COMPUTATIONAL PROPERTIES OF SPATIAL LOGICS IN THE REAL PLANE

A thesis submitted to the University of Manchester
for the degree of Doctor of Philosophy
in the Faculty of Engineering and Physical Sciences

2008

By
Aled Alun Griffiths
School of Computer Science

Contents

Word Count: 36,066

List of Tables

List of Figures

Abstract

Spatial logics are formal languages whose predicate and function symbols are interpreted as geometric relations and properties. In order to use these logics to perform automated reasoning on spatial data, we must have formal procedures which can decide the satisfiability of the formulae of these logics. However, *first order* spatial logics are typically undecidable and thus have no such formal procedure. By restricting the syntax of a spatial logic in certain ways, we can achieve languages which are decidable.

This thesis provides a new and consolidated survey of spatial logics and their complexity, and examines the effect of syntactic restriction on a particular family of spatial logics called topological constraint languages. The thesis also contributes two complexity results. The first is a considerably simplified proof of the NP membership of a spatial logic called RCC8. The second contribution is a new complexity result for the RCC8 language with the addition of a topological connectedness predicate.

Declaration

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright

i. The author of this thesis (including any appendices and/or schedules to this thesis) owns any copyright in it (the "Copyright") and s/he has given The University of Manchester the right to use such Copyright for any administrative, promotional, educational and/or teaching purposes.

ii. Copies of this thesis, either in full or in extracts, may be made only in accordance with the regulations of the John Rylands University Library of Manchester. Details of these regulations may be obtained from the Librarian. This page must form part of any such copies made.

iii. The ownership of any patents, designs, trade marks and any and all other intellectual property rights except for the Copyright (the "Intellectual Property Rights") and any reproductions of copyright works, for example graphs and tables ("Reproductions"), which may be described in this thesis, may not be owned by the author and may be owned by third parties. Such Intellectual Property Rights and Reproductions cannot and must not be made available for use without the prior written permission of the owner(s) of the relevant Intellectual Property Rights and/or Reproductions.

iv. Further information on the conditions under which disclosure, publication and exploitation of this thesis, the Copyright and any Intellectual Property Rights and/or Reproductions described in it may take place is available from the Head of School of School of Computer Science (or the Vice-President).

Acknowledgements

Firstly, I would like to thank my supervisor, Dr. Ian Pratt-Hartmann. Without his guidance and patience, this thesis would never have been completed.

For all their love and support, I must thank my parents. Without their encouragement, I would never have begun the PhD in the first place, and would certainly never have finished it - it is to them that I dedicate this thesis.

Thanks to Allan for all the illuminating discussions in the first half of my PhD, and thanks to Don for chess, chats, advice, and proofreading.

Finally, thanks to Manu, for everything, *ti amo*.

Chapter 1

Introduction

Space is a fundamental aspect of our perception of reality. For a human, it is trivial to perform inference with the spatial information that we receive through our senses. However, automating this process of inference, so that it may be performed by a computer is far from trivial. In order that we may automate such inferences, we must have formal languages with which we can describe spatial structures. Furthermore, given information expressed in such a language, we must have formal procedures which can tell us which conclusions can be inferred from the information.

Traditionally, the mathematical view of space is of a collection of points. Spatial relations and properties are said to hold between either the points themselves, or sets of the points - subsets of the space. We may impose a metric on this set of points, and refer to each point by a numerical coordinate. Traditional, or *quantitative* spatial representation involves storing the coordinates of a set of objects, and performing inference then involves drawing inferences about the spatial relationships from this raw numerical data.

However, there are problems associated with this quantitative approach to space. Firstly, we have a difficult issue of whether points physically exist. We cannot physically produce a point as evidence of their existence, and space is certainly not perceived as a collection of points. We must at least concede that the notion of space as points is mathematically convenient, but nevertheless an abstraction from what is perceived. We also have more practical problems regarding the quantitative approach. Performing inference on data consisting of lists of numerical coordinates can be very difficult. Although numerical coordinates are an ideal notation for determining things such as which of two object is closest to a third object, or what the distance between two objects is, they are quite far removed from the actual structure of a space, and are quite unsuitable for some kinds of questions. Determining whether one object contains another, or whether the surfaces of two objects intersect can require many numerical calculations, and if this process of inference is automated, answering these questions could be computationally expensive.

The alternative to quantitative spatial representation is *qualitative* spatial representation. Instead of points we take subsets, or *regions*, of a space as the primitive entities. Thus we represent spatial information by recording the qualitative spatial relations which hold between these regions. We are concerned solely with the qualitative relations and properties that belong to the field of mereotopology. Mereotopology is the name given to the study of relations and structures from two areas of mathematics, Mereology and Topology. Mereology is the study of part-whole relationships. Topology is the study of geometric properties which are preserved under continuous change.

A *spatial logic* is a formal language whose formulae are interpreted over a class of geometric structure. We will use the term spatial logic to mean a qualitative spatial logic, that is, variables are interpreted over elements corresponding to the regions of a space, and predicate and function symbols are interpreted as mereotopological relations and properties. Because it is already used in the literature of spatial logics, we use the term *topological inference* to describe the process of logical inference when restricted to spatial logics.

One of the problems central to performing inference with formulae of a spatial logic, is that of deciding the *satisfiability* of a formula. We say that a formula is satisfiable, if in the class over which the language is interpreted, there exists a structure in which the formula is true. For a given spatial logic, we call the problem of deciding the satisfiability of a formula the *satisfiability problem* of that spatial logic. If there is a formal procedure for deciding the satisfiability problem of a spatial logic, then we say that the logic is *decidable*, otherwise we call the logic *undecidable*.

In order to guarantee that we can perform topological inference with a spatial logic, the language must be decidable. Furthermore, the formal procedures for deciding the satisfiability of the formulae of these spatial logics must be of low complexity that is, they must execute in a reasonable time. One way to achieve spatial logics of lower complexity is to place limitations on the syntax of our language which reduce the range of formulae we are able to express. The main challenge in producing practical spatial logics is to strike a balance between having a language which can express all the things we need, while still having a decision procedure which is not too computationally complex. This thesis examines the effect of certain restrictions and expansions to the syntax of spatial logics on the complexity of those logics. We examine in detail a family of spatial logics called topological constraint languages, and pay close attention to the methods by which their satisfiability problems are solved.

If our aim is to be able to perform automated spatial reasoning which behaves at all like human reasoning, then we may wish to have the ability restrict our definition of regions to subsets of Euclidean spaces, as opposed to abstract many-dimensional spaces, which may bear no relation to the physical world. For this reason, this thesis looks at the general problem of topological inference, but with a special emphasis on topological inference over the two dimensional Euclidean plane.

This thesis aims to provide a new and consolidated survey of spatial logics focusing on the class of logics called topological constraint languages. In particular, we examine the effect that syntactic restriction has on the complexity of these logics. We also contribute new complexity results for two of these topological constraint languages. The first result is a simplified complexity proof of the NP membership of the spatial logic RCC8. This proof is completely different from the existing ones, and allows a considerably simpler proof to be made. The RCC8 result is shown by proving that a restricted fragment of RCC8 is decidable in NLOGSPACE. We can also give simpler proofs of a number of side results, in particular involving the applicability of using methods from relation algebras to solve the satisfiability problem of RCC8. The second contribution is a completely new complexity result for the RCC8 language with the addition of a connectedness operator. By using results from a solution to the string graph problem (see Chapter 3) we show that this language is in NP.

The structure of this thesis is as follows. Firstly, Chapter 2 contains basic mathematical preliminaries, and also serves the purpose of fixing the mathematical notation used throughout the rest of the thesis. In Chapter 3 we introduce the graph theoretic notion of the *string graph problem*, and we provide an outline of the proof of its solution. We introduce the topic of spatial logic in Chapter 4, and formally define the languages and models which we investigate over the following chapters. Chapter 5 introduces a range of spatial logics, showing in detail how they are related to one another in terms of syntactic restrictions, and examining effect these restrictions have on the complexity of the logics. Finally, Chapter 6 investigates the effect of restricting the interpretation of a number of topological constraint languages to regions of the Euclidean plane.

In terms of the contribution of this thesis, Chapters 2 and 3 simply provide background material, while Chapters 4 and 5 provide the consolidated survey of spatial logics, and Chapter 6 provides both the simplified RCC8 complexity result, and the RCC8 with connectedness complexity result.

Chapter 2

Preliminaries

This chapter introduces some of the basic concepts that will be used throughout the rest of the thesis. This will involve the topics of Modal Logic, Topology, Boolean Algebra, and Computational Complexity. This chapter can be skipped entirely, if the reader is already familiar with these topics. It also provides a description of the notational conventions used throughout the thesis, and for this purpose, the chapter may be useful as a reference.

2.1 Modal Logic

A very comprehensive exploration of modal logics can be found in [BdRV01]. Syntactically, we can view modal logics as a propositional logic with an additional operator.

Modal logics were first proposed as a formal way of representing systems involving ideas of *necessity* and *possibility*. However, depending on the interpretation of the modal operator, the logics may be given other semantics. They can, for example, represent relations of *knowledge* and *belief*.

There is a simple, yet powerful, semantics for modal logics, called *relational semantics*. Kripke proved that a certain class of relational structures is complete with respect to modal logic [Kri59], thus these semantics are often called *Kripke* semantics.

In propositional logic, the value of a variable is either true or false. The value depends on the definition of the valuation function of an interpretation. Modal logic allows us to describe a structure where the value of a variable depends on the circumstances in which the variable is encountered. We call this kind of structure a *Kripke frame*.

Definition 2.1.1. A *Kripke frame* is a pair $\langle W, \rightarrowtail \rangle$ where W is a non-empty set, and $\rightarrowtail$ is a binary relation, $\rightarrowtail \subseteq W \times W$. Elements of W are known as *worlds* or *nodes*. The relation $\rightarrowtail$ is known as the *accessibility* or *reachability* relation. If $w \rightarrowtail w'$ then we say that world w' is accessible from w.

Definition 2.1.2. Given a Kripke frame $\langle W, \rightarrowtail \rangle$ a *valuation function* is a map

$$\eta : \mathcal{P} \to \mathbb{P}(W)$$

which assigns to each propositional variable $p \in \mathcal{P}$ those worlds in which p is true. We say that a propositional variable p *holds* in a world w if $w \in \eta(p)$.

Definition 2.1.3. A *Kripke model* is a triple $\langle W, \rightarrowtail, \eta \rangle$ where $\langle W, \rightarrowtail \rangle$ is a Kripke frame and η is a valuation function on that frame.

The value of a variable depends, therefore, on both the valuation function of the Kripke model, and on whichever world of that model we choose to interpret the variable in.

As mentioned before, modal logic is syntactically very similar to propositional logic, but with the addition of a unary operator. We call this operator the *modal* operator and write it as $\Diamond$. We will use the symbol $\Box$ as shorthand for $\neg\Diamond\neg$.

A modal formula is a finite sequence of symbols, built with the following rules: any propositional variable p_i is a formula, if φ and ψ are formulae, then so are $\neg\varphi$, $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$, $(\varphi \to \psi)$, $\Box\varphi$, and $\Diamond\varphi$.

The symbols $\wedge$, $\vee$, $\neg$, and $\to$ have the same meanings as they do in propositional logic. The symbols $\Box$ and $\Diamond$ traditionally stand for, respectively, "necessity" and "possibility". However, we will now formally define their interpretations within a Kripke model.

Definition 2.1.4. Let $\mathfrak{M} = \langle W, \rightarrowtail, \eta \rangle$ be a Kripke model. Given a world $w \in W$, we define the *satisfaction relation*, $\models$, as follows:

$$\begin{array}{lll}
\mathfrak{M} \models_w p & \text{iff} & w \in \eta(p) \\
\mathfrak{M} \models_w \neg\varphi & \text{iff} & \mathfrak{M} \not\models_w \varphi \\
\mathfrak{M} \models_w \varphi \vee \psi & \text{iff} & \mathfrak{M} \models_w \varphi \ \text{ or } \ \mathfrak{M} \models_w \psi \\
\mathfrak{M} \models_w \Diamond\varphi & \text{iff} & \text{there exists } w' \in W \text{ such that } w \rightarrowtail w' \text{ and } \mathfrak{M} \models_{w'} \varphi
\end{array}$$

We say that φ is *satisfiable in* $\mathfrak{M}$ if and only if there exists a $w \in W$ such that $\mathfrak{M} \models_w \varphi$. If no such world in the model exists, then φ is said to be *unsatisfiable in* $\mathfrak{M}$. If every world satisfies φ, then φ is said to be *valid in* $\mathfrak{M}$.

Furthermore, given a modal formula, φ, we say that φ is *satisfiable* if there exists a model $\mathfrak{M}$ such that φ is satisfiable in $\mathfrak{M}$. If there is no such model, then φ is *unsatisfiable*.

The standard modal logic is known as K, after Kripke, and is characterised by the set of all Kripke frames. In these frames, the $\rightarrowtail$ relation obeys the following axiom, where φ, ψ are modal formulae.

K: $\Box(\varphi \rightarrow \psi) \rightarrow (\Box\varphi \rightarrow \Box\psi)$

We can restrict the class of Kripke frames which characterise a modal logic with the following axioms.

T: $\Box\varphi \rightarrow \varphi$

4: $\Box\varphi \rightarrow \Box\Box\varphi$

5: $\Diamond\varphi \rightarrow \Box\Diamond\varphi$

The T axiom ensures that the $\rightarrowtail$ relation is reflexive. The 4 axiom ensures that the $\rightarrowtail$ relation is transitive. The 5 axiom ensures that the $\rightarrowtail$ relation is Euclidean. There are of course many other axioms which can restrict the class of Kripke frames, however these are the only ones we are interested in.

We refer to a modal logic by the axioms which characterise the logic, for example $KT4$ is axiomatised by the axioms K, T & 4. We are only interested in the logics $KT4$ and $KT5$, or as they are usually called, $S4$ and $S5$.

2.1.1 Universal S4

We can increase the expressiveness of $S4$ by adding two additional modal operators, written $\exists$ and $\forall$. The interpretation of these operators is given below (extending Definition 2.1.4).

$$
\begin{aligned}
\mathfrak{M} \models_w \forall\varphi \quad &\text{iff} \quad \text{for every } w' \in W,\ \mathfrak{M} \models_{w'} \varphi,\\
\mathfrak{M} \models_w \exists\varphi \quad &\text{iff} \quad \text{there exists a } w' \in W \text{ such that } \mathfrak{M} \models_{w'} \varphi.
\end{aligned}
$$

We call this extension of $S4$, *universal* $S4$, or $S4_U$. Strictly speaking, $S4_U$ is a bimodal logic - a hybrid of $S4$ and $S5$. An $S4_U$ frame consists of two reachability relations, one for the $S4$ model operator, and the other for the $S5$ modal operator. The $S5$ modal operator normally splits the worlds of a frame into equivalence classes, however we use a single class to encompass all the worlds of the frame. Therefore we will simply view $S4_U$ as $S4$ with the addition of *first-order*-like quantifiers.

2.2 Topology

The reader is assumed to have some familiarity with topology, as the following definitions are mainly for the purpose of fixing notational standards. If a slower introduction to topology is needed, both [New64] and [Kah75] provide a good introduction, whereas a more advanced treatment can be found in [Kel60]. Topology is a branch of mathematics that involves the study of geometric relations that are preserved through continuous transformation, such as stretching or warping.

Topological Space

Definition 2.2.1. Given a set X, we can define a *topological space* on this set as a pair $\langle X,\mathcal{U}\rangle$, where $\mathcal{U} \subseteq \mathbb{P}(X)$ such that the following hold.

1. $\varnothing \in \mathcal{U}$ and $X \in \mathcal{U}$.
2. If $u_1, u_2, \ldots, u_n \in \mathcal{U}$ then $u_1 \cap u_2 \cap \cdots \cap u_n \in \mathcal{U}$.
3. If for any $\mathcal{U}' \subseteq \mathcal{U}$, then $\bigcup_{u\in\mathcal{U}'} u \in \mathcal{U}$.

This set of open sets $\mathcal{U}$ is called the *topology* on X.

Let $\langle X,\mathcal{U}\rangle$ be a topological space.

Definition 2.2.2. Let $p \in X$, $v \subseteq X$, and $u \in \mathcal{U}$. A *neighbourhood* of p is a set v which contains an open set u containing p.

Definition 2.2.3. A *basis* for X is a set $B \subseteq \mathcal{U}$ such that every open set in $\mathcal{U}$ is a union of elements of B.

Definition 2.2.4. We say that X is T_1 if and only if, for all $p, p' \in X$, there exist $u, u' \in \mathcal{U}$ such that $p \in u$, $p' \in u'$, $p \notin u'$, and $p' \notin u$.

Definition 2.2.5. We say that X is T_2 or *Hausdorff* if and only if, for any $p, p' \in X$ (where $p \neq p'$) there is a neighbourhood v of p and a neighbourhood v' of p' such that $v \cap v' = \varnothing$.

Definition 2.2.6. We say that X is an *Alexandroff* space if and only if for any $\mathcal{U}' \subseteq \mathcal{U}$, then $\bigcap_{u \in \mathcal{U}'} u \in \mathcal{U}$.

Definition 2.2.7. A *cover* of a set $v \subseteq X$ is a collection of subsets of X whose union contains v. A cover is open if each of its elements is open.

Definition 2.2.8. We say that $v \subseteq X$ is *compact* if for every open cover of v there is a finite subcover of v.

Definition 2.2.9. Let $v \subseteq X$. We define the following set.

$$\mathcal{U}_v = \{\, v \cap u \mid u \in \mathcal{U} \,\}$$

This set, $\mathcal{U}_v$, fulfills all three conditions on a topological space. Therefore $\mathcal{U}_v$ is a topology on v. We call $\langle v, \mathcal{U}_v \rangle$ a *relative* or *subspace* topology for the subset v of the space X.

Interior, Closure, Boundary and Complement

If $\langle X, \mathcal{U} \rangle$ is a topological space and $v \subseteq X$, then:

Definition 2.2.10. The *complement* of v is $X \setminus v$ which we write as $-v$.

Definition 2.2.11. The *interior* of v is the largest open set contained in v, which we write as v°.

Definition 2.2.12. The *closure* of v is the smallest closed subset of X, or member of the set of complements of $\mathcal{U}$, containing v, which we write as v^-. From this definition, we can see that v^- is expressible in terms of the interior of v.

$$v^- = -((-v)^\circ)$$

Definition 2.2.13. The *boundary* of v, written v^∂, is defined as follows.

$$v^\partial = v^- \cap -(v^\circ)$$

Definition 2.2.14. The set v is *regular open* if and only if $v = ((v)^-)^\circ$ and is *regular closed* if and only if $v = ((v)^\circ)^-$. We denote the set of regular open subsets of X by $RO(X)$, and the set of regular closed subsets of X by $RC(X)$.

Definition 2.2.15. We say that X is *semi-regular* if it has a basis of regular open sets. We say that X is *weakly regular* if it is semi-regular and, for any non-empty open set $u \in \mathcal{U}$ there exists a non-empty open set u' such that $(u')^- \subseteq v$.

Definition 2.2.16. We call the largest open subset of X which is disjoint from v the *pseudocomplement* of v, which we write as v^*.

It is clear that $v^* = X \setminus v^-$ and $v^{**} = (v^-)^\circ$. Hence, v is regular if and only if $v = v^{**}$.

Observation 2.2.17. Let $\langle X, \mathcal{U} \rangle$ be a topological space. Note that the following properties hold, for every $v, v' \subseteq X$.

(i) $X^\circ = X$

(ii) $v^\circ \subseteq v$

(iii) $(v^\circ)^\circ = v^\circ$

(iv) $v^\circ \cap v'^\circ = (v \cap v')^\circ$

Note the similarity between (i), (ii), (iii), (iv) and the modal logic axioms K, T, and 4.

Definition 2.2.18. We define an *interior operator*, $i : \mathbb{P}(X) \to \mathbb{P}(X)$, as a function which maps a subset of X to its interior. If the conditions in Observation 2.2.17 are satisfied by the mapping i, then we say that i defines a topology on X. Therefore a topological space can also be defined by a pair (X, i) where i is an interior operator.

Connectedness

Definition 2.2.19. Given a topological space $\langle X, \mathcal{U} \rangle$, we say that the space is *disconnected*, if there exist $u, u' \in \mathcal{U}$ such that:

1. $u \neq \varnothing$ and $u' \neq \varnothing$
2. $u \cup u' = X$
3. $u \cap u' = \varnothing$

If a space is not disconnected, then it is *connected*. A subset $v \subseteq X$ is said to be connected if it is connected under the subspace topology (of v in X).

Components

Definition 2.2.20. Given a topological space $\langle X, \mathcal{U} \rangle$ and a subset $v \subseteq X$, a *component* of v is a maximal connected subset of v. Every set has at least one component; the empty set is the only component of itself; all components of a nonempty set are nonempty. A set is connected if and only if it has exactly one component.

2.2.1 Geometric Topology

Let $\langle X, \mathcal{U} \rangle$ be a topological space.

Definition 2.2.21. We say that X is *locally Euclidean* if there is a $n \in \mathbb{N}$ such that every point in X has a neighbourhood which is homeomorphic to the Euclidean space $\mathbb{R}^n$.

Definition 2.2.22. We say that X is a *manifold* if X is a locally Euclidean Hausdorff space.

Definition 2.2.23. We say that X is a *surface* if it is a two-dimensional manifold.

Definition 2.2.24. A surface S is *orientable* if and only if there is no continuous function $f : D \times [0, 1] \to S$ from the product of a disc and the unit interval $[0, 1]$ to the surface such that $f(j, i) = f(k, i)$ only if $j = k$ for all $i \in [0, 1]$, and there is a reflection function r such that $f(d, 0) = f(r(d), 1)$ for all $d \in D$.

Definition 2.2.25. A *triangulation* of a surface is the partitioning of the surface into a set of triangles such that each triangle side is entirely shared by two adjacent triangles.

Definition 2.2.26. A *curve* is a continuous function $f : [0, 1] \to X$.

A *plane curve* is a curve whose codomain is the Euclidean plane. We say that a curve is *simple* if $f(i) = f(j) \Rightarrow i = j$. We call a curve a *loop* if $f(0) = f(1)$. A simple loop is called a *Jordan curve*.

Definition 2.2.27. The *interior* of a curve f is the set $\{f(i) \mid 0 < i < 1\}$.

Now, we state two well known results.

Theorem 2.2.28 (Jordan curve theorem)**.** *Let f be a Jordan curve in the Euclidean plane. Then $\mathbb{R}^2 \setminus f$ has two components, an* inside*, and an* outside*, both of which have f as their boundary.*

Theorem 2.2.29 (Jordan-Schönflies theorem)**.** *Let f be a Jordan curve in the Euclidean plane. The closure of one of the components of $\mathbb{R}^2 \setminus f$ is homeomorphic to the open unit disc.*

2.3 Graph Theory

Definition 2.3.1. A *graph* is a pair $G = (V, E)$ of sets such that $E \subseteq V^2$.

We call the elements of V *vertices*, and the elements of E *edges*. For any edge $e \in E$ where $e = (v, v')$, we call the vertices v and v' the *endpoints* of e, and say that e is *incident* on both v and v'.

Definition 2.3.2. An *embedding* of a graph is a pair of functions (f, g) with $f : V \to \mathbb{R}^2$ and $g : E \to \mathcal{C}$, where $\mathcal{C}$ is the set of simple plane curves, such that the following hold. For each $e \in E$ where $e = (v, v')$, $g(e)$ is a simple curve with endpoints $f(v)$ and $f(v')$, and for all $v'' \in V$ the interior of the curve $g(e)$ does not contain the point $f(v'')$.

Definition 2.3.3. We say that a graph (V, E) is *planar* if it has an embedding (f, g) such that for each $e, e' \in E$, the interiors of the curves $g(e)$ and $g(e')$ do not intersect.

2.4 Boolean Algebras

This subsection presents basic definitions of Boolean algebra concepts, mainly for the purpose of fixing notational standards. Koppelberg [Kop89] provides a comprehensive reference for Boolean algebras.

Definition 2.4.1. A *Boolean algebra* is a structure $\langle A, +, \cdot, -, 0, 1\rangle$, consisting of a set of elements, A, two binary operations, $+$ and $\cdot$, a unary operation, $-$, and two constants, 0 and 1, such that the following holds (where $a, b, c \in A$):

$$\begin{aligned} a + -a &= 1 & a \cdot -a &= 0 \\ a + (b + c) &= (a + b) + c & a \cdot (b \cdot c) &= (a \cdot b) \cdot c \\ a + b &= b + a & a \cdot b &= b \cdot a \\ a + (a \cdot b) &= a & a \cdot (a + b) &= a \\ a \cdot (b + c) &= (a \cdot b) + (a \cdot c) & a + (b \cdot c) &= (a + b) \cdot (a + c) \end{aligned}$$

These axioms define a natural partial order $\leq$ over the set A, where we say that $a \leq b$ if and only if $a + b = b$.

Definition 2.4.2. An *atom* in a Boolean algebra is a nonzero element a such that there is no element b such that $0 < b < a$. A Boolean algebra is *atomic* if every nonzero element of the algebra is above an atom.

Let A be a Boolean algebra.

Definition 2.4.3. For $B \subseteq A$ and $b \in B$, we say that b is a *lower bound* of B if $b \leq b'$ for every $b' \in B$. We say that b is a *greatest lower bound* of B if b is a lower bound of B and $b' \leq b$ holds for each lower bound b' of B. Likewise, we say that b is an *upper bound* of B if $b' \leq b$ for every $b' \in B$. And we say that b is a *least upper bound* of B if b is an upper bound of B and $b \leq b'$ holds for each upper bound b' of B.

Definition 2.4.4. We say that A is *complete* if, for each $B \subseteq A$, B has both a least upper bound and a greatest lower bound.

Definition 2.4.5. We call $B \subseteq A$ a *sub-algebra* of A, if the restriction of $\leq_A$ to B satisfies the axioms of 2.4.1. We say that B is a *dense* sub-algebra of A if, for every $a \in A$ with $0 < a$, there exists $b \in B$ with $0 < b \leq a$.

The following is a well known theorem, a proof of which can be found in [Joh82].

Theorem 2.4.6. *Let X be a topological space. The set of regular open sets in X, written $RO(X)$, forms a Boolean algebra with top and bottom defined by* $1 = X$

and $0 = \varnothing$*, and Boolean operations defined by* $u + u' = ((u \cup u')^-)^\circ$, $u \cdot u' = u \cap u'$ *and* $-u = (X - u)^\circ$*, where* $u, u' \in RO(X)$.

Similarly, the set of regular closed sets in X*, written* $RC(X)$*, also forms a Boolean algebra, the difference being that the Boolean operations are defined by* $u + u' = u \cup u'$, $u.u' = ((u \cap u')^\circ)^-$ *and* $-u = (X - u)^-$.

2.5 Computational Complexity

Computers are particularly suited for solving complicated problems which can be broken down into simple, repetitive actions. An algorithm is simply a description of how to break down a complicated problem into many smaller ones. If an algorithm always produces a correct result for a given problem, then the algorithm is said to *solve* the problem. Complexity theory studies the factors affecting the performance of computers executing algorithms which solve problems. In order to investigate this further, we will introduce a formal model of computation known as a *Turing machine*.

2.5.1 Turing Machines

Informally, a Turing machine is a tape beginning with a leftmost 'start' cell which infinitely extends to the right, with a read/write head that has a state and which can move left and right along the tape, and a program in the form of a table which, given the current state of the head and the current symbol under the head, directs the head to write a symbol to the tape, move one step left or right, then sets the head to a new state. The following definitions are taken from Papadimitriou [Pap94], which should be consulted for more information on computational complexity.

Definition 2.5.1. A *deterministic* Turing machine is a quadruple $M = (K, \Sigma, \delta, k)$, where K is a finite set of states, Σ is a finite alphabet, δ is a transition function mapping the set $K \times \Sigma$ to $(K \cup \{halt, yes, no\}) \times \Sigma \times \{left, right, wait\}$, and $k \in K$ is the initial state. We always assume that $K \cap \Sigma = \varnothing$, and that Σ contains symbols representing *blank* and *first*.

An *input* to a Turing machine is a finite string of symbols from Σ not containing the blank symbol, with the start symbol as the leftmost of the string. The input represents the contents of the tape before the execution begins. Initially, the head of the machine is positioned on the leftmost symbol of the string (the start symbol).

A *configuration* of a Turing machine is a quadruple (l, S, T, n) consisting of the current state of the machine $l \in K$, a string of symbols representing the contents of the tape to the left of (and including) the head, S, a string of symbols representing the contents of the tape to the right of the head, T, and a *step count* $n \in \mathbb{N}$. If a string T' is the input to a Turing machine, then the initial configuration of that machine is $(k, start, T', 0)$.

Let (l, Ss, tT, n) be a configuration of a Turing machine where $s, t \in \Sigma$ and S and T are strings over Σ, and $\delta(l, s) = (l', s', D)$. If D is $left$, let $S' = S$ and let $T' = s'tT$. If D is $right$, let $S' = Ss't$ and let $T' = T$. If D is $wait$, let $S' = Ss'$ and let $T' = tT$. Then we say that a Turing machine with configuration (l, Ss, tT, n) *yields* the configuration $(l', S', T', n+1)$ in one step. This is extended to "yields in n steps" in the obvious way.

We interpret this "yields" relation as the execution of the program δ on a given input. The machine starts in configuration $(k, start, S, 0)$ for some string over Σ, S, and each successive configuration is yielded according to δ, until the Turing machine enters one of the states *halt*, *yes*, or *no*, when we say that the machine has *halted*, and execution terminates. If the Turing machine enters the state *yes*, then we say that the machine has *accepted* its input, and if the machine enters the state *no*, then we say it has *rejected* its input. If a Turing machine finishes in either *yes* or *no*, then we say that the *output* of the machine is *yes* or *no* respectively, if the machine finishes in the state *halt*, then the output of the machine is the the machine's string at the time of termination.

