; the interested reader is referred to [13] for details.

We recall the definition of the *Infinitary Compatible Reachability Logic* (ICRL) proposed in [13] that provides a logical characterisation of CoPa-bisimilarity.

Definition 8 (Infinitary Compatible Reachability Logic - ICRL).

The abstract language of ICRL is defined by:

$$\Phi ::= p \mid \neg \Phi \mid \bigwedge_{i \in I} \Phi_i \mid \vec{\zeta} \Phi_1[\Phi_2] \mid \overleftarrow{\zeta} \Phi_1[\Phi_2]$$

where p ranges over AP and I ranges over a collection of index sets. The satisfaction relation for all QdCMs \mathcal{M} , points $x \in \mathcal{M}$, and ICRL formulas Φ is defined recursively on the structure of Φ as follows:

$$\begin{aligned} \mathcal{M}, x \models_{\text{ICRL}} p &\Leftrightarrow x \in \mathcal{V}(p) \\ \mathcal{M}, x \models_{\text{ICRL}} \neg \Phi &\Leftrightarrow \mathcal{M}, x \models_{\text{ICRL}} \Phi \text{ does not hold} \\ \mathcal{M}, x \models_{\text{ICRL}} \bigwedge_{i \in I} \Phi_i &\Leftrightarrow \mathcal{M}, x \models_{\text{IRL}} \Phi_i \text{ for all } i \in I \\ \mathcal{M}, x \models_{\text{ICRL}} \vec{\zeta} \Phi_1[\Phi_2] &\Leftrightarrow \text{path } \pi \text{ and index } \ell \text{ exist such that } \pi(0) = x, \\ &\quad \mathcal{M}, \pi(\ell) \models_{\text{ICRL}} \Phi_1, \text{ and } \mathcal{M}, \pi(j) \models_{\text{ICRL}} \Phi_2 \text{ for } j \in [0, \ell) \\ \mathcal{M}, x \models_{\text{ICRL}} \overleftarrow{\zeta} \Phi_1[\Phi_2] &\Leftrightarrow \text{path } \pi \text{ and index } \ell \text{ exist such that } \pi(\ell) = x, \\ &\quad \mathcal{M}, \pi(0) \models_{\text{ICRL}} \Phi_1, \text{ and } \mathcal{M}, \pi(j) \models_{\text{ICRL}} \Phi_2 \text{ for } j \in (0, \ell]. \end{aligned}$$

Logical equivalence with respect to ICRL is defined as expected.

Definition 9 (ICRL-equivalence - $\simeq_{\text{ICRL}}^{\mathcal{M}}$). For CM $\mathcal{M} = (X, \mathcal{C}, \mathcal{V})$, the equivalence relation $\simeq_{\text{ICRL}}^{\mathcal{M}} \subseteq X \times X$ is defined as: $x_1 \simeq_{\text{ICRL}}^{\mathcal{M}} x_2$ if and only if for all ICRL formulas Φ , it holds that $\mathcal{M}, x_1 \models_{\text{ICRL}} \Phi$ if and only if $\mathcal{M}, x_2 \models_{\text{ICRL}} \Phi$.

The following result establishes the relationship between CoPa-bisimilarity and ICRL-equivalence [13].

Theorem 2. For every QdCM \mathcal{M} it holds that ICRL-equivalence $\simeq_{\text{ICRL}}^{\mathcal{M}}$ coincides with CoPa-bisimilarity $\rightleftharpoons_{\text{CoPa}}^{\mathcal{M}}$.

In the remainder of the paper, since we are concerned with *finite* models only, we confine to the finitary fragment of ICRL, i.e. the fraction where I ranges over a collection of finite index sets. Furthermore, we will refrain from writing the superscript \mathcal{M} in $\text{q}\rightleftharpoons_{\text{CoPa}}^{\mathcal{M}}$ and $\simeq_{\text{ICRL}}^{\mathcal{M}}$, when this will not cause confusion.

In this work, given a QdCM $\mathcal{M} = (X, \mathcal{C}_R, \mathcal{V})$, we aim at running the model checking algorithm of [18] on the quotient of X with respect to \simeq_{ICRL} . The remainder of this paper is devoted to explain how to compute such set. It is a natural question at this point, whether the minimal model exists in the class of QdCMs. In other words, one needs to show that the set of equivalence classes of \simeq_{ICRL} can be endowed with a quasi-discrete closure operator, in such a way that logical truth is preserved and reflected. We do so in Proposition 1 below.

