Generating Relation Algebras for Qualitative Spatial Reasoning

Prathap Siddavaatam

Department of Computer Science

Submitted in partial fulfillment
of the requirements for the degree of

Master of Science

Faculty of Mathematics and Science, Brock University
St. Catharines, Ontario

Abstract

Basic relationships between certain regions of space are formulated in natural language in everyday situations. For example, a customer specifies the outline of his future home to the architect by indicating which rooms should be close to each other. Qualitative spatial reasoning as an area of artificial intelligence tries to develop a theory of space based on similar notions. In formal ontology and in ontological computer science, mereotopology is a first-order theory, embodying mereological and topological concepts, of the relations among wholes, parts, parts of parts, and the boundaries between parts.

We shall introduce abstract relation algebras and present their structural properties as well as their connection to algebras of binary relations. This will be followed by details of the expressiveness of algebras of relations for region based models.

Mereotopology has been the main basis for most region based theories of space. Since its earliest inception many theories have been proposed for mereotopology in artificial intelligence among which Region Connection Calculus is most prominent. The expressiveness of the region connection calculus in relational logic is far greater than its original eight base relations might suggest. In the thesis we formulate ways to automatically generate representable relation algebras using spatial data based on region connection calculus. The generation of new algebras is a two pronged approach involving splitting of existing relations to form new algebras and refinement of such newly generated algebras. We present an implementation of a system for automating aforementioned steps and provide an effective and convenient interface to define new spatial relations and generate representable relational algebras.

Acknowledgements

I would like to express my sincere gratitude to many people who made this thesis possible.

I have been greatly indebted in the preparation of this thesis to my supervisor, Dr. Michael Winter. Thanks to his engaging personality and enthusiasm and moreover his great efforts to clarify concepts and simplify terms, helped to make mathematics fun for me. Throughout the period of my masters studies he provided encouragement, sound advice, exemplary teaching and a whole lot of nice ideas. I would have been lost without him.

Additionally I would like to thank my supervisory committee members - Dr. Sheridan Houghten and Dr. Beatrice M. Ombuki-Berman who reviewed my work promptly in spite of their busy schedules.

I am deeply grateful to my wife Richa and son Aarav, who always encouraged me despite the rigors of graduate studies. Without their emotional support in my life I would have never been able to dedicate my time and effort to my studies and this research in pursuit of my goals.

Lastly, and most importantly, I wish to thank my late grandmother Jayalakshmamma. She raised me, supported me, taught me about life and loved me. Her words of inspiration and encouragement still lingers in me today. I dedicate this thesis to her.

Prathap

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The relational methods have been the basis for many conceptual and methodological tools in computer science since the mid-1970's. The plethora of applications of relational methods has been widely demonstrated by series of RelMiCS seminars (*International Seminar on Relational Methods in Computer Science*). In fact relation algebra has been a reference point for analyzing, modeling or resolving several computer science problems such as program specification, heuristic approaches for program derivation, automatic prover design, database and software decomposition, program fault tolerance, testing, data abstraction and information coding, and more importantly in the area of *qualitative spatial reasoning (QSR)*. We refer to [5, 34, 43] for a detailed overview.

QSR being an important branch of Artificial Intelligence predominantly deals with the qualitative aspects of representing and reasoning about spatial entities. A majority of the contemporary qualitative spatial reasoning is based on the behavior of "part of" and "connection" (or "contact") relations in different domains [8, 23], and thereby the objectivity of study in QSR has the important components of expressive power, consistency and complexity of relational reasoning.

The origins of *region-based* theories of space, also referred to as *pointless* geometries, date back to De Laguna [11] and to Whitehead [49]. The most notable feature of pointless approach is that it considers the classical notion of *point* (and the related notions of *line* and *surface*) as being too abstract to be considered as basic primitive notions in the theory of space. Rather than points, these theories take the primitives an alternate notion of a *spatial region* and define binary relations between such regions known as *connection relation*. These theories can then be used to represent space in the context of (qualitative) spatial reasoning. This abets humans to perceive physical world as a mathematical model.

There are many possible applications of QSR; some of which are well known such as reasoning about physical systems in the traditional domain of Qualitative Reasoning. Other works are motivated by the necessity of giving a semantics to natural language spatial expressions, e.g., [48], which tend to be predominantly qualitative rather than quantitative (consider prepositions such as 'in', 'on' and 'through'). Another large and growing application area is Geographical Information Systems: there is a need for qualitative spatial query languages such as those described in [6] and requires tools for navigation as in the case of [40]. Other applications include specifying syntax and semantics of Visual Programming languages [9].

Mereotopology - consisting of some topological notion of *contact* and a mereological notion of *parthood* - is the common core to most region-based theories of space. Broadly speaking, mereotopology is a hybrid of topology, a mathematical model for connectedness and contact, and mereology, a description of the parthood relationship. It is an indispensable part of any comprehensive framework for qualitative spatial reasoning with regard to the characteristic of topology and its close relation to how humans perceive space. The history of QSR can be traced back to works in the science of phenomena (Husserl, 1913; Whitehead, 1920, 1929)- what we call today "commonsensical" in artificial intelligence. Other motivations to study the topological and mereological relations of space include-an appealing alternative to set theory or point-set topology, or as a region-based alternative to Euclidean geometry. In addition, mereotopology is generally flexible and can be applied to many fields whose primary purpose may not be of spatial character.

Since its earliest inception, many possible theories have been proposed for mereotopology, the Region Connection Calculus (RCC) [7] being the most prominent, originated from Clarke's theory [4]. RCC was first proposed by Randell et al. in [37, 39], with an intention to describe a logical framework for mereotopology. Later, it was shown [47] that models of the RCC are isomorphic to so-called Boolean connection algebras (or Boolean contact algebras), i.e., Boolean algebras together with a binary contact relation C satisfying certain axioms. Since lattices and Boolean algebras in particular are well-known mathematical structures, this approach led to an intensive study of the properties of the RCC including several topological representation theorems [12, 18, 20, 47].