We take a *problem* Π to be a class of questions, and refer to specific questions of Π as *instances* of the problem Π. If the instances of Π are problems which are answered with a *yes* or *no*, then we say that Π is a *decision problem*.

We can encode instances of a problem Π as strings. If Π is a decision problem, then we say that a Turing machine *solves* these Π if it accepts the encoded *yes* instances, and rejects the encoded *no* instances. Otherwise, we say that a Turing machine *solves* Π if its output is always a suitably encoded string of the correct answer to the instance of Π in question. Given a problem Π, if there is a Turing machine which solves Π, then we say that Π is *decidable*, otherwise it is undecidable.

If M is a quadruple satisfying Definition 2.5.1 in every way except that δ

is a relation, $\delta \subseteq (K \times \Sigma) \times ((K \cup \{halt, yes, no\}) \times \Sigma \times \{left, right, wait\})$, instead of being a function from $K \times \Sigma$ to $(K \cup \{halt, yes, no\}) \times \Sigma \times \{left, right, wait\}$, then we say that M is a *non-deterministic* Turing machine. Since δ in a non-deterministic Turing machine is a relation, each configuration of a non-deterministic Turing machine could yield multiple configurations. Based on the particular choice of configuration at each execution step, a Turing machine could produce different answers to the same instance of a problem. We say that a non-deterministic Turing machine solves an instance of a problem if there is at least one run of execution which terminates successfully (it does not matter if other runs terminate unsuccessfully). The machine fails to solve an instance only when every run of execution terminates unsuccessfully.

2.5.2 Complexity

We can see that intuitively some problems are 'harder' than others. But, we must have a formal way of showing this, and we must specify formally what 'harder' means. By formalising the model of computation in the form of a Turing machine, we are able to clearly see that some problems are more expensive, in terms of time, and space, to solve. We measure time as the number of steps a Turing machine requires to arrive at a correct solution, and we measure space in terms of the number of symbols on a tape needed to arrive at a correct solution. We express the time or space required by a Turing machine M to solve a problem as a function of the size of the input to M, and in complexity theory, we are interested in which general class of functions this function belongs, for example, logarithmic, polynomial, and so on. We use the so called 'Big O' notation to denote which class these input functions belong to.

Definition 2.5.2. Let $f : \mathbb{N} \rightarrow \mathbb{N}$ and $g : \mathbb{N} \rightarrow \mathbb{N}$ be functions. We say that $f(n) = O(g(n))$ if there are positive integers c and n_0 such that for all $n > n_0$, $f(n) \leq c.g(n)$.

Let $f : \mathbb{N} \rightarrow \mathbb{N}$ be a function, then given any instance of a problem Π which is encoded as input of size n to a Turing machine M, if M terminates in $O(f(n))$ steps, then we say that M implements an algorithm of time complexity $O(f(n))$ which solves Π. Likewise, let $g : \mathbb{N} \rightarrow \mathbb{N}$ be a function, then given any instance of a problem Π which is encoded as input of size n to a Turing machine M, if M terminates having used $O(g(n))$ cells of the tape, then we say that M implements an algorithm of space complexity $O(g(n))$ which solves Π. Note that the input and output strings are not counted when considering the space complexity, we only consider the space used during the algorithm.

In order to study the difficulty of a problem Π, we must consider only the instance of Π which is the worst case in terms of difficulty. By examining the time and space taken for M to reach a solution, as functions of the size of the input to M, we achieve an *upper bound* on the computational complexity of Π.

We can split complexity classes into two kinds, those of time complexity, and those of space complexity. Time complexity classes are those classes whose problems have a known upper bound in terms of time complexity, with no restriction on space complexity. Likewise, space complexity classes are those classes whose problems have a known upper bound on their space complexity, with no restriction on time complexity.

The time complexity classes that we encounter during this thesis are as follows.

PTIME is the set of problems which are solvable by a deterministic Turing machine in time which is bounded by a polynomial function of the size of the input. The class **PTIME** is normally abbreviated to **P**.

NPTIME is the set of problems which are solvable by a nondeterministic Turing machine in time which is bounded by a polynomial function of the size of the input. The class **NPTIME** is normally abbreviated to **NP**.

EXPTIME is the set of problems which are solvable by a deterministic Turing machine in time which is bounded by a exponential function of the size of the input. The class **EXPTIME** is normally abbreviated to **EXP**.

NEXPTIME is the set of problems which are solvable by a nondeterministic Turing machine in time which is bounded by a exponential function of the size of the input. The class **NEXPTIME** is normally abbreviated to **NEXP**.

As these classes represent upper bounds on complexity, they form the following hierarchy.

$$P \subseteq NP \subseteq EXP \subseteq NEXP$$

It is not known whether any of these individual relations are strict subset or equality, although it is known that $P \neq EXP$. We are also interested in the following space complexity classes.

LOGSPACE is the set of problems which are solvable by a deterministic Turing machine in space which is bounded by a logarithmic function of the size of the input. The class **LOGSPACE** is normally abbreviated to **L**.
NLOGSPACE is the set of problems which are solvable by a nondeterministic Turing machine in space which is bounded by a logarithmic function of the size of the input. The class **NLOGSPACE** is normally abbreviated to **NL**.

PSPACE is the set of problems which are solvable by a deterministic Turing machine in space which is bounded by a polynomial function of the size of the input.

These space classes fit into the hierarchy of time classes in the following way.

$$L \subseteq NL \subseteq P \subseteq NP \subseteq PSPACE \subseteq EXP \subseteq NEXP$$

Now we define the concept of a reduction from one decision problem to another. Roughly speaking, a reduction is the transformation of one problem into another problem.

Definition 2.5.3. We say that a decision problem Π is reducible to the problem Π' if for every instance π of Π, there is an *efficient* algorithm which encodes π as $\pi' \in \Pi'$ such that π and π' have the same answer.

What we mean by *efficient* varies slightly depending on the complexity class that Π belongs to. If Π is an NL or P problem, the reduction must be in L. For all other complexity classes we consider, the reduction must in P.

Given a complexity class C, we say that a problem Π is C-hard if every problem in C is reducible to Π. A problem Π is C-complete if $\Pi \in C$ and Π is C-hard. For every class C, the C-complete problems are the *hardest* group of problems for that class. Many problems in C' may also have algorithms in C, however all of the problems which are C-complete do not have algorithms in C, unless $C = C'$.

The *satisfiability* problem for first order logic is undecidable, however the satisfiability problem for the modal logic $S4$ is in $PSPACE$, and the satisfiability problem for propositional logic is in NP. Both of these satisfiability problems are complete for their respective complexity classes, and in fact, propositional logic satisfiability is one of the most famous examples of NP-completeness, see [Coo71].

For the remainder of this thesis, we investigate the *topological inference* problem. That is, the problem of determining the satisfiability of logics which are interpreted over topological structures.

Chapter 3

String Graphs

This chapter introduces background information regarding a graph theoretic problem called the *string graph problem.* The string graph problem was recently solved independently by Schaefer & Štefankovič [SŠ04] and by Pach & Tóth [PT02]. In this chapter we give an outline of the proof of the decidability of string graphs given by Schaefer & Štefankovič.

3.1 String Graph Problem

First we define some concepts needed in order to introduce the string graph problem. The following definitions, theorems and corresponding proofs are all taken from [SŠ04], except where otherwise stated.

Definition 3.1.1. Given a collection of curves $C_1, \ldots, C_n$ in the plane, the corresponding *intersection graph* is as follows.

$$(\{v_1, \ldots, v_n\}, \{(v_i, v_j) \mid C_i \text{ and } C_j \text{ intersect, for all } i < j \leq n\})$$

The *size* of a collection of curves is the number of intersection points. A graph isomorphic to the intersection graph of a collection of curves in the plane is called a *string graph.* Note that C_i and C_j may intersect more than once, though this does not affect the intersection graph.

The string graph problem is as follows: given a graph G, is G a string graph? Alternatively, we can rephrase the problem. Can we draw a set of curves in the plane, such that only the curves we specify intersect?

Before we investigate this further, we need to define some more concepts.

Definition 3.1.2. Let $G = (V, E)$ be a graph, and let $H \subseteq \binom{E}{2}$ (where $\binom{E}{2}$ is the set of unordered pairs of E), we call this pair (G, H) an abstract topological graph, or AT-graph.

Definition 3.1.3. We call a drawing D in the plane of G a *weak realization* of (G, H) if only pairs of edges which are in H intersect in D. We call (G, H) *weakly realizable* if it has a weak realization. Note that in a weak realization the pairs of edges in H do not have to intersect.

Definition 3.1.4. We say that a drawing D of G is a *realization* of (G, H), and say that (G, H) is *realizable* if exactly the edges in H intersect in D.

Let $c_s(G)$ be the size of a smallest (i.e. smallest number of intersections) set of curves whose intersection graph is isomorphic to G, then we define $c_s(m) = max\{c_s(G) \mid G \text{ has } m \text{ edges }\}$. Let $c_w(G, H)$ be the smallest number of intersections in a weak realization of (G, H), let $c_w(G) = max\{c_w(G, H) \mid (G, H) \text{ has a weak realization }\}$, and let $c_w(m) = max\{c_w(G) \mid G \text{ has } m \text{ edges }\}$. Similarly, we can define $c_r(G, H)$, $c_r(G)$ and $c_r(m)$ for realizations. It is simple to see that $c_w(m) \leq c_r(m)$.

We will now examine an overview of a proof that a solution to the string graph problem belongs to the NEXP time complexity class. The method we will follow is the one given in Schaefer & Štefankovič [SŠ04].

As shown in [Kra91], the string graph problem can be reduced to AT-graph weak realizability, as follows.

Theorem 3.1.5. *Given a graph* $G = (V, E)$*, let* $G' = (V \cup E, \{(u, e) \mid u \in e \in E\})$ *and let* $H = \{((u, e), (v, f)) \mid \{u, v\} \in E\}$*. Then* G *is a string graph if and only if* (G', H) *is weakly realizable.*

As a result, we have the following bound $c_s(m) \leq 4c_w(2m) + 2m$, see [Kra91] for more details.

Our overall aim here is to prove that if an AT-graph has a weak realization, then it has a weak realization of a certain (maximum) size. It will follow, therefore, that if a graph G is weakly realizable as its AT-graph (Theorem 3.1.5) within the maximum bound, then G is a string graph.

First we need to prove the finiteness of string graphs; we can take the following result from [KGK86].

Lemma 3.1.6. *A string graph can be realized by a family of polygonal arcs with a finite number of intersections.*

As a result $c_s(G)$ is a finite number, if G is a string graph. Given a system of curves, $(C_i)_{i \in I}$, and an alphabet of size $|I|$, we can assign each curve in the system a letter from the alphabet, and we can, therefore, encode the intersections of each curve as a word of this alphabet. If we encode each curve as a word of an alphabet, the following lemma shows that each of these words has a property which we can use to determine a bound on the size of the words.

Lemma 3.1.7 ([SŠ04])**.** *Every word of length at least 2^n over an alphabet of size n contains a non-trivial subword in which every character occurs an even number of times.*

Proof. Let $\Sigma = \{1, \ldots, n\}$ be an alphabet of length n, and let $w \in \Sigma^*$ be a word of that alphabet, $|w| \geq 2^n$. To every $i \in \{0, \ldots, 2^n\}$ assign a vector v_i in $\mathbb{Z}_2^n$ whose jth coordinate is the parity of the number of occurrences of the symbol j in the prefix of w of length i (v_0 is the all-zero vector). Since there are $2^n + 1$ indices, but only 2^n vectors in $\mathbb{Z}_2^n$, there are $0 \leq i < j \leq 2^n$ such that $v_i = v_j$. Since each successive v_i will have one digit different to v_{i-1}, $j > i + 1$ and so, the non-trivial subword of w starting in position $i + 1$ and ending in position j fulfils the conditions of the lemma. □

The following theorem states that the number of intersections along a curve of a weak realization of a graph is bounded by an exponential value of the number of edges of the graph.

Theorem 3.1.8 ([SŠ04])**.** *Let G be a graph with m edges, $H \subseteq \binom{E}{2}$ such that (G, H) is weakly realizable, and let D be a weak realization of (G, H) with the minimal number of intersections. Then for any edge $e \in G$ there are fewer than 2^m intersections on the curve realizing e in D.*

Proof. We will prove this by contradiction by assuming that we have a minimal (in terms of the number of intersections) weak realization of (G, R) with an edge e that has more than $2^m - 1$ intersections.

Lemma 3.1.7 shows us that we can choose a segment of e, in which e is intersected only an even number of times by other curves in the system. We then draw a 'window' around this segment of the curve, containing no other intersections of

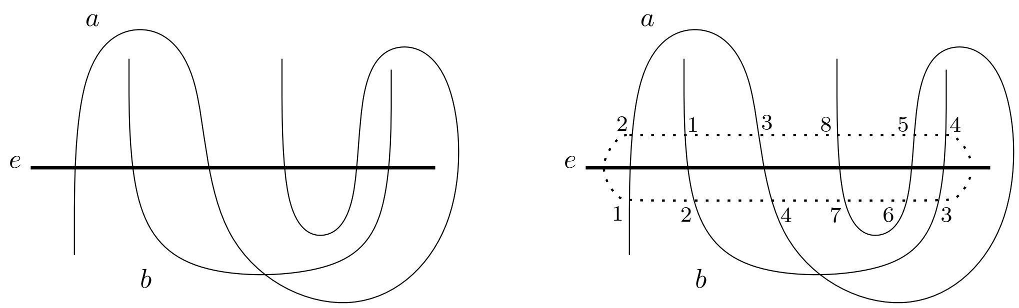

Figure 3.1: Drawing a window around the intersections.

D (see Figure 3.1). This is possible because D is finite by Lemma 3.1.6. Let $2n_f$ ($n_f \in \mathbb{N}$) be the number of intersections of any curve f with the curve e inside the window. For each edge f assign numbers $1, 2, \ldots, 4n_f$ to the intersection with the window, in the order they appear along f (choose an arbitrary orientation of f), again see Figure 3.1.

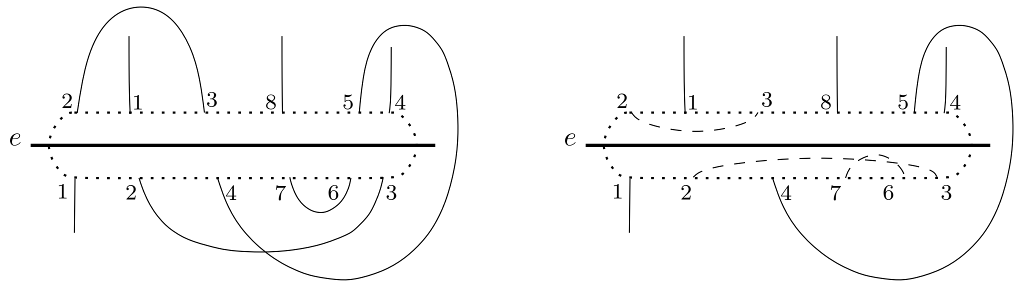

Figure 3.2: Performing circular inversion on sections of the curves.

We can assume that the window is a circle (by application of the Jordan-Schoenflies theorem, [MT01]), and that e is a straight line passing through the centre, and for each curve f, the window intersection points $2i-1$ and $2i$ (along f) are mirror images of each other (with e as the mirror), for $i \in \{1, \ldots, 2n_f\}$. Now, we remove everything inside the window, except the line e, see first diagram of Figure 3.2. For each edge f, there is a curve (segment of f) between intersections $4i-2$ and $4i-1$ lying outside the window ($i \in \{1, \ldots, n_f\}$). Use circular inversion along the window to bring all of these segments inside the window, see second diagram of Figure 3.2. Now mirror the curves inside the window along e, see first diagram of Figure 3.3.

There will now be connections, for every edge f, between intersections $4i-3$ and $4i$, $i \in \{1, \ldots, n_f\}$, inside the window. Now with reference to the first diagram of Figure 3.3, we demonstrate how to construct a new version of the curve f. Take,

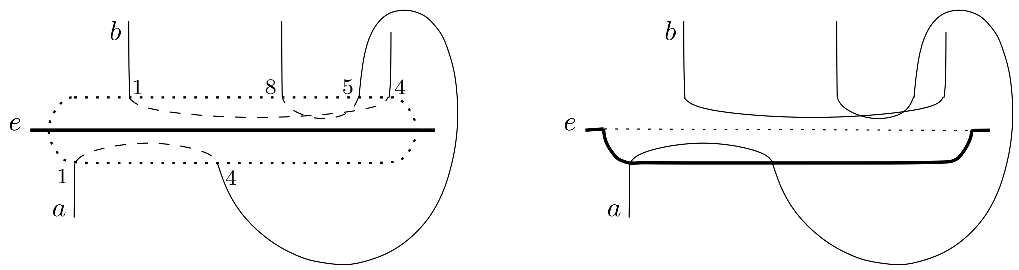

Figure 3.3: Mirror along e, then shift e to minimise intersections.

for example, curve a: we start at intersection 1 (which is connected to the start point of the curve), continue inside the window to intersection 4, move outside the window from intersection 4 to intersection 5, back inside the window through to intersection 8, and outside the window along to the endpoint of the curve. This new version of f still connects its two endpoints, hence the need for f to intersect e an even number of times.

Every intersection between curves which happens inside the window corresponds to an intersection outside the window, hence this drawing is still a weak realization of G with respect to R. As we only require a *weak* realization, it does not matter if some intersections between curves have been lost.

It is possible that a curve brought inside the window by circular mirroring may intersect e, thus increasing the intersections along e. We have certainly decreased the number of intersections with the boundary of the window by half; we can split the boundary of the window into two arcs, according to where e intersects the boundary, and one of these arcs has half (or less) of the intersections that e did originally. We can, therefore, re-route e through the path of the arc with the least intersections, see second diagram of Figure 3.3.

We have reduced the number of intersections of any curve f along a segment of e from $2n_f$ to less than or equal to n_f, thus contradicting the assumption that D was of minimal size. □

We now have the following corollary:

Corollary 3.1.9 ([SŠ04])**.** *String graph recognition is in NEXP.*

Proof. Theorem 3.1.8 shows that $c_w(m) \leq m2^m$, and since $c_s(m) \leq 4c_w(2m)+2m$, then we can state that $c_s(m) \leq 8m.2^{2m}+2m$. Therefore given a graph $G = (V, E)$, if G is a string graph then there is a collection of curves of size $N = O(2^m)$ whose intersection graph is isomorphic to G. The drawing of this collection of curves can be considered as a planar graph with at most N vertices.

By a result of Schnyder [Sch90], there is a drawing of this graph on an $N \times N$ grid. We can construct all possible planar graphs of up to N vertices, with the curves of the collection being represented by disjoint sets of edges, and intersections between curves being represented by the vertices of the graph. Each curve is represented by a set of edges which form a continuous non-intersecting path in these graphs. Two curves intersect if their edge sets have elements which are adjacent to a common vertex. By this definition of curve intersection, given a planar graph G' of up to N vertices, we can partition the edges of this graph into $|V|$ disjoint edge sets and compute an intersection graph for each of these partitionings. If any of these intersection graphs are isomorphic to G, then G is a string graph. □

However, Corollary 3.1.9 is not the result we are aiming for. We can reduce the complexity bound to NP, as we shall see. First, we must introduce some results about *word equations*.

3.1.1 Word Equations

Let Σ be an alphabet of symbols, and Θ a disjoint alphabet of variables.

Definition 3.1.10. A *word equation* $u = v$ is a pair of words such that $(u, v) \in (\Sigma \cup \Theta)^* \times (\Sigma \cup \Theta)^*$.

Definition 3.1.11. Let $u = v$ be a word equation. A *solution* to $u = v$ is a morphism $h : (\Sigma \cup \Theta)^* \to \Sigma^*$, such that $h(a) = a$ for all $a \in \Sigma$ and $h(u) = h(v)$.

Definition 3.1.12. An LZ-encoding of a solution h to a word equation is the sequence of LZ-encodings of $h(x)$ for all $x \in \Theta$.

Theorem 3.1.13. *([GKPR96]) Let $u = v$ be a word equation. Given an LZ-encoding of a morphism h, we can check whether h is a solution of the equation in time polynomial in $|LZ(h)|$.*

Theorem 3.1.14. *([PR98]) Let $u = v$ be a word equation with lengths specified by function f. Assume $u = v$ has a solution respecting f. Then, there is a solution h respecting the lengths such that $|LZ(h)|$ is polynomial in the size of a binary encoding of f and the size of the equation. Moreover, the lexicographically least solution can be found in P time.*

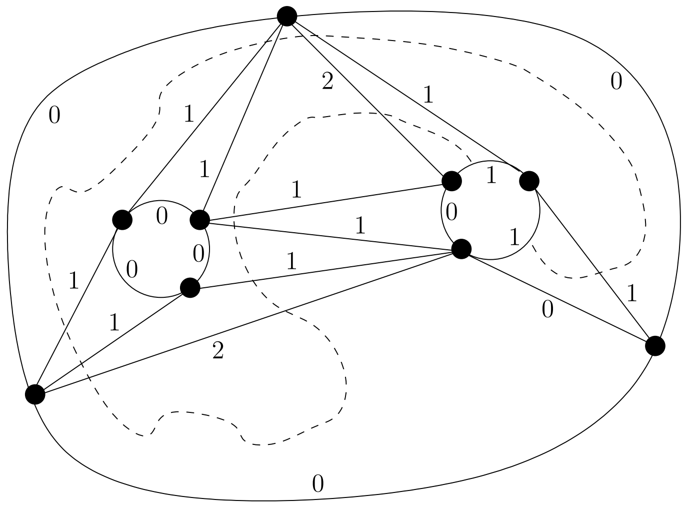

Figure 3.4: Example of a triangulation numbering.

Finally, we take the following result which follows from Gasieniec et al. [GKPR96].

Lemma 3.1.15. *Given an LZ-encoding LZ(w) of a word w and a letter $a \in \Sigma$, we can compute the number of occurrences of a in w in time polynomial in $|LZ(w)|$.*

3.1.2 Main String Graph Result

Let M be a compact orientable (see Definition 2.2.24) surface with a boundary.

Definition 3.1.16. A *properly embedded arc* (in M) is an arc whose endpoints are on the boundary of M, and whose internal points are in the interior of M.

Let T be a planar graph which forms a triangulation (see Definition 2.2.25)of M, with E_T being the edge set of T.

Definition 3.1.17. We say that an arc is *normal* with respect to T if all intersections with T are transversal, and if the arc enters a triangle via one edge, and leaves the triangle via a different edge.

We make the following claim.

Lemma 3.1.18. *Let γ be a properly embedded arc in M. There is always an isotopically equivalent arc which is normal with respect to T.*

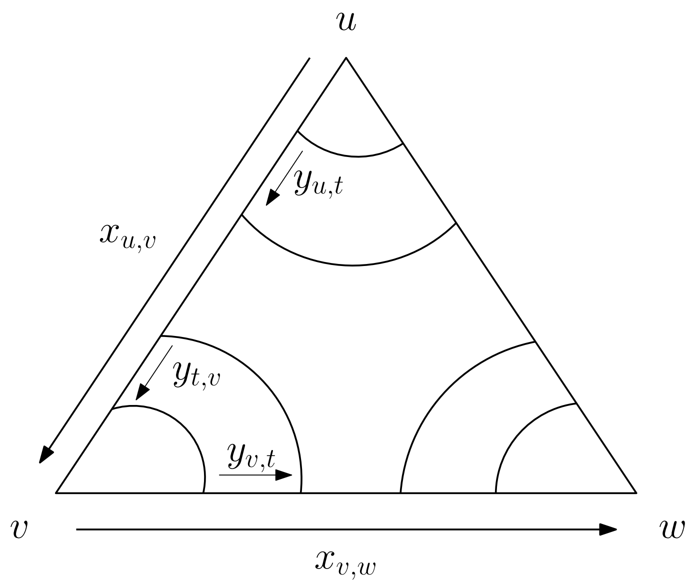

Figure 3.5: Triangle variables.

Given a properly embedded arc γ which is normal with respect to T, we can label each edge of the triangulation with the number of intersections of gamma with that edge, see Figure 3.4. We say that a numbering $\ell : E_T \to \mathbb{N}$ is *valid* if there is a properly embedded arc, which is normal with respect to T, which intersects each edge $e \in T$, $\ell(e)$ times. And we say that γ *realizes* ℓ. Note that all arcs which realize a given numbering are isotopically equivalent.

Let ℓ be a valid numbering. The sum of the labels of edges from $E_T \cap M^\partial$ is 2. For each triangle $t \in T$ the labels a, b, c of edges of t satisfy $a + b \geq c$, $a + c \geq b$, $b + c \geq a$ and $a + b + c$ is even. These conditions are necessary for validity, but not sufficient. We call a labeling satisfying these conditions *semi-valid*. Any semi-valid labeling defines a properly-embedded arc and a (possibly empty) set of closed curves.

For each oriented edge $(u, v) \in T$ there is a variable $x_{u,v}$ encoding the order in which the curves intersect on (u, v). Let $t \in T$ be a triangle with vertices u, v, w. We add six variables $y_{t,u}, y_{t,v}, y_{t,w}, y_{u,t}, y_{v,t}, y_{w,t}$ as shown in Figure 3.5. We associate the following set of equations with the triangulation T.

$$x_{u,v} = y_{u,t} y_{t,v} \qquad x_{v,u} = y_{v,t} y_{t,u}$$
$$x_{v,w} = y_{v,t} y_{t,w} \qquad x_{w,v} = y_{w,t} y_{t,v}$$
$$x_{u,w} = y_{u,t} y_{t,w} \qquad x_{w,u} = y_{w,t} y_{t,u}$$

Lemma 3.1.19 ([SSŠ03]). *Given a numbering ℓ, we can test whether ℓ is valid in polynomial time.*

Proof. First we verify if ℓ is semi-valid, and reject ℓ if it is not. Let $\sigma = \{a, b\}$, we take the set of equations associated with T over σ, and for each edge $e = (u, v) \in E_T$ we specify that $|x_{u,v}| = \ell(e)$. For each edge $e = (u, v) \in E_T \cap M^\partial$ we specify $x_{u,v} = b^{\ell(e)}$.

We claim that if ℓ is valid, then the set of equations associated with T has a unique solution. Take the properly embedded arc γ which realizes ℓ, number the intersections of γ with T in the order in which they occur on γ. Each intersection corresponds to a position in some variable. By induction on the number of intersections it follows that each position in every variable is forced to be b.

On the other hand, let us assume that ℓ is not valid. Because it is semi-valid, there is a solution to the set of word equations. However, a lexicographically smallest solution will now contain the letter a, which corresponds to one of the members of the set of closed curves defined by a semi-valid labeling.

Because of Theorem 3.1.14 we can compute the lexicographically least solution in polynomial time, and we can check by Lemma 3.1.15 that it does not contain any occurrences of a. So, by solving the set of equations associated with T, we can check if ℓ is valid. □

Lemma 3.1.20 ([SSŠ03]). *Let γ_1, γ_2 be properly embedded arcs which realize the numberings ℓ_1, ℓ_2. If γ_1 and γ_2 do not intersect, then we can verify that $i(\gamma_1, \gamma_2) = 0$ in polynomial time. Moreover, if the verification concludes that $i(\gamma_1, \gamma_2) = 0$ then γ_1 and γ_2 are isotopically disjoint.*

We now introduce a topological variant of the graph weak realizability problem. Say we have a weakly realizable graph (G, H). Then, let M be the surface obtained by drilling a hole, for each vertex of the graph G, in the Euclidean plane. Now, a set S of properly embedded arcs on M is called a weak realization with holes, if, for each edge between a pair of vertices in the graph, we have a properly embedded arc connecting the two holes representing those vertices, and, for each pair of edges not in H, the arcs representing those edges are isotopically disjoint.

Lemma 3.1.21 ([SSŠ03]). *Let (G, R) be an AT-graph. The graph (G, R) is weakly realizable if and only if it has a weak realization with holes.*

Lemma 3.1.22 ([SSŠ03])**.** *Let G be a graph with m edges and n vertices. Assume that (G, H) has a weak realization with holes. Let M be the surface obtained from the plane by drilling $|V|$ holes. Let T be a minimal triangulation of M. Then, there is a weak realization with holes of (G, H) in M such that there are at most 2^{12n+m} intersections on each edge of T.*

Theorem 3.1.23 ([SSŠ03])**.** *The weak realizability problem is in NP.*

Proof. Let (G, R) be an AT-graph with $n = |V_G|$ and $m = |E_G|$. We show that deciding whether (G, R) has a weak realization with holes lies in NP. Since Lemma 3.1.21 shows the equivalence of weak realizability and weak realizability with holes, this proves the result.

Suppose (G, R) has a weak realization with holes. Let T be a minimal triangulation of M, Lemma 3.1.22 implies that there is a weak realization with holes in which every edge of T is intersected at most 2^{12n+m} times. So every edge e of G can be represented by an arc γ with a numbering $\ell_\gamma : E_T \to \{0, \ldots, 2^{12n+m}\}$. By Lemma , we can assume that two arcs γ_1, γ_2 representing two edges $(e, f) \notin R$ are disjoint. To verify weak realizability with holes of (G, R), it is sufficient to guess for each edge e of G a numbering $\ell_e : E_T \to \{0, \ldots, 2^{12n+m}\}$ of T (note that the numbering has size polynomial in G). Then we check that all guessed numberings are valid and verify that for every $(e, f) \notin R$ the curves representing e and f are isotopically disjoint. Both of these tasks can be performed in polynomial time, by Lemma 3.1.19 and Lemma 3.1.20. The verification succeeds if and only if (G, R) has a weak realization with holes, so this implies that weak realizability with holes can be verified in NP. □

3.2 Conclusion

This chapter has given an outline of the solution for the string graph problem, and for the AT-graph weak realizability problem. All of the results in this chapter have come from Schaefer & Štefankovič [SŠ04], and Schaefer, Sedgwick, and Štefankovič [SSŠ03]. These results are used later on in this thesis, in Chapter 6.