Proposition 1. Given a QdCM $\mathcal{M} = (X, \mathcal{C}_R, \mathcal{V})$, let X_{\min} be the set of equivalence classes of X modulo \simeq_{ICRL} . Let R' be the relation $\{(x, y) \in X \times X \mid y \in \mathcal{C}_R(\{x\})\}$. Let R_{\min} be the relation $\{(\alpha, \beta) \in X_{\min} \times X_{\min} \mid \exists x \in \alpha. \exists y \in \beta. (x, y) \in R'\}$. For each atomic proposition p , let $\mathcal{V}_{\min}(p) = \{\alpha \in X_{\min} \mid \exists x \in \alpha. x \in \mathcal{V}(p)\}$. Let $\mathcal{M}_{\min} = (X_{\min}, \mathcal{C}_{R_{\min}}, \mathcal{V}_{\min})$. Then for each x in X and for each formula Φ , we have $\mathcal{M}, x \models \Phi \iff \mathcal{M}_{\min}, [x] \models \Phi$, where $[x]$ is the equivalence class of x modulo \simeq_{ICRL} .

4 From QdCMs to Labelled Transition Systems

In this section we show how a finite QdCM can be encoded as an LTS in such a way that the images of points that are CoPa-bisimilar in the QdCM are mapped to branching bisimilar states in the LTS and viceversa. A simplification of the encoding is possible for the special case of QdCMs where the relation underlying the closure operator is symmetric.

4.1 General Encoding for Finite CMs

The encoding takes a finite QdCM as input and produces an LTS as output. To illustrate the various steps in the encoding, we use the QdCM in Fig. 4 and its LTS encoding in Fig. 5 as a running example. Let $\mathcal{M} = (X, \mathcal{C}, \mathcal{V})$ be a QdCM and R the binary relation on X that underlies the closure operator \mathcal{C} . The output

$\text{LTS}(\mathcal{M}) = (S, \text{Act}, \rightarrow)$ of the encoding of \mathcal{M} is an LTS where we can identify two parts, the *direct* part and the *converse* part. Roughly speaking, the direct part corresponds to R , whereas the converse part corresponds to the converse of R , i.e. R^{-1} . Both parts consist of the same number of states as the number of points in X .

More specifically, the set of states of the direct part is the set $\{\vec{x} \mid x \in X\} \subset S$, i.e. for each point in X there is a state in S in $\text{LTS}(\mathcal{M})$. We decorate it with an arrow from left to right to emphasise that it belongs to the direct part. Moreover, for all $x, x' \in X$, whenever $x' \in \mathcal{C}(\{x\})$ — i.e. $x R^= x'$ — and $x' \neq x$, there is a transition from \vec{x} to \vec{x}' in $\text{LTS}(\mathcal{M})$. In particular, if x and x' satisfy the *same* set of atomic proposition letters, i.e. $\mathcal{V}^{-1}(\{x\}) = \mathcal{V}^{-1}(\{x'\})$, then an internal transition $\vec{x} \xrightarrow{\tau} \vec{x}'$, is generated. If, instead, there is a change in the set of atomic proposition letters satisfied by x' with respect to those satisfied by x , then the transition $\vec{x} \xrightarrow{\text{ch}} \vec{x}'$ is generated, where **ch** signals such a **change**. In addition, for each $x \in X$, the actual proposition letters p satisfied by x are encoded as self-loops $\vec{x} \xrightarrow{p} \vec{x}$.

The set of states of the converse part is the set $\{\bar{x} \mid x \in X\} \subset S$ and the right-to-left arrows witness it. Moreover, for all $x, x' \in X$, whenever $x \in \mathcal{C}(\{x'\})$ — i.e. $x (R^=)^{-1} x'$ — and $x' \neq x$, there is a transition from \bar{x}' to \bar{x} in $\text{LTS}(\mathcal{M})$. For what concerns the labels of such transitions in the converse part the same rules apply as those for the direct part. The encoding of satisfaction of atomic propositions by self-loops does not need to be repeated in the converse part.

Finally, from each state \vec{x} in the direct part there is a transition, labelled by **cv**, leading to the corresponding state \bar{x} in the converse part, i.e. $\vec{x} \xrightarrow{\text{cv}} \bar{x}$ and, similarly, from each state \bar{x} in the converse part there is a transition, labelled by **dr**, leading to the corresponding state \vec{x} in the direct part, i.e. $\bar{x} \xrightarrow{\text{dr}} \vec{x}$. The translation is formalised in Definition 10.