In the RCC theory, the *Jointly Exhaustive and Pairwise Disjoint* (JEPD) set of topological relations known as RCC-8 were identified as being of particular importance. RCC-8 contains relations "x is disconnected from y", "x is externally connected to y","x partially overlaps y","x is equal to y","x is tangential proper part of y","x is

non-tangential proper part of y", and the inverses of the latter two relations. In addition this kind of categorization of topological relations was independently given by Egenhofer [22] in the context of Geographical Information Systems. RCC-8 supports the notion of a composition table (refer to Definition 14) since it is a JEPD. Such a composition table appeared first [10] in its most basic form.
Following the lead of [10], Düntsch et al. [13, 16, 17] initiated using methods of relation algebras to study contact relations and explore their expressive power with respect to topological domains. This idea led to the RCC-8 composition table, i.e., a relation algebra based on eight atomic relationship between two regions [38]. Several refinements of the eight atomic relations produced new algebras up to 25 atoms [15, 16, 17]. Each time new relations were obtained by splitting certain atoms from the previous algebra into two new relations and simultaneously removing certain entries in the composition table for one of the new atoms. It is easy to verify that there are more atoms within the current set of 25 atoms that can be split. But the manual computation of the resulting algebra tends to be impracticable. This is precisely the reason the results in this thesis are outlined so that it details a mechanism to overcome these obstacles. We have developed a computer program written in the functional programming language Haskell to automate these manual tasks.

The method of splitting in relational algebras was formulated by H. Andréka et al. [2]. In [2] the authors adapted an existing method already known in the theory of cylindric algebras, originating with L. Henkin [28], which was used to obtain nonrepresentable cylindric algebras from representable ones. Their approach uses a condition of splittability on the atoms in question in order ensure associativity of the composition operation after splitting. Contradictorily, this is violated by all the RCC tables in consideration starting with RCC-11 or also known as the complemented closed disc algebra [13, 15]. We will show a theorem for splitting atoms in a more general setting than [2], by overcoming some of these obstacles and exploring ways to accomodate functional elements like bijections during the splitting process. We refer to [2, 32, 42, 43] for the terminology and notation including all the theoritical aspects used in this thesis.

The remainder of the thesis is structured as follows. In Chapter 2 we begin with the topological background of mereotopology and delve into fundamentals of Relation Algebras (RAs) and present their structural properties as well as their connection to algebras of binary relations. We will discuss using methods of relation algebras to study contact relations and explore their expressive power with respect to topological domains. We will recall the definitions of atom structure and complex algebras which

are essential tools in implementing any algorithm on relation algebras. In Chapter 3 we describe the existing theory of splitting atoms in relation algebras and also present our approach to splitting. We will provide refinement conditions so that the result of a splitting is indeed a relation algebra coupled with some interesting examples. In Chapter 4 we focus on the functionality details of such a system and conclude the thesis in Chapter 5 with potential for future research work.

Chapter 2

Background

As a formal logical system *first-order logic* is widely used in mathematics, philosophy, linguistics and computer science. First-order logic is distinguished from propositional logic by its use of quantifiers; each interpretation of first-order logic includes a domain of discourse over which the quantifiers range. A sentence in first-order logic is written in the form $P(x)$, where P is the predicate and x is the subject, represented as a variable. Complete sentences are logically combined and manipulated according to the same rules as those used in Boolean algebra. We employ first-order logic for all the axiomatizations used in this chapter and thereafter. Both the unary and binary relations are considered predicates where as subset and superset relations are considered binary predicates in logic. Interiors (Int), closures (Cl) and complements denote functions.

As an area of QSR, mereotopology aims at developing formalisms for reasoning about spatial entities [3, 36, 35]. This chapter covers the representational framework of mereotopology as a structure comprising of three parts: a topological part, an algebraic part and a relational (mereological) part. We begin with relevant background on topological aspects of mereotopology specifically related to point-set topology. Mereological relations such as "part-of", "overlap","non-tangential inclusion" and others can be defined in terms of the connection relation. In particular, this connection relation associates some pointless geometries to the field of mereotopology [47]. RCC consists of a set of axioms which are intended to characterize spatial regions from a qualitative perspective. The axioms constrain two binary operations, sum and product, a unary operation, complement, and a binary relation, connection. Sums and products of regions correspond to unions and intersections of regions. The complement of a region is that region outside it, and two regions are connected if they overlap or touch. We describe these axioms in Section 2.4.7.

It is well known for some time, that the expressiveness of reasoning with basic operations on binary relations is equal to the expressive power of the three variable fragment of the first order logic with at most binary relations [46]. Thus it is imperative to use relation algebraic methods, initiated by Tarski [45], to study contact relations in their own right, and explore their expressive power with respect to topological domains. Egenhofer and Sharma [24] introduced the concept of relation algebras into spatial reasoning. Düntsch [13] states Egenhofer's reason as follows:

> "Spatial databases will benefit from the composition table of topological relations if it is applied during data acquisition to integrate independently collected topological information and to derive new topological knowledge; to detect consistency violations among spatial data about some otherwise non-evident topological facts; or during query processing, when spatial queries are less expensive to be executed or involve less objects."

We refer to [32] for a deeper insight on various aspects of relation algebras including its history.
The following section outlines the idea of Düntsch and Winter [18] who characterized the models of region connection calculus in terms of topological spaces. They concluded that the earlier assumptions of the topological representations of RCC being regular spaces is in fact too strong and that even regularity alone is not always satisfied. This is attributed to various possible ways of constructing topological spaces from a model of the RCC. The algebraic part usually comprises of an atomless Boolean algebra or in general terms, an orthocomplemented lattice, both without the smallest element.

Topological Spaces

2.1.1 Properties

We begin with basic definitions for topological spaces as found in standard literature. For more details on general topology we refer to [33]. The intention of the definitions and concepts are meant to emphasize the characterization of the models of RCC from a topological perspective.