Chapter 4

Spatial Logics and Reasoning

In order to study instances of the topological inference problem, we must first examine the spatial logics with which we specify the instances of the problem. This chapter introduces the concept of a spatial logic and defines the structures which we interpret these logics over. We take a model theoretic approach to examining the computational properties of the topological inference problem, that is, we investigate the relationship between the spatial logics and the topological structures we interpret them over. This chapter, provides the first part of a new and consolidated survey of spatial logics, which is one of the major contributions of this thesis.

4.1 Spatial Logic

A spatial logic is a formal language whose formulae are interpreted over a class of geometric structures. The variables of our language are interpreted as the primitives of our geometric structures and the predicate and function symbols are interpreted as various geometric relations and properties. We are interested in *qualitative* spatial logics, which are concerned with mereotopological properties and relations, as opposed to *quantitative* spatial logics, which are concerned with quantifiable spatial properties and relations, such as size and distance. Mereotopology is a combination of two fields of mathematics. *Mereology*, which is the study of part-whole relationships, and *Topology*, which was briefly introduced in Section 2.2. Furthermore, as mentioned in Chapter 1, we restrict ourselves primarily to qualitative spatial logics which are interpreted over structures which inhabit the Euclidean plane.

It is difficult to identify a beginning of the development of qualitative spatial logics. Traditionally, the primitive units of a space are taken to be its points. But, the very notion of a 'point' is something which does not fit in very well with our perception of space. If each physical object occupies a set of points, then a question arises. Is this set topologically open, or closed? So, in order to avoid these difficult philosophical issues, perhaps we should consider other kinds of entities as our primitive units.

One of the first systems of geometry to consider alternative primitives to the point was Whitehead's 'point-free geometry'. In [Whi19] and [Whi20], Whitehead presented a system of spatiotemporal ordering and measurement where 'regions' are the primitive entity. While Whitehead's system appears to have some flaws, certain parts of it were used by de Laguna ([dL22a], [dL22b]) to describe some standard geometrical concepts in [dL22c]. De Laguna does not attempt to construct a complete system of geometry, but simply to show "the possibility of a geometry" based on the concepts he defines. Inspired by de Laguna's work, Whitehead proposed a modified system [Whi29], based on a topological relation which he called 'extensive connection'. Although Whitehead doesn't specify a specific domain for interpretation, the 'extensive connection' relation seems to behave similarly to the relation which holds between two subsets of a topological space whose closures intersect. Somewhat counter-intuitively, Whitehead's relation does not allow for a region to be connected to itself.

Perhaps the first fully realised 'spatial logic' was presented by Tarski in [Tar56] (this is actually a summary of an address Tarski gave to a mathematical conference in 1927). The primitives over which the variables of this logic range are *sets* of regions, or *solids* as he called them, thus making this language a second-order logic. According to the way Tarski's regions are defined, these regions are simply regular closed subsets of $\mathbb{R}^3$. The language has two predicates, one interpreted as the property of being a 'sphere', and the other interpreted as the 'part-of' relation. A complete axiomatization of the theory is provided, and it is shown that all models of the theory are isomorphic to the standard interpretation of the language over $\mathbb{R}^3$. This is done by showing that the points in $\mathbb{R}^3$ can be represented by sets of converging solids (or spheres).

While Tarski's results would seem to provide a sound foundation for the further development of spatial logics, they were largely ignored for a long time, in favour of Whitehead's work. Clarke [Cla81] proposed a 'calculus of individuals' based on Whitehead's 'extensive connection' relation. Clarke calls this relation, simply, 'connection', and makes the modification of permitting a region to be connected to itself. Although Clarke suggests that the language is taken to range over 'spatio-temporal' regions, the system is presented as an 'uninterpreted calculus' and so the purpose of the axiomatization is a little unclear, as there can be no proof of correctness with regards to an interpretation. Clarke's axioms appear to suggest that there is some distinction between open and closed regions, as he defines both the 'connection' relation and an 'overlap' relation, the latter seems to be the relation which holds between regions when their interiors intersect.

As we are interested in the topological inference problem, we are interested in spatial logics mainly in relation to the structures we interpret them over.

4.1.1 Models

Although Clarke's calculus was presented without any specific interpretation in mind, it is a natural question to wonder what kind of structures are models for the axiomatization that was provided. Biacino & Gerla [BG91] investigated this and found that Clarke's axioms characterise the complete atomless Boolean algebras, thus in a sense the class of non-empty regular open subsets of a topological space are models for the axiomatization. Unfortunately, as defined over this domain, the connection and overlap relations are equivalent. This leaves Clarke's calculus being able to express mereological relations, but not topological ones, this renders it unsuitable as a spatial logic.

Inspired by Allen's temporal calculus [All83], the Region Connection Calculus, or RCC [RCC92b] was an attempt to correct Clarke's calculus (especially the issues raised by Biacino & Gerla) and to create a spatio-temporal logic, based on a connection relation. As with Clarke's calculus, an axiomatization was provided but with no proof of correctness with respect to any particular interpretation. The RCC has received considerable attention and we shall examine parts of it in more detail in Chapters 5 and 6.

The rest of this chapter looks at an approach to spatial logic which is much more in the tradition of Tarski. Instead of choosing a language, creating a set of axioms which *seem* to correspond with intuition, and then finding out if our system is modelled by any kind of space in existence, we begin by first examining the kind of geometric structures that we wish to interpret our spatial logics over. We now introduce the concept of a *mereotopology*.

4.1.2 Mereotopologies

One of the first questions we must answer is: How do we define the regions which our language ranges over? Since we are restricting our attention to topological inference in the real plane, we assume from now on that our regions are subsets of the real plane. And as we are mainly concerned with topological inference in spatial structures that may occur in the physical world, as mentioned in Chapter 1, we may wish to restrict what kind of subsets that we consider to be regions.

We call our choice of what defines the regions of a space, the *mereotopology* of the space. The definitions and corresponding examples in the rest of this subsection are taken from [PH07].

Definition 4.1.1. Given a topological space X, we define a *mereotopology over* X to be a Boolean sub-algebra M of $RO(X)$ such that, if u is an open subset of X and $p \in u$, there exists $m \in M$ such that $p \in m \subseteq u$. We refer to the elements of M as *regions*.

The real plane contains many subsets that could not possibly be occupied by a physical object, for example, and we may wish our domain of regions to not include such subsets.

Consider that although open and closed sets are well-defined mathematical concepts, they are difficult to understand intuitively, and many difficult questions arise as a result of this distinction. For example, take a physical object (in three dimensional Euclidean space) - is the subset of the space that this object occupies open, or closed? In other words - do objects contain their boundary points? We may wish to bypass questions of this nature by considering either open, or closed, subsets to be regions.

Additionally, as a result of Theorem 2.4.6 it is common to restrict attention to only *regular open* subsets. This guarantees us an algebra which is closed under intersection, union and complement of regions. Therefore, for our first example of a mereotopology, we choose the set of all regular open subsets of $\mathbb{R}^2$.

Example 4.1.2. The set $RO(\mathbb{R}^2)$ is a mereotopology over $\mathbb{R}^2$.

Proof. Given $p \in o \subseteq \mathbb{R}^2$ such that o is open, let u, v be disjoint open subsets of $\mathbb{R}^2$ such that $p \in u$ and $\mathbb{R}^2 \setminus o \subseteq v$. Since v is open, $u^- \cap v = \varnothing$, whence $u^- \subseteq o$, and so $p \in (u^-)^0 \subseteq o$. But $(u^-)^0 \in RO(\mathbb{R}^2)$. Hence $RO(\mathbb{R}^2)$ is a mereotopology. □

The mereotopology $RO(\mathbb{R}^2)$ avoids any difficulty regarding the distinction between open and closed sets, however it still contains *every* regular open subset of $\mathbb{R}^2$. Many of these subsets form regions which no physical object could possibly inhabit. For example, subsets with infinitely convoluted boundaries. If our aim is to represent regions which are representative of objects, or regions which exist in the physical world, then we may want to further restrict our definition of regions.

Given any straight line in the plane, we can cut the plane into two halves. These halves are regular open, each being the pseudocomplement (Definition 2.2.16) of the other, and are called *half-planes*.

Definition 4.1.3. A *basic polygon* in $\mathbb{R}^2$ is the intersection of finitely many half-planes in $\mathbb{R}^2$. A *polygon* in $\mathbb{R}^2$ is the sum, in $RO(\mathbb{R}^2)$ of a finite set of basic polygons. We denote the set of polygons in $\mathbb{R}^2$ by $ROP(\mathbb{R}^2)$.

Example 4.1.4. The set $ROP(\mathbb{R}^2)$ is a mereotopology over $\mathbb{R}^2$.

Proof. We need only show that $ROP(\mathbb{R}^2)$ is closed under the Boolean operations. This is obvious given the distribution laws for $RO(\mathbb{R}^2)$. □

We now present another mereotopology which will be used later to show that changing the definition of the regions does not necessarily affect the spatial logic.

If a line is defined by an equation such as $ax + by + c = 0$, where a, b and c are rational numbers, we call it a *rational line*; if a half-plane is bounded by a rational line, we call it a *rational half-plane*. A pair of rational lines can intersect only at points with rational coordinates.

Definition 4.1.5. A *rational basic polygon* is the intersection of finitely many rational half-planes. A *rational polygon* is the sum, in the Boolean algebra $RO(\mathbb{R}^2)$, of a finite set of rational basic polygons. We denote the set of rational polygons in $\mathbb{R}^2$ by $ROQ(\mathbb{R}^2)$.

Example 4.1.6. The set $ROQ(\mathbb{R}^2)$ is a mereotopology over $\mathbb{R}^2$.

Proof. Similar to Example 4.1.4. □

4.1.3 First Order Spatial Logic

Now, we have introduced structures which we have called *mereotopologies* which are essentially sets of elements which we refer to as *regions*. We now look at the construction of logics whose variables range over the elements (regions) of a mereotopology.

First, we must choose what spatial operations and relations we wish to be able to express. By choosing which spatial primitives we include, we select a signature, σ, for our logic. Given a signature σ, we denote the first-order language over σ by $\mathcal{L}(\sigma)$.

So, we choose a signature, σ, of our language, and we define the predicates, functions and constants in σ over the chosen mereotopology M. The resulting construction, which we write $M(\sigma)$, is known as a *structure*. A structure along with an interpretation function, allows us to determine the truth value of a given formula.

Let $M(\sigma)$ be a structure, and let α be a valuation function, mapping variables to regions of the mereotopology. If φ is true in $M(\sigma)$ under α, then we say that $M(\sigma)$ *satisfies* φ under α. If there exists an α such that $M(\sigma) \models_\alpha \varphi$, then we say that φ is *satisfiable in* $M(\sigma)$, and write $M(\sigma) \models \varphi$. If there exists a $M(\sigma)$ such that φ is satisfiable in $M(\sigma)$, then we say that φ is *satisfiable*.

Definition 4.1.7. We call a $\mathcal{L}(\sigma)$ formula with no free variables a *sentence*. Given an $\mathcal{L}(\sigma)$ sentence, φ, if $M(\sigma) \models_\alpha \varphi$ for some assignment α, then $M(\sigma) \models_\alpha \varphi$ for any assignment α, and we write $M(\sigma) \models \varphi$.

Definition 4.1.8. Given a structure $M(\sigma)$, the set of all $\mathcal{L}$ formulas φ such that $M(\sigma) \models \varphi$ is known as the *$\mathcal{L}$-theory* of $M(\sigma)$, written $Th(M(\sigma))$.

By choosing the signature σ of our language, and by defining what the regions of our mereotopology M are, we define the structure $M(\sigma)$, which is our model of space, or *spatial ontology*. Quoted from [PH01a]: *... a spatial ontology is a model of what we think space is like at the level of regions: it tells us what regions exist and what properties those regions have.*

The theory of a spatial ontology is simply all the true statements about the ontology. A theory $Th(M(\sigma))$ is clearly characterised by the spatial ontology $M(\sigma)$ that is, the choices relating to the definition of regions and the composition of the signature directly affect the contents of this theory. Given the theory of an ontology $M(\sigma)$, a number of questions are raised. What other ontologies characterise this theory? And how are these ontologies related to $M(\sigma)$? Are we able to further restrict the definition of the regions which make up M, while still characterising $Th(M(\sigma))$? The first of these questions is one of many investigated in [PH01a], and it is shown that the spatial ontologies $ROP(c, \leq)$ and $ROQ(c, \leq)$ have the same theory.

An *axiom system* of a spatial ontology is a set of axioms and rules of inference whose consequences are exactly the theory of the spatial ontology. Having a complete axiom system for a spatial ontology allows us to determine, for any given formula, whether that formula is true or satisfiable in the spatial ontology. Given a spatial ontology, an important question is whether its theory can be *axiomatised*, and if so, what are its axioms?

Choice of Signature

The operations $+$, $\cdot$ and $-$ are so commonly included in spatial logics, that they will be present in the signatures implicitly. Thus we write $\mathcal{L}(C)$ instead of $\mathcal{L}(+, \cdot, -, C)$. We take the symbols 0, 1, $+$, $\cdot$, $-$ and $\leq$ to have their usual (Boolean algebra) interpretations. We take the binary predicate C to denote a standard contact relation, that is, C holds between two regions if their topological closures intersect. The interpretation of these symbols is given in Table 4.1.

Symbol	Definition
0	$(0)^M = \varnothing$
1	$(1)^M = X$
$+$	$+^M(x, y) = ((x \cup y)^-)^\circ$
$\cdot$	$\cdot^M(x, y) = x \cap y$
$-$	$-^M(x) = X \setminus x^-$
C	$C^M = \{(x, y) \in M^2 : x^- \cap y^- \neq \varnothing\}$

Table 4.1: Definitions of simple constants and functions in a mereotopology, M, over topological space X $(x, y \in M \subseteq \mathbb{P}(X))$

We now look at the structures involving just a C contact relation.

4.1.4 Contact Relations

Since Whitehead's extensive connection relation, there have been a number of investigations of spatial logics using *contact relations*. The standard topological interpretation of a contact relation is that it holds between two subsets of a topological space, if their closures have a non empty intersection. We now give a more general definition of a contact relation.

Definition 4.1.9. We say that a binary relation C is a *contact relation* if it satisfies the following.

C1: $\forall x(C(x,x))$.

C2: $\forall x \forall y(C(x,y) \to C(y,x))$.

C3: $\forall x \forall y(\forall z(C(z,x) \leftrightarrow C(z,y)) \to x = y)$.

These axioms correspond to the axioms given by Clarke for his 'calculus of individuals' [Cla81].

Now we introduce the structures that result when these contact relations are taken to range over elements of Boolean algebras. The following definition is taken from [BD07].

Definition 4.1.10. Let A be a Boolean algebra, and let C be a binary relation on A. The pair $\langle A, C\rangle$ is a *Boolean contact algebra* if C has the following properties (for all $x, y, z \in A$).

BCA0: $C(x,y) \to x, y \neq 0$.

BCA1: $x \neq 0 \to C(x,x)$.

BCA2: C is symmetric.

BCA3: $C(x,y)$ and $y \leq z \to C(x,z)$.

BCA4: $C(x,y+x) \to C(x,y)$ or $C(x,z)$.

We are also interested in cases of C that have the following properties.

(Ext): $(C(x,z) \to C(y,z)) \to x \leq y$.

(Con): $x \notin \{0,1\} \to C(x,-x)$.

Note that C is a contact relation by Definition 4.1.9.

In [DW05], the following observations are made about BCAs in relation to regular closed subsets of a topological space, but they are also true for any mereotopology.

Observation 4.1.11. If $\langle X, U \rangle$ is a topological space, and M is a mereotopology over X, with C_U being a standard topological contact relation over M then the following hold.

1. $M(C_U) \models$ BCA0-4.
2. $M(C_U) \models$ (Ext) if and only if X is weakly regular (see Definition 2.2.15).
3. $M(C_U) \models$ (Con) if and only if X is connected (see Definition 2.2.19).

That is, the BCA0-4 axioms seem to capture the behaviour of a topological contact relation. However, there is a much stronger result regarding contact relations and these axioms.

BCA Representation Theorem

Similar to Stone's theorem, Düntsch & Winter give a representation theorem for BCAs in [DW05].

Theorem 4.1.12. *Each BCA $\langle A, C \rangle$ is isomorphic to a dense substructure of some regular closed algebra $\langle RC(X), C_U \rangle$, where $\langle X, U \rangle$ is a weakly regular T_1 space, and C is the restriction of C_U to A.*

As a result of the observations (similar to 4.1.11) in [DW05], we have the following theorem.

Theorem 4.1.13. *If $\langle X, U \rangle$ is a weakly regular T_1 space, and A is a dense subalgebra of $RC(X)$, with C being the restriction of the standard contact relation on $RC(X)$, then $\langle A, C \rangle$ is a BCA.*

And therefore, we have the following result.

Theorem 4.1.14. *The axioms of BCAs are complete with respect to the class of dense substructures of regular closed algebras of weakly regular T_1 spaces with standard contact.*

4.1.5 BCA Undecidability

The satisfiability problem of first order logic is undecidable. Similarly, first-order spatial logics are generally undecidable. The first order theory of the RCC (see

Section 4.1.1) was first observed to be undecidable by Gotts, Gooday, and Cohn [GGC96] and it was shown by Dornheim [Dor98] that the first-order theory of polygons in the real plane is undecidable.

We now present a proof of the undecidability of the first-order theory of BCAs by reducing the tiling problem to BCA language satisfiability. Dornheim's proof used the Post correspondence problem to show that first order mereotopology over the regular closed polygons was undecidable. First, we make some definitions that we use as shorthand, to improve the readability of the proof.

1. $EC(x, y) \equiv_{def} C(x, y) \wedge y \leq (-x)$
2. $O(x, y) \equiv_{def} x.y \neq \varnothing$
3. $Co(x, y) \equiv_{def} cc(x) \wedge x \leq y \wedge \neg\exists z(cc(z) \wedge x < z \leq y)$
4. $cc(x) \equiv_{def} \neg\exists x_1, x_2(x_1 \neq \varnothing \wedge x_2 \neq \varnothing \wedge x_1 + x_2 = x \wedge \neg C(x_1, x_2))$

We take formulae 1,2, and 3 to represent external contact, overlapping, and 'maximal component of' relations respectively and we take formula 4 to represent the property of having a connected closure.

Additionally, we make the following abbreviations.

$$\begin{aligned} J(x, y, X, K) \equiv_{def} & Co(x, X) \wedge Co(y, X) \wedge \\ & \exists j(Co(j, K) \wedge EC(x, j) \wedge O(j, y)) \end{aligned}$$

$$\begin{aligned} D(x, z, X) \equiv_{def} & Co(x, X) \wedge Co(z, X) \wedge \\ & \exists y_v(Co(y_v, X) \wedge J(x, y_v, X, V) \wedge J(y_v, z, X, H)) \wedge \\ & \exists y_h(Co(y_h, X) \wedge J(x, y_h, X, H) \wedge J(y_h, z, X, V)) \end{aligned}$$

The J relation states that x is connected to y by a (joiner) maximal component of K. The D relation ensures diagonal consistency between the tiles, that is, for any two tiles x and z, if z is above the tile to the right of x, then z is also to the right of the tile above x.

Now, let $\Phi(X, V, H, Y_1, \ldots, Y_N)$ be the following set of twelve formulae, with free variables: $X, V, H, Y_1, \ldots, Y_N$. We will use upper case letters for free variables, and lower case letters for bound variables.

1. $X \neq \varnothing \wedge \exists x(Co(x, X))$

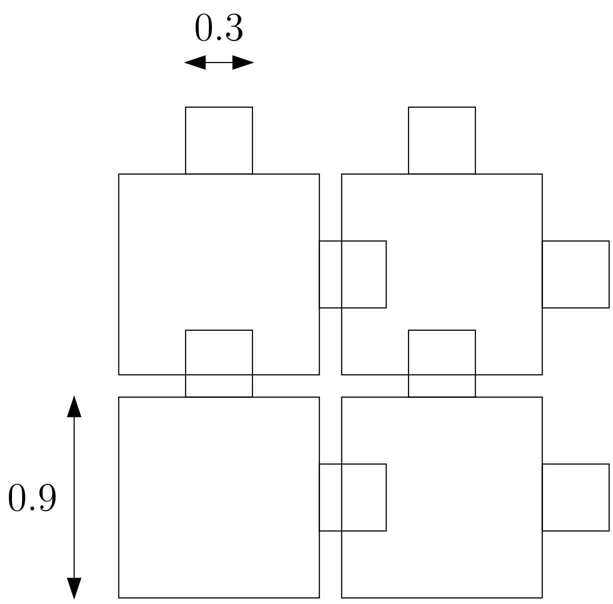

Figure 4.1: Layout of the tiles.

2. $X = Y_1 + \cdots + Y_N \wedge \bigwedge_{\substack{1 \le i,j \le N \\ i \ne j}} \neg C(Y_i, Y_j)$

3. $\forall x\big(Co(x, X) \to \neg Co(x, H) \wedge \neg Co(x, V)\big)$

4. $\neg C(V, H)$

5. $\forall x\big(Co(x, X) \to \exists y(J(x, y, X, V) \wedge \forall z(J(x, z, X, V) \to z = y))\big)$

6. $\forall x\big(Co(x, X) \to \exists y(J(x, y, X, H) \wedge \forall z(J(x, z, X, H) \to z = y))\big)$

7. $\forall x\big(Co(x, X) \to \exists y(D(x, y, X))\big)$

8. $\neg \exists x_1, x_2\big(J(x_1, x_2, X, V) \wedge (\bigvee_{\substack{1 \le i,j \le N \\ (i,j) \notin \mathcal{V}}} (x_1 \le Y_i \wedge x_2 \le Y_j))\big)$

9. $\neg \exists x_1, x_2\big(J(x_1, x_2, X, H) \wedge (\bigvee_{\substack{1 \le i,j \le N \\ (i,j) \notin \mathcal{H}}} (x_1 \le Y_i \wedge x_2 \le Y_j))\big)$

The strategy is that we construct all the tiles such that each tile has two 'joining' tiles which connect it to the tile above, and to the right, respectively. See Figure 4.1.

Lemma 4.1.15. *Given a tiling T for $(\mathcal{C}, \mathcal{V}, \mathcal{H})$ where $\mathcal{C} = c_1, \ldots, c_N$, we can construct $S, M, N, P_1, \ldots, P_N \in RC(\mathbb{R}^2)$ such that*

$$RC(\mathbb{R}^2) \vDash \Phi[S, M, N, P_1, \ldots, P_N].$$

Proof. Let S be the region consisting of the union of all square regions of $RC(\mathbb{R}^2)$, of side-length 0.9 whose bottom left corner is at a co-ordinate (i, j) of $\mathbb{R}^2$, for all $(i, j) \in \mathbb{N}^2$. We call the maximal components of S *squares*.

Let M be the region consisting of the union of all square regions of side-length 0.3 whose bottom left corner is at co-ordinate $(i + 0.3, j + 0.9)$ of $\mathbb{R}^2$, for all $(i, j) \in \mathbb{N}^2$. We call the maximal components of M *vertical joiners*.

Let N be the region consisting of the union of all square regions of side-length 0.3 whose bottom left corner is at a co-ordinate $(i + 0.9, j + 0.3)$ of $\mathbb{R}^2$, for all $(i, j) \in \mathbb{N}^2$. We call the maximal components of N *horizontal joiners*.

We construct a function $\tau : \mathbb{N}^2 \to S$ by setting $\tau(i, j)$ to be the square region (maximal component of S) whose bottom left co-ordinate is at (i, j). Let

$$P_k = \sum\{\tau(i, j) : i, j \in \mathbb{N} \text{ and } T(i, j) = c_k\}.$$

Now we show that the formulae of Φ are satisfied.

1. By the fact that S is not empty.

2. By the definition of P_k (and the fact that T is well-defined and total) $Y_1, \ldots, Y_N$ are pairwise disjoint, and X is the union of $Y_1, \ldots, Y_N$.

3. By the definition of S, M and N, no square can also be a joiner.

4. By the definition of M and N, a joiner cannot be both horizontal and vertical.

5. By the definition of S, M and N, each square is vertically joined to a distinct unique square.

6. By the definition of S, M and N, each square is horizontally joined to a distinct unique square.

7. By the definition of S, M, N and τ, for each square (i, j), the square in vertical contact $(i, j + 1)$ is in horizontal contact with $(i + 1, j + 1)$, while the square in horizontal contact $(i + 1, j)$ is below $(i + 1, j + 1)$.

8. By the definition of P_k and the fact that T respects $\mathcal{V}$.

9. By the definition of P_k and the fact that T respects $\mathcal{H}$.

□

Lemma 4.1.16. *If there exists a structure* $\mathfrak{A}$ *and* $S, M, N, P_1, \ldots, P_N \in A$ *such that*

$$\mathfrak{A} \models \Phi[S, M, N, P_1, \ldots, P_N],$$

then there exists a tiling T *for* $(\mathcal{C}, \mathcal{V}, \mathcal{H})$*, where* $\mathcal{C} = c_1, \ldots, c_N$*.*

Proof. First, we define a mapping $\sigma : \mathbb{N}^2 \to S$ such that:

$$\langle \sigma(i,j), \sigma(i+1,j) \rangle \in \mathcal{H}$$
$$\langle \sigma(i,j), \sigma(i,j+1) \rangle \in \mathcal{V}.$$

Formula 1 ensures that S is non-empty and has at least one maximal component, so we may choose a maximal component $s \subseteq S$ and set $\sigma(0,0) = s$.

We define the following sets.

$$I_v = \{(a,b) : \mathfrak{A} \models J[a,b,S,M] \text{ where } a,b \text{ are maximal components of } S\}$$
$$I_h = \{(a,b) : \mathfrak{A} \models J[a,b,S,N] \text{ where } a,b \text{ are maximal components of } S\}$$
$$I_d = \{(a,b) : \mathfrak{A} \models D[a,b,S] \text{ where } a,b \text{ are maximal components of } S\}$$

1. If $\sigma(0,j)$ is defined but $\sigma(0,j+1)$ isn't, then set $\sigma(0,j+1) = a'$ where $(\sigma(0,j), a') \in I_v$.

   To show that a' is unique we must show that for each a there is a unique b such that $\mathfrak{A} \models J[a,b,S,M]$. By formula 5, there exists exactly one square which is vertically joined to a.

2. If $\sigma(i,0)$ is defined but $\sigma(i+1,0)$ isn't, then set $\sigma(i+1,0) = a'$ where $(\sigma(i,0), a') \in I_h$.

   To show that a' is unique we must show that for each a there is a unique b such that $\mathfrak{A} \models J[a,b,S,N]$. By formula 6, there exists exactly one square which is horizontally joined to a.

3. If $\sigma(i, j)$ is defined but $\sigma(i + 1, j + 1)$ isn't, then set $\sigma(i + 1, j + 1) = a'$ where $(\sigma(i, j), a') \in I_d$

   To show that a' is unique we must show that for each a there is a unique b such that $\mathfrak{A} \models D[a, b, S]$. By forumla 7, there is at least one such b. By formulae 5 and 6, there is at most one such b.

Now we define $T : \mathbb{N}^2 \to \mathcal{C}$ by setting $T(i, j) = c_k$ where $\sigma(i, j)$ is a maximal component of P_k, for all $i, j \in \mathbb{N}$. Now we show that:

1. T is well defined. Assume the opposite, then $\sigma(i, j)$ is a maximal component of P_k and $\sigma(i, j)$ is a maximal component of P_l, so $P_k.P_l \neq \varnothing$. However, by formula 2, $\neg C(P_k, P_l)$.

2. T respects $\mathcal{V}$ and $\mathcal{H}$. Assume the opposite. Suppose T does not respect $\mathcal{V}$. There exist $i, j \in \mathbb{N}$ such that $\langle T(i, j), T(i, j + 1)\rangle \notin \mathcal{V}$, it is straighforward to show that $\mathfrak{A} \models J[\sigma(i, j), \sigma(i, j + 1), S, M]$. However, this contradicts formula 8. The proof that T respects $\mathcal{H}$ is similar.

□

We have the following theorem.

Theorem 4.1.17. *The formula $\Phi(S, M, N, P_1, \ldots, P_N)$ is satisfiable if and only if there exists a tiling T for $(\mathcal{C}, \mathcal{V}, \mathcal{H})$ where $\mathcal{C} = c_1, \ldots, c_N$.*

Proof. By Lemmas 4.1.15 and 4.1.16. □

4.2 Conclusion

Many questions regarding spatial logics remain unanswered. However, this thesis focuses on questions relating to the computational properties of a theory of space. For instance, is there an algorithm which can determine whether any given sentence is a member of that theory? If so, how complex is its satisfiability problem?

Given a formula in a spatial logic and a specific domain over which the variables of that logic may range, is there an assignment of variables to elements of the domain such that the formula is satisfied? This is the *topological inference problem.*

In this chapter, we introduced many of the concepts that we make extensive use of throughout the rest of the thesis. But the overall aim of the chapter has been to introduce the topological inference problem so that we may now examine the computational properties of this problem.

We have seen that the first order language with a contact relation is undecidable, however there are many less expressive languages which are decidable. The following chapters are concerned with investigating the computational properties of these languages.

Chapter 5

Satisfiability

This chapter forms the second part of the new survey of spatial logics provided by this thesis. In this chapter, we look at different approaches to solving the topological inference problem. We start with with a simple spatial logic called $\mathcal{T}$, giving a proof of its equivalence to the modal logic $S4$. The rest of the chapter examines some languages which are related to $\mathcal{T}$. We look at $\mathcal{TCC}$ which is a superset of $\mathcal{T}$, and over this chapter and the next, we look at a series of languages which are progressively stronger restrictions of $\mathcal{T}$, including RCC8.