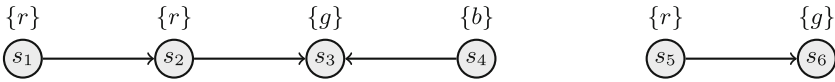


Fig. 4. A finite QdCM

Definition 10 (Encoding Finite CMs into LTSs). Let $\mathcal{M} = (X, \mathcal{C}, \mathcal{V})$ be a finite CM. Define labelled transition system $\text{LTS}(\mathcal{M})$ as follows. $\text{LTS}(\mathcal{M}) = (S, \text{Act}, \rightarrow)$ where: (i) $S = \{\vec{s} \mid s \in X\} \cup \{\bar{s} \mid s \in X\}$; (ii) $\text{Act} = \text{AP} \cup \{\tau, \text{dr}, \text{cv}, \text{ch}\}$, where $\{\tau, \text{dr}, \text{cv}, \text{ch}\} \cap \text{AP} = \emptyset$; (iii) the transition relation \rightarrow contains exactly the following transitions:

$$\begin{array}{ll}
 \vec{s} \xrightarrow{p} \vec{s} & \text{for all } p \in \text{AP} \text{ and } s \in \mathcal{V}(p) \\
 \vec{s} \xrightarrow{\text{cv}} \bar{s} & \text{for all } s \in X \\
 \bar{s} \xrightarrow{\text{dr}} \vec{s} & \text{for all } s \in X \\
 \vec{s} \xrightarrow{\tau} \vec{s}', \bar{s}' \xrightarrow{\tau} \bar{s} & \text{if } s' \in \mathcal{C}(\{s\}) \setminus \{s\} \text{ and } \mathcal{V}^{-1}(\{s\}) = \mathcal{V}^{-1}(\{s'\}) \\
 \vec{s} \xrightarrow{\text{ch}} \vec{s}', \bar{s}' \xrightarrow{\text{ch}} \bar{s} & \text{if } s' \in \mathcal{C}(\{s\}) \setminus \{s\} \text{ and } \mathcal{V}^{-1}(\{s\}) \neq \mathcal{V}^{-1}(\{s'\}).
 \end{array}$$

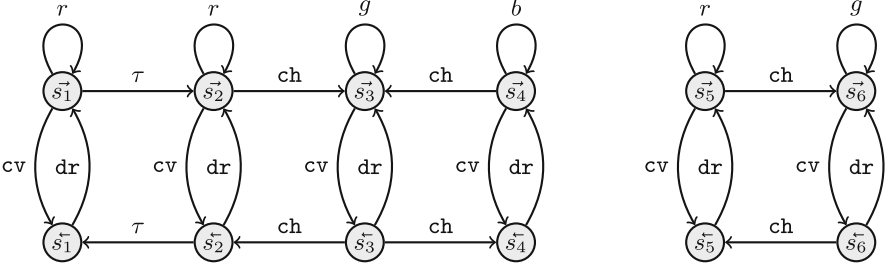


Fig. 5. LTS resulting from the application of the encoding defined in Definition 10 to the QdCM of Fig. 4.

The following lemma states an interesting property of the output of the encoding. Such a property turns out to be useful for the proof of the main result, asserting correctness of the encoding with respect to CoPa-bisimilarity.

Lemma 1. *Let $\mathcal{M} = (X, \mathcal{C}, \mathcal{V})$ be a finite CM. It holds that*

$$\vec{s} \leftrightarrow_b \vec{t} \text{ if and only if } \vec{s} \leftrightarrow_b \vec{t} \text{ for all } s, t \in X.$$

The proof of Lemma 1 builds on the following technical result.

Lemma 2. *Let $\mathcal{M} = (X, \mathcal{C}, \mathcal{V})$ be a finite CM. It holds that*

$$\text{if } \vec{s} \xrightarrow{\tau} \vec{s}' \text{ and } \vec{s} \leftrightarrow_b \vec{s}', \text{ then } \vec{s}' \xrightarrow{\tau} \vec{s} \text{ and } \vec{s} \leftrightarrow_b \vec{s}'.$$

The proof of Lemma 2 goes by induction on the *depth* of a direct state. In general, for an LTS, silent transitions $\xrightarrow{\tau^*}$ split the state space in τ -strongly connected components (τ -SCCs): states s and s' are in the same τ -SCC if both $s \xrightarrow{\tau^*} s'$ and $s' \xrightarrow{\tau^*} s$. Moreover, in the case of a finite LTS, τ -SCCs are well-ordered; a τ -SCC C is less than τ -SCC C' if $s' \xrightarrow{\tau^*} s$ for some $s' \in C'$, $s \in C$, but not the other way around. The depth of a state is then defined as the number of τ -SCCs a path of τ -transitions passes through to reach a so-called bottom τ -SCC.

Below we formulate the main theorem, showing that two points s and t in the QdCM are CoPa-bisimilar if and only if the corresponding states \vec{s} and \vec{t} are branching bisimilar, where the LTS is obtained by applying the encoding defined in Definition 10.

Theorem 3. *Let $\mathcal{M} = (X, \mathcal{C}, \mathcal{V})$ be a finite CM. Then, for all $s, t \in X$ we have*

$$s \rightleftharpoons_{\text{CoPa}} t \text{ if and only if } \vec{s} \leftrightarrow_b \vec{t}.$$

For a proof from left to right one defines a CoPa-bisimulation on \mathcal{M} obtained from branching bisimulation in $\text{LTS}(\mathcal{M})$: put $s B t$ if $\vec{s} \leftrightarrow_b \vec{t}$ for points $s, t \in X$. Lemma 2 is used to reduce the proof of obligation for requirement 3 of Definition 7 to the case of its requirement 2. A proof right to left is more direct; a branching bisimulation R defined by having $\vec{s} R \vec{t}$ and $\vec{s} R \vec{t}$ in case $s \rightleftharpoons_{\text{CoPa}} t$.