Definition 1. *Let X be a non-empty set. A collection $\mathcal{T}$ of subsets of X is said to be a topology on X if*

i *X and the empty set $\emptyset$, belong to $\mathcal{T}$,*

ii *the union of any (finite or infinite) number of sets in $\mathcal{T}$ belongs to $\mathcal{T}$, and*

iii *the intersection of any two sets in $\mathcal{T}$ belongs to $\mathcal{T}$.*

The pair $(X,\mathcal{T})$ is called a topological space.

Example 1. *Let $X = \{ a,b,c,d,e,f \}$ and $\mathcal{T}_1 = \{X,\emptyset,\{a\},\{c,d\},\{a,c,d\},\{b,c,d,e,f\}\}$. Then $\mathcal{T}_1$ is a topology on X as it satisfies all the conditions of Definition 1.*

Example 2. *Let $X = \{ a,b,c,d,e\}$ and $\mathcal{T}_2 = \{X,\emptyset,\{a\},\{c,d\},\{a,c,e\},\{b,c,d\}\}$. Then $\mathcal{T}_2$ is not a topology on X as the union $\{c,d\} \cup \{a,c,e\} = \{a,c,d,e\}$ of two members of $\mathcal{T}_2$ does not belong to $\mathcal{T}_2$; that is, $\mathcal{T}_2$ does not satisfy condition (ii) of Definition 1.*

Definition 2. *Let $(X,\mathcal{T})$ be any topological space. Then the members of $\mathcal{T}$ are said to be open sets.*

Proposition 3. *If $(X,\mathcal{T})$ is any topological space, then*

i *X and $\emptyset$ are open sets,*

ii *the union of any (finite or infinite) number of open sets is an open set and*

iii *the intersection of any finite number of open sets is an open set.*

Definition 4. *Let $(X,\mathcal{T})$ be a topological space. A subset S of X is said to be a closed set in $(X,\mathcal{T})$ if its complement in X, namely $X \setminus S$, is open in $(X,\mathcal{T})$.*

Proposition 5. *If $(X,\mathcal{T})$ is any topological space, then*

i *X and $\emptyset$ are closed sets,*

ii *the intersection of any (finite or infinite) number of close sets is a closed set and*

iii *the union of any finite number of close sets is a close set.*

2.1.2 Regular Closed Sets

An illustration of open and close sets is shown in Figure 2.1. In the Figure 2.1 the points (x,y) satisfying $x^2 + y^2 = r^2$ are located on the circumference where as the points less than r^2 are located interior of the region. The interior points form an open set and the union of both the interior and exterior points forms a close set.

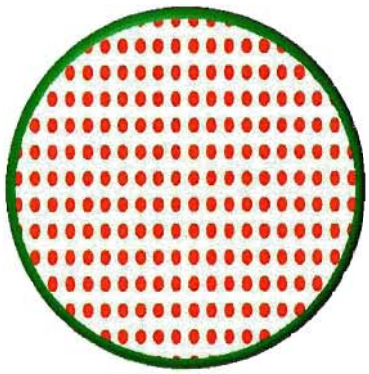


Figure 2.1: Regular region

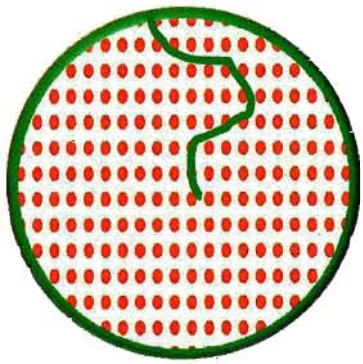


Figure 2.2: Non-regular region

Definition 6. *For any subset S of X, the interior of S denoted as* `Int(S)` *is the largest open set contained in S i.e.,*

$$Int(S) = \bigcup_{\substack{T_{open} \subseteq S \\ T_{open} \in \mathcal{T}}} T_{open} \tag{2.1}$$

Definition 7. *For any subset S of X, the interior of S denoted as* `Cl(S)` *is the smallest closed set contained in S i.e.,*

$$Cl(S) = \bigcap_{\substack{T_{closed} \supset S \\ T_{closed} \in \mathcal{T}}} T_{closed} \tag{2.2}$$

Let $x, y \in \mathcal{T}$, then x and y are called *separated* if $Cl(x) \cap y = x \cap Cl(y) = \emptyset$. A non-empty open set x is called *connected* if it is not the union of two separated nonempty open sets. A set $u \subseteq X$ is called *regular open* if $u = Int(Cl(u))$, and *regular closed*, if $u = Cl(Int(u))$. The set complement of a regular open set is regular closed and vice versa. Figure 2.1 and Figure 2.2 show examples for regular and non-regular regions respectively. It is worth to note that we will be mentioning regions models only in terms of regular closed sets in this thesis.

Boolean Contact Algebra

We begin this section with the basic notions of Boolean algebras.

Definition 8. *A structure $\mathfrak{B} = \langle B, +, ^- \rangle$ of type $\langle 2, 1 \rangle$ is called a Boolean algebra (BA) iff it satisfies the following $(\forall x, y, z \in B)$:*

B1 $x + y = y + x$.

B2 $x + (y + z) = (x + y) + z$.

B3 $\overline{\overline{x} + y} + \overline{\overline{x} + \overline{y}} = x$

In addition to the *union* $+$ and *complementation* $^-$ operators of a Boolean algebra we can define an *intersection* $\cdot$ by $x \cdot y = \overline{\overline{x} + \overline{y}}$. Furthermore the *empty element* 0 and the *universal element* 1 are defined as $0 = x \cdot \overline{x}$ and $1 = x + \overline{x}$ for an arbitrary $x \in B$ respectively. A Boolean algebra is also equipped with a *partial order* $\leq$ definable as $x \leq y$ iff $x + y = y$ or, equivalently, $x \leq y$ iff $x \cdot y = x$.

The distributive identity holds for any given Boolean algebra i.e., for all x, y, z we have $x \cdot (y + z) = (x \cdot y) + (x \cdot z)$ and $x + (y \cdot z) = (x + y) \cdot (x + z)$.