5.1 Modal Logic and Topology

In Observation 2.2.17, we noted the similarity between properties of a topological space and the $S4$ modal logic axioms. In fact, there is a much deeper connection between the $S4$ modal logic and topological spaces. McKinsey & Tarski ([McK41] and [MT44]) show that when the modal $\Diamond$ operator is taken to mean topological closure, $S4$ is complete with respect to the interpretation over topological spaces.

The most common interpretation of modal logics is the relational semantics, using Kripke structures, as discussed in Section 2.1. The logic $S4$ is known to be complete with respect to the Kripke frames whose reachability relation is reflexive and transitive. There is a strong connection between $S4$ Kripke frames and a type of topological space known as an *Alexandroff* space (see Definition 2.2.6). First, we introduce the following definition.

Definition 5.1.1. Let $\mathfrak{F} = \langle W, \rightarrowtail \rangle$ be an $S4$ Kripke frame (so $\rightarrowtail$ is reflexive and transitive). Given $W' \subseteq W$, we define the following set.

$$\mathbb{I}_{\mathfrak{F}}(W') = \{\ w \in W \mid \forall w' : w \rightarrowtail w' \implies w' \in W'\ \}$$

We can see that $\mathbb{I}^{\mathfrak{F}}$ is an interior operator according to Definition 2.2.18, and so defines a topology on W. We call $\langle W, \mathbb{I}^{\mathfrak{F}} \rangle$ the *topological space defined by* $\mathfrak{F}$.

It is straightforward to show that the topological space defined by an $S4$ Kripke frame is an Alexandroff space. In fact, given a topological space X, if we define a relation $x \rightarrowtail x'$ for all $x, x' \in X$ if and only if $x \in \{x'\}^-$, then it is easy to show that $\rightarrowtail$ is reflexive and transitive if and only if X is an Alexandroff space.

This implies that there is a 1-1 correspondence between $S4$ Kripke frames, and Alexandroff spaces. And since every *finite* topological space is an Alexandroff space, there is a 1-1 correspondence between finite $S4$ Kripke frames and finite topological spaces. Essentially, what this means is that the theory of the $S4$ language over finite topological spaces is equivalent to the more standard theory of $S4$ over finite Kripke frames.

5.1.1 Simple Topological Constraints

We will now introduce a simple topological language, which was first introduced by Nutt ([Nut99]). We shall call this topological constraint language $\mathcal{T}$.

Definition 5.1.2. We define the terms of $\mathcal{T}$ as follows.

1. Every variable is a term.

2. If s is a term, so are $-s$, s^-, and s°.

3. If s and t are terms, so are $s \cap t$ and $s \cup t$.

The domain of the $\mathcal{T}$ language is the set $\mathbb{P}(X)$ where $\langle X, \mathcal{U} \rangle$ is a topological space. As usual, we denote the domain of this language by the symbol U. We extend the interpretation functions of this language over the connectives $-$, $\cup$ and $\cap$ and the interior and closure operators as follows (where s and t are terms).

$\alpha(-s) = X \setminus (\alpha(s))$

$\alpha(s^-) = (\alpha(s))^-$ and $\alpha(s^\circ) = (\alpha(s))^\circ$

$\alpha(s \cap t) = \alpha(s) \cap \alpha(t)$ and $\alpha(s \cup t) = \alpha(s) \cup \alpha(t)$

Thus the interpretation of a term s is the element of U described by s. We define the constraints of $\mathcal{T}$ as follows.

Definition 5.1.3. If s and t are terms, then $s = t$ is a constraint and so is $s \neq t$.

Let the language $\mathcal{T}$ be the set of all $\mathcal{T}$ formulae.

Definition 5.1.4. Let $\varphi, \psi \in \mathcal{T}$. We define a $\mathcal{T}$ formula as follows.

1. Every $\mathcal{T}$ constraint is a formula.
2. $\neg\varphi$ is a formula.
3. $\varphi \wedge \psi$ is a formula.
4. $\varphi \vee \psi$ is a formula.

The operators $\wedge$, $\vee$ and $\neg$ are interpreted as the logical operators conjunction, disjunction, and negation respectively. We may refer to a conjunction of formulae as a *set of $\mathcal{T}$ constraints.*

Now we will define what it means for a $\mathcal{T}$ formula to be satisfiable.

Definition 5.1.5. We define a *topological model* to be a pair (U, α) which consists of a set U of subsets of a topological space and a function α mapping variables to the elements of U.

Definition 5.1.6. Let s and t be $\mathcal{T}$ terms and let $\varphi, \psi \in \mathcal{T}$. Given a topological model $\mathfrak{A} = (U, \alpha)$ we say the following.

If φ is $s = t$, then $\mathfrak{A}$ *satisfies* φ iff $\alpha(s) = \alpha(t)$.
If φ is $s \neq t$, then $\mathfrak{A}$ *satisfies* φ iff $\alpha(s) \neq \alpha(t)$.

If $\mathfrak{A}$ satisfies a $\mathcal{T}$ formula, φ, then we write $\mathfrak{A} \models \varphi$. We can extend $\models$ over Boolean combinations of $\mathcal{T}$ formulae in the following way.

$$\begin{array}{lll}
\mathfrak{A} \models \neg\varphi & \text{iff} & \mathfrak{A} \not\models \varphi \\
\mathfrak{A} \models \varphi \wedge \psi & \text{iff} & \mathfrak{A} \models \varphi \text{ and } \mathfrak{A} \models \psi \\
\mathfrak{A} \models \varphi \vee \psi & \text{iff} & \mathfrak{A} \models \varphi \text{ or } \mathfrak{A} \models \psi \\
\mathfrak{A} \models \varphi \rightarrow \psi & \text{iff} & \mathfrak{A} \not\models \varphi \text{ or } \mathfrak{A} \models \psi
\end{array}$$

Given a $\mathcal{T}$ formula, φ, we say that φ is *satisfiable* if there is a topological model, $\mathfrak{A}$, such that $\mathfrak{A} \models \varphi$.

Using the connection discussed in earlier on in the chapter, Nutt showed the $\mathcal{T}$ satisfiability problem to be equivalent to the $S4$ satisfiability problem, and therefore, that the $\mathcal{T}$ satisfiability problem is PSPACE-complete.

Now, we give a proof of the equivalence of the $S4$ satisfiability problem with the $\mathcal{T}$ satisfiability problem (over arbitrary subsets of a topological space).

5.1.2 Translating Topological Constraints

First, we define the translation of the language $\mathcal{T}$ into $S4_U$. Given a term s, we denote the translation into $S4_U$ of s by $(s)^*$. We translate variables of $\mathcal{T}$ into a corresponding proposition letter, $(x_i)^* = p_i$, where x_i is a variable of $\mathcal{T}$, and p_i is a proposition letter. We then structurally extend our translation over the terms of $\mathcal{T}$ as follows (where s and t are terms, and the symbols $\neg$, $\wedge$ and $\Box$ represent $S4_U$ symbols):

1. $(-s)^* = \neg(s)^*$,
2. $(s \cup t)^* = (s)^* \wedge (t)^*$,
3. $(s^\circ)^* = \Box(s)^*$.

It is trivial to extend this translation to cover formulae of $\mathcal{T}$ (where s is a term, and $\forall$ and $\exists$ are the $S4_U$ universal modal operators):

1. $(s = \top)^* = \forall(s)^*$,
2. $(s \neq \top)^* = \neg\forall(s)^*$,
3. $(s = \bot)^* = \neg\exists(s)^*$,
4. $(s \neq \bot)^* = \exists(s)^*$.

This kind of translation was first proposed by Bennett [Ben96] specifically for formulae of a more restricted fragment of $\mathcal{T}$ called RCC8, which we shall see later on in this chapter. We will now show the correctness of this translation with respect to the formulae of $\mathcal{T}$, which is more general than RCC8.

First we must show that if we have a $\mathcal{T}$ formula φ and a topological model $\mathfrak{A}$, then we can construct an $S4_U$ Kripke model $\mathfrak{M}_\mathfrak{A}$ such that $\mathfrak{A} \models \varphi$ if and only if $\mathfrak{M}_\mathfrak{A} \models \varphi^*$.

Lemma 5.1.7. *Given a finite topological model* $\mathfrak{A} = \langle (X, U), \alpha \rangle$, *let* $\mathfrak{M}_{\mathfrak{A}} = \langle X, \rightarrowtail, \eta \rangle$, *where*

$$v \rightarrowtail v' \text{ if and only if } v \in \{v'\}^- \text{ (for } v, v' \in X\text{)},$$

and

$$\eta(p_i) = \alpha(x_i),$$

for all variables x_i. *Then* $\mathfrak{M}_{\mathfrak{A}}$ *is an* S4-*Kripke model.*

Proof. $\mathfrak{M}_{\mathfrak{A}}$ is an *S4*-Kripke model if and only if the relation $\rightarrowtail$ is reflexive and transitive. Since $v \in \{v\}^-$, $\rightarrowtail$ is reflexive. Suppose (i) $v \in \{v'\}^-$ and (ii) $v' \in \{v''\}^-$ for some $v'' \in X$. By (ii), $v' \subseteq \{v''\}^-$, $\{v'\}^- \subseteq (\{v''\}^-)^-$ and $\{v'\}^- \subseteq \{v''\}^-$. By (i), $v \in \{v'\}^- \subseteq \{v''\}^-$, i.e. $v \in \{v''\}^-$, therefore $\rightarrowtail$ is transitive. □

Lemma 5.1.8. *If* s *is a* $\mathcal{T}$ *term, then for every* $w \in X$,

$$\mathfrak{M}_{\mathfrak{A}} \models_w s^* \text{ if and only if } w \in s^{\mathfrak{A}}.$$

Proof. By Structural induction on s^* (where t_1 and t_2 are terms):

1. If s is a variable, x_i
   $w \in {x_i}^{\mathfrak{A}} \Leftrightarrow w \in \alpha(x_i) \Leftrightarrow w \in \eta(p_i) \Leftrightarrow \mathfrak{M}_{\mathfrak{A}} \models_w p_i \Leftrightarrow \mathfrak{M}_{\mathfrak{A}} \models_w s^*$

2. If s is $t_1 \cap t_2$

$$\begin{aligned} w \in (t_1 \cap t_2)^{\mathfrak{A}} &\Leftrightarrow w \in {t_1}^{\mathfrak{A}} \text{ and } w \in {t_2}^{\mathfrak{A}} \\ &\Leftrightarrow \mathfrak{M}_{\mathfrak{A}} \models_w (t_1)^* \text{ and } \mathfrak{M}_{\mathfrak{A}} \models_w (t_2)^* \qquad (IH) \\ &\Leftrightarrow \mathfrak{M}_{\mathfrak{A}} \models_w (t_1)^* \wedge (t_2)^* \\ &\Leftrightarrow \mathfrak{M}_{\mathfrak{A}} \models_w (t_1 \cap t_2)^* \\ &\Leftrightarrow \mathfrak{M}_{\mathfrak{A}} \models_w s^* \end{aligned}$$

3. If s is $-t_1$
   $w \in s^{\mathfrak{A}} \Leftrightarrow w \in (-t_1)^{\mathfrak{A}} \Leftrightarrow w \notin {t_1}^{\mathfrak{A}} \Leftrightarrow \mathfrak{M}_{\mathfrak{A}} \not\models_w (t_1)^* \ (IH) \Leftrightarrow \mathfrak{M}_{\mathfrak{A}} \models_w s^*$

4. If s is t_1^-

$$\begin{aligned} \mathfrak{M}_{\mathfrak{A}} \models_w s^* &\Rightarrow \exists w' \text{ such that } w \rightarrowtail w' \text{ and } \mathfrak{M}_{\mathfrak{A}} \models_{w'} (t_1)^* \\ &\Rightarrow w \rightarrowtail w' \text{ and } w' \in {t_1}^{\mathfrak{A}} \\ &\Rightarrow w \in w'^- \text{ and } w' \in {t_1}^{\mathfrak{A}} \\ &\Rightarrow w \in ({t_1}^{\mathfrak{A}})^- = t_1^{-\mathfrak{A}}. \end{aligned}$$

Conversely, suppose $w \in t_1^{-\mathfrak{A}} = (t_1{}^{\mathfrak{A}})^-$. Since $\mathfrak{A}$ is finite, $(t_1{}^{\mathfrak{A}}) = \{w_1', \ldots, w_k'\}$ hence $(t_1{}^{\mathfrak{A}})^- = w_1'^- \cup \cdots \cup w_k'^-$. Without loss of generality, $w \in w_1'^-$. Hence wRw_1', and (IH) $\mathfrak{M}_{\mathfrak{A}} \models_{w_1'} (t_1)^*$. Hence $\mathfrak{M}_{\mathfrak{A}} \models_w \Diamond(t_1)^*$, i.e. $\mathfrak{M}_{\mathfrak{A}} \models_w s^*$

□

Lemma 5.1.9. *If φ is a formula of $\mathcal{T}$ and $\mathfrak{A}$ is a finite topological model, then $\mathfrak{M}_{\mathfrak{A}} \models \varphi^*$ if and only if $\mathfrak{A} \models \varphi$.*

Proof. Will show for the case that φ is $s = \top$, where s is a term.
$s = X \Leftrightarrow \forall u \in X\ u \in s^{\mathfrak{A}} \Leftrightarrow \forall u \in X\ u \in (s)^*$ (by Lemma 5.1.8) $\Leftrightarrow \forall (s)^*$. □

Lemma 5.1.10. *Given a $\mathcal{T}$ formula φ, as defined in Definition 5.1.3, then φ is satisfiable in a topological model if and only if it is satisfiable in a finite topological model*

Proof. The *small model property* in Section 4.2 of [PH01b] shows this for $\mathcal{TCC}$ (see Section 5.2) formulae of which $\mathcal{T}$ formulae are a restricted case. □

Theorem 5.1.11. *Let $\mathfrak{A}$ be a topological model, let φ be a $\mathcal{T}$ formula. We can construct an $S4_U$ Kripke model $\mathfrak{M}_{\mathfrak{A}}$ such that $\mathfrak{A} \models \varphi$ if and only if $\mathfrak{M}_{\mathfrak{A}} \models \varphi^*$.*

Proof. Lemma 5.1.9 proves this theorem if $\mathfrak{A}$ is finite. Lemma 5.1.10 shows that if φ is satisfiable, it is satisfiable in a finite topological model. □

Now we must show that if we have a $\mathcal{T}$ formula φ and a $S4_U$ Kripke model $\mathfrak{M}$ then we can construct a topological model $\mathfrak{A}_{\mathfrak{M}}$ such that $\mathfrak{M} \models \varphi^*$ if and only if $\mathfrak{A}_{\mathfrak{M}} \models \varphi$.

Lemma 5.1.12. *Let $\mathfrak{M} = \langle W, \rightarrowtail, \eta \rangle$ be an $S4$-Kripke model, and let $\mathbb{I}_{\mathfrak{M}}$ be the topology defined by $\mathfrak{M}$, as in Lemma 5.1.1. Let $\mathfrak{A}_{\mathfrak{M}} = \langle (W, \mathbb{I}_{\mathfrak{M}}), \alpha \rangle$ be the topological model such that the valuation function α is defined as*

$$\alpha(x_i) = \eta(p_i) \text{ for all proposition letters } p_i.$$

Then, for every $w \in W$,

$$w \in s^{\mathfrak{A}_{\mathfrak{M}}} \text{ if and only if } \mathfrak{M} \models_w s^*;$$

where s is a $\mathcal{T}$ term.

Proof. Proof is similar in structure to that given in Lemma 5.1.8, except for the fourth part.

4. If s^* is $\Box(t_1)^*$:

$$\begin{aligned}
\mathfrak{M} \models_w \Box(t_1^*) &\Leftrightarrow \forall w' : w \rightarrowtail w' \implies \mathfrak{M} \models_{w'} (t_1)^* \\
&\Leftrightarrow w \rightarrowtail w' \implies w' \in {t_1}^{\mathfrak{A}_{\mathfrak{M}}} && \text{(IH)} \\
&\Leftrightarrow w \in \mathbb{I}_{\mathfrak{M}}({t_1}^{\mathfrak{A}_{\mathfrak{M}}}) && \text{(def. interior op, Lemma 5.1.1)} \\
&\Leftrightarrow w \in (\mathbb{I}_{\mathfrak{M}} t_1)^{\mathfrak{A}_{\mathfrak{M}}}
\end{aligned}$$

□

Theorem 5.1.13. *Let $\mathfrak{M}$ be an $S4_U$ Kripke model, let φ^* be a modal formula. We can construct a topological model $\mathfrak{A}_{\mathfrak{M}}$ such that $\mathfrak{M} \models \varphi^*$ if and only if $\mathfrak{A}_{\mathfrak{M}} \models \varphi$.*

Proof. Proof is similar to Lemma 5.1.9. □

Corollary 5.1.14. *A $\mathcal{T}$ formula φ is satisfiable if and only if its corresponding $S4_U$ translation, φ^*, is satisfiable.*

Proof. Direct result of Theorems 5.1.11 and 5.1.13. □

Corollary 5.1.14 proves the correctness of the translation from the $\mathcal{T}$ to the $S4_U$ modal logic. The following theorem can be found in Halpern & Moses [HM92].

Theorem 5.1.15. *Let $\varphi_1, \ldots, \varphi_m, \psi_1, \ldots, \psi_n$ be $S4_U$ formulae. Then*

$$\forall\varphi_1 \wedge \cdots \wedge \forall\varphi_m \wedge \exists\psi_1 \wedge \cdots \wedge \exists\psi_n \tag{5.1}$$

is satisfiable if and only if for each $j \leq m$ the formula

$$\varphi_1 \wedge \cdots \wedge \varphi_m \wedge \psi_j \tag{5.2}$$

is satisfiable.

Observe that all the $S4_U$ formulae resulting from the $\mathcal{T}$ translations are of the form given in formula (5.1), such that the formula given in (5.2) will be an $S4$ formula. The satisfiability of $S4$ formulae is known to be PSPACE-complete [HM92], so the following corollary is immediate.

Corollary 5.1.16. *The satisfiability problem of $\mathcal{T}$ formulae is PSPACE-complete.*

5.2 Topological Component Counting

Now we look at the satisfiability problem for a spatial logic called $\mathcal{TCC}$[1]. This language is a superset of $\mathcal{T}$, with the addition of a predicate which is interpreted as a kind of topological connectedness property.

As with the $\mathcal{T}$ language, the domain U of $\mathcal{TCC}$ will be the subsets of a topological space X, and we let α be a function mapping variables to elements of U. We denote the interpretation of a term s in the topological space by $\alpha(s)$. The interpretation of a variable x is the element of U which is mapped to x by α. The interpretation is extended over the connectives $-$, $\cup$ and $\cap$ and the interior and closure operators exactly as with $\mathcal{T}$.

Definition 5.2.1. We define a $\mathcal{TCC}$ constraint as follows (where s and t are $\mathcal{TCC}$ terms).

1. $s = t$ is a constraint.
2. $s \neq t$ is a constraint.
3. $c^{\leqslant k}(s)$ is a constraint (if k is a binary numeral and $k \geqslant 0$).
4. $c^{\geqslant k}(s)$ is a constraint (if k is a binary numeral and $k \geqslant 1$).

Although the parameter k is a binary numeral, in order to simplify the notation, we will treat k as the integer value which the binary numeral represents.

Note that any type 1 or 2 constraint can be converted to the form $s = \top$ or $s \neq \top$ (using $s = t \Leftrightarrow (-s \cup t) \cap (s \cup -t) = \top$).

We define the formulae of $\mathcal{TCC}$ in exactly the same way as $\mathcal{T}$ (Definition 5.1.4) with the addition of permitting type 3 and type 4 constraints.

Now we define what it means for a $\mathcal{TCC}$ formula to be satisfiable.

Definition 5.2.2. Let the language $\mathcal{TCC}$ be the set of all $\mathcal{TCC}$ formulae, let s and t be $\mathcal{TCC}$ terms, and let $\varphi, \psi \in \mathcal{TCC}$. Given a topological model $\mathfrak{A}$, we say that:

if φ is $s = t$, then	$\mathfrak{A}$ *satisfies* φ	iff $\alpha(s) = \alpha(t)$,
if φ is $s \neq t$, then	$\mathfrak{A}$ *satisfies* φ	iff $\alpha(s) \neq \alpha(t)$,
if φ is $c^{\leqslant n}(s)$, then	$\mathfrak{A}$ *satisfies* φ	iff $\alpha(s)$ has at most n components,
if φ is $c^{\geqslant n}(s)$, then	$\mathfrak{A}$ *satisfies* φ	iff $\alpha(s)$ has at least n components.

[1]Rough abbreviation of "topological constraint language with component counting"

If $\mathfrak{A}$ satisfies a $\mathcal{TCC}$ formula, φ, then we write $\mathfrak{A} \models \varphi$. We can extend $\models$ over Boolean combinations of $\mathcal{TCC}$ formulae in the following way:

$$\begin{array}{lll} \mathfrak{A} \models \neg\varphi & \text{iff} & \mathfrak{A} \not\models \varphi, \\ \mathfrak{A} \models \varphi \wedge \psi & \text{iff} & \mathfrak{A} \models \varphi \text{ and } \mathfrak{A} \models \psi, \\ \mathfrak{A} \models \varphi \vee \psi & \text{iff} & \mathfrak{A} \models \varphi \text{ or } \mathfrak{A} \models \psi, \\ \mathfrak{A} \models \varphi \rightarrow \psi & \text{iff} & \mathfrak{A} \not\models \varphi \text{ or } \mathfrak{A} \models \psi. \end{array}$$

Given a $\mathcal{TCC}$ formula, φ, we say that φ is *satisfiable* if there exists a topological model, $\mathfrak{A}$, such that $\mathfrak{A} \models \varphi$.

$\mathcal{TCC}$ examples

Here are some examples of what we can express in $\mathcal{TCC}$:

Example 5.2.3. Let s, t be terms.

$$s \cap t \neq \bot$$

Let $\mathfrak{A} = (U, \alpha)$ be a model of this formula. Then $\alpha(s)$ and $\alpha(t)$ intersect, or overlap.

Example 5.2.4. Let s, t be terms.

$$s \cap t = \bot \ \wedge \ s^- \cap t^- \neq \bot$$

Let $\mathfrak{A} = (U, \alpha)$ be a model of this formula. Then $\alpha(s)$ and $\alpha(t)$ touch, i.e. their closures overlap, but $\alpha(s)$ and $\alpha(t)$ do not overlap.

Example 5.2.5. Let s be a term.

$$c^{\leqslant 1}(s) \ \rightarrow \ c^{\leqslant 1}(s^-)$$

Any topological model is a model of this formula, as the closure of any connected set is connected.

Example 5.2.6. Let r, s, t be terms.

$$c^{\leqslant 1}(r) \ \wedge \ r = s \cup t \ \wedge \ s^- \cap t^- = \varnothing \ \wedge \ r \neq \varnothing \ \wedge \ t \neq \varnothing$$

This formula is unsatisfiable, as there is no topological model under which a connected set may be split into two non-empty disjoint sets.

We will now give an outline of the proof of membership in the NEXP time complexity class for the satisfiability problem of the $\mathcal{TCC}$ language. The full proof can be found in [PH01b]. First, we give a brief outline of a proof of the *small model property* (which is presented fully in [PH01b]).

Let $\langle X,\mathcal{U}\rangle$ be a topological space, and let $\mathbf{B}$ be a finite set of subsets of X, such that $B \in \mathbf{B} \Rightarrow B^{-} \in \mathbf{B}$. Define a binary relation $\sim$ on X by setting $a \sim b$ if, for all $B \in \mathbf{B}$, $a \in B \Leftrightarrow b \in B$.

Definition 5.2.7. Let $G = X/\sim$ and define a function $f : X \to G$ as

$$f(a) = \{b \in X \mid b \sim a\}$$

Note that if $a \in X$ and $B \in \mathbf{B}$, then $a \in B$ if and only if $f(a) \in f(B)$. Also, note that $|G| < 2^{|\mathbf{B}|}$.

Let us define a binary relation $\to$ on G as

$$v \to v' \text{ if and only if, for all } B \in \mathbf{B},\ v \in f(B) \Rightarrow v' \in f(B^{-})$$

Definition 5.2.8. The reflexive, transitive, directed graph, $\langle G, \to\rangle$ can be regarded as a topological space with a closure operator $^{-G}$ for any $H \subseteq G$, defined as follows:

$$H^{-G} = \{\ v' \in G \mid v \to v' \text{ for some } v \in H\ \}.$$

For the duration of this section, we will use $^{-}$ to denote the closure operator in the topological space X, and we will use $^{-G}$ to denote the closure operator in the topological space G. If we are defining a topological space with a closure operator, then from now on, instead of writing $\langle X,\mathcal{U}\rangle$, we will abbreviate it to merely X.

The following lemmas establish properties of the function $f : X \to G$. Proofs for the following can be found in [PH01b].

Lemma 5.2.9. *The function $f : X \to G$ is continuous.*

Given a $\mathcal{TCC}$ formula, the size of φ, i.e. the number of symbols occurring in φ, is written as $|\varphi|$.

Let φ be a set of $\mathcal{TCC}$ constraints, which contains no constraints of type 4. Let $\mathfrak{A} = \langle X, \alpha\rangle$ such that $\mathfrak{A} \models \varphi$, and given a $\mathcal{TCC}$ term, s, let $s^{\mathfrak{A}}$ be the interpretation of term s in $\mathfrak{A}$.

Let

$$\begin{aligned} \mathbf{B}' &= \{ s^{\mathfrak{A}} \mid s \text{ is a term occurring in } \varphi \} \\ \mathbf{B} &= \mathbf{B}' \cup \{ B^- \mid B \in \mathbf{B}' \} \end{aligned}$$

Defined in this way, $\mathbf{B}$ is a finite set of subsets of X, such that $B \in \mathbf{B} \Rightarrow B^- \in \mathbf{B}$. And defining $f : X \to G$ as above, let $f(\mathfrak{A})$ be the structure $\langle f(X), f \circ \alpha \rangle$.

Lemma 5.2.10. *For each term s in φ,*

$$f(s^{\mathfrak{A}}) \quad = \quad s^{f(\mathfrak{A})}.$$

Theorem 5.2.11. *If $\mathfrak{A} \vDash \varphi$, and f is defined as above, then $f(\mathfrak{A}) \vDash \varphi$.*

Proof. The following proof is presented unchanged from what is presented in [PH01b]. We will present the proof for types 1, 2 and 3 separately. Let s and t be $\mathcal{TCC}$ terms.

Type 1: $\mathfrak{A} \models s = t \Rightarrow s^{\mathfrak{A}} = t^{\mathfrak{A}} \Rightarrow f(s^{\mathfrak{A}}) = f(t^{\mathfrak{A}}) \Rightarrow s^{f(\mathfrak{A})} = t^{f(\mathfrak{A})}$ by Lemma 5.2.10 $\Rightarrow f(\mathfrak{A}) \vDash s = t$.

Type 2: $\mathfrak{A} \models s \neq t \Rightarrow s^{\mathfrak{A}} \neq t^{\mathfrak{A}} \Rightarrow f(s^{\mathfrak{A}}) \neq f(t^{\mathfrak{A}})$ by Definition 5.2.7 (since both $s^{\mathfrak{A}}$ and $t^{\mathfrak{A}}$ are in $\mathbf{B}$) $\Rightarrow s^{f(\mathfrak{A})} \neq t^{f(\mathfrak{A})}$ by Lemma 5.2.10 $\Rightarrow f(\mathfrak{A}) \vDash s \neq t$.

Type 3: $\mathfrak{A} \models c^{\leqslant k}(s) \Rightarrow s^{\mathfrak{A}}$ has at most k components $\Rightarrow f(s^{\mathfrak{A}})$ has at most k components, by Lemma 5.2.9 $\Rightarrow s^{f(\mathfrak{A})}$ has at most k components, by Lemma 5.2.10 $\Rightarrow f(\mathfrak{A}) \vDash c^{\leqslant k}(s)$.

□

Corollary 5.2.12. *If φ is a satisfiable set of constraints of types 1,2 and 3, then φ is satisfied in a structure of size bounded by $2^{2|\varphi|}$.*

Proof. By Theorem 5.2.10, and the fact that $|\mathbf{B}|$ has at most $2|\varphi|$ elements meant that $f(\mathfrak{A})$ is of maximum size $2^{2|\varphi|}$. □

Building Models

Corollary 5.2.12 tells us that if a set of $\mathcal{TCC}$ constraints, of types 1, 2 or 3, is satisfiable, then it is satisfiable in a finite model; it also tells us the maximum possible size of this model. Given a model, $\mathfrak{A}$, which satisfies a $\mathcal{TCC}$ constraint set, the function f will generate a finite model from $\mathfrak{A}$ which satisfies the constraint set.

Let φ be a set of $\mathcal{TCC}$ constraints which contains no constraints of type 4. Let $\mathfrak{A}$ be a topological model such that $\mathfrak{A} \models \varphi$.

Let us also define another set:

$$\begin{aligned} \mathbf{T}' &= \{ \ s \mid s \text{ is a term occurring in } \varphi \ \} \\ \mathbf{T} &= \mathbf{T}' \cup \{ \ s^- \mid s \in \mathbf{T}' \ \} \end{aligned}$$

in this case, s^- is a new term resulting from applying the closure operator $^-$ to the term s. The set $\mathbb{P}(\mathbf{T})$ is the set of subsets of $\mathbf{T}$. Where S is a set of terms, $\{s_1, \ldots, s_n\}$, $\bigcap S$ represents the intersection of each element of the set, i.e. $s_1 \cap \cdots \cap s_n$.