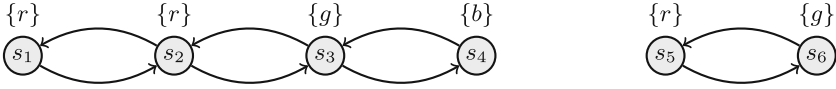


Fig. 6. A symmetric finite QdCM

4.2 Optimised Encoding for Symmetric Finite CMs

For symmetric finite CMs a simplified version of the encoding can be given. Symmetric QdCMs naturally emerge as representations of digital images where points are related via an adjacency relation as discussed in Sect. 1.

Definition 11 (Encoding Symmetric Finite CMs into LTSs). Let $\mathcal{M} = (X, \mathcal{C}, \mathcal{V})$ be a symmetric finite CM. Define $LTS_{sym}(\mathcal{M}) = (S, Act, \rightarrow)$ where: (i) $S = \{\vec{s} \mid s \in X\}$; (ii) $Act = AP \cup \{\tau, ch\}$; (iii) the transition relation \rightarrow contains exactly the following transitions:

$$\begin{aligned} \vec{s} &\xrightarrow{p} \vec{s} \text{ for all } p \in AP \text{ and } s \in \mathcal{V}(p) \\ \vec{s} &\xrightarrow{\tau} \vec{s}' \text{ if } s' \in \mathcal{C}(\{s\}) \setminus \{s\} \text{ and } \mathcal{V}^{-1}(\{s\}) = \mathcal{V}^{-1}(\{s'\}) \\ \vec{s} &\xrightarrow{ch} \vec{s}' \text{ if } s' \in \mathcal{C}(\{s\}) \text{ and } \mathcal{V}^{-1}(\{s\}) \neq \mathcal{V}^{-1}(\{s'\}). \end{aligned}$$

As an example, consider the symmetric finite QdCM of Fig. 6 and its LTS encoding in Fig. 7, obtained with the encoding given in Definition 11. It is easy to see that a symmetric QdCM with n nodes and t transitions leads to an LTS with n nodes and $t + n$ transitions.

Theorem 4. Let $\mathcal{M} = (X, \mathcal{C}, \mathcal{V})$ be a symmetric finite CM. Then, for all $s, t \in X$ we have: $s \rightleftharpoons_{CoPa} t$ if and only if $\vec{s} \rightleftharpoons_b \vec{t}$.

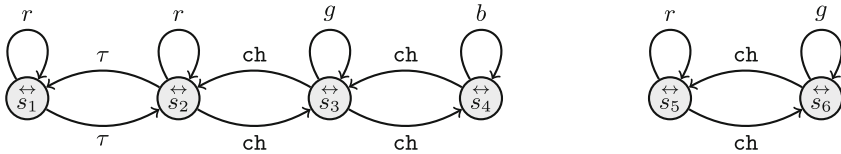


Fig. 7. LTS resulting from the application of the encoding in Definition 11 to the symmetric QdCM of Fig. 6.

The 2D maze in Fig. 9a, part of our feasibility study, exemplifies the significance of CoPa-minimization on images. Each node of the graph represents an area of interest in the image: exit (green), walls (black), walking areas (white) and starting points (blue). The three white nodes, as an example, represent three different kinds of white walking areas: the ones from which neither an exit nor a starting point can be reached (without crossing walls), the ones from which a starting point can be reached (but not the exit), and the ones from which a starting point and the exit can be reached.

5 Feasibility Study

In this section we provide an experimental validation of the theory presented in the previous sections. In particular, a prototype implementation of the encoding of Sect. 4 is introduced and applied to two representative benchmark examples.

5.1 Implementation

The encoding of Sect. 4.1 has been implemented as Free and Open Source Software, derived from the sources of the spatial model checker `GraphLogicA`, handling general finite QdCMs, in order to reuse its model loading functionality.

Procedure. The tool converts a spatial model — either an image (e.g. `png`), or a general graph written in a simple `json` format — to an LTS in the `aut` file format, which is one of the LTS formats accepted by the `mCRL2` tool suite [10]. For images, the optimised encoding described in Sect. 4.2 is used. The resulting LTS is minimised using the efficient branching bisimulation minimisation algorithm [24] implemented in `mCRL2`. This last step results in a minimised LTS in `aut` format which can be used for spatial model checking with `GraphLogicA`, after a simple conversion back to the `json` format. For measuring the model checking speed-up, in our toolchain, we use `GraphLogicA` for checking the minimal model, and `VoxLogicA` to check the full model⁴.