2.2.3 Properties

We will use some basic topological properties required to describe Boolean contact algebras using following lemma.

Lemma 2.2.1. *Let $RC(X)$ be a collection of regular closed sets of $(X, \mathcal{T})$, then $RC(X)$ is a complete Boolean algebra under set inclusion. Then we have:*

1. $v + w = v \cup w$.
2. $v \cdot w = Cl(Int(v \cap w))$.
3. $v^* = Cl(\overline{v})$.

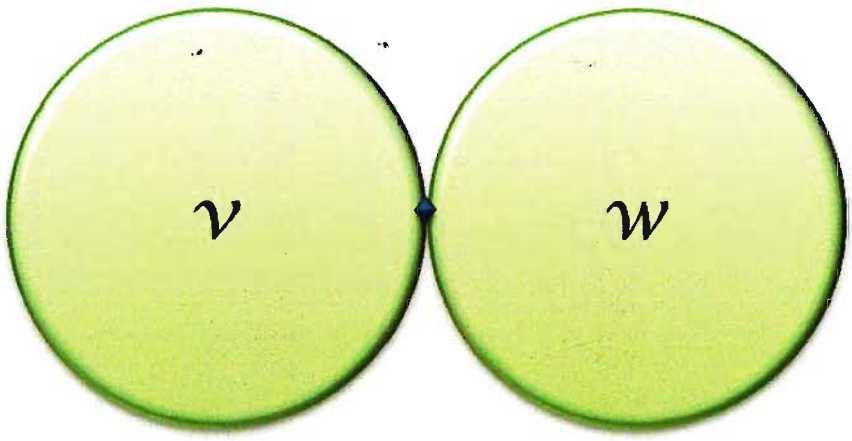


Figure 2.3: Regions v and w in contact by means of a point.

In the Lemma 2.2.1 it is important to note that $v \cdot w \subseteq v \cap w$. This condition can be demonstrated pictorially by Figure 2.3. In Figure 2.3 we observe that although $v \cdot w = \emptyset$, the regions v and w are still in contact with each other by means of a single point i.e., $v \cap w \neq \emptyset$. Using such a notion we define a *contact relation* C on RC(X) as follows.

Definition 9. *[19] A standard model of regions are the regular closed sets of a regular T_1 topological space $(X, \mathcal{T})$ with a relation C defined on RC(X) by*

$$\boxed{uCv \iff u \cap v \neq \emptyset} \tag{2.3}$$

C will be called the `standard (topological) contact` *on X.*

2.2.4 Definitions

The origins of Boolean contact algebras can be traced back to the works of [11, 49] who tend to use regions instead of points as the basic entity of geometry. The notion of "connection"(or "contact") of regions was crucial in all the theories which is basically a *reflexive* and *symmetric* relation C among non-empty regions, satisfying an additional extensionality axiom [14]. In order to formalize mereological structures which are essentially complete Boolean algebras B except the smallest element together with Whitehead's connection relation C, Clarke proposed additional axioms such as *compatibility* and *summation* in [4]. Subsequently RCC [38], also supposed that each proper non-zero region was connected to its complement. Boolean algebras of regular closed sets of regular T_1 spaces along with Whitehead's connection $uCv \iff u \cap v \neq \emptyset$ serves as standard models for such "connection algebras". Stell [44] axiomatized RCC as a Boolean algebra enhanced with *contact relation* C satisfying the axioms in [38].

Definition 10. *[21] Let $Rel(B)$ denote the set of all the binary relations for any given set B. Suppose that $C \in Rel(B)$ satisfying the following properties: for all x, y, z in B we have,*

C_0. $0\ \overline{(C)}\ x$ (Null disconnectedness).

C_1. $x \neq 0 \Rightarrow xCx$ (Reflexivity).

C_2. $xCy \Rightarrow yCx$ (Symmetry).

C_3. xCy *and* $y \leq z \Rightarrow xCz$ (Compatibility).

C_4. $xC(y+z) \Rightarrow (xCy \ or \ xCz)$ (Summation).

C_5. $C(x) = C(y) \Rightarrow x = y$ (Extensionality).

C_6. $(xCz \ or \ yCz^*) \Rightarrow xCy$ (Interpolation).

C_7. $(x \neq 0 \wedge x \neq 1) \Rightarrow xCx^*$ (Connection).

A binary relation C on a Boolean algebra $B\,\langle 0, 1, +, \cdot, * \rangle$ satisfying $C_0 - C_4$ is called a *contact relation.* The structure $\langle B, 0, 1, +, \cdot, *, C \rangle$ is known as a *Boolean contact algebra.* C is called *extensional contact relation* if it satisfies $C_0 - C_5$. If C satisfies C_7, we call it *connected.*

Relation Algebraic Properties

2.3.5 Definitions

Relation algebras have become a core entity for researchers in the area of qualitative spatial reasoning. As we already discussed in the prelude to this chapter, a large part of the contemporary spatial reasoning is based on investigations of the properties of "part of" relations and their extensions to "contact relation" in various domains. Relation algebra is an equational formalism from which it can be inferred to access the existence of relations, given several basic operations, such as Boolean operations on relations, relational composition and converse. Every equation in relation algebra corresponds to its individual theorem, and if these relations are finite, it is possible to construct a *composition table* (refer Definition 14) which serves as a look-up table for such relations.
In this section we will use the notion and basic definitions from [32]. In particular, we use the varieties NA and RA of nonassociative and associative relation algebras.

Definition 11. *A structure* $\mathfrak{A} = \langle A, +, \cdot, ^-, 0, 1, ,^\smile, ; , 1' \rangle$ *of type* $\langle 2, 2, 1, 0, 0, 1, 2, 0 \rangle$ *is called a relation algebra (RA) iff it satisfies the following:*

R_1 $\langle A, +, \cdot, ^-, 0, 1 \rangle$ *is a Boolean algebra.*

R_2 $\langle A, ; , ,^\smile 1' \rangle$ *is an involuted monoid i.e.,*

R_{2a} $\langle A, ; , 1' \rangle$ *is a semigroup with identity* $1'$.