Now we can give a slightly different definition for the equivalence relation; let us define an equivalence relation, $\sim$, on U as $a \sim b$ if, for all $s \in \mathbf{T}$, $a \in s^{\mathfrak{A}} \Leftrightarrow b \in T^{\mathfrak{A}}$. Now, if $G = U/\sim$, then for each equivalence class, $e \in G$, we can say that e is defined by a set of terms, S, where S is the set of all and only those $s \in \mathbf{T}$ such that for every point $p \in e$, $p \in s^{\mathfrak{A}}$.

Thus, we can define a function $g : G \to \mathbb{P}(\mathbf{T})$

$$g(e) = \{s \mid \forall p \in e, \ p \in s^{\mathfrak{A}}\}.$$

which maps each equivalence class, e, to the set of terms which define e.

Because $\mathbb{P}(\mathbf{T})$ contains the sets of all possible combinations of elements of $\mathbf{T}$, we might say that it also contains the definitions for all possible equivalence classes.

However, because $\mathbb{P}(\mathbf{T})$ contains all possible combinations of elements of $\mathbf{T}$, it also contains combinations which cannot possibly define an equivalence class. For example, if $\mathbf{T} = \{x \cap y, x, y, (x \cap y)^-, x^-, y^-\}$, then the equivalence class represented by the set $\{x\}$ is invalid, as it is not possible, given a point p, that $p \in x^{\mathfrak{A}}$ and $p \notin (x^-)^{\mathfrak{A}}$. Likewise, the equivalence class represented by the set $\{x \cap y, x, (x \cap y)^-, x^-\}$ is invalid, as it is not possible, given a point, that p, $p \in (x \cap y)^{\mathfrak{A}}$ and $p \notin y^{\mathfrak{A}}$

Given an element, S, of $\mathbb{P}(\mathbf{T})$, we define

$$S^{max} = S \cup \{-s \mid s \in (\varphi \setminus S)\}.$$

For every element in S, S^{max} contains the element or the negation of the element.

Given that S is a set of terms, saying that S is topologically consistent is equivalent to saying that

$$\bigcap S \neq \varnothing$$

is satisfiable. Since the above is a type 2 constraint, the problem of determining the consistency of S is an instance of *S4*-SAT (which is in the PSPACE complexity class).

Definition 5.2.13. We say that a set of terms, S, defines an *equivalence class* if and only if S^{max} is topologically consistent.

We will now define a set which consists of all the combinations of $\mathbf{T}$ which define a valid equivalence class.

Definition 5.2.14. Given a set of terms, $\mathbf{T}$, the set of all possible valid equivalence classes is defined as:

$$\mathbf{S}^{eq} = \{\ S \mid S \in \mathbb{P}(\mathbf{T}) \text{ and } (\bigcap S^{max} \neq \varnothing) \text{ is } S4\text{-satisfiable }\}.$$

Now, we can state that $g(G) \subseteq \mathbf{S}^{eq}$.

Let φ be a set of $\mathcal{TCC}$ constraints of types 1,2 and 3, and let $\mathbf{S}^{eq}$ be the corresponding set of all possible equivalence classes for the set of terms in φ. For every model, $\mathfrak{A} = \langle X, \alpha \rangle$, if $\mathfrak{A} \models \varphi$ then

$$g(f(X)) \subseteq \mathbf{S}^{eq}.$$

We can define a topological model from each subset of $\mathbf{S}^{eq}$ as follows.

For each $Q \in \mathbb{P}(\mathbf{S}^{eq})$:

We define a binary relation, $\rightarrow$, on Q as follows:

$$S \rightarrow S' \text{ if and only if, for all } s \in \mathbf{T},\ s \in S \Rightarrow s^{-} \in S'$$

We define a closure operator, $^{-Q}$, on Q as follows, where $V \subseteq Q$:

$$V^{-Q} = \{\ S' \mid S \to S' \text{ for some } S \in V\ \}$$

We define a valuation function $\alpha^Q : \mathcal{V} \to \mathbb{P}(Q)$, given a variable p:

$$\alpha^Q(p) = \{\ S \in Q \mid p \in S\ \}.$$

Definition 5.2.15. Thus, we define a set of topological models generated from $\mathbf{S}^{eq}$ as

$$\mathfrak{T} = \{\ \langle Q, \alpha^Q \rangle \mid Q \in \mathbb{P}(\mathbf{S}^{eq})\ \}.$$

Given a set, φ, of $\mathcal{TCC}$ constraints of types 1,2 and 3, if φ is satisfiable, then there exists a $\mathfrak{A} \in \mathfrak{T}$ such that $\mathfrak{A} \models \varphi$.

The satisfiability of $\mathcal{TCC}$ formulae containing types 1, 2 and 3 constraints is defined in definition 5.2.2.

Type 4 Constraint Removal

So far, we are still not able to decide the satisfiability of a set of constraints containing type 4 constraints, that is, a constraint of the form:

$$c^{\geqslant k}(s).$$

The following lemma is presented as an example in [PH01b]; however, it cannot be accomplished in polynomial time; a more complicated method, which can be accomplished in polynomial time, is also described in the paper.

Let $z_1, \ldots, z_k$ be variables not occurring in the term s, then:

Lemma 5.2.16. *The formula $c^{\geqslant k}(s)$ is equisatisfiable with the formula φ given by:*

$$s = \bigcup_{1 \leqslant i \leqslant k} z_i \ \wedge \ \bigwedge_{1 \leqslant i \leqslant k} z_i \neq \varnothing \ \wedge \ \bigwedge_{1 \leqslant i < j \leqslant k} s \cap {z_i}^- \cap {z_j}^- = \varnothing$$

Proof. This is proved in [PH01b]. □

So, we can determine satisfiability of $\mathcal{TCC}$ formulae in NEXP time. First, we have shown that if a set of constraints is topologically satisfiable, then they are satisfiable in a finite model. Then we have shown how to build a set of 'candidate' models for the set of constraints. These models are of exponential size ($2^{2|\varphi|}$) to the set of constraints, and can be tested for satisfiability (in exponential time). If a model is found which satisfies the constraint set, then the whole $\mathcal{TCC}$ constraint set is, obviously, satisfiable.

When we compare the $\mathcal{TCC}$ language with the $\mathcal{T}$ language, we can see that syntactically, the $\mathcal{TCC}$ language is the same language with the addition of the component counting constraints. But, we also see that the addition of this added expressiveness comes with a large increase in complexity - from PSPACE to NEXPTIME. Indeed, if we take the $\mathcal{TCC}$ language, but remove the ability for the connectedness predicate to specify a number of components (we may call this the $\mathcal{TC}$ language), then the satisfiability problem of this language is in EXPTIME (see [KPHWZ08] for details and proof).

It is also interesting to compare the different methods for solving the respective satisfiability problems for $\mathcal{TCC}$ and $\mathcal{T}$. The satisfiability problem of the $\mathcal{T}$ language in Subsection 5.1.2 was simply shown to be equivalent to the *S4*-satisfiability problem. In a sense, this translation to modal logic cuts many topological links, and certainly makes it harder to expand the language with more topological constraints. On the other hand, the $\mathcal{TCC}$-satisfiability problem was solved in a way which remains grounded in topology - thus, adding topological constraints (in this case, for component counting) was much more straightforward.

It should also be noted that the results for both these languages are for when they are interpreted over arbitrary subsets of a topological space. We currently have no results regarding the complexity of these logics when restricted to subsets of the Euclidean plane.

Now we shall examine languages which are restrictions of $\mathcal{T}$, starting with one which is a fragment of the undecidable first order theory of BCAs (see Chapter 4).

5.3 Boolean Contact Algebras

Let us now define another restricted spatial logic, which we will call $\mathcal{BC}$. The language $\mathcal{BC}$ has only one binary predicate, C, a contact relation (see Section 4.1.4), which may hold between Boolean combinations of variables.

Definition 5.3.1. We define the terms of $\mathcal{BC}$ as follows.

1. Every variable is a term.

2. If s is a term, so is $-s$.

3. If s and t are terms, so are $s \cap t$ and $s \cup t$.

The domain U of this language are the regular open (or regular closed) subsets of a topological space X, and we let α be a function mapping variables to elements of the U, as with $\mathcal{T}$. The interpretation α is extended over the connectives $-$, $\cup$ and $\cap$ as it is with $\mathcal{T}$. We define the constraints of $\mathcal{BC}$ as follows.

Definition 5.3.2. If s and t are terms, then $s = t$ is a constraint, so is $s \neq t$, and so is $C(s, t)$.

We interpret the C predicate as a standard topological contact relation. Since the contact relation can be expressed by the $\mathcal{T}$ constraint $s^- \cap t^- \neq \varnothing$, we can easily see that $\mathcal{BC}$ is a restriction of $\mathcal{T}$. Note that since the variables of $\mathcal{BC}$ range over regular closed or open sets, every $\mathcal{BC}$ term s would be represented by the $\mathcal{T}$ term $s^{\circ -}$ (for regular closed) or $s^{-\circ}$ (for regular open). This is extended over Boolean connectives in the obvious way.

Moreover, the theory of the BC language over regular open (or regular closed) subsets of a topological space is equivalent to the existential theory of BCAs. That is, the theory of the language of BCAs where every variable is implicitly taken to be existentially quantified.

Now, we introduce three more spatial logics, one of whom, called BRCC8, is equivalent to $\mathcal{BC}$ and which has some complexity results which allow us to easily achieve complexity results for $\mathcal{BC}$.

5.3.1 Region Connection Calculus

With the aim of developing a spatial counterpart to Allen's interval algebra, Randell et al. developed Clarke's 'calculus of individuals based on connection' [Cla81], into the *Region Connection Calculus*. Influenced by Hayes's *naïve physics* [Hay79], the RCC aimed to provide a framework for spatial reasoning using regions instead of points as the primitive entity. The intention of taking regions as primitive is to create a system which corresponds much closer to the human perception of space and in particular, to jettison the 'tricky' notion of points in a space.

The language of the RCC is simply first order logic with a single binary 'contact' predicate C. A set of axioms was also provided for the RCC language

$$\begin{array}{rl} P: & \{(u,v) \in RC(\mathbb{R}^2) \mid \forall w \in RC(\mathbb{R}^2),\ C(w,u) \implies C(w,v)\} \\ PP: & \{(u,v) \in RC(\mathbb{R}^2) \mid P(u,v) \text{ and } \neg P(v,u)\} \\ O: & \{(u,v) \in RC(\mathbb{R}^2) \mid \exists w \in RC(\mathbb{R}^2) \text{ s.t. } P(w,u) \text{ and } P(w,v)\} \end{array}$$

Table 5.1: Three relations definable in terms of contact.

which corresponds to the BCA0-4 axioms, together with the (Ext) and (Con) axioms (see Definition 4.1.10). As a result, Theorem 4.1.14 means that we have the following completeness theorem for RCC.

Theorem 5.3.3. *The axioms for RCC are complete with respect to the class of dense substructures of regular closed algebras of weakly regular T_1 spaces with the standard contact relation.*

However, for our purposes, we are only interested in the RCC language. Although the intention with RCC was to avoid point set topology, we can view the contact predicate as being interpreted over the relation which includes all pairs of regions whose closures have a non-empty intersection.

The language of the RCC is known to be undecidable over a number of domains, including $RO(\mathbb{R}^2)$ and $ROP(\mathbb{R}^2)$ ([Dor98]). As mentioned previously, RCC was developed in order to provide a spatial counterpart to Allen's interval algebra and, with this in mind, a particular fragment of RCC called RCC8 was proposed.

RCC8

Let us now formally define a 'contact' relation over the set of regular closed subsets of $\mathbb{R}^2$.

$$C = \{(u,v) \in RC(\mathbb{R}^2) \mid u \cap v \neq \varnothing\}$$

Using this contact relation we can define more specific relations, and we can see in Table 5.1 how we can define relations representing the mereotopological relations part-of (P), proper part-of (PP), and overlaps (O). These relations, together with the C relation, allow the definition of a further eight more specific relations as defined in Table 5.2 (the names stand for disconnected, external contact, partial overlap, equality, tangential proper part, and non-tangential proper part, respectively). These eight relations have direct parallels in the eight relations Egenhofer defines in [Ege91]; DC is '*disjoint*', EQ is '*equal*', $NTPP$ is '*inside*', TPP is '*covered*', EC is '*meet*' and PO is '*overlap*'. The language

$DC:$	$\{(u,v) \in RC(\mathbb{R}^2) \mid \neg C(u,v)\}$
$EC:$	$\{(u,v) \in RC(\mathbb{R}^2) \mid C(u,v) \text{ and } \neg O(u,v)\}$
$PO:$	$\{(u,v) \in RC(\mathbb{R}^2) \mid O(u,v) \text{ and } \neg P(u,v) \text{ and } \neg P(v,u)\}$
$EQ:$	$\{(u,v) \in RC(\mathbb{R}^2) \mid P(u,v) \text{ and } P(v,u)\}$
$TPP:$	$\{(u,v) \in RC(\mathbb{R}^2) \mid PP(u,v) \text{ and } \exists w \in RC(\mathbb{R}^2) \text{ s.t. } (EC(w,u) \text{ and } EC(w,v))\}$
$TPP^{-1}:$	$\{(u,v) \in RC(\mathbb{R}^2) \mid TPP(v,u)\}$
$NTPP:$	$\{(u,v) \in RC(\mathbb{R}^2) \mid PP(u,v) \text{ and } \neg\exists w \in RC(\mathbb{R}^2) \text{ s.t. } (EC(w,u) \text{ and } EC(w,v))\}$
$NTPP^{-1}:$	$\{(u,v) \in RC(\mathbb{R}^2) \mid NTPP(v,u)\}$

Table 5.2: The eight RCC8 relations.

whose formulae consist of conjunctions of disjunctions of predicates which are interpreted as the relations in Table 5.2, over existentially quantified variables, is called RCC8.

It is trivial to see that the RCC8 relations can all be expressed in terms of $\mathcal{T}$ formulae, as follows.

$$
\begin{aligned}
DC(x_i,x_j) &= (x_i \cap x_j) = \bot \\
EC(x_i,x_j) &= (x_i \cap x_j) \neq \bot \wedge (x_i^\circ \cap x_j^\circ) = \bot \\
PO(x_i,x_j) &= (x_i^\circ \cap x_j^\circ) \neq \bot \wedge (x_i^\circ \cap \neg x_j) \neq \bot \wedge \\
&\quad\ (\neg x_i \cap x_j^\circ) \neq \bot \\
EQ(x_i,x_j) &= (x_i \leftrightarrow x_j) = \top \\
TPP(x_i,x_j) &= (\neg x_i \cap x_j) = \top \ \wedge (x_i \cap (\neg x_j)^-) \neq \bot \wedge \\
&\quad\ (\neg x_i \cap x_j) \neq \bot \\
NTPP(x_i,x_j) &= (\neg x_i \cap x_j^\circ) = \top \wedge (\neg x_i \cap x_j) \neq \bot
\end{aligned}
$$

We will return to the RCC8 language in Chapter 6, but for now we introduce an expansion of RCC8.

5.3.2 BRCC8

Wolter and Zakharyaschev [WZ00] extend the RCC8 language, interpreted over $RC(\mathbb{R}^2)$, to allow predicates to hold between Boolean combinations of regions. They call this language BRCC8. We can state, for instance, that the intersection of x and y overlaps with z, "$PO(x \cap y, z)$", or that the complement of x is a non-tangential proper part of y, "$NTPP(-x, y)$". Note that this allows the whole space $\top$, and the empty region $\bot$ to feature in BRCC8 formulae. Here is an example of what we can express in BRCC8.

Example 5.3.4. Let x, y, z be variables.

$$\begin{gathered} TPP(x,y) \wedge EQ(-x, -y \cup z) \wedge DC(-y, z) \wedge \\ \neg EQ(x, \bot) \wedge \neg EQ(y, \bot) \wedge \neg EQ(z, \bot) \\ \neg EQ(x, \top) \wedge \neg EQ(y, \top) \wedge \neg EQ(z, \top) \end{gathered}$$

In any model of this formula, the region z is mapped to will be a hole of the region y is mapped to.

We can easily express $\mathcal{BC}$ constraints in terms of BRCC8 constraints, since $C(s,t) \equiv \neg DC(s,t)$, $s = t \equiv EQ(s,t)$, and $s \neq t \equiv \neg EQ(s,t)$. And we can easily express BRCC8 constraints in terms of $\mathcal{BC}$ constraints, as follows (for regular open subsets of a topological space).

$$\begin{array}{lcl} DC(s,t) & \equiv & \neg C(s,t) \\ PO(s,t) & \equiv & s \cap t \neq \bot \wedge s \cap -t \neq \bot \wedge -s \cap t \neq \bot \\ EC(s,t) & \equiv & C(s,t) \wedge s \cap t = \bot \\ TPP(s,t) & \equiv & s \subset t \wedge C(s,-t) \wedge s \neq t \\ NTPP(s,t) & \equiv & s \subset t \wedge \neg C(s,-t) \wedge s \neq t \\ EQ(s,t) & \equiv & s = t \end{array}$$

For regular closed subsets of a topological space, the translation is identical except for the following cases.

$$\begin{array}{lcl} PO(s,t) & \equiv & s \cap t \neq \bot \wedge s \cap -t \neq \bot \wedge -s \cap t \neq \bot \wedge \neg EC(s,t) \\ EC(s,t) & \equiv & C(s,t) \wedge -s \cup -t = \top \end{array}$$

This language is less expressive than $\mathcal{T}$, and as a result, we can achieve an NP-completeness result. We are also interested in the effect that interpreting spatial logics over the Euclidean plane has on the complexity of those spatial logics. In the case of BRCC8, the complexity rises from NP-complete, as we shall see. Consider the following example (from [WZ00]).

Example 5.3.5. Consider the following BRCC8 formula.

$$\begin{aligned} \varphi = & EQ(x_1 \cup x_2, x_3) \wedge NTPP(x_1, x_3) \wedge NTPP(x_2, x_3) \wedge \\ & \neg EQ(x_3, \top) \wedge \neg EQ(x_1, \bot) \wedge \neg EQ(x_2, \bot) \wedge \neg EQ(x_3, \bot) \end{aligned}$$

The formula φ can be satisfied in the discrete topological space consisting of three points $\{p_1, p_2, p_3\}$, with the identity function as interior operator. If we map the variables x_1 and x_2 to $\{p_1\}$ and $\{p_2\}$, respectively, and map the variable x_3 to $\{p_1, p_2\}$, then we can see, by the definition of the EQ and $NTPP$ relations, that this assignment satisfies this formula. Now, suppose that φ has a model α in a topological space $\langle X, \mathcal{U}\rangle$. The region $\alpha(x_1) \cup \alpha(x_2)$ is included in the interior of $\alpha(x_3)$. On the other hand, it coincides with $\alpha(x_3)$. Hence $\alpha(x_3)$ is both closed and open. It follows that X is the union of two disjoint non-empty open sets, $\alpha(x_3)$ and $X \setminus \alpha(x_3)$, so $\langle X, \mathcal{U}\rangle$ is not connected. So, φ is not satisfiable in $\mathbb{R}^n$ for *any* $n \geq 1$.

So, now we will present an outline of the NP-completeness for BRCC8 over subsets of a topological space (and PSPACE for subsets of the Euclidean plane). We must preserve the fact that variables range over the domain of the regular closed subsets of $\mathbb{R}^2$. So, where x_i is a variable, and s and t are terms (or Boolean combinations of variables), we extend the translation as follows.

1. $(x_i)^* = p_i$,
2. $(-s)^* = \Diamond\Box\neg s^*$,
3. $(s \cup t)^* = \Diamond\Box(s^* \vee t^*)$,
4. $(s \cap t)^* = \Diamond\Box(s^* \wedge t^*)$.

Now, we will show how the BRCC8 relations can be expressed in $S4_U$. For each BRCC8 relation $P(s,t)$, we associate a translated formula $(P(s,t))^*$, defined as follows (where s and t are terms).

$$
\begin{array}{lcl}
(DC(s,t))^* & = & \neg\exists(s^* \wedge t^*) \\
(EC(s,t))^* & = & \exists(s^* \wedge t^*) \wedge \neg\exists(\Box s^* \wedge \Box t^*) \\
(PO(s,t))^* & = & \exists(\Box s^* \wedge \Box t^*) \wedge \exists(\Box s^* \wedge \neg t^*) \wedge \exists(\neg s^* \wedge \Box t^*) \\
(EQ(s,t))^* & = & \forall(s^* \leftrightarrow t^*) \\
(TPP(s,t))^* & = & \forall(\neg s^* \wedge t^*) \wedge \exists(s^* \wedge \Diamond\neg t^*) \wedge \exists(\neg s^* \wedge t^*) \\
(NTPP(s,t))^* & = & \forall(\neg s^* \wedge \Box t^*) \wedge \exists(\neg s^* \wedge t^*)
\end{array}
$$

These obviously follow the translation which was proved correct in Corollary 5.1.14. Given a BRCC8 formula, φ, we denote the result of replacing each occurrence of an BRCC8 relation $P(s,t)$ with its translation $(P(s,t))^*$ by φ^*. We translate a conjunction of BRCC8 formulae φ as follows.

$$\varphi^\dagger = \varphi^* \wedge \bigwedge_{x_i \in var(\varphi)} (p_i \Leftrightarrow \Diamond\Box p_i)$$

A demonstration of NP-completeness for satisfiability of BRCC8 in arbitrary topological spaces is given in [WZ00], however this does not hold for connected topological spaces, and therefore neither for $\mathbb{R}^2$ (see Example 5.3.5). We will now give an outline of the proof that the satisfiability problem of BRCC8 in the Euclidean plane (as a consequence for $\mathbb{R}^n$ where $n \geq 1$) is still decidable. The complexity grows from NP-complete up to PSPACE.

Definition 5.3.6. A partial order $\langle V, S\rangle$ is of *depth* ≤ 1 if and only if V can be represented as the disjoint union of two sets, V_1 and V_0, in such a way that S is the reflexive closure of a subset of $V_1 \times V_0$. The points in V_i are said to be of depth i.

The *width* of a partial order is the maximum number of successors any element may have (via the ordering).

Definition 5.3.7. A partial order of depth ≤ 1 and width ≤ 2 is called a *quasisaw*, and a Kripke model based on a quasisaw is called a *quasisaw model.*

Definition 5.3.8. A frame $\mathfrak{F} = \langle W, \rightarrowtail\rangle$ is *connected* if, for any two points $x, y \in W$ we have $x(\rightarrowtail \cup \rightarrowtail^{-1})^{\to} y$, where $x(\rightarrowtail \cup \rightarrowtail^{-1})^{\to} y$ is the transitive closure of the relation $\rightarrowtail \cup \rightarrowtail^{-1}$.

The following lemmas and theorems are taken from Wolter & Zakharyaschev [WZ00]. Only outlines of the full proofs are presented here; the reader is referred to the source in the event that more information is required.

Lemma 5.3.9. *Every $S4_U$-formula satisfiable in a connected topological space is satisfiable in a finite connected frame.*

Proof. Given an $S4_U$-formula φ and a topological model $\mathfrak{A} = (X, \mathcal{U}, \alpha)$, we build a frame, which is shown to be connected if $\mathfrak{A}$ is connected, and which satisfies φ.

First we partition the topological space with an equivalence relation $\sim$ on X. We take $v \sim w$ if and only if for every subformula ψ of φ, we have $v \in \psi^{\mathfrak{A}}$ if and only if $w \in \psi^{\mathfrak{A}}$. Let $W = \{[v] : v \in u\}$ where $[v] = \{w : w \sim v\}$. Let $\rightarrowtail = \{([v], [w]) :$ for every subformula $\Box\psi$ of φ, $w \in (\Box\psi)^{\mathfrak{A}}$ whenever $v \in (\Box\psi)^{\mathfrak{A}}$ $\}$.

It should be clear that $\mathfrak{F} = \langle W, \rightarrowtail \rangle$ is an $S4$ Kripke frame; i.e. $\rightarrowtail$ is reflexive and transitive. Furthermore, $\mathfrak{F}$ is connected if $\mathfrak{A}$ is connected, and it can be shown that the formula φ is satisfiable in $\mathfrak{F}$. □

Lemma 5.3.10. *If a BRCC8 formula φ is satisfiable in a finite connected frame, then the translated formula $\varphi^\dagger$ is satisfiable in a finite quasisaw model.*

Proof. We can construct a partial order $\mathfrak{F}$ of depth ≤ 1 which satisfies φ, under some valuation η, such that $\mathfrak{F}$ is connected, and every point of depth 1 has at least two proper successors.

Let X be the set of all pairs $\{x, y\}$ of distinct points of $\mathfrak{F}$ of depth 0, for which there is a $z \in W$ with $z \rightarrowtail x$ and $z \rightarrowtail y$. For each $\{x, y\} \in X$ we make a new point $u_{x,y}$ and define a new frame $\mathfrak{F}' = \langle W', \rightarrowtail' \rangle$ where $W' = \{w \in W :$ depth of w is $0\}$ and $\rightarrowtail'$ is the reflexive closure of $\{(u_{x,y}, x) : \{x, y\} \in U\}$ and we define a valuation function η', for each proposition letter p, $x \in \eta'(p)$ if and only if there exists a $y \in W'$ of depth 0 such that $x \rightarrowtail y$ and $y \in \eta(p)$.

It can be shown that $\varphi^\dagger$ is satisfiable in $\mathfrak{F}$ under the valuation η'. □

Theorem 5.3.11. *A BRCC8 formula φ is satisfiable in $\mathbb{R}$ if and only if it is satisfiable in a finite quasisaw of size $\leq 2^{c.l(\varphi)}$, where c is some constant value and $l(\varphi)$ is the number of occurrences of predicate symbols in φ.*

Proof. Given a connected topological space, like $\mathbb{R}$, which satisfies φ, Lemma 5.3.9 shows us that there is a finite connected frame which satisfies φ, Lemma 5.3.10 then shows us that there is a finite saw model which satisfies φ.

For the other direction, Lemma 5.3.10 tells us that $\varphi^\dagger$ is satisfied in a saw model $\mathfrak{N}$, this can be transformed into a model $\mathfrak{M} = \langle \mathbb{Z}, \rightarrowtail, \mathfrak{v} \rangle$ which satisfies $\varphi^\dagger$ such that $\rightarrowtail = \{(x, y) : x = y$ or there exists $n \in \mathbb{Z}$ with $x = 2n$ and $y \in \{2n-1, 2n+1\}\}$. We can define η' in $\mathbb{R}$ as: $\eta'(p) = \bigcup\{(2n, 2n+2) : 2n+1 \in \eta(p)\} \cup \{2n : 2n \in \eta(p)\}$ for all proposition letters p.

It can be shown that $\varphi^\dagger$ is satisfied in the topological model $(\mathbb{R}, \eta')$. □

Because of Theorem 5.3.11, we have the following theorem:

Theorem 5.3.12. *The satisfiability problem for BRCC8 formulae in $\mathbb{R}$ is decidable in PSPACE.*

Because of Savitch's theorem, it is sufficient to present a nondeterministic polynomial space algorithm. We present an outline of an algorithm which consists of two parts. The first part 'guesses' a quasisaw model $\mathfrak{M}$ which satisfies a formula φ, and which consists of a maximum of $9l(\varphi)$ points, generates a new set of sentences:

$$L = \{\neg\exists \in sub(\varphi) \mid \mathfrak{M} \models \neg\exists\varphi\} \cup \{p_i \leftrightarrow \Diamond\Box p_i \mid X_i \in var(\varphi)\},$$

(where $sub(\varphi)$ is the set of subformulae of φ, and $var(\varphi)$ is the set of variables letters which occur in φ) and generates a set Π of all pairs of points of depth 0 in $\mathfrak{M}$ that are not connected by the quasisaw.

The second part of the algorithm checks if pairs from Π can be connected by a quasisaw model with $\leq 2^{c.l(\varphi)}$ points which validates L, thus meaning the original formula φ is satisfiable.

Theorem 5.3.13. *Satisfiability of BRCC8 formulae (interpreted over subsets of* $\mathbb{R}$*) is PSPACE-complete.*

Proof. See [WZ00] for the proof. □

These theorems have only shown decidability of the satisfiability problem of BRCC8 formulae in $\mathbb{R}$, however decidability in $\mathbb{R}^n$ for $(n \geq 1)$ coincides with decidability in $\mathbb{R}$. This may not seem intuitively correct, however it is due to the fact that the property of internal connectedness of regions is not preserved over the translation to modal logic. As a consequence, we also have the results that the $\mathcal{BC}$ satisfiability problem over regular closed subsets of a topological space is NP-complete, and over regular closed subsets of the Euclidean plane it is PSPACE-complete.

5.4 Conclusion

This chapter has presented a family of related spatial logics. The language $\mathcal{TCC}$ is the most expressive, with the (briefly mentioned) $\mathcal{TC}$ and $\mathcal{T}$ being simple restrictions of that language. These restrictions in expressiveness cause a reduction in the complexity of the satisfiability problems for the respective languages. While the satisfiability problem of $\mathcal{TCC}$ is in NEXP-time, the satisfiability problems of $\mathcal{TC}$ and $\mathcal{T}$ are in EXP-time, and PSPACE, respectively.

We have also looked at a restricted version of the language of BCAs, which we called $\mathcal{BC}$, which is a subset of $\mathcal{T}$, and whose satisfiability problem is decidable in NP-time. The special case of the $\mathcal{BC}$ satisfiability problem, where the language is interpreted over subsets of the Euclidean plane is slightly more complex, being PSPACE.

The following chapter begins by introducing a type of language which is a restriction of the $\mathcal{BC}$ language. We are interested in the effect of these syntactic restrictions on the computational complexity of these languages, and in particular, what happens when we interpret them over the Euclidean plane.