Fig. 8. Monoscope test pattern Philips PM5544

5.2 Experimental Setup

The procedure described above was used to produce the minimised LTSs shown in Fig. 9b, of the image of a maze of Fig. 9a, and to minimise the classical *Philips 5544* monoscope test pattern, shown in Fig. 8. Our tests have been run on a workstation equipped with an Intel Core™ i9 9900 K and 32 gb of RAM.

⁴ Note that `VoxLogicA` is inherently much faster than `GraphLogicA` as it is specialised for images, exploiting state-of-the-art imaging libraries and automatic parallelisation. This poses a further challenge to the speed-up via minimisation and is the reason why we use `VoxLogicA` instead of `GraphLogicA` for the full model.

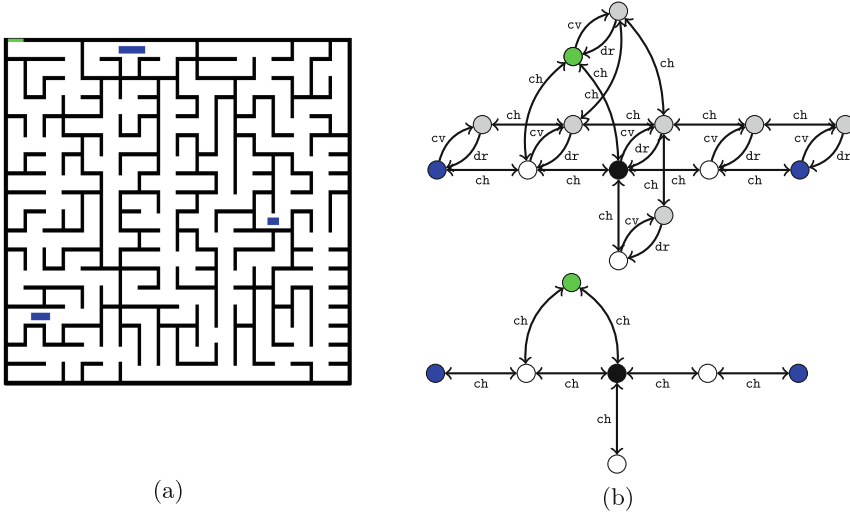


Fig. 9. An image of a 2D maze (9a), its minimal LTS using the general encoding of Definition 10 (9b - top), and that obtained using the optimised encoding of Definition 11 (9b - bottom). For readability, self-loops labelled by atomic propositions are not shown; the corresponding states are shown in the colour represented by the omitted label; symmetric transition pairs are drawn as doubly-headed arrows.

Full data, source code and tools needed to reproduce the experiments can be found in a Zenodo repository [16].

Test Images. For experimental evaluation, the two images have been rescaled at various resolutions. The “name” column of Table 1 indicates the vertical resolution of each image. The maze image is square, therefore the horizontal resolution coincides with the vertical one. The monoscope has a 16:9 ratio, thus, e.g., the horizontal resolution of mono-1080 is 1920 pixels.

Logical Specifications. For the maze image, the model checking specification consists of the computation of three reachability-based formulas, identifying: 1) the white points from which both a blue point and a green point can be reached (roughly, the white paths connecting blue points to the exit), via the formula $\zeta_{blue}[white] \wedge \zeta_{green}[white]$; 2) the blue points from which there is no white path to the exit (via a similar formula); 3) the blue points from which, instead, an exit can be reached (again, using reachability). For the monoscope pattern, the logical specification is more artificial, as it has been designed to be more demanding in terms of computation time (both model checkers have linear complexity in the number of sub-formulas). A single property Φ has been designed, characterising the points from which very specific paths start, crossing a number of different colours in a specific order, using 16 nested reachability

constraints, in the shape $\vec{\zeta}'(\vec{\zeta}'(\vec{\zeta}'(\dots)[green]))[cyan])[yellow]$, where $\vec{\zeta}'\phi_1[\phi_2]$ is defined as $\neg\phi_1 \wedge \vec{\zeta}'\phi_1[\phi_2]$.

5.3 Results and Discussion

Table 1 reports the results for each test image, for the logical properties specified in Sect. 5.2. Even though some models are equivalent (for instance, all the minimised versions of the maze), we have re-run all the phases of our experiment for each image, including model checking of the minimal model, as we do not test for equality of models for each pair of images in our experiment (which would yield a quadratic number of tests).