R_{2b} $a^{\smile\smile} = a$, $(a \ ; \ b)^\smile = b^\smile \ ; \ a^\smile$.

$\mathbf{R_3}$ *For all $x, y, z \in A$ the following formulae are equivalent:*

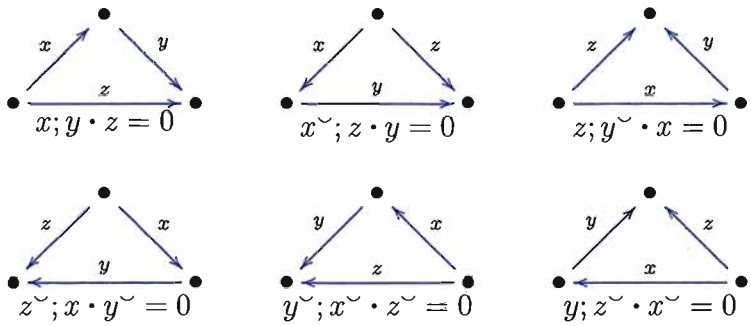
$$x \,;\, y \cdot z = 0 \iff x^{\smile} \,;\, z \cdot y = 0 \iff z \,;\, y^{\smile} \cdot x = 0.$$

We say that $\mathfrak{A}$ is a nonassociative relation algebra (NA) if $\mathfrak{A}$ is a structure satisfying all of the axioms above except associativity of the composition operation ;, i.e., $\mathbf{R_2}$ is weakened by only requiring that 1' is a neutral element for composition. Also in our definitions we adopt complementation notation $^-$ instead of $*$ for historical reasons.

We adopt the regular convention to denote the diversity element $\overline{1'}$ by 0'.

The notion of a subalgebra is as usual, and we will denote the fact that $\mathfrak{B}$ is a subalgebra of $\mathfrak{A}$ by $\mathfrak{B} \subseteq \mathfrak{A}$.

Oriented triangles can be used to visualize $\mathbf{R_3}$ and its immediate consequence the so-called *cycle law*. It states that the following properties are equivalent:



Besides the cycle law above, we will need some basic properties which are summarized in the following lemma. A proof can be found in any of [45, 30, 46, 32, 41].

Lemma 2.3.2. *Let $\mathfrak{B}$ be a relation algebra, and let $x, y, z \in \mathfrak{B}$. Then we have:*

1. $0^{\smile} = 0, 1^{\smile} = 1, (1')^{\smile} = 1'$.
2. $x \leq x; x^{\smile}; x$.
3. $x; (y + z) = x; y + x; z$.
4. *if* $x^{\smile}; x \leq 1'$, *then* $x; (y \cdot z) = x; y \cdot x; z$.
5. *if* $x^{\smile}; x \leq 1'$, *then* $x; \overline{y} \leq \overline{x; y}$.

A relation x is called univalent (or functional) iff $x^{\smile}; x \leq 1'$. It is called injective if $x^{\smile}$ is univalent. A univalent and injective relation is called bijective. We denote the set of all bijections (bijective relations) of a relation algebra $\mathfrak{B}$ by $\mathrm{Bij}\mathfrak{B}$. On addition,

we say that x is total if $1' \leq x; x^\smile$ and surjective if $x^\smile$ is total. A function (or map) is a univalent and total relation.

In the next lemma we have summarized some properties of atoms in relation algebras that will be needed in this thesis. We will denote the set of atoms of a relation algebra $\mathfrak{B}$ by $\mathrm{At}\mathfrak{B}$.

Lemma 2.3.3. *Let $\mathfrak{B}$ be a relation algebra, and $x, y, z \in \mathrm{At}\mathfrak{B}$. Then we have:*

1. *There is an atom $i \leq 1'$ with $x; i = x$.*
2. *$z \leq x; y$ iff $y \leq x^\smile; z$ iff $x \leq z; y^\smile$ iff $z^\smile \leq y^\smile; x^\smile$ iff $y^\smile \leq z^\smile; x$ iff $x^\smile \leq y; z^\smile$.*
3. *If y is a bijection and $x; y \neq 0$, then $x; y$ is an atom.*

Again, a proof can be found in any of [29, 30, 31, 32].

Of particular interest are integral relation algebras. They form basic building block in constructing arbitrary algebras. For details on their importance, we refer to [50, 51].

Definition 12. *A relation algebra $\mathfrak{A}$ is called integral iff for all $x, y \in \mathfrak{A}$, $x; y = 0$ implies that $x = 0$ or $y = 0$.*

It is well-known that the property of being integral is equivalent to the fact that the identity is an atom of $\mathfrak{A}$. Another equivalent property is the requirement that all relations of the algebra are total, i.e., $1' \leq x; x^\smile$.

Notice that (1) and (3) of Lemma 2.3.3 become trivial in integral relation algebras because the identity is an atom respectively every relation is total.

An important example of relation algebra is the *full algebra of binary relations* on the underlying set U, denoted by structure $\langle Rel(U), \cup, \cap, \bar{}, \emptyset, V, ;, \smile, 1' \rangle$ of type $\langle 2, 2, 1, 0, 0, 1, 2, 0 \rangle$ where

Rel(U) - the set of all binary relations on U.

$\cup$ - the union operation on U.

$\cap$ - the intersection operation on U.

$\bar{}$ - the complement operation on U.

$\emptyset$ - empty relation on U.

V - is the product U x U, a universal relation.

$\smile$ - the relational converse.

; - the relational composition.

1' - the identity relation on U.

These full algebras are large even in case of small sets. The cardinality of Rel(U) is given by $2^{|u|^2}$.