Chapter 6

Topological Constraint Languages

This chapter provides two of the main contributions of this thesis, in the form of two separate complexity results. The first is a new and simplified proof of the NP membership of RCC8, given in Section 6.4, with Theorem 6.4.11. The second contribution is a new complexity result for the RCC8 language with a connectedness predicate, which is given in Section 6.4.3, with Corollary 6.4.19.

In this chapter, we continue our investigation into the effect of restricting the syntax of a language on the computational complexity of the language. This chapter introduces a particular approach to solving the topological inference problem called *constraint satisfaction*. In particular, we are interested in the languages used in these constraint satisfaction problems. Some of these languages have been shown to be decidable, and decidable in low complexity classes. We will investigate some of these languages which are restrictions of the $\mathcal{T}$, $\mathcal{BC}$ language group. First, we introduce the concept of a relation algebra.

6.1 Relation Algebras

Given a set U, a binary relation R on U is a subset of $U \times U$ (relations may be empty). If two elements $u, v \in U$ are in a relation R, we write $R(u, v)$, as a shorthand for $\langle u, v\rangle \in R$.

Now, we will introduce two operators on relations, *composition* and *converse*.

Definition 6.1.1. The *composition* $R \circ R'$ of two binary relations is defined as:

$$R \circ R' = \{\langle u, w\rangle \mid \exists v[R(u, v) \text{ and } R'(v, w)]\}$$

Definition 6.1.2. The *converse* $R^{\smile}$ of a binary relation is defined as:

$$R^{\smile} = \{\langle v, u\rangle \mid R(u, v)\}$$

Additionally, we define the following property of a set of relations:

Definition 6.1.3. A set of relations $\mathcal{R}$ is said to be *jointly-exhaustive pairwise-disjoint* if and only if $\mathcal{R}$ forms a partition of $U \times U$.

Systems of relations were first studied by de Morgan [dM64]. However, it was Tarski [Tar41] who first defined the concept of a *relation algebra*.

Definition 6.1.4. A *Relation algebra* is a structure $\langle A, +, \cdot, -, 0, 1, \circ, ^{\smile}, 1'\rangle$ where $\langle A, +, \cdot, -, 0, 1\rangle$ is a Boolean algebra, and the following conditions hold (for $u, v, w \in A$):

1. $u \circ (v \circ w) = (u \circ v) \circ w$.
2. $(u + v) \circ w = (u \circ w) + (v \circ w)$.
3. $u \circ 1' = u$.
4. $u^{\smile\smile} = u$.
5. $(u + v)^{\smile} = u^{\smile} + v^{\smile}$.
6. $(u \circ v)^{\smile} = u^{\smile} \circ v^{\smile}$.
7. $(u^{\smile} \circ -(u \circ v)) \leq -v$.

Let A be a relation algebra. If A is finite then it is, in the Boolean algebra sense, atomic, and the structure of the relation algebra can be determined by specifying the behaviour of the composition and converse operations. The specification of the composition operation is usually given in the form of a *composition table* which is simply a two-dimensional array. The rows and columns of this array are labelled by the atoms of the relation algebra, with the cells of the array containing sets of the atoms.

Given a cell $\langle R, R'\rangle$ in the composition table, if the value of the cell is a set of atoms $T_1, \ldots, T_k$ (for some $k \in \mathbb{N}$), then $R \circ R' = T_1 + \ldots + T_k$ and the following hold:

$$\forall u, v, w \in A\ (R(u, v) \wedge R'(v, w) \implies T_1(u, w) \vee \ldots \vee T_k(u, w))$$
$$\forall u, w \in A\ (T_i(u, w) \implies \exists v \in A\ (R(u, v) \wedge R'(v, w)))\ \text{(where } i \leq k\text{)}$$

Now, we introduce constraint satisfaction problems and the constraint languages used to express these problems.

6.2 Constraint Satisfaction Problems

The *constraint satisfaction problem* is a type of automated reasoning problem which originally arose in the field of Artifical Intelligence and was first identified as a class of problem by Montanari [Mon74]. In general terms, a constraint satisfaction problem is simply the problem of finding a solution to a set of constraints. We use the term 'constraint' here in a very general sense. These constraints can be of any type, but will obviously be restricted to those which are relevant to whatever the particular domain of our problem is.

Definition 6.2.1. A *constraint language* is a formal language consisting of a set of variables $x_1, x_2, \ldots$ together with a set of predicates $P_1, P_2, \ldots$. If $x_1, \ldots, x_n$ are variables and P a predicate, then we call $P(x_1, \ldots, x_n)$ an atom, and we call a conjunction of atoms a constraint language *formula*. We call a constraint language *binary*, if the predicates are all binary predicates.

Definition 6.2.2. A *constraint satisfaction problem* (CSP) consists of a constraint language $\mathcal{C}$, a domain U, and a finite set of relations $\mathcal{R}$ defined over U. The predicates of the constraint language are interpreted, by $\mathcal{R}$, over U such that the following conditions hold.

1. The relations in $\mathcal{R}$ must be non empty and pairwise disjoint.

2. The union of the relations in $\mathcal{R}$ is the universal relation of the domain.

3. The interpretations of the predicates of the constraint language are unions of the elements of $\mathcal{R}$.

We call a CSP *binary* if the constraint language is binary.

Since the relations of a CSP must be pairwise disjoint, without loss of generality, for binary CSPs, we may assume that there is at most one predicate defined over any pair of variables, x_i, x_j, in a constraint language formula, and we refer to this predicate as P_{ij}.

An *instance* of a CSP consists of a formula in the constraint language (to be interpreted over the domain of the CSP). Let $u_1, \ldots, u_n$ be elements of the domain of a CSP and let the predicate P be interpreted as S, a union of relations in $\mathcal{R}$. Then, as a shorthand, we will overload P and say that $P(u_1, \ldots, u_n)$ holds if and only if $(u_1, \ldots, u_n) \in S$.

We call an instance of a CSP *explicit* if for each of its atoms, $P(x_1, \ldots, x_n)$, P is interpreted as one of the elements of $\mathcal{R}$.

Definition 6.2.3. A *solution* to an instance of a CSP is a function, $\alpha : \mathcal{V} \to U$, mapping the variables of the formula to elements of the domain such that, if $P_i(x_1, \ldots, x_n)$ is an atom of the formula, then $P_i(\alpha(x_1), \ldots, \alpha(x_n))$ holds. If an instance of a CSP has a solution then we say that the instance is *satisfiable*.

Let φ be an instance of a CSP, then without loss of generality we can assume that φ takes the form,

$$\varphi = \bigwedge_{i,j \leq n} P_{ij}(x_i, x_j)$$

where φ involves n variables. Now we give the definition of an important property of an instance of a constraint satisfaction problem.

Definition 6.2.4. An instance of a binary CSP is *path consistent* if and only if, for all $i, j, k \leq n$ where $P_{ij}(x_i, x_j)$, $P_{ik}(x_i, x_k)$, and $P_{kj}(x_j, x_j)$ are atoms of the instance, and for all $u, u'' \in U$ where $P_{ij}(u, u'')$ holds, there exists $u' \in U$ such that $P_{ik}(u, u')$ and $P_{kj}(u', u'')$ hold.

Let us consider a well known example of a constraint language, the interval algebra, introduced by Allen [All83]. The language of the interval algebra consists of a finite set of variables $x_1, \ldots, x_n$, a set of thirteen binary predicates (*equals*, *ends*, *during*, *begins*, *overlaps*, *meets*, *before*, and their inverses) and the Boolean connectives $\wedge, \vee, \implies, \neg$. The well formed formulae of this language are those which consist of Boolean combinations of the thirteen predicates, those predicates having a pair of variables as arguments. This language is usually interpreted over non-empty connected closed subsets of $\mathbb{R}$, with the thirteen relations

$$\begin{array}{rl} ends: & \{([q,r],[s,t]) \mid s < q < r = t, q,r,s,t \in \mathbb{R}\} \\ during: & \{([q,r],[s,t]) \mid q < s < t < r, q,r,s,t \in \mathbb{R}\} \\ begins: & \{([q,r],[s,t]) \mid q = s < r < t, q,r,s,t \in \mathbb{R}\} \\ overlaps: & \{([q,r],[s,t]) \mid q < s < r < t, q,r,s,t \in \mathbb{R}\} \\ meets: & \{([q,r],[s,t]) \mid q < r = s < t, q,r,s,t \in \mathbb{R}\} \\ before: & \{([q,r],[s,t]) \mid q < r < s < t, q,r,s,t \in \mathbb{R}\} \end{array}$$

Table 6.1: Interval algebra relations.

being interpreted as given in Table 6.1.

The interval algebra gives us a language in which we can describe structures of temporal intervals. The usefulness of these constraint languages is largely determined by the existence of a decision procedure for determining the satisfiability of the instances of these CSPs. For the interval algebra, which is certainly decidable, a decision procedure was provided in [All83]. This made use of some properties of relation algebras, which we shall now look at.

6.2.1 Relation Algebras and CSPs

We can rephrase Definition 6.2.4 in terms of relation algebras.

Definition 6.2.5. An instance of a binary constraint satisfaction problem is *path consistent* if and only if $P_{ik} \subseteq P_{ij} \circ P_{jk}$ for all $i, j, k \leq n$.

Given a binary CSP with its set of relations $\mathcal{R}$, if there exists a *finite* binary relation algebra whose atoms are the elements of $\mathcal{R}$, then the satisfiability of an instance of this binary CSP implies the path consistency of that instance (see [Mon74]), but not necessarily the other way around. However, for some binary CSPs, path consistency does imply satisfiability, for example the interval algebra over connected closed subsets of the reals (or intervals) is one of these languages.

Obviously, for these decidable constraint languages, it is important to have an efficient way to determine path consistency. Most existing algorithms for determining path consistency are based on the simple operation of calculating $P_{ik} \cap (P_{ij} \circ P_{jk})$, for all $i, j, k \leq n$. See Figure 6.1. This relation intersection operation is called the *triangle operation* in Ladkin & Maddux [LM94]. This operation is said to *stabilize* if $P_{ik} \cap (P_{ij} \circ P_{jk}) = P_{ik}$. A constraint satisfaction problem is obviously path consistent if every triangle operation stabilizes (that is, $P_{ik} \cap (P_{ij} \circ P_{jk}) = P_{ik}$, for all $i, j, k \leq n$). Given an instance of a CSP, the algorithm given by van Beek [vB92] simply performs the triangle operation on

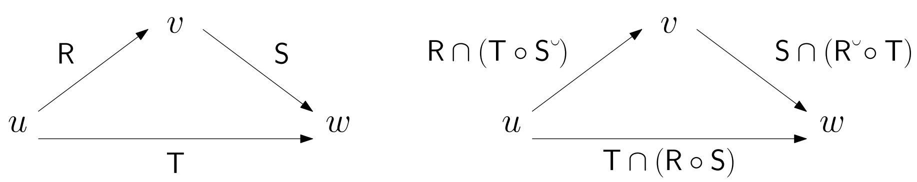

Figure 6.1: The triangle operation.

every tuple of variables in the formula, until each tuple stabilizes. The van Beek algorithm computes the path consistency of an instance of a CSP in time $O(n^3)$.

However, any algorithm performing the triangle operation is only guaranteed to terminate if the set of base relations is *jointly-exhaustive pairwise-disjoint*, and is the set of atoms of a *finite* relation algebra. Then, $P_{ij} \circ P_{jk}$ is a union of base relations and $P_{ik} \cap (P_{ij} \circ P_{jk})$ is either a union of base relations or empty.

Otherwise, the triangle operation $P_{ik} \cap (P_{ij} \circ P_{jk})$ may lead to a relation which is not a union of base relations and the triangle operation is then not guaranteed to terminate.

The interval algebra is jointly-exhaustive pairwise-disjoint and its relations are the atoms of a finite relation algebra, which means that there is a decision procedure for the path consistency (and therefore satisfiability) of interval algebra formulae. In fact, this satisfiability problem is NP-complete [All83].

6.2.2 Algebraic Closure

We now introduce a variant of the relational composition operator called *weak composition*, see [DWM01] and [Dün05].

Definition 6.2.6. Let $\mathcal{R}$ be a set of relations on a set U and let $R, R' \in \mathcal{R}$. We define *weak* relational composition $\circ_w$ as follows.

$$R \circ_w R' = \bigcup \{T \in \mathcal{R} \mid (R \circ R') \cap T \neq \varnothing\}$$

As with composition, weak composition is usually specified in the form of a table. A *weak* composition table is a two dimensional array whose rows and columns are labelled by the atomic relations, with the cells of the array containing sets of the atoms. Given a cell $\langle R, R' \rangle$ in a weak composition table, if the value of the cell is a set of atoms $T_1, \ldots, T_k$ (for some $k \in \mathbb{N}$), then $R \circ_w R' = T_1 + \ldots + T_k$

and the following holds.

$$\forall u, v, w \in A(R(u, w) \wedge R'(w, v) \implies T_1(u, v) \vee \ldots \vee T_k(u, v))$$

We can produce an algorithm for determining the algebraic closure of an instance of a CSP by modifying the van Beek algorithm to perform the triangle operation with weak composition.

6.3 Topological Constraint Languages

We now introduce a particular class of constraint satisfaction problem, that of the *topological* constraint satisfaction problem. We say that a CSP is a topological CSP if the relations over which the constraint language is interpreted are topological (or mereotopological) relations. We call the languages used in these topological CSPs, *topological constraint languages*.

The constraint languages used in topological CSPs are restricted forms of spatial logic. Determining the satisfiability of the formulae of a spatial logic is typically a very computationally complex, and sometimes impossible, task. In Chapter 4 we saw that the first order language with only a contact relation was undecidable. In order to make this task more practical, we need some way to reduce the computational complexity of these languages. One way to achieve this is to accept a reduction in the expressiveness of the languages we use.

This reduction in expressiveness can be achieved by limiting the languages in some way. For instance by removing syntactic features, imposing conditions on the language, or by restricting the domain of the language. All of these methods can be effective in achieving a lower complexity for the satisfiability problem of a language. However we focus on the removal of syntactic features from the language and the fragments of the language this produces. We have already seen in Chapter 4, that the highly restricted language (equivalent to $S4$) was decidable in PSPACE.

Because of their inherently restricted syntax, topological constraint languages make ideal candidates for practical applications of spatial logics. As a result they have received considerable attention. We now look at two similar topological constraint languages whose satisfiability problems have been presented as topological constraint problems.

$$
\begin{aligned}
\textit{disjoint}: &\quad \{(u,v) \in D(\mathbb{R}^2)^2 \mid u \cap v = \varnothing\} \\
\textit{equal}: &\quad \{(u,v) \in D(\mathbb{R}^2)^2 \mid u = v\} \\
\textit{inside}: &\quad \{(u,v) \in D(\mathbb{R}^2)^2 \mid u^\circ \subset v^\circ \text{ and } u^\partial \cap v^\partial = \varnothing\} \\
\textit{cover}: &\quad \{(u,v) \in D(\mathbb{R}^2)^2 \mid u^\circ \subset v^\circ \text{ and } u^\partial \cap v^\partial \neq \varnothing\} \\
\textit{meet}: &\quad \{(u,v) \in D(\mathbb{R}^2)^2 \mid u^\circ \cap v^\circ = \varnothing \text{ and } u \cap v \neq \varnothing\} \\
\textit{overlap}: &\quad \{(u,v) \in D(\mathbb{R}^2)^2 \mid u^\circ \cap v^\circ = \varnothing \text{ and } u^\circ \cap (-v)^\circ \neq \varnothing \text{ and } (-u)^\circ \cap v^\circ \neq \varnothing\}
\end{aligned}
$$

Table 6.2: Egenhofer topological relations.

6.3.1 Egenhofer's Topological Relations

Egenhofer's language was motivated specifically by a need for qualitative spatial logics for geographical information systems. The language consists of a set of variables, eight binary predicates (*disjoint*, *equal*, *inside*, *contains*, *cover*, *covered*, *meet*, and *overlap*) and the Boolean conjunction ($\wedge$) operator. The satisfiability problem of this language is presented in [Ege91] as a topological constraint problem whose domain is the set of regions of the real plane homeomorphic to the closed unit disc (which we call $CD(\mathbb{R}^2)$). The predicates are interpreted over the relations given in Table 6.2. The well formed formulae of this language are simply conjunctions of the binary constraints whose arguments are pairs of variables.

Significantly, these eight relations are the atoms of a finite relation algebra ([LY03]) which means that a path consistency algorithm using the triangle operation will terminate. Egenhofer [Ege91] presents a composition table with the aim of providing a means of deciding the satisfiability of formulae of this language. However, at this point we reach a problem first raised in [GPP95]. Given an instance of Egenhofer's topological CSP with a formula φ, the satisfiability of the instance is not equivalent to the satisfiability of φ interpreted over $CD(\mathbb{R}^2)$. Take the following formula (to save on space, all pairs of variables not specified to be in *meet* are in the *disjoint* relation) with the variables $A, \ldots, E$ and $AB, AC, AD, AE, BC, \ldots$.

$$
\begin{aligned}
&\textit{meet}(A, AB) \wedge \textit{meet}(A, AC) \wedge \textit{meet}(A, AD) \wedge \textit{meet}(A, AE) \wedge \\
&\textit{meet}(B, AB) \wedge \textit{meet}(B, BC) \wedge \textit{meet}(B, BD) \wedge \textit{meet}(B, BE) \wedge \\
&\textit{meet}(C, AC) \wedge \textit{meet}(C, BC) \wedge \textit{meet}(A, CD) \wedge \textit{meet}(A, CE) \wedge \\
&\textit{meet}(D, AD) \wedge \textit{meet}(D, BD) \wedge \textit{meet}(D, CD) \wedge \textit{meet}(D, DE) \wedge \\
&\textit{meet}(E, AE) \wedge \textit{meet}(E, BE) \wedge \textit{meet}(E, CE) \wedge \textit{meet}(E, DE)
\end{aligned}
$$

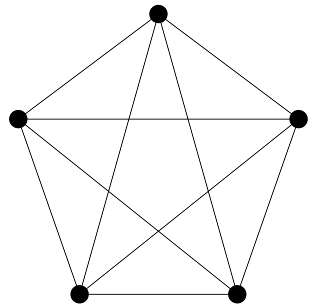

Figure 6.2: The K^5 graph.

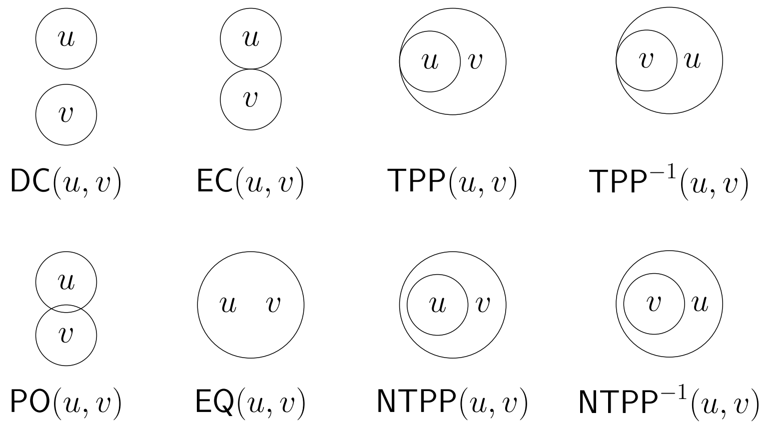

Figure 6.3: The eight mereotopological relations of RCC8.

This formula is path consistent, but not satisfiable. A quick check of the composition table given in [Ege91] shows that no tuple of these relations violates the path consistency of the formula. However, it is impossible to choose fifteen internally connected regions in the plane such that this formula is satisfied, for the same reason that the K^5 graph (Figure 6.2) cannot be drawn in the plane such that no edges intersect each other. The variables A, B, C, D, E can be thought of as representing the vertices of a K^5 graph, with the variables $AB, AC, AD, AE, \ldots$ representing the edges.

It is clear for Egenhofer's language that we have a CSP that cannot be solved with path consistency methods. The problem is that path consistency ensures satisfiability only for every subset of three relations of the formula of the CSP. Because of planarity constraints in the real plane, path consistency is not sufficient.

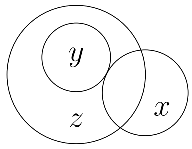

Figure 6.4: Model of Formula 6.1.

6.3.2 Region Connection Calculus

The RCC8 language (from Section 5.3.1) fulfills all our conditions of being a topological constraint language. If we interpret the RCC8 language over regular closed subset of a topological space, interpreting the binary predicates as shown in Figure 6.3 (over the relations given in Table 5.2), then we have the RCC8 CSP. Note that these relations are jointly-exhaustive pairwise-disjoint. In [RCC92a], Randell et al. observed that these eight relations are present in the relation algebra generated by the contact relation in any RCC model. Furthermore, Renz [Ren98] showed that, for RCC8 formulae, satisfiability over regular closed (or open) subsets of arbitrary topological spaces coincides with satisfiability in the Euclidean plane (in fact, in $\mathbb{R}^n$ for $n > 1$).

Unfortunately, when interpreted over the regular closed subsets of a topological space, there is no *finite* relation algebra with the eight RCC8 relations as its atoms ([LY03]). As a result, performing the triangle operation on a tuple involving the RCC8 relations is not guaranteed to terminate, as the relational composition operator can lead to a relation which is not a union of the RCC8 base relations. Furthermore, for the RCC8 CSP, we can easily show that path consistency is stronger than satisfiability, by giving an example of a formula which is satisfiable (in $RC(\mathbb{R}^2)$) but not path consistent. Take the following example (from [LW06]).

Example 6.3.1.

$$PO(x,z) \wedge EC(x,y) \wedge NTPP(y,z) \tag{6.1}$$

This instance of the RCC8 CSP is clearly satisfiable (see Figure 6.4). However, we can choose elements of the domain of this CSP, which clearly show that the formula is not path consistent. Let $u, v, w \in RC(\mathbb{R}^2)$ be pairwise disjoint, let a be the union of u and v, and c be the union of u and w. Obviously $PO(a,c)$ holds. We can clearly see from Figure 6.5 that there can exist no $b \in RC(\mathbb{R}^2)$

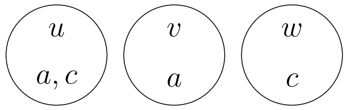

Figure 6.5: Elements of $RC(\mathbb{R}^2)$ that are not path consistent.

such that both $EC(a, b)$ and $NTPP(b, c)$ hold.

If we restrict the domain to the regions homeomorphic to the closed (or open) unit discs ($CD(\mathbb{R}^2)$), then the RCC8 relations *are* the atoms of a finite relation algebra, [LY03]. However, over this domain, we have the same problem as we've seen with Egenhofer's language. Because of planarity constraints in the real plane, path consistency of an instance of the RCC8 CSP (over $CD(\mathbb{R}^2)$) is not equivalent to the satisfiability of the formula of that instance. It is possible that there are also other domains that result in the RCC8 relations being atoms of a finite relation algebra, but very few results are known.

Instead of path consistency, we can consider the weaker notion of algebraic closure. We can compute the algebraic closure of an instance of the RCC8 CSP by using weak composition tables of the RCC8 relations. These tables have appeared many times in the literature (see for example [RN99]), but have usually been mistakenly presented as proper (full) composition tables.

Given two binary relations over the domain of an instance of the RCC8 CSP, then for any u, v, w such that $R(u, v)$ and $R'(v, w)$, $R \circ_w R'$ is the set of all possible relations which may hold between u and w. So it is clear that satisfiability of an instance of the RCC8 CSP implies algebraic closure. But for algebraic closure to provide a useful means of determining satisfiability for RCC8, we must also show that algebraic closure implies satisfiability.

In Section 7 of Renz & Nebel [RN99], there is a result showing that 'path-consistency' is equivalent to satisfiability for RCC8. However, the table of relations (given in [RN99], amongst other places) is not a composition table, but a *weak* composition table, and what the result *actually* shows is that the property of being algebraic closed is equivalent to satisfiability for a *fragment* of RCC8. We will now give a very brief summary of this proof, [RN99] should be consulted for the full proof.

First a translation is given to transform the RCC8 CSP into the S4-satisfiability problem which is in turn translated into a propositional logic satisfiability problem, see Bennett [Ben94]. For every instance of the RCC8 CSP we have a corresponding propositional logic formula, which is satisfiable if and only if the instance of the RCC8 CSP is satisfiable. We call the fragment of RCC8 whose formulae translate into Horn formulae (conjunctions of clauses with at most one positive literal), $\mathcal{H}_8$. We call the CSP involving only formulae which belong to $\mathcal{H}_8$, the $\mathcal{H}_8$ CSP.

The proof then shows that, for every instance of the $\mathcal{H}_8$ CSP, deciding if the instance is algebraic closed is equivalent to deciding if the instance is satisfiable. We can convert an RCC8 formula into at most 3^m (where m is the number of constraints in the formula) $\mathcal{H}_8$ formulae, and if any of those $\mathcal{H}_8$ are satisfiable (or path consistent), then the RCC8 formula is satisfiable. Simpler proofs of this result have been given by Bennett [Ben97], and [Ben98], but as a result of the theorems in Section 6.4, we are able to present a considerably simpler proof of this result, in Theorem 6.4.13.

6.4 Complexity of RCC8

Instead of using relation algebraic techniques, we will proceed to show that the complexity of RCC8 can be established simply, using a model theoretic approach, simply by building a model whose size has a clear upper bound.

So, we present a decision procedure for the satisfiability problem of a restricted fragment of RCC8, which we call $\mathcal{E}$. This leads to a simple proof of the NP completeness of the RCC8 CSP.

6.4.1 Explicit RCC8

First, we define an $\mathcal{E}$ formula to be a conjunction of expressions of the form $R(x, y)$, where R is one of

$$\{PO, DC, EC, EQ, NTPP, NTPP^{-1}, TPP, TPP^{-1}, \top\}$$

and x and y are variables, and at most one relation is specified between any (unordered) pair of variables.

If we let φ be an $\mathcal{E}$ formula and V be the set of variables in φ, then an interpretation of φ, in a topological space X, consists of an assignment $\alpha : V \rightarrow$

$\mathbb{P}(X)$, mapping the variables of φ to elements of $\mathbb{P}(X)$.

Now we will define precisely how the eight topological relations are interpreted in the language.

Definition 6.4.1. Let φ be an $\mathcal{E}$ formula. We say that an interpretation α of φ is a model of φ if and only if, for each conjunct $R(x, y)$ of φ, the following conditions hold.

1. If R is PO then $\alpha(x)^\circ \cap \alpha(y)^\circ \neq \varnothing$, $\alpha(x)^\circ \cap (-\alpha(y))^\circ \neq \varnothing$, and $(-\alpha(x))^\circ \cap \alpha(y)^\circ \neq \varnothing$.
2. If R is DC then $\alpha(x)^- \cap \alpha(y)^- = \varnothing$.
3. If R is EC then $\alpha(x)^\partial \cap \alpha(y)^\partial \neq \varnothing$ and $\alpha(x)^\circ \cap \alpha(y)^\circ = \varnothing$.
4. If R is EQ then $\alpha(x) = \alpha(y)$.
5. if R is $NTPP$, $\alpha(x)^\circ \subset \alpha(y)^\circ$ and $\alpha(x)^\partial \cap \alpha(y)^\partial = \varnothing$,
6. If R is $NTPP^{-1}$ then $\alpha(y)^\circ \subset \alpha(x)^\circ$ and $\alpha(x)^\partial \cap \alpha(y)^\partial = \varnothing$.
7. If R is TPP then $\alpha(x)^\circ \subset \alpha(y)^\circ$ and $\alpha(x)^\partial \cap \alpha(y)^\partial \neq \varnothing$.
8. If R is TPP^{-1} then $\alpha(y)^\circ \subset \alpha(x)^\circ$ and $\alpha(x)^\partial \cap \alpha(y)^\partial \neq \varnothing$.
9. If R is $\top$, R is satisfied by any pair of variables.

If there exists a model of φ, then we say that φ is *satisfiable*.

We can always convert an $\mathcal{E}$ formula into a simplified form, as follows.

Lemma 6.4.2. *Let φ be an $\mathcal{E}$ formula with variables $x_1, \ldots, x_n$. We can convert φ in polynomial time into an $\mathcal{E}$ formula of the form*

$$\bigwedge_{1 \leq i < j \leq n} R_{ij}(x_i, x_j) \tag{6.2}$$

(where $R_{ij} \in \{PO, DC, EC, NTPP, TPP, \top\}$) with exactly the same models as φ. We say that a formula of the form (6.2) *is* simple.

Proof. First, we consider the case where φ is satisfiable. Any conjuncts in φ which are $NTPP^{-1}(x_i, x_j)$ can simply be replaced by the conjunct $NTPP(x_j, x_i)$ (similarly for TPP^{-1}). If φ is satisfiable, then the numbering of the variables can always be changed such that $j < i$. If no relation is specified in φ between variables x_i and x_j, then we simply add the conjunct $\top(x_i, x_j)$. For any conjuncts $EQ(x_i, x_j)$ in φ, we simply replace every occurrence of x_j in φ with x_i. The relations DC, EC, and PO are symmetric, so can easily be changed such that $i < j$, for $R_{ij}(x_i, x_j)$.

In the case that φ is unsatisfiable, the above changes will preserve the unsatisfiability of φ. □

Definition 6.4.3. Suppose φ is a simple $\mathcal{E}$ formula and x, y are variables of φ. A *chain* in φ from x to y is a sequence of conjuncts (of φ)

$$R_1(x_1, x_2), R_2(x_2, x_3), \ldots, R_m(x_m, x_{m+1})$$

where $x_1 = x$, $x_{m+1} = y$, and R_i is either TPP or $NTPP$. The chain is *strict* if some R_i is $NTPP$.

As a shorthand, we write $x \to y$ if there is a chain from x to y or if $x = y$, and we write $x \Rightarrow y$ if there is a strict chain from x to y.

The following properties hold in every model of an $\mathcal{E}$ formula.