The obtained speed-up is noteworthy, especially for large images, as shown in the right-most column in Table 1. The longer formula used in the monoscope test demonstrates that minimisation clearly pays off when multiple formulas are checked on the same model, which is common in formal verification. For the larger images, the model checking time for the full model is substantially longer than the sum of the conversion, minimisation, backwards conversion, and model checking of the minimal model. For the maze example, the minimal model has the same size (actually, it is byte-by-byte the same file) for each resolution. The monoscope test, on the other hand, is designed to highlight artifacts in images. The original image is the one of 1080 pixel height and when downsampled at various resolutions, some lines disappear (specifically, belonging to the vertical bars close to the middle of the image), yielding different reachability properties, and therefore a more varied setup for our tests. We report the times both excluding and including input/output, for completeness. The intermediate file size for the `aut` files may be very large, thus saving and parsing times mask the effective computation. In perspective, the computation time is more relevant, as in the future intermediate files will be avoided altogether, by constructing `mCRL2` models directly in memory, using its programming interfaces.

As expected, with larger image sizes, the speed-up obtained in model checking becomes more prominent (again with reference to Table 1 the speed-up of model checking for the largest image is 22). Large images are particularly relevant in 3D medical imaging, which will be the subject of future work exploring the potential impact of bisimulation-based techniques to this novel application domain.

Ongoing work, also taking into account the results presented in [34], is devoted to translating spatial-logic properties to the language of `mCRL2` in order to use its state-of-the-art model checking techniques to verify spatial properties of directed graphs, in order to leverage the obtained speed-up even further.

Table 1. Results. All times are in seconds, rounded to two decimals. In order: conversion time from `png` to `aut`, without and with I/O; number of states, transitions, and `aut` file size of full model; minimisation time, without and with I/O; number of states and transitions of minimal model; time to convert the minimal model back to `json`; time for model checking the full model with `VoxLogicA`, and the minimal model with `GraphLogicA`; model checking speed-up.

Name	Conversion		Full model			Minimisation					Model checking		
	Time	t.w.IO	States	Transitions	Aut file size	Time	t.w.IO	States	Trans.	t.blck	ch. full	ch. min	Speedup
maze-128	0.34	0.34	16.00 K	142.50 K	2.47 MiB	0.00	0.03	7	21	0.36	0.49	0.39	1.25
maze-256	0.41	0.43	64.00 K	573.00 K	10.35 MiB	0.02	0.12	7	21	0.29	0.46	0.44	1.06
maze-512	0.34	0.78	256.00 K	2.24 M	44.55 MiB	0.08	0.49	7	21	0.30	0.45	0.47	0.97
maze-1024	0.39	1.28	1.00 M	8.99 M	184.34 MiB	0.32	2.06	7	21	0.31	0.51	0.38	1.34
maze-2048	0.46	4.12	4.00 M	35.98 M	793.73 MiB	1.31	8.10	7	21	0.34	0.82	0.41	1.98
maze-4096	0.87	21.91	16.00 M	143.95 M	3.27 GiB	5.37	33.32	7	21	0.33	1.81	0.45	4.01
maze-8192	2.20	173.55	64.00 M	575.91 M	13.63 GiB	21.53	135.45	7	21	0.29	5.34	0.42	12.77
mono-130	0.32	0.38	30.47 K	272.05 K	4.83 MiB	0.01	0.06	155	899	0.29	0.52	0.44	1.17
mono-260	0.31	0.62	121.88 K	1.07 M	20.27 MiB	0.03	0.26	315	1841	0.32	0.54	0.49	1.09
mono-540	0.35	0.90	506.25 K	4.44 M	90.33 MiB	0.16	1.01	460	2766	0.30	0.78	0.51	1.52
mono-1080	0.40	5.00	1.98 M	17.78 M	384.28 MiB	0.62	4.08	945	6965	0.32	1.57	0.57	2.75
mono-2160	0.57	43.87	7.91 M	71.16 M	1.55 GiB	2.45	16.74	945	6965	0.33	4.14	0.64	6.48
mono-4320	1.58	30.72	31.64 M	284.70 M	6.65 GiB	9.88	67.52	945	6965	0.72	14.96	0.65	22.87

6 Conclusions and Future Work

A practical minimisation method has been proposed for CoPa-bisimilarity for finite quasi-discrete closure models. The latter are a convenient theoretical foundation for spatial model checking. The method relies on an encoding of closure models onto LTSs such that an existing efficient algorithm for branching bisimilarity can be used to obtain a minimal model. The encoding has been proven correct, in the sense that two points in the CM are CoPa-bisimilar if and only if the states they are mapped into by the encoding are branching bisimilar. Spatial model checking can be performed exploiting the logical characterisation of CoPa-bisimilarity by the ICRL logic. A feasibility study has been performed to provide insight in the potential of the minimisation method for its use in the analysis of, possibly large, 2D images, in preparation of its envisioned use in spatial model checking in the medical domain. First results confirm that a very promising speed-up of spatial model checking can be obtained for single formulas, also for images of huge, but realistic, size. Minimisation also clearly pays off when multiple formulas are checked on the same model, which is common in formal verification. In such scenario, the model checking time for the full model is substantially longer than the sum of the conversion, minimisation, backwards conversion, and model checking of the minimal model, even in the current prototype setting.