Example 3. *Consider a full algebra Rel(3) where 3 = {0, 1, 2}. Rel(3) has 512 elements and 9 atoms. Then the atoms of Rel(3) are given by* { ⟨0,0⟩, ⟨0,1⟩, ⟨0,2⟩, ⟨1,0⟩, ⟨1,1⟩, ⟨1,2⟩, ⟨2,0⟩, ⟨2,1⟩, ⟨2,2⟩ }

Definition 13. *For a given subset B of Rel(U) closed under the distinguished operations* ; *and* $\smile$ *and contains* 1' *is called an algebra of binary relations (BRA) on U.*

Suppose R is a *binary relation* on set U and is a subset of V i.e., a set of ordered pairs $\langle x,y\rangle$ where $x,y \in U$. We usually substitute xRy instead of writing $\langle x,y\rangle \in R$. The *domain* and *range* of R are defined by

$$\begin{aligned} \mathrm{dom}(R) &= \{x \in U : (\exists y \in U) xRy\}, \\ \mathrm{ran}(R) &= \{x \in U : (\exists y \in U) yRx\}. \end{aligned} \tag{2.4}$$

In addition we let $R(x) = \{\ y \in U : xRy\ \}$. Most of the properties for relations can be expressed by equations (or inclusions) among relations, for example

$$\begin{aligned} R \text{ is reflexive} &\iff (\forall x)\ xRx, \\ &\iff 1' \subseteq R. \end{aligned} \tag{2.5}$$

$$\begin{aligned} R \text{ is symmetric} &\iff (\forall x, y)\ [xRx \iff yRx], \\ &\iff R = R^{\smile}. \end{aligned} \tag{2.6}$$

$$\begin{aligned} R \text{ is transitive} &\iff (\forall x, y, z)\ [xRy \wedge yRx \implies xRz], \\ &\iff R\ ;\ R \subseteq R. \end{aligned} \tag{2.7}$$

and

$$\begin{aligned} R \text{ is extensional} &\iff (\forall x, y)\ [R(x) = R(y) \implies x = y], \\ &\iff [-(R\ ;\ -R^{\smile}) \cap -(R^{\smile}\ ;\ -R)] \subseteq 1'. \end{aligned} \tag{2.8}$$

Notice that all the above formulae contain at most three variables in them. This is not a mere coincidence but specifically due to the Theorem 2.3.4 stated below.

Theorem 2.3.4. *[26, 46]*

(i) *The first order properties of binary relations on a set U that can be expressed by equations using the operators* $\langle \cap, \cup, -, \emptyset, V, ;, \smile, 1' \rangle$ *are exactly those which can be expressed with at most three distinct variables.*

(ii) *If B is a collection of binary relations on U, then the closure of B under the operations* $\langle \cap, \cup, -, \emptyset, V, ;, \smile, 1' \rangle$ *is the set of all binary relations on U which are definable in the (language of the) relational structure* $\langle U, B \rangle$ *by first order formulae using at most three variables, two of which are free.*

The algebra of binary relations (BRA) on U is a subalgebra of $Rel(U)$ denoted by $B \leq Rel(U)$. If $R_i \in Rel(U)$ for $i \in I$, we refer the BRA generated by $R_i : i \in I$ by $\langle R_i : i \in I \rangle$. Suppose $X \in \langle R_i : i \in I \rangle$, we say that X is a relation algebra - definable by $R_i : i \in I$. A relation algebra $\mathfrak{B}$ is called *representable* if it is isomorphic to a subalgebra of a product of full algebras of binary relations.

Suppose $\mathfrak{B}$ is a complete and atomic relation algebra specifically if $\mathfrak{B}$ is finite, then the actions of the Boolean operators are uniquely determined by the atoms. In order to determine the structure of $\mathfrak{B}$, it is sufficient to specify the composition and converse operations. In case of an atomic BRA, it is more appropriate to mention composition operation with the aid of *composition table* (CT), which, for any two atomic relations R_i and R_j, specifies the relation $R_i ; R_j$ purely in terms of its constituent atomic relations. Considering these factors, a composition table may be defined as follows,

Definition 14. *[15] A composition table is a mapping* $\mathrm{CT} : At(\mathfrak{B}) \times At(\mathfrak{B}) \longrightarrow 2^{At(\mathfrak{B})}$ *such that for all R, S, T we have* $\mathrm{CT}(R, S) \iff T \subseteq R ; S$ *and iff* $\mathfrak{B}$ *is atomic, we have*

$$\boxed{R ; S = \bigcup \mathrm{CT}(R, S)} \tag{2.9}$$

Note that a CT can be described by a matrix, whose rows and columns are labelled by the atoms and an entry $\langle P, Q \rangle$ is the set of atoms contained in $P ; Q$. For such an algebra $\mathfrak{B}$, if 1' is an atom, we discard column and row pertaining to 1' in its composition table.

BRA is one among the many instances in the class of relation algebras (refer to Lemma 2.3.2), which may be seen as an abstraction of algebras of binary relations [45].

We will employ the BRA definition from Definition 13 to a define a *weak composition* as follows.

Definition 15. *[15] If $\Re = (R_i)_{i \in I}$ is a partition of V such that $\Re$ is closed under converse, and either $R_i \subseteq 1'$ or $R_i \cap 1' = \emptyset$ for all $i \in I$, we define the weak composition as $;_w : \Re \times \Re \longrightarrow 2^{\Re}$ of $\Re$ as the mapping*

$$\boxed{R_i ;_w R_j = S \in \Re : S \cap (R_i ; R_j) \neq \emptyset}. \tag{2.10}$$

Example 4. *Consider the weak composition table for RCC-8 relations as shown in Table 2.1. It is shown as a (7,7) matrix where an entry (i, j) contains a list of atoms below the composition of relations x_i and x_j i.e., $(x_i ; x_j)$. Suppose $x_i = EC$ and $x_j = TPP$, we get resultant composition by the cell entry which contains the atoms $(EC, PO, TPP, NTPP)$.*

From Definition 15, we have $R_i ;_w R_j \subseteq R_i ; R_j$ and in essence if equality holds everywhere i.e., if $;_w = ;$ then such a weak composition table is termed as *extensional.*