Lemma 6.4.4. *If φ is an $\mathcal{E}$ formula with a model α, then for all $i, j \leq n$, if $i \to j$ then $\alpha(x_i)^\circ \subset \alpha(x_j)^\circ$.*

Lemma 6.4.5. *If φ is an $\mathcal{E}$ formula with a model α, then for all $i, j \leq n$, if $i \Rightarrow j$ then $\alpha(x_i)^\partial \cap \alpha(x_j)^\partial = \varnothing$.*

Now we define the decision procedure for the satisfiability of $\mathcal{E}$ formulae.

Definition 6.4.6. We define a function SAT which takes a $\mathcal{E}$ formula with variables $x_1, \ldots, x_n$ as an argument. The function SAT returns $false$ if:

1. there exists $i, j \leq n$ such that $i \to j$ and $R_{ij} \in \{PO, DC, EC\}$, or
2. there exists $i, j \leq n$ such that $i \Rightarrow j$ and $R_{ij} \in \{TPP\}$, or
3. there exists $i, k, l \leq n$ such that $i \to k$, $i \to l$, and $R_{kl} \in \{DC, EC\}$, or
4. there exists $i, j, k, l \leq n$ such that ($i \to k$ and $j \to l$) or ($i \to l$ and $j \to k$) and $R_{ij} \in \{PO\}$ and $R_{kl} \in \{DC, EC\}$, or

5. there exists $i, j, k, l \leq n$ such that ($i \to k$ and $j \to l$) or ($i \to l$ and $j \to k$) and $R_{ij} \in \{EC\}$ and $R_{kl} \in \{DC\}$, or

6. there exists $i, j, k, l \leq n$ such that ($i \Rightarrow k$ and $j \to l$) or ($i \to k$ and $j \Rightarrow l$) or ($i \Rightarrow l$ and $j \to k$) or ($i \to l$ and $j \Rightarrow k$) and $R_{ij} \in \{EC\}$ and $R_{kl} \in \{EC\}$.

Otherwise, the function returns *true*.

Now we must prove the correctness of the SAT function in Definition 6.4.6.

Lemma 6.4.7. *Let φ be a simple $\mathcal{E}$ formula. If φ is satisfiable, then* SAT(φ) *will return* true.

Proof. Assume φ is satisfiable. Let α be a model of φ. Let $s_i = \alpha(x_i)$, for all i.

We see from SAT that there are exactly six situations where *false* may be returned. We show that if any of these situations had occurred, then φ could not have been satisfiable.

1. In this case SAT will return *false* if, for any $i \leq n$, there exists a j such that $i \to j$ and $R_{ij} \in \{PO, DC, EC\}$. Since $i \to j$, by Lemma 6.4.4, $s_i^\circ \subset s_j^\circ$. However, if R_{ij} is PO, $s_i^\circ \cap (-s_j)^\circ \neq \varnothing$, and if R_{ij} is DC or EC, $s_i^\circ \cap s_j^\circ = \varnothing$.

2. In this case SAT will return *false* if, for any $i \leq n$, there exists a j such that $i \Rightarrow j$ and $R_{ij} \in \{TPP\}$. Since $i \Rightarrow j$, by Lemma 6.4.5 $s_i^\partial \cap s_j^\partial = \varnothing$ however, since R_{ij} is TPP, $s_i^\partial \cap s_j^\partial \neq \varnothing$.

3. In this case SAT will return *false* if, for any $k, l \leq n$ such that $R_{kl} \in \{DC, EC\}$, there exists a $i \leq n$ such that $i \to k, l$. Since R_{kl} is either DC or EC, $s_k^\circ \cap s_l^\circ = \varnothing$, but since, by Lemmas 6.4.4 & 6.4.5, $s_i^\circ \subset s_k^\circ$ and $s_i^\circ \subset s_l^\circ$, $\varnothing \neq s_i^\circ \subseteq s_k^\circ \cap s_l^\circ$.

4. In this case SAT will return *false* if, for any $k, l \leq n$ such that $R_{kl} \in \{DC, EC\}$, there exists $i, j \leq n$ such that $R_{ij} \in \{PO\}$, $i \to k$ and $j \to l$ or $i \to l$ and $j \to k$. Since R_{kl} is either DC or EC, $s_k^\circ \cap s_l^\circ = \varnothing$. Without loss of generality, we can take $s_i^\circ \subset s_k^\circ$ and $s_j^\circ \subset s_l^\circ$, and therefore $s_i^\circ \cap s_j^\circ \subseteq s_k^\circ \cap s_l^\circ$. However, since R_{ij} is PO, $s_i^\circ \cap s_j^\circ \neq \varnothing$, so $s_k^\circ \cap s_l^\circ \neq \varnothing$.

5. In this case SAT will return *false* if, for any $k, l \leq n$ such that $R_{kl} \in \{DC\}$, there exists $i, j \leq n$ such that $R_{ij} \in \{EC\}$, $i \to k$ and $j \to l$ or $i \to l$ and $j \to k$. Without loss of generality, we can take $s_i^\circ \subset s_k^\circ$ and $s_j^\circ \subset s_l^\circ$. Since

R_{ij} is EC, $s_i^\partial \cap s_j^\partial \neq \varnothing$. And since R_{kl} is DC, $s_k^- \cap s_l^- = \varnothing$, which implies $s_i^- \cap s_j^- = \varnothing$, and therefore $s_i^\partial \cap s_j^\partial = \varnothing$.

6. In this case SAT will return *false* if, there exists $k, l \leq n$ such that $R_{kl} \in \{EC\}$ and there exists $i, j \leq n$ such that $R_{ij} \in \{EC\}$, such that $i \Rightarrow k$ and $j \rightarrow l$ or $i \rightarrow k$ and $j \Rightarrow l$ or $i \Rightarrow l$ and $j \rightarrow k$ or $i \rightarrow l$ and $j \Rightarrow k$. As R_{kl} is EC, then $s_k^\circ \cap s_l^\circ = \varnothing$, and since R_{ij} is EC, then $s_i^\partial \cap s_j^\partial \neq \varnothing$. Since EC is symmetric, without loss of generality, we can assume that $i \Rightarrow k$, so $s_i^\circ \subset s_k^\circ$ and $s_i^\partial \cap s_k^\partial = \varnothing$, and therefore $s_i^\partial \subset s_k^\circ$, $(s_i^\partial \cap s_j^\partial) \subset s_k^\circ$, and $s_j^\partial \cap s_k^\circ \neq \varnothing$ (because $s_i^\partial \cap s_j^\partial \neq \varnothing$). Since s_k° is an open set $s_j^\circ \cap s_k^\circ \neq \varnothing$ therefore $s_l^\circ \cap s_k^\circ \neq \varnothing$.

□

Lemma 6.4.8. *Let φ be a simple $\mathcal{E}$ formula. If* SAT(φ) *returns* true*, given φ as input, then φ has a model in $RO(\mathbb{R})$.*

Proof. First, for $i < j \leq n$, let $s = 10((i-1)n + (j-1))$, and we define the following sets:

$$A_{ij} = \{x \in \mathbb{R} \mid s < x < s + 10\}.$$

Now, for each conjunct $R_{ij}(x_i, x_j)$ of φ, we define D_i and D_j the following way.

A1: If R_{ij} is PO then:

$$D_{ij} = \{x \in \mathbb{R} \mid s + 2 < x < s + 4\}$$
$$D_{ji} = \{x \in \mathbb{R} \mid s + 3 < x < s + 5\}$$

A2: If R_{ij} is DC or $\top$ then:

$$D_{ij} = \{x \in \mathbb{R} \mid s + 2 < x < s + 4\}$$
$$D_{ji} = \{x \in \mathbb{R} \mid s + 6 < x < s + 8\}$$

A3: If R_{ij} is EC then:

$$D_{ij} = \{x \in \mathbb{R} \mid s + 2 < x < s + 4\}$$
$$D_{ji} = \{x \in \mathbb{R} \mid s + 4 < x < s + 6\}$$

A4: If R_{ij} is $NTPP$ then:

$$D_{ij} = \{x \in \mathbb{R} \mid s+2 < x < s+4\}$$
$$D_{ji} = \{x \in \mathbb{R} \mid s+1 < x < s+5\}$$

A5: If R_{ij} is TPP then:

$$D_{ij} = \{x \in \mathbb{R} \mid s+2 < x < s+4\}$$
$$D_{ji} = \{x \in \mathbb{R} \mid s+2 < x < s+5\}$$

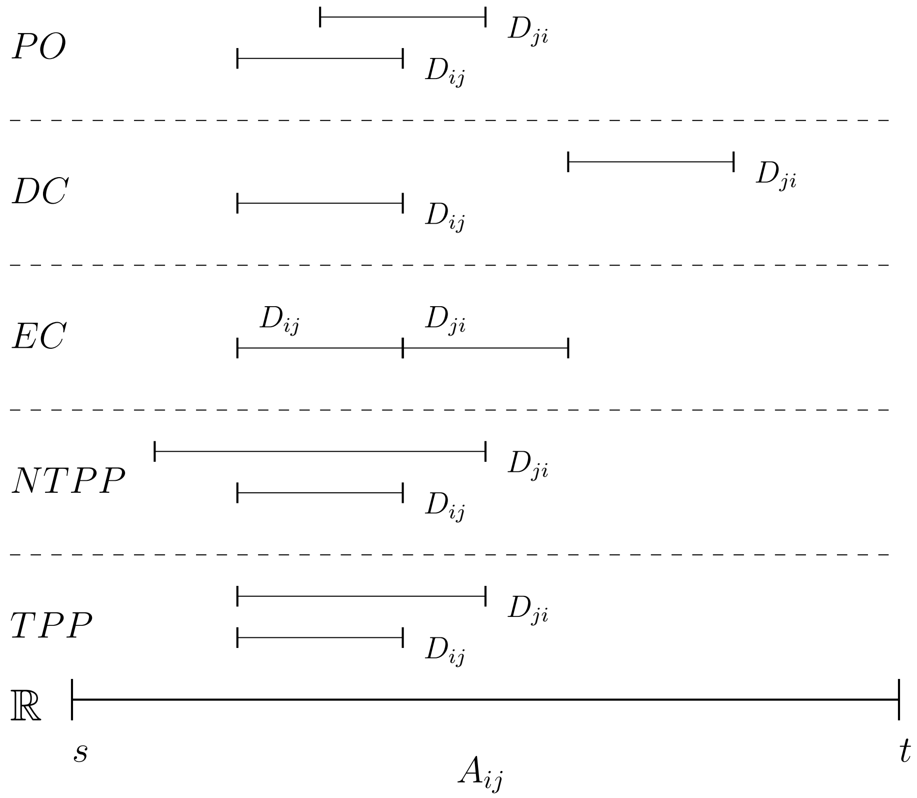

Figure 6.6: Sets produced by A1-A5, within each A_{ij} region.

Note that $A_{ij} \cap (D_k)^- = \varnothing$ if $k \neq i$ and $k \neq j$ (for all $i, j, k \leq n$). Let $\epsilon = \frac{1}{n+1}$. For each $i \leq n$:

$$D_i = \bigcup_{j \leq n} D_{ij}$$

And for each $j \leq n$:

$$C_j = \{x \in \mathbb{R} \mid x \in D_i \text{ and } i \rightarrow j \text{ or } \exists y \in D_i \text{ where } |x - y| < (j - i)\epsilon \text{ and } i \Rightarrow j\}$$

Then, we define an interpretation $\alpha(x_i) = C_i$ (for all $i \leq n$). We claim that α is a model of φ. We show that, for all i, j ($1 \leq i < j \leq n$) the sets C_i and C_j stand in the relation R_{ij}. There are five cases for R_{ij} (we can simply ignore $\top$):

PO: We show that $C_i^\circ \cap C_j^\circ \neq \varnothing$, $C_i^\circ \cap (-C_j)^\circ \neq \varnothing$ and $(-C_i)^\circ \cap C_j^\circ \neq \varnothing$. By the definitions of D_i and D_j, $C_i^\circ \cap C_j^\circ \neq \varnothing$ holds. Since $i < j$, then $C_i^\circ \cap (-C_j)^\circ \neq \varnothing$ holds by the definitions of D_i, C_i and C_j. And since $i \nrightarrow j$, by case (1) of Sat, $(A_{ij} \cap C_i)^\circ \cap (A_{ij} \cap -C_j)^\circ \neq \varnothing$ and so $(-C_i)^\circ \cap C_j^\circ \neq \varnothing$.

DC: We must show that $C_i^- \cap C_j^- = \varnothing$. Note that $i \nrightarrow j$, by case (1) of Sat. By definition of D_i and D_j, $D_i^- \cap D_j^- = \varnothing$. For $C_i^- \cap C_j^- \neq \varnothing$ to hold, there must exist $k, l \leq n$ where $k \rightarrow i$ and $l \rightarrow j$ such that either $PO(x_k, x_l)$ or $EC(x_k, x_l)$ is a conjunct of φ. If $PO(x_k, x_l)$ were a conjunct of φ, then case (4) of Sat would have returned false. Likewise, if $EC(x_k, x_l)$ were a conjunct of φ, then case (5) of Sat would have returned false.

EC: We must show that $C_i^\partial \cap C_j^\partial \neq \varnothing$ and $C_i^\circ \cap C_j^\circ = \varnothing$. Trivially $D_i^\partial \cap D_j^\partial \neq \varnothing$, and since $i \nrightarrow j$, by case (1) of Sat, we can see by the definitions of D_i, D_j, C_i, and C_j, that $C_i^\partial \cap C_j^\partial \neq \varnothing$. By definition of D_i and D_j, $D_i^\circ \cap D_j^\circ = \varnothing$. So, for $C_i^\circ \cap C_j^\circ \neq \varnothing$ to hold, there must exist $k, l \leq n$ where $k \rightarrow i$ and $l \rightarrow j$ such that $PO(x_k, x_l)$ is a conjunct of φ, or where either $k \Rightarrow i$ and $l \rightarrow j$ or $k \rightarrow i$ and $l \Rightarrow j$ such that $EC(x_k, x_l)$ is a conjunct of φ. If $PO(x_k, x_l)$ were a conjunct of φ, then case (4) of Sat would have returned false. If $EC(x_k, x_l)$ were a conjunct of φ, then case (6) of Sat would have returned false.

NTPP: We must show that $C_i^\circ \subset C_j^\circ$ and $C_i^\partial \cap C_j^\partial = \varnothing$. Since $i \Rightarrow j$ (and $i < j$), then for all $k \leq n$ if $k \rightarrow i$, then $k \Rightarrow j$, and so by the definition of C_i and C_j, the closure of every maximally connected component of C_i will be contained in the interior of a maximally connected component of C_j, thus $C_i^\circ \subset C_j^\circ$ and $C_i^\partial \cap C_j^\partial = \varnothing$.

TPP: We must show that $C_i^\circ \subset C_j^\circ$ and $C_i^\partial \cap C_j^\partial \neq \varnothing$. Since $i \rightarrow j$ (and $i < j$), then for all $k \leq n$ if $k \rightarrow i$, then $k \rightarrow j$, and so by the definition of C_i

and C_j, the interior of every maximally connected component of C_i will be contained in the interior of a maximally connected component of C_j, thus $C_i^\circ \subset C_j^\circ$. By the definition of D_i and D_j, $D_i^\partial \cap D_j^\partial \neq \varnothing$ and $D_i^\circ \subset D_j^\circ$. By the definition of C_j, $(A_{ij} \cap C_j) = D_j$, and since $i < j$, $(A_{ij} \cap C_i) = D_i$, so $(A_{ij} \cap D_i)^\partial \cap (A_{ij} \cap D_j)^\partial \neq \varnothing$, and since $(A_{ij} \cap C_i)^\partial \cap (A_{ij} \cap C_j)^\partial \subset A_{ij}^\circ$, then $C_i^\partial \cap C_j^\partial \neq \varnothing$.

□

We now state the main result.

Theorem 6.4.9. *Given an $\mathcal{E}$ formula φ as input,* Sat(φ) *will return* true *if and only if φ is satisfiable in $RO(\mathbb{R}^2)$.*

Proof. For the *if* direction, Lemma 6.4.7 shows that if φ is satisfiable, then Sat will return *true*. For the *only if* direction, Lemma 6.4.8 shows that if Sat returns *true*, then φ is satisfiable. □

Theorem 6.4.10. *The satisfiability problem of $\mathcal{E}$ over $RO(\mathbb{R}^2)$ is in NLOGSPACE*[1]*.*

Proof. We show this by reducing the Sat function to a graph reachability problem. Given a simple $\mathcal{E}$ formula φ with variables $x_1, \ldots, x_n$, we define a set of vertices $V = \{v_1, \ldots, v_n\}$ and two sets of edges as follows.

$$E = \{(v_i, v_j) \mid NTPP(x_i, x_j) \text{ or } TPP(x_i, x_j) \text{ are conjuncts of } \varphi\}$$
$$E' = \{(v_i, v_j) \mid TPP(x_i, x_j) \text{ is a conjunct of } \varphi\}$$

Also, we define two graphs $G = (V, E)$ and $G' = (V, E')$, and we define two operators such that $i \succ j$ if and only if v_j is reachable from v_i in G, and $i \succcurlyeq j$ if and only if v_j is reachable from v_i in G, but not in G'. Now, each of the six cases of Sat (see Definition 6.4.6) can be dealt with simply. For the first case, if $PO(x_i, x_j)$, $DC(x_i, x_j)$, or $EC(x_i, x_j)$ are conjuncts of φ, and if $i \succ j$, then the function Sat would return *false*. The other cases can be dealt with similarly.

To show that this reduction can be computed in space *logn*, we can build a Turing machine which, for each pair of variables i and j calculates if $i \succ j$ and $i \succcurlyeq j$ hold. Essentially, such a Turing machine would use four counters, two to

[1] This observation is thanks to R. Kontchakov.

record i and j, another to traverse the parthood relations (TPP and $NTPP$) i is involved in, and another to record whether a TPP relation has been traversed. Since we consider at most two pairs of variables at once, in clauses 4, 5, and 6, we need at most eight counters. □

Theorem 6.4.11. *The satisfiability problem of RCC8 over RO*($\mathbb{R}^2$) *is in NP-time.*

Proof. By Corollary 6.4.10, the $\mathcal{E}$ satisfiability problem is in NLOGSPACE. Every RCC8 formula is equisatisfiable with a disjunction of at most 8^{n^2} $\mathcal{E}$ formulae, where n is the number of variables in the $RCC8$ formula. We can nondeterministically choose one of these 8^{n^2} $\mathcal{E}$ formulae, and verify if it is satisfiable - this can be done in NP time. □

Lemma 6.4.12. *Given an* $\mathcal{E}$ *formula* φ *as input, if* SAT(φ) *returns* false, *then* φ *is not algebraically closed.*

Proof. We see from SAT that there are exactly six situations where *false* may be returned. We show that if any of these situations had occurred, then φ cannot be algebraically closed. For any $i, j \leq n$, if $i \rightarrow j$, then although neither $NTPP(x_i, x_j)$ nor $TPP(x_i, x_j)$ may necessarily be conjuncts of φ, since $i \rightarrow j$, for the purposes of this proof we can assume that one of them is, without affecting the satisfiability of φ.

1. In this case, $R_{ij} \in \{NTPP, TPP\}$ *and* $R_{ij} \in \{PO, DC, EC\}$, however this is impossible, and therefore φ cannot be algebraically closed.

2. In this case, $R_{ij} \in \{NTPP\}$ *and* $R_{ij} \in \{TPP\}$, however this is impossible, and therefore φ cannot be algebraically closed.

3. In this case, $R_{ik} \in \{NTPP, TPP\}$, $R_{il} \in \{NTPP, TPP\}$, and $R_{kl} \in \{DC, EC\}$. We can see that $R_{li} \not\subseteq (R_{ik} \circ_w R_{kl})$, since $\{NTPP^{-1}, TPP^{-1}\} \not\subseteq \{DC, EC\}$, and therefore φ cannot be algebraically closed.

4. In this case either ($R_{ik} \in \{NTPP, TPP\}$, $R_{jl} \in \{NTPP, TPP\}$, $R_{ij} \in \{PO\}$, $R_{kl} \in \{DC, EC\}$) or ($R_{il} \in \{NTPP, TPP\}$, $R_{jk} \in \{NTPP, TPP\}$, $R_{ij} \in \{PO\}$, $R_{kl} \in \{DC, EC\}$). Without loss of generality, we can consider the case where the former set of constraints hold. The proof for the other set of constraints is analogous. By looking at an RCC8 weak composition

table, we can see that $(R_{jl} \circ_w R_{lk}) \cap (R_{ji} \circ_w R_{ik}) = \varnothing$ since $\{DC, EC\} \cap \{PO, TPP, NTPP\} = \varnothing$, and so φ cannot be algebraically closed.

5. In this case either ($R_{ik} \in \{NTPP, TPP\}$, $R_{jl} \in \{NTPP, TPP\}$, $R_{ij} \in \{EC\}$, $R_{kl} \in \{DC\}$) or ($R_{il} \in \{NTPP, TPP\}$, $R_{jk} \in \{NTPP, TPP\}$, $R_{ij} \in \{EC\}$, $R_{kl} \in \{DC\}$). Without loss of generality, we can consider the case where the former set of constraints hold. The proof for the other set of constraints is analogous. By looking at an RCC8 weak composition table, we can see that $(R_{jl} \circ_w R_{lk}) \cap (R_{ji} \circ_w R_{ik}) = \varnothing$ since $\{DC\} \cap \{EC, PO, TPP, NTPP\} = \varnothing$, and so φ cannot be algebraically closed.

6. In this case either ($R_{ik} \in \{NTPP\}$, $R_{jl} \in \{NTPP, TPP\}$, $R_{ij} \in \{EC\}$, $R_{kl} \in \{EC\}$) or ($R_{ik} \in \{NTPP, TPP\}$, $R_{jl} \in \{NTPP\}$, $R_{ij} \in \{EC\}$, $R_{kl} \in \{EC\}$) or ($R_{il} \in \{NTPP\}$, $R_{jk} \in \{NTPP, TPP\}$, $R_{ij} \in \{EC\}$, $R_{kl} \in \{EC\}$) or ($R_{il} \in \{NTPP, TPP\}$, $R_{jk} \in \{NTPP\}$, $R_{ij} \in \{EC\}$, $R_{kl} \in \{EC\}$). Without loss of generality, we can consider the case where the former set of constraints hold. The proofs for the other sets of constraints are analogous. By looking at an RCC8 weak composition table, we can see that $(R_{jl} \circ_w R_{lk}) \cap (R_{ji} \circ_w R_{ik}) = \varnothing$ since $\{DC, EC\} \cap \{PO, TPP, NTPP\} = \varnothing$, and so φ cannot be algebraically closed.

□

Theorem 6.4.13. *For the RCC8 CSP (over $ROP(\mathbb{R}^2)$), algebraic closure is equivalent to satisfiability.*

Proof. Satisfiability implies algebraic closure, and Lemma 6.4.12 shows that algebraic closure implies satisfiability. □

This result has already been provided by Renz [RN99], but the proof given here is considerably simpler. Bennett [BIC97], and Renz [RL05] have asked questions about the usefulness of algebraic closure and weak composition regarding the topological inference problem. It is clear that algebraic closure cannot be shown to be equivalent to satisfiability for some topological constraint language without a careful examination of the relationship between the models of the language, and the language itself.

6.4.2 AT-graph Realizability

We now investigate a solution to the satisfiability problem of the $\mathcal{E}$ language by reducing the satisfiability problem to one of AT-graph weak realizability. First, we will examine the case of $\mathcal{E}$ formulae interpreted over regions homeomorphic to the open unit discs of the real plane (we shall use the symbol $OD(\mathbb{R}^2)$ to denote this set).

The following lemmas and theorem are taken from [SŠ04]. Lemma 6.4.15 is presented here in a considerably expanded form, with a few small errors corrected.

Lemma 6.4.14. *Let φ be an $\mathcal{E}$ formula. If φ has a model in $OD(\mathbb{R}^2)$, then φ has a model in $OD(\mathbb{R}^2)$ in which the number of contact points on the boundary of each region is bounded by the square of the number of variables in φ.*

Proof. The bound is a square of the number of variables, as there is potentially a contact point for each pair of variables. The full proof can be found in [SŠ04]. □

Lemma 6.4.15. *The satisfiability problem of $\mathcal{E}$ over $OD(\mathbb{R}^2)$ NP-reduces to the AT-graph weak realizability problem. That is, for every $\mathcal{E}$ formula φ we can in NP compute AT-graphs (G, R) such that φ is satisfiable in $OD(\mathbb{R}^2)$ if and only if one of the (G, R) is weakly realizable.*

Proof. We can assume that φ is in simple form (see Lemma 6.4.2). Now we describe the structure of the AT-graphs. Firstly, the graph has vertices z,z_1,z_2, and z_3 which are connected to each other by edges which may not intersect *any* other edges. For each variable x_i of φ, there is a vertex c_i and a circle graph B_i with at least three vertices. For each B_i $(i \leq n)$ if e_1, e_2 are edges of B_i, then e_1 may not intersect e_2. For each vertex v in B_i $(i \leq n)$, v is connected to c_i, z_1, z_2, and z_3 with edges which may not intersect B_i. Furthermore, for each $i \leq n$, no edge with an endpoint c_i may intersect an edge with an endpoint z_1, z_2, or z_3. We say that a vertex v is an *in-x_i-witness* if it does not belong to B_i and is adjacent to c_i using an edge which does not intersect B_i. We say that a vertex v is an *out-x_i-witness* if it does not belong to B_i and is adjacent to z_1, z_2, and z_3 using edges which do not intersect B_i. Now, the rest of the structure of the (G, R) AT-graphs is determined by φ. For each conjunct of φ, R_{ij} (which is the relation specified to hold between variables x_i and x_j) which can be one of the following six cases.

PO: Then, B_i and B_j may share vertices and the edges of B_i may cross the edges of B_j. Also, B_i must contain an in-x_j-witness and an out-x_j-witness, and B_j must contain an in-x_i-witness and an out-x_i-witness.

DC: Then, B_i and B_j may not share vertices, and the edges of B_i may not cross the edges of B_j. Also, B_i must contain an out-x_j-witness, and B_j must contain an out-x_i-witness.

EC: Then, B_i and B_j must share at least one vertex, but the edges of B_i may not cross the edges of B_j. Subtracting the vertices of B_j from B_i splits B_i into a set of paths. Each of these paths must contain an out-x_j-witness. Similarly, $B_j \setminus B_i$ splits B_j into a set of paths, each of these paths must contain an out-x_i-witness.

NTPP: Then, B_i and B_j may not share vertices, and the edges of B_i may not cross the edges of B_j. Also, B_i must contain an in-x_j-witness.

TPP: Then, B_i and B_j must share at least one vertex, but the edges of B_i may not cross the edges of B_j. Also, $B_i \setminus B_j$ splits B_i into a set of paths, each of these paths must contain an in-x_j-witness.

$\top$: Then, B_i and B_j may share vertices and the edges of B_i may cross the edges of B_j.

Now, we show that if such a (G, R) is realizable, then φ is satisfiable in $OD(\mathbb{R}^2)$. First, we can assume that the vertex z lies outside of the triangle z_1, z_2, z_3. As a result, all other vertices and edges must lie inside the triangle. For all $i \leq n$, c_i must lie inside the region enclosed by B_i, with the vertices z_1, z_2, and z_3 being outside. Now, we define a function α which maps each variable x_i to the region enclosed by B_i. For each conjunct of φ we will show that the relations hold under α.

1. For *PO*, since B_i contains an in-x_j-witness and B_j contains an in-x_i-witness, then $\alpha(x_i)^\circ \cap \alpha(x_j)^\circ \neq \varnothing$, and since B_i contains an out-x_j-witness, $\alpha(x_i)^\circ \cap (-\alpha(x_j))^\circ \neq \varnothing$, and since B_j contains an out-x_i-witness, $(-\alpha(x_i))^\circ \cap \alpha(x_j)^\circ \neq \varnothing$.

2. For *DC*, since B_i and B_j may not share vertices, and since the edges of B_i and B_j may not cross, the fact that B_i contains an out-x_j-witness and that B_j contains an out-x_i-witness means that $\alpha(x_i)^- \cap \alpha(x_j)^- = \varnothing$.

3. For EC, recall that $B_i \setminus B_j$ splits B_i into a set of paths, and each path contains an out-x_j-witness (likewise for B_j and its out-x_i-witnesses), and since the edges of B_i and B_j may not cross, then $\alpha(x_i)^\circ \cap \alpha(x_j)^\circ = \varnothing$, and since B_i must share at least one vertex with B_j, $\alpha(x_i)^\partial \cap \alpha(x_j)^\partial \neq \varnothing$.

4. For $NTPP$, since B_i and B_j may not share vertices, and since the edges of B_i and B_j may not cross, the fact that B_i contains an in-x_j-witness means that $\alpha(x_i)^\circ \subset \alpha(x_j)^\circ$ and $\alpha(x_i)^\partial \cap \alpha(x_j)^\partial = \varnothing$.

5. For TPP, recall that $B_i \setminus B_j$ splits B_i into a set of paths, and each path contains an in-x_j-witness, and since the edges of B_i and B_j may not cross, $\alpha(x_i)^\circ \subset \alpha(x_j)^\circ$, and since B_i must share at least one vertex with B_j, $\alpha(x_i)^\partial \cap \alpha(x_j)^\partial \neq \varnothing$.

6. For $\top$, since B_i and B_j may share vertices, and since the edges of B_i and B_j may cross, there are no conditions to be violated.

So, given a weakly realizable AT-graph (G, R) which satisfies the previous conditions, we can build a model of φ in $OD(\mathbb{R}^2)$.