Future work aims at further optimisations of the representations of the models, an integration of the toolchain and the visualisation of the results of checking the minimised model by mapping them back to the original image. The basic ingredients for such a mapping, i.e. the sets of states in the equivalence classes of the bisimulation, are readily available using the `mCRL2` tool suite [10].

Acknowledgements. We thank the anonymous reviewers for their valuable suggestions for improvement of this work.

References

1. Aiello, M.: Spatial Reasoning: Theory and Practice. Ph.D. thesis, Institute of Logic, Language and Computation, University of Amsterdam (2002)
2. Aiello, M.: The topo-approach to spatial representation and reasoning. *AIIA NOTIZIE* (4) (2003)
3. Aiello, M., Pratt-Hartmann, I., van Benthem, J. (eds.): *Handbook of Spatial Logics*. Springer, Berlin, Heidelberg (2007). <https://doi.org/10.1007/978-1-4020-5587-4>
4. Aubert-Broche, B., Griffin, M., Pike, G., Evans, A., Collins, D.: Twenty new digital brain phantoms for creation of validation image data bases. *IEEE Trans. Med. Imaging* **25**(11), 1410–1416 (2006). <https://doi.org/10.1109/TMI.2006.883453>
5. Banci Buonamici, F., Belmonte, G., Ciancia, V., Latella, D., Massink, M.: Spatial logics and model checking for medical imaging. *Int. J. Softw. Tools Technol. Transf.* **22**(2), 195–217 (2020). <https://doi.org/10.1007/s10009-019-00511-9>
6. 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.) *9th IEEE/ACM International Conference on Formal Methods in Software Engineering, FormaliSE@ICSE 2021, Madrid, Spain, 17–21 May 2021*, pp. 1–12. IEEE (2021). <https://doi.org/10.1109/FormaliSE52586.2021.00007>
7. 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, Cham (2019). https://doi.org/10.1007/978-3-030-30985-5_7
8. 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, Cham (2019). https://doi.org/10.1007/978-3-030-17462-0_16
9. Bezhanishvili, N., Ciancia, V., Gabelaia, D., Grilletti, G., Latella, D., Massink, M.: Geometric model checking of continuous space. *Log. Methods Comput. Sci.* **18**(4) (2022). (4:7)2022. <https://doi.org/10.46298/lmcs-18>, <https://lmcs.episciences.org/10348>
10. Bunte, O., Groote, J.F., Keiren, J.J.A., Laveaux, M., Neele, T., de Vink, E.P., Wesselink, W., Wijs, A., Willemse, T.A.C.: The mCRL2 toolset for analysing concurrent systems. In: Vojnar, T., Zhang, L. (eds.) *TACAS 2019*. LNCS, vol. 11428, pp. 21–39. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17465-1_2
11. Caires, L., Cardelli, L.: A spatial logic for concurrency (Part I). *Inf. Comput.* **186**(2), 194–235 (2003). [https://doi.org/10.1016/S0890-5401\(03\)00137-8](https://doi.org/10.1016/S0890-5401(03)00137-8)
12. Cardelli, L., Gordon, A.D.: Anytime, anywhere: modal logics for mobile ambients. In: Wegman, M.N., Reps, T.W. (eds.) *POPL 2000, Proceedings of the 27th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, Boston, Massachusetts, USA, 19–21 January 2000*, pp. 365–377. ACM (2000). <https://doi.org/10.1145/325694.325742>
13. Ciancia, V., Latella, D., Massink, M., de Vink, E.P.: Back-and-forth in space: on logics and bisimilarity in closure spaces. In: Jansen, N., Stoelinga, M., van den Bos, P. (eds.) *A Journey from Process Algebra via Timed Automata to Model Learning*. LNCS, vol. 13560, pp. 98–115. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-15629-8_6