;	DC	EC	PO	TPP	$TPP^{\smile}$	NTPP	$NTPP^{\smile}$
DC	DC, EC, PO TPP, $TPP^{\smile}$, 1' NTPP, $NTPP^{\smile}$	DC, EC, PO TPP NTPP	DC, EC, PO TPP NTPP	DC, EC, PO TPP NTPP	DC	DC, EC, PO TPP NTPP	DC
EC	DC, EC, PO $TPP^{\smile}$ $NTPP^{\smile}$	DC, EC, PO TPP, $TPP^{\smile}$ 1'	DC, EC, PO TPP NTPP	EC, PO TPP NTPP	DC, EC	PO TPP NTPP	DC
PO	DC, EC, PO $TPP^{\smile}$ $NTPP^{\smile}$	DC, EC, PO $TPP^{\smile}$ $NTPP^{\smile}$	DC, EC, PO TPP, $TPP^{\smile}$, 1' NTPP, $NTPP^{\smile}$	PO TPP NTPP	DC, EC, PO $TPP^{\smile}$ $NTPP^{\smile}$	PO TPP NTPP	DC, EC, PO $TPP^{\smile}$ $NTPP^{\smile}$
TPP	DC	DC, EC	DC, EC, PO TPP NTPP	TPP NTPP	DC, EC, PO TPP, $TPP^{\smile}$ 1'	NTPP	DC, EC, PO $TPP^{\smile}$ $NTPP^{\smile}$
$TPP^{\smile}$	DC, EC, PO $TPP^{\smile}$ $NTPP^{\smile}$	EC, PO $TPP^{\smile}$ $NTPP^{\smile}$	PO $TPP^{\smile}$ $NTPP^{\smile}$	PO TPP, $TPP^{\smile}$	$TPP^{\smile}$ $NTPP^{\smile}$	PO TPP NTPP	$NTPP^{\smile}$
NTPP	DC	DC	DC, EC, PO TPP NTPP	NTPP	DC, EC, PO TPP NTPP	NTPP	DC, EC, PO TPP, $TPP^{\smile}$, 1' NTPP, $NTPP^{\smile}$
$NTPP^{\smile}$	DC, EC, PO $TPP^{\smile}$ $NTPP^{\smile}$	PO $TPP^{\smile}$ $NTPP^{\smile}$	PO $TPP^{\smile}$ $NTPP^{\smile}$	PO $TPP^{\smile}$ $NTPP^{\smile}$	$NTPP^{\smile}$	PO TPP, $TPP^{\smile}$, 1' NTPP, $NTPP^{\smile}$	$NTPP^{\smile}$

Table 2.1: RCC-8 Composition Table.

The concept of *residuation* of a relation algebra plays a predominant role in most mereological parts of spatial relations. In fact the "part of" in meroelogical terms turns out to be the right residual of the "connection" C relation.

Suppose that $\mathfrak{B}$ is a relation algebra and $a, b \in \mathfrak{B}$. The equations $a ; x = b$ and $x ; a = b$ may not always be resolvable but there are always elements $a \setminus b$ and b / a

known as "right" and "left residual" of b such that it satisfies: $a \,;\, x \leq b \Longleftrightarrow x \leq a \setminus b$ and $x \,;\, a \leq b \Longleftrightarrow x \leq b \,/\, a$ i.e., $a \setminus b$ is the greatest solution of $a \,;\, x \leq b$ and $b \,/\, a$ is the greatest solution of $x \,;\, a \leq b$. In relation algebraic terms these residuals are computed by,

$$a \setminus b = -(a^{\smile} \,;\, -b), \tag{2.11a}$$

$$b \,/\, a = -(b \,;\, a^{\smile}). \tag{2.11b}$$

If a = b, then we consider only right (left) residual of a. These residuals have the following properties:

Lemma 2.3.5. *[17]*

(1) *$a \setminus a$ and $a \,/\, a$ are reflexive and transitive.*

(2) *If a is reflexive, then $a \setminus a \leq a$.*

(3) *If a is symmetric, then $a \setminus a \leq a$ iff $(a \setminus a)^{\smile} \,;\, (a \setminus a) \leq a$.*

Region Connection Calculus

2.4.6 RCC Description

The Region Connection Calculus (RCC) made its debut in [39] as an alternative system similar to the mereological and quasi-Boolean counterpart of Clarke's calculus of individuals, the differing point being the definition of complementation. RCC is intended to provide a logical framework for incorporating spatial reasoning into artificial intelligence systems.

Clarke [4] generalized Leśniewski's classical mereology by considering the "contact" relation C as the basic structural element. The duopoly - a set U of nonempty regions and a binary contact relation C on U, forms the basis of an RCC model.

Recall from Definition 10 that a binary relation C on a set of nonempty regions U is called "extensional contact relation" if it satisfies the axioms $C_0 - C_5$ i.e., C needs to fulfil:

$$C \text{ is reflexive and symmetric,} \tag{2.12a}$$

$$C(x) = C(y) \Longrightarrow x = y. \tag{2.12b}$$

and it was shown in [17] that the extensionality axiom C_5 may be replaced by

$$C \setminus C \text{ is antisymmetric.} \tag{2.13}$$

An important definable relation in terms of contact C is normally interpreted as the *part* relation,

$$xPy \iff C(x) \subseteq C(y), \text{or} \tag{2.14a}$$

$$P \stackrel{def}{=} C \setminus C \tag{2.14b}$$

We can infer from Lemma 2.3.5 and Equation 2.13 that P is a *partial order.*

It is evident from Equation 2.14a that we may as well denote xPy by $x \leq y$. We also write PP for the asymmetric part of P, i.e., $PP = P \cap -1'$.