Now we show that if φ has a model in $OD(\mathbb{R}^2)$, then there exists a weakly realizable AT-graph (G, R) satisfying the previous conditions whose size is polynomially bound by the number of variables in φ. By Lemma 6.4.14, φ has a model in which the number of contact points is at most n^2. We choose a region Z which contains $\alpha(x_i)$ (for all $i \leq n$). On Z^∂, we choose three points z_1, z_2, and z_3, and we choose a point z from the exterior of Z, and we connect z to z_1, z_2, and z_3 with edges which are outside Z. We choose a point c_i in each $\alpha(x_i)$ (for all $i \leq n$), and we select at least three points (for B_i) on each $\alpha(x_i)^\partial$ including any contact points, (note that we can draw edges between pairs of vertices in B_i completely contained within $\alpha(x_i)^\partial$). We connect c_i to each point in B_i with edges contained within $\alpha(x_i)$, and we connect each point in B_i to the points z_1, z_2, and z_3 with edges in $Z \setminus \alpha(x_i)$. For each conjunct of φ we will show that we can choose in/out witnesses such that the conditions on (G, R) hold.

1. For PO, we choose a point in $\alpha(x_i)^\partial \cap \alpha(x_j)^\circ$ as an in-x_j-witness on B_i, a point in $\alpha(x_i)^\partial \cap (-\alpha(x_j))^\circ$ as an out-x_j-witness on B_i, and we choose a point in $\alpha(x_j)^\partial \cap \alpha(x_i)^\circ$ as an in-x_i-witness on B_j, a point in $\alpha(x_j)^\partial \cap (-\alpha(x_i))^\circ$ as an out-x_i-witness on B_j.

2. For DC, we choose a point in $\alpha(x_i)^\partial \cap (-\alpha(x_j))^\circ$ as an out-x_j-witness on B_i and a point in $\alpha(x_j)^\partial \cap (-\alpha(x_i))^\circ$ as an out-x_i-witness on B_j. Since $\alpha(x_i)^- \cap \alpha(x_j)^- = \varnothing$, then B_i and B_j may not share any vertices, neither may the edges of B_i and B_j intersect.

3. For EC, $\alpha(x_i)^\partial \setminus \alpha(x_j)^\partial$ splits $\alpha(x_i)^\partial$ into a number of connected subsets, in each of these subsets we choose a point as an out-x_j-witness, likewise for $\alpha(x_j)^\partial$ and out-x_i-witnesses. Since $\alpha(x_i)$ and $\alpha(x_j)$ have at least one contact point, B_i and B_j will share at least one vertex. By Lemma 6.4.14, $\alpha(x_i)$ and $\alpha(x_j)$ have exactly one contact point, so that the edges of B_i and B_j will not intersect.

4. For $NTPP$, we choose a point in $\alpha(x_i)^\partial \cap \alpha(x_j)^\circ$ as an in-x_j-witness on B_i. Since $\alpha(x_i)^\partial \cap \alpha(x_j)^\partial = \varnothing$, then B_i and B_j may not share any vertices, neither may their edges intersect.

5. For TPP, $\alpha(x_i)^\partial \setminus \alpha(x_j)^\partial$ splits $\alpha(x_i)^\partial$ into a number of connected subsets, in each of these subsets we choose a point as an in-x_j-witness. Since $\alpha(x_i)$ and $\alpha(x_j)$ have at least one contact point, B_i and B_j will share at least one vertex. By Lemma 6.4.14, $\alpha(x_i)$ and $\alpha(x_j)$ have exactly one contact point, so that the edges of B_i and B_j will not intersect.

6. For $\top$, no in/out witnesses are required, vertices are permitted to be shared, and edges are permitted to cross.

So, we can see that given a model of φ in $OD(\mathbb{R}^2)$, we can build a suitable weakly realizable AT-graph (G, R). ☐

Theorem 6.4.16. *The satisfiability of an $\mathcal{E}$ formula over $OD(\mathbb{R}^2)$ can be decided in NP.*

Proof. Lemma 6.4.15 allows us in NP to translate the satisfiability of $\mathcal{E}$ formulas to the weak realizability of some AT-graph (G, R). So, since the weak realizability of an AT-graph (G, R) can be decided in NP (by Theorem 3.1.23), the satisfiability of $\mathcal{E}$ formulas over $OD(\mathbb{R}^2)$ can also be decided in NP. ☐

6.4.3 Adding Connectedness to $\mathcal{E}$

Earlier on in this chapter, we saw the $\mathcal{TC}$ language, which is the result of adding a connectedness predicate to the $\mathcal{T}$ language, as well as a number of restricted fragments of the $\mathcal{T}$ language. Although we know that adding the connectedness predicate to $\mathcal{T}$ increases the complexity from PSPACE to EXP [KPHWZ08], we have no results regarding the effect of adding this predicate to any fragments of $\mathcal{T}$.

We define the language $\mathcal{E}_c$ to be the result of taking the language $\mathcal{E}$ defined in Chapter 5, and adding a single unary predicate c, interpreted as connectedness. Given an $\mathcal{E}_c$ formula φ, we denote by $\varphi_{\mathcal{E}}$ the formula which contains only the $\mathcal{E}$-constraints from φ.

Note that the proof of Lemma 6.4.14 from [SŠ04] can also be simply modified to work for $\mathcal{E}_c$ interpreted over $ROP(\mathbb{R}^2)$. We now reduce the $\mathcal{E}_c$ satisfiability problem over $ROP(\mathbb{R}^2)$ to the $\mathcal{E}$ satisfiability problem over $OD(\mathbb{R}^2)$. Note that by Theorem 6.4.9, there always exists a *small* model of any satisfiable $\mathcal{E}$ formula which has a size (the number of components) of $2n^3$. That is, each variable is mapped to a region consisting of at most $2n^2$ components.

So, given an $\mathcal{E}_c$ formula φ, we describe a transformation of φ to an $\mathcal{E}$ formula φ^*, such that φ is satisfiable in $ROP(\mathbb{R}^2)$ if and only if φ^* is satisfiable in $OD(\mathbb{R}^2)$. We start with $\varphi^* = \varphi$, and we will systematically replace the conjuncts as follows. For each $i \leq n$ such that $c(x_i)$ is not a conjunct of φ, we create m ($m = 2n^2$) new variables $x_{i_1}, \ldots, x_{i_m}$. Now, for each $j \leq n$ such that $R_{ij}(x_i, x_j)$ or $R_{ji}(x_j, x_i)$ is a conjunct of φ, we replace the conjunct as follows.

1. If R_{ij} (or R_{ji}) is DC, without loss of generality, we replace the conjunct $DC(x_i, x_j)$ with the following conjunction.

$$\bigwedge_{k \leq m} DC(x_{i_k}, x_j) \wedge \bigwedge_{l < l' \leq m} DC(x_{i_l}, x_{i_{l'}})$$

2. If R_{ij} (or R_{ji}) is PO, without loss of generality, we replace the conjunct $PO(x_i, x_j)$ with the following conjunction (for some $k \leq m$).

$$PO(x_{i_k}, x_j) \wedge \bigwedge_{k' \leq m, k' \neq k} \top(x_{i_{k'}}, x_j) \wedge \bigwedge_{l < l' \leq m} DC(x_{i_l}, x_{i_{l'}})$$

3. If R_{ij} (or R_{ji}) is EC, without loss of generality, we replace the conjunct

$EC(x_i, x_j)$ with the following conjunction (for some $k \leq m$).

$$EC(x_{i_k}, x_j) \wedge \bigwedge_{k' \leq m, k' \neq k} DC(x_{i_{k'}}, x_j) \wedge \bigwedge_{l < l' \leq m} DC(x_{i_l}, x_{i_{l'}})$$

4. (i): If R_{ij} is $NTPP$, we replace the conjunct $NTPP(x_i, x_j)$ with the following conjunction.

$$\bigwedge_{k \leq m} NTPP(x_{i_k}, x_j) \wedge \bigwedge_{l < l' \leq m} DC(x_{i_l}, x_{i_{l'}})$$

(ii): If R_{ji} is $NTPP$, we replace the conjunct $NTPP(x_j, x_i)$ with the following conjunction (for some $k \leq m$).

$$NTPP(x_j, x_{i_k}) \wedge \bigwedge_{k' \leq m, k' \neq k} \top(x_j, x_{i_{k'}}) \wedge \bigwedge_{l < l' \leq m} DC(x_{i_l}, x_{i_{l'}})$$

5. (i): If R_{ij} is TPP, we replace the conjunct $TPP(x_i, x_j)$ with the following conjunction (for some $k \leq m$).

$$TPP(x_{i_k}, x_j) \wedge \bigwedge_{k' \leq m, k' \neq k} NTPP(x_{i_{k'}}, x_j) \wedge \bigwedge_{l < l' \leq m} DC(x_{i_l}, x_{i_{l'}})$$

(ii): If R_{ji} is TPP, we replace the conjunct $TPP(x_j, x_i)$ with the following conjunction (for some $k \leq m$).

$$TPP(x_j, x_{i_k}) \wedge \bigwedge_{k' \leq m, k' \neq k} \top(x_j, x_{i_{k'}}) \wedge \bigwedge_{l < l' \leq m} DC(x_{i_l}, x_{i_{l'}})$$

6. If R_{ij} (or R_{ji}) is $\top$, without loss of generality, we replace the conjunct $\top(x_i, x_j)$ with the following conjunction.

$$\bigwedge_{k \leq m} \top(x_{i_k}, x_j) \wedge \bigwedge_{l < l' \leq m} DC(x_{i_l}, x_{i_{l'}})$$

Lemma 6.4.17. *We can transform an $\mathcal{E}_c$ formula φ in polynomial time, to an $\mathcal{E}$ formula φ^* such that φ is satisfiable in $ROP(\mathbb{R}^2)$ if and only if φ^* is satisfiable in $OD(\mathbb{R}^2)$.*

Proof. Firstly, if we have a model of φ in $ROP(\mathbb{R}^2)$, we can build a model in

$OD(\mathbb{R}^2)$ of φ^*. It is straightforward to see that for each variable x_i where $c(x_i)$ is a conjunct of φ, we can take an open disc subset of $\alpha(x_i)$ such that all relations in φ^* involving x_i are preserved, and assign $\alpha^*(x_i)$ respectively to this subset. Similarly, for each variable x_j where $c(x_j)$ is not a conjunct of φ, we can take a set of open disc subsets of $\alpha(x_j)$, corresponding to each x_{j_k} variable (where $k \leq 2n^2$), such that every relation in φ^* involving x_{j_k} is preserved, and assign $\alpha^*(x_{j_k})$ respectively to these subsets.

Secondly, if we have a model of φ^*, α^*, in $OD(\mathbb{R}^2)$, we can easily produce a model of φ in $ROP(\mathbb{R}^2)$. For each $i \leq n$ where $c(x_i)$ is a conjunct of φ, we set $\alpha(x_i) = \alpha^*(x_i)$, and for each $i \leq n$ where $c(x_i)$ is not a conjunct of φ, we set $\alpha(x_i) = \bigcup_{k \leq m} \alpha^*(x_{i_k})$. It is straightforward to see that α is a model of φ in $ROP(\mathbb{R}^2)$. □

Theorem 6.4.18. *The satisfiability problem of the $\mathcal{E}_c$ language over $ROP(\mathbb{R}^2)$ is in NP time.*

Proof. By Theorem 6.4.16, the satisfiability problem of the $\mathcal{E}$ language over $OD(\mathbb{R}^2)$ is decidable in NP time. By Lemma 6.4.17, we can reduce the satisfiability problem of the $\mathcal{E}_c$ language over $ROP(\mathbb{R}^2)$ to the satisfiability problem of the $\mathcal{E}$ language over $OD(\mathbb{R}^2)$. □

We can now extend this result to the language which results from adding a unary connectedness predicate to $RCC8$.

Corollary 6.4.19. *The satisfiability problem of $RCC8$ with connectedness over $ROP(\mathbb{R}^2)$ is in NP time.*

Proof. By the same reasoning as Theorem 6.4.11, each $RCC8$ with connectedness formula is equisatisfiable with at most 8^{n^2} $\mathcal{E}_c$ formulae. We can nondeterministically choose one of these, and verify its satisfiability in NP time. □

6.5 Conclusion

This chapter has provided two new results on the complexity of the RCC8 language and two related languages $\mathcal{E}$, and $\mathcal{E}_c$.

First, this chapter introduced relation algebras, which were first introduced by Tarski [Tar41]. Then, we introduced a class of problems involving systems of relations called *constraint satisfaction problems*. Constraint satisfaction problems are well known in the field of computer science, and in particular in artificial intelligence (see for example [Mac77]), and there have been efforts to apply techniques for solving constraint satisfaction problems to the topological inference problem. If we restrict our spatial logics down to languages which fulfill the requirements of constraint languages, then the topological inference problem for these languages becomes a constraint satisfaction problem.

There has been considerable attention paid to applying one particular technique from constraint satisfaction, called *path consistency*, to the topological inference problem, see [RN99]. However, path consistency required a very specific property of the system of relations, which many of these spatial logics did not have. As a result of this, applying constraint satisfaction techniques to the topological inference problem is difficult at best. The class of topological constraint languages is still a useful class of restricted spatial logics, however, and one which contains a number of languages of low complexity. One of the most famous of these topological constraint languages is called RCC8, and the second half of this chapter gave a number of complexity results regarding this language.

First, we provide a complexity result showing a restricted fragment of RCC8, called $\mathcal{E}$, to be in NLOGSPACE, which gives a simple proof of NP-time membership for RCC8. This result allowed us to give a simple proof that the relation algebraic property of algebraic closure can be used to determine the satisfiability of RCC8, and although this result has appeared in [RN99], the result we present is considerably simpler. Then, we gave a simplified and corrected version of the proof that the complexity of $\mathcal{E}$ over internally connected subsets of the Euclidean plane is NP-complete - the original proof, with minor errors, appeared in [SŠ04]. Finally, we expanded the previous proof to cover a new language $\mathcal{E}_c$ which includes a predicate which is interpreted as being the property of a region being internally connected.

Chapter 7

Conclusion

This thesis has investigated the computational complexity of the satisfiability problems of a class of spatial logics called topological constraint languages. In order to perform automated reasoning on spatial data, we need spatial logics whose satisfiability problems are in low complexity classes. However, spatial logics are extremely computationally complex. First order spatial logics are typically undecidable. The expressiveness of a spatial logic is very closely linked to its complexity. One way to achieve spatial logics with practically computable satisfiability problems is to restrict the expressiveness of our logics. Topological constraint languages are the result of a particular kind of syntactical restriction.

This thesis provides an introduction and survey of spatial logics and in particular, topological constraint languages, with attention to the relationship between the constraint languages and the models of these languages. In Chapter 4 we introduced an algebraic structure which incorporates a topological relation, called a Boolean contact algebra. The starting point in this thesis for spatial logics is the first order language of these Boolean contact algebras. We gave a proof of the undecidability of this language, and then introduced a series of increasingly stronger restrictions of the language, in the form of a group of topological constraint languages.

In Chapter 5, we gave a survey of the approaches to solving the satisfiability problem of various spatial logics. The strong connection between modal logic and topology has been known since McKinsey & Tarski [MT44], and we started with the spatial logic $\mathcal{T}$ which is equivalent to the modal logic $S4$. We gave a simple proof of the equivalence of $\mathcal{T}$ and $S4$, and then examined a superset of $\mathcal{T}$ which adds the ability to place restrictions on the number of components a region may have, called $\mathcal{TCC}$, and gave a brief outline of its membership in the NEXP time complexity class. Then, we looked at a languages which are progressively stronger restrictions of $\mathcal{T}$. We showed that the BRCC8 language is equivalent to the existential theory of Boolean connection algebras, and gave an outline of the proof of the membership of BRCC8 in NP (for arbitrary topological spaces) and in PSPACE (for the Euclidean plane).

In Chapter 6, we introduced topological constraint languages in terms of relation algebras. We highlighted the problems associated with using techniques from relation algebras to determine the satisfiability problems of topological constraint languages. In particular, we explained that a property of relation algebras called 'path consistency' is not necessarily equivalent to the satisfiability problem for topological constraint languages. The main result of this thesis concerns RCC8, and a fragment of RCC8 which we call $\mathcal{E}$. The formulae of RCC8 are expressible in terms of an exponential number of $\mathcal{E}$ formulae, and we showed that the $\mathcal{E}$ satisfiability problem is in NLOGSPACE. We also provided a very simple decision procedure for the satisfiability problem of $\mathcal{E}$ formulae. The NLOGSPACE complexity result also allows us to provide an easy proof of the equivalence of a technique similar to 'path consistency' of $\mathcal{E}$ formulae to the $\mathcal{E}$ satisfiability problem. We then gave a thorough and expanded proof of the NP completeness of the $\mathcal{E}$ satisfiability problem over the domain of regions homeomorphic to the open unit discs by using the string graph result given in Chapter 3. Furthermore, we expanded on this graph theoretic result to solve the problem of a language, based on $\mathcal{E}$, which allows constraints to be made on the connectedness of regions.

7.1 Further Work

A fairly simple result can be obtained by extending Theorem 6.4.18. Instead of simply adding a connectedness predicate to the language $\mathcal{E}$, we could fairly easily add a predicate similar to that of the $\mathcal{TCC}$ language, which allows component counting. Adding such a predicate would almost certainly cause the complexity of $\mathcal{E}_c$ to increase, but it may possibly still remain within NP.

Additionally, in terms of syntactic features, we have focused solely on languages with either a contact, or a connectedness predicate, using other geometric relations would result in a whole family of different logics, see for example spatial logics with convexity [DGC99].

Many questions remain completely open, and we will only consider those which are directly related to the content of the thesis. This thesis has had a special emphasis on topological inference in the Euclidean plane, but many of the complexity results regarding the satisfiability problems of spatial logics are concerned only with arbitrary topological spaces.

In Chapter 6, we saw results regarding $\mathcal{E}$ over the domain of regions homeomorphic to the open unit disc $OD(\mathbb{R}^2)$, as well as an extension of $\mathcal{E}$ to allow restrictions to be placed on the connectedness of regions, called $\mathcal{E}_c$. Although the results for $\mathcal{E}$ easily translate to results for RCC8, any such results for BRCC8/$\mathcal{BC}$ are far from trivial. And similarly, results regarding adding a connectedness predicate to $\mathcal{E}$ translate to allow the addition of the predicate to RCC8, but what effect the addition of a connectedness predicate to BRCC8 would have on the complexity of BRCC8 is unknown.

Another interesting area to investigate would be the expressiveness of these spatial logics (see [PS00], [Dav06]). Adding and removing syntactic features of a language can be a rather imprecise way of modifying these languages. A full analysis of the effect of particular syntactic restrictions on the expressiveness of these languages could provide additional insights into the languages, and allow us to identify more languages in low complexity classes.

Bibliography

[All83] J. F. Allen. Maintaining knowledge about temporal intervals. *Commun. ACM*, 26(11):832–843, 1983.

[BD07] B. Bennett and I. Düntsch. Axioms, algebras and topology. In *Handbook of Spatial Logics*, chapter 3. Springer, 2007.

[BdRV01] P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic*. Cambridge University Press, 2001.

[Ben94] B. Bennett. Spatial reasoning with propositional logic. In *Proceedings of the 4th International Conference on Knowledge Representation and Reasoning*, pages 51–62. Morgan Kaufmann, 1994.

[Ben96] B. Bennett. Modal logics for qualitative spatial reasoning. *Journal of the Interest Group on Pure and Applied Logic*, 4:23–45, 1996.

[Ben97] B. Bennett. *Logical representations for automated reasoning about spatial relationships*. PhD thesis, School of Computing, The University of Leeds, 1997.

[Ben98] B. Bennett. Determining consistency of topological relations. *Constraints*, 3(2&3):213–225, 1998.

[BG91] L. Biacino and G. Gerla. Connection structures. *Notre Dame Journal of Formal Logic*, 32(2):242–247, 1991.

[BIC97] B. Bennett, A. Isli, and A.G. Cohn. When does a composition table provide a complete and tractable proof procedure for a relational constraint language. In *Proceedings of the IJCAI-97 workshop on Spatial and Temporal Reasoning*, Nagoya, Japan, 1997.

[Cla81] B. L. Clarke. A calculus of individuals based on 'connection'. *Notre Dame Journal of Formal Logic*, 22(3):204–218, 1981.

[Coo71] S. Cook. The complexity of theorem proving procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, pages 151–158, 1971.

[Dav06] E. Davis. The expressivity of quantifying over regions. *Journal of Logic and Computation*, 16(6):891–916, 2006.

[DGC99] E. Davis, N. Gotts, and A. G. Cohn. Constraint networks of topological relations and convexity. *Constraints*, 4(3):241–280, 1999.

[dL22a] T. de Laguna. The nature of space – i. *The Journal of Philosophy*, 19(15):393–407, 1922.

[dL22b] T. de Laguna. The nature of space – ii. *The Journal of Philosophy*, 19(16):421–440, 1922.

[dL22c] T. de Laguna. Point, line and surface as sets of solids. *The Journal of Philosophy*, 19(17):449–461, 1922.

[dM64] A. de Morgan. On the syllogism: Iv, and on the logic of relations. *Transactions of the Cambridge Philosophical Society*, 10:331–358, 1864.

[Dor98] C. Dornheim. Undecidability of plane polygonal mereotopology. In L. Schubert A. Cohn and S. Shapiro, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the 6th International Conference (KR'98)*, pages 342–353. Morgan Kaufmann, 1998.

[Dün05] I. Düntsch. Relation algebras and their application in temporal and spatial reasoning. *Artificial Intelligence Review*, 23:315–357, 2005.

[DW05] I. Düntsch and M. Winter. A representation theorem for boolean contact algebras. *Theoretical Computer Science*, 347(3):498–512, 2005.

[DWM01] I. Düntsch, H. Wang, and S. McCloskey. A relation-algebraic approach to the region connection calculus. *Theoretical Computer Science*, 255(1-2):63–83, 2001.

[Ege91] M. J. Egenhofer. Reasoning about binary topological relations. In *SSD '91: Proceedings of the Second International Symposium on Advances in Spatial Databases*, pages 143–160, London, UK, 1991. Springer-Verlag.

[GGC96] N. M Gotts, J. M. Gooday, and A. G. Cohn. A connection based approach to common-sense topological description and reasoning. *The Monist*, 79(1):51–75, 1996.

[GKPR96] L. Gasieniec, M. Karpinski, W. Plandowski, and W. Rytter. Efficient algorithms for Lempel-Ziv encoding (extended abstract). In *SWAT '96: Proceedings of the 5th Scandinavian Workshop on Algorithm Theory*, pages 392–403, London, UK, 1996. Springer-Verlag.

[GPP95] M. Grigni, D. Papadias, and C. H. Papadimitriou. Topological inference. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence, Montreal*, pages 901–905, 1995.

[Hay79] P. J. Hayes. The naive physics manifesto. In D. Michie, editor, *Expert Systems in the Micro-Electronic Age*, pages 242–270. Edinburgh University Press, 1979.

[HM92] J. Y. Halpern and Y. Moses. A guide to completeness and complexity for modal logics of knowledge and belief. *Artificial Intelligence*, 54(3):319–380, 1992.

[Joh82] P. T. Johnstone. *Stone Spaces*. Cambridge University Press, 1982.

[Kah75] D.W. Kahn. *Topology: An Introduction to the Point-Set and Algebraic Areas*. Williams & Wilkins Company, Baltimore, 1975.

[Kel60] J. L. Kelley. *General Topology*. Van Nostrand, Princeton (New Jersey, USA), 1960.

[KGK86] J. Kratochvíl, M. Goljan, and P. Kučera. String graphs. *Rozpravy Československ Akad. Věd Řada Mat. Přírod. Věd.*, 3(96):1–96, 1986.

[Kop89] S. Koppelberg. *Handbook of Boolean Algebras*, volume 1. North-Holland, Amsterdam, 1989.

[KPHWZ08] R. Kontchakov, I. Pratt-Hartmann, F. Wolter, and M. Zakharyaschev. Topology, connectedness, and modal logic. In *Proceedings of AiML 2008*, 2008.

[Kra91] J. Kratochvíl. String graphs. ii. recognizing string graphs is np-hard. *Journal of Combinatorial Theory, Series B*, 52:67–78, 1991.

[Kri59] S. Kripke. A completeness theorem in modal logic. *Journal of Symbolic Logic*, 24:1–14, 1959.

[LM94] P. B. Ladkin and R. D. Maddux. On binary constraint problems. *J. ACM*, 41(3):435–469, 1994.

[LW06] S. Li and H. Wang. RCC8 binary constraint network can be consistently extended. *Artificial Intelligence*, 170:1–18, 2006.

[LY03] S. Li and M. Ying. Region connection calculus: Its models and composition table. *Artificial Intelligence*, 145:121–146, 2003.

[Mac77] A. K. Mackworth. Consistency in networks of relations. *Artificial Intelligence*, 8(1):99–118, 1977.

[McK41] J. C. C. McKinsey. A solution of the decision problem for the Lewis systems S2 and S4, with an application to topology. *Journal of Symbolic Logic*, 6:117–134, 1941.

[Mon74] U. Montanari. Networks of constraints: Fundamental properties and applications to picture processing. *Information Sciences*, 7(2):95–132, 1974.

[MT44] J. C. C. McKinsey and A. Tarski. The algebra of topology. *Annals of Mathematics*, 45:141–191, 1944.

[MT01] B. Mohar and C. Thomassen. *Graphs on Surfaces*. John Hopkins University Press, Baltimore, 2001.

[New64] M. H. A. Newman. *Elements of the Topology of Plane Sets of Points*. Cambridge University Press, Cambridge, 1964.

[Nut99] W. Nutt. On the translation of qualitative spatial reasoning problems into modal logics. In *KI '99: Proceedings of the 23rd Annual German Conference on Artificial Intelligence*, pages 113–124, London, UK, 1999. Springer-Verlag.

[Pap94] C. H. Papadimitriou. *Computational Complexity.* Addison Wesley, 1994.

[PH01a] I. Pratt-Hartmann. Empiricism and rationalism in region-based theories of space. *Fundamenta Informaticae*, 46:159–186, 2001.

[PH01b] I. Pratt-Hartmann. A topological constraint language with component counting. *Journal of Applied Non-Classical Logics*, 11(3-4):1–90, 2001.

[PH07] I. Pratt-Hartmann. First-order mereotopology. In *Handbook of Spatial Logics*, chapter 2. Springer, 2007.

[PR98] W. Plandowski and W. Rytter. Application of lempel-ziv encodings to the solutions of word equations. In *Proceedings of the 25th International Colloquium on Automata, Languages and Programming*, volume 1443 of *Lecture Notes in Computer Science*, pages 731–742, 1998.

[PS00] I. Pratt and D. Schoop. Expressivity in polygonal, plane mereotopology. *Journal of Symbolic Logic*, 65(2):822–838, 2000.

[PT02] J. Pach and G. Tóth. Recognizing string graphs is decidable. In *GD '01: Revised Papers from the 9th International Symposium on Graph Drawing*, pages 247–260, London, UK, 2002. Springer-Verlag.

[RCC92a] D. A. Randell, A. G. Cohn, and Z. Cui. Computing transitivity tables: A challenge for automated theorem provers. In D. Kapur, editor, *Proceeding of the 11th International Conference on Automated Deduction (CADE-11)*, volume 607 of *LNAI*, pages 786–790. Springer, Saratoga Springs, NY., 1992.

[RCC92b] D. A. Randell, Z. Cui, and A. G. Cohn. A spatial logic based on regions and connection. In *Proc. 3rd Int. Conf. on Knowledge*

Representation and Reasoning, pages 165–176, San Mateo, 1992. Morgan Kaufmann.

[Ren98] J. Renz. A canonical model of the region connection calculus. In *Proceedings of the 6th International Conference on Knowledge Representation and Reasoning*, pages 330–341. Morgan Kaufmann, 1998.

[RL05] J. Renz and G. Ligozat. Weak composition for qualitative spatial and temporal reasoning. In *Proceedings of the Eleventh International Conference on Principles and Practice of Constraint Programming (CP'05)*, 2005.

[RN99] J. Renz and B. Nebel. On the complexity of qualitative spatial reasoning: a maximal tractable fragment of the region connection calculus. *Artif. Intell.*, 108(1-2):69–123, 1999.

[Sch90] W. Schnyder. Embedding planar graphs on the grid. In *SODA '90: Proceedings of the first annual ACM-SIAM symposium on Discrete algorithms*, pages 138–148, Philadelphia, PA, USA, 1990. Society for Industrial and Applied Mathematics.

[SŠ04] M. Schaefer and D. Štefankovič;. Decidability of string graphs. *J. Comput. Syst. Sci.*, 68(2):319–334, 2004.

[SSŠ03] M. Schaefer, E. Sedgwick, and D. Štefankovič;. Recognizing string graphs in np. *J. Comput. Syst. Sci.*, 67(2):365–380, 2003.

[Tar41] A. Tarski. On the calculus of relations. *The Journal of Symbolic Logic*, 6(3):73–89, 1941.

[Tar56] A. Tarski. Foundations of the geometry of solids. In *Logic, Semantics, Metamathematics*, chapter 2, pages 24–29. Clarendon Press, Oxford, 1956.

[vB92] P. van Beek. Reasoning about qualitative temporal information. *Artificial Intelligence*, 58(1-3):297–321, 1992.

[Whi19] A. N. Whitehead. *An Enquiry Concerning the Principles of Natural Knowledge*. Cambridge University Press, Cambridge, 1919.

[Whi20] A. N. Whitehead. *The Concept of Nature.* Cambridge University Press, Cambridge, 1920.

[Whi29] A. N. Whitehead. *Process and Reality.* The MacMillan Company, New York, 1929.

[WZ00] F. Wolter and M. Zakharyaschev. Spatial reasoning in RCC-8 with boolean region terms. In W. Horn, editor, *Proceedings of the Fourteenth European Conference on Artificial Intelligence, ECAI 2000, Berlin, Germany*, pages 244–248. IOS Press, 2000.