14. Ciancia, V., Gilmore, S., Grilletti, G., Latella, D., Loretì, M., Massink, M.: Spatio-temporal model checking of vehicular movement in public transport systems. *Int. J. Softw. Tools Technol. Transf.* **20**(3), 289–311 (2018). <https://doi.org/10.1007/s10009-018-0483-8>
15. Ciancia, V., Groote, J.F., Latella, D., Massink, M., de Vink, E.P.: Minimisation of spatial models using branching bisimilarity (Extended Version) (2022). <https://doi.org/10.32079/ISTI-TR-2022/027>, CNR-ISTI Technical report TR-2022-027
16. Ciancia, V., Groote, J.F., Latella, D., Massink, M., de Vink, E.P.: Minimisation of Spatial Models using Branching Bisimilarity - Validation code and data (2022)
17. Ciancia, V., Latella, D., Loretì, M., Massink, M.: Specifying and verifying properties of space. In: Diaz, J., Lanese, I., Sangiorgi, D. (eds.) TCS 2014. LNCS, vol. 8705, pp. 222–235. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44602-7_18
18. Ciancia, V., Latella, D., Loretì, M., Massink, M.: Model checking spatial logics for closure spaces. *Log. Methods Comput. Sci.* **12**(4) (2016). [https://doi.org/10.2168/LMCS-12\(4:2\)2016](https://doi.org/10.2168/LMCS-12(4:2)2016)
19. Ciancia, V., Latella, D., Massink, M.: Embedding RCC8D in the collective spatial logic CSLCS. In: Boreale, M., Corradini, F., Loretì, M., Pugliese, R. (eds.) Models, Languages, and Tools for Concurrent and Distributed Programming. LNCS, vol. 11665, pp. 260–277. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-21485-2_15
20. Ciancia, V., Latella, D., Massink, M., Paškauskas, R., Vandin, A.: A tool-chain for statistical spatio-temporal model checking of bike sharing systems. In: Margaria, T., Steffen, B. (eds.) ISO/LA 2016. LNCS, vol. 9952, pp. 657–673. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-47166-2_46
21. Cohn, A.G., Renz, J.: Qualitative spatial representation and reasoning. In: van Harmelen, F., Lifschitz, V., Porter, B.W. (eds.) Handbook of Knowledge Representation, Foundations of Artificial Intelligence, vol. 3, pp. 551–596. Elsevier (2008). [https://doi.org/10.1016/S1574-6526\(07\)03013-1](https://doi.org/10.1016/S1574-6526(07)03013-1)
22. Galton, A.: A generalized topological view of motion in discrete space. *Theor. Comput. Sci.* **305**(1–3), 111–134 (2003). Elsevier. [https://doi.org/10.1016/S0304-3975\(02\)00701-6](https://doi.org/10.1016/S0304-3975(02)00701-6)
23. van Glabbeek, R.J., Weijland, W.P.: Branching time and abstraction in bisimulation semantics. *J. ACM* **43**(3), 555–600 (1996). <https://doi.org/10.1145/233551.233556>
24. Groote, J.F., Jansen, D.N., Keiren, J.J.A., Wijs, A.: An $O(m \log n)$ algorithm for computing stuttering equivalence and branching bisimulation. *ACM Trans. Comput. Log.* **18**(2), 13:1–13:34 (2017). <https://doi.org/10.1145/3060140>
25. Haghghi, 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.) Proceedings of the 18th International Conference on Hybrid Systems: Computation and Control, HSCC 2015, Seattle, WA, USA, 14–16 April 2015, pp. 189–198. ACM (2015). <https://doi.org/10.1145/2728606.2728633>
26. Jansen, D.N., Groote, J.F., Keiren, J.J.A., Wijs, A.: An $O(m \log n)$ algorithm for branching bisimilarity on labelled transition systems. In: TACAS 2020. LNCS, vol. 12079, pp. 3–20. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45237-7_1

27. Linker, S., Papacchini, F., Sevegnani, M.: Analysing spatial properties on neighbourhood spaces. In: Esparza, J., Král', D. (eds.) 45th International Symposium on Mathematical Foundations of Computer Science, MFCS 2020, 24–28 August 2020, Prague, Czech Republic. LIPIcs, vol. 170, pp. 66:1–66:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2020). <https://doi.org/10.4230/LIPIcs.MFCS.2020.66>
28. Loreti, M., Quadrini, M.: A spatial logic for a simplicial complex model. CoRR abs/2105.08708 (2021). <https://arxiv.org/abs/2105.08708>
29. Milner, R.: The Space and Motion of Communicating Agents. Cambridge University Press, Cambridge (2009)
30. Nenzi, L., Bortolussi, L., Ciancia, V., Loreti, M., Massink, M.: Qualitative and quantitative monitoring of spatio-temporal properties with SCTL. Log. Methods Comput. Sci. **14**(4) (2018). [https://doi.org/10.23638/LMCS-14\(4:2\)2018](https://doi.org/10.23638/LMCS-14(4:2)2018)
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, Dordrecht (2007). https://doi.org/10.1007/978-1-4020-5587-4_12
32. Tsigkanos, C., Pasquale, L., Ghezzi, C., Nuseibeh, B.: Ariadne: topology aware adaptive security for cyber-physical systems. In: Bertolino, A., Canfora, G., Elbaum, S.G. (eds.) 37th IEEE/ACM International Conference on Software Engineering, ICSE 2015, Florence, Italy, 16–24 May 2015, vol. 2, pp. 729–732. IEEE Computer Society (2015). <https://doi.org/10.1109/ICSE.2015.234>
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)
34. Zeven, F.: Spatial Model Checking with mCRL2. Master's thesis, Eindhoven University of Technology (2022)