Mereological Relation	Connection Interpretation	Definition using Ordered Pairs
$DC \stackrel{def}{=} -C$	x is disconnected from y	$x\overline{C}y$
$P \stackrel{def}{=} C \setminus C$	x is part of y	$\forall z[zCx \rightarrow zCy]$
$PP \stackrel{def}{=} P \cap -1'$	x is proper part y	xPy and $x\overline{P}y$
$O \stackrel{def}{=} P^{\smile}\,;P$	x overlaps y	$\exists z[zPx$ and $zPy]$
$PO \stackrel{def}{=} O \cap (P \cup P^{\smile})$	x partially overlaps y	xOy and $x\overline{P}y$ and $y\overline{P}x$
$EC \stackrel{def}{=} C \cap -O$	x is externally connected to y	xCy and $x\overline{O}y$
$TPP \stackrel{def}{=} PP \cap (EC\,;EC)$	x is a tangential proper part of y	$xPPy$ and $\exists z[xECz$ and $zECy]$
$NTPP \stackrel{def}{=} PP \cap -TPP$	x is non-tangential proper part of y	$xPPy$ and $-(\exists z[xECz$ and $zECy])$
$DR \stackrel{def}{=} -O$	x is discrete from y	$x\overline{O}y$

Table 2.2: Relations between regions defined in terms of "C".

Further definitions and intended meaning of the relations on C can be nicely summarized by Table 2.2. Among these relations, the set $\{DC\ , EC\ , PO\ , TPP\ , TPP^{\smile}, NTPP, NTPP^{\smile}\}$ are depicted in the Figure 2.4 and its composition table is given by Table 2.1. These relations belong to the domain of set of all nonempty closed disks in the Euclidean plane defined by Definition 9 where $C = EC \cup PO \cup TPP \cup TPP^{\smile} \cup NTPP \cup NTPP^{\smile} \cup 1'$.

2.4.7 RCC Axioms

RCC basically uses a contact relation C which satisfies the conditions 2.12a and 2.12b.

Figure 2.4: **Closed Disk Relations.**

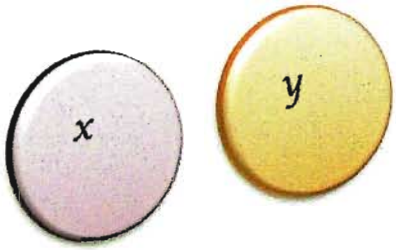


Figure 2.5: x**DC**y

Figure 2.6: x**EC**y

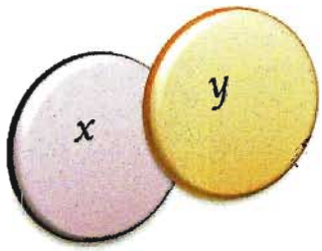


Figure 2.7: x**PO**y

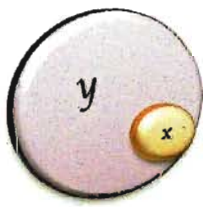


Figure 2.8: x**TPP**y

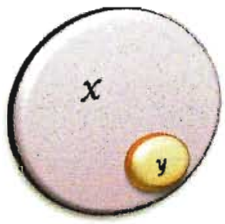


Figure 2.9: x**TPP**$^{\smile}$y

Figure 2.10: x**NTPP**y

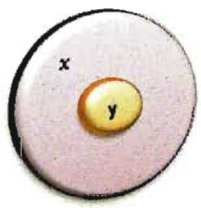
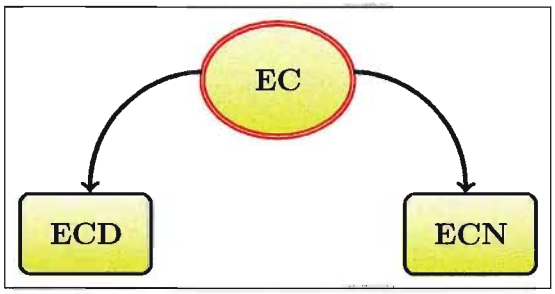


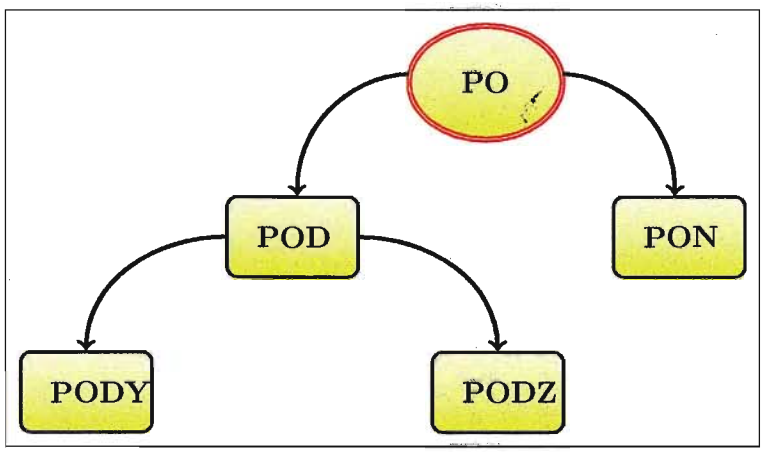
Figure 2.11: x**NTPP**$^{\smile}$y

Definition 16. *[27, 44] A model of the Region Connection Calculus consists of a base set* $U = R \cup N$ *such that* R, N *are disjoint, a distinguished element* $1 \in R$*, a unary operation* $* : R_0 \to R_0$*, where* $R_0 \stackrel{def}{=} R \setminus \{1\}$*, a binary operation* $+ : R \times$

$R \to R$, another binary operation $\cdot : R \times R \to R \cup N$, and a binary operation C on R.



(a) **Splitting of atom EC**



(b) **Splitting of atom PO**

Figure 2.12: Atom splitting in RCC-8

We use the properties from Definition 16 to satisfy the following axioms which make use of the relations derived from C as defined in the Table 2.2. Also, for all the axioms we assume $|U| \geq 2$ to forego trivialities.

RCC$_1$ $(\forall x \in R)\ xCx$.

RCC$_2$ $(\forall x, y \in R)\ [xCy \implies yCx]$.

RCC$_3$ $(\forall x \in R)\ xC1$.

RCC$_4$ $(\forall x \in R, y \in R_0)$,

RCC$_{4a}$ $xCy^* \iff x\overline{NTPP}y$,

RCC$_{4b}$ $xOy^* \iff x\overline{P}y$.

RCC$_5$ $(\forall x, y, z \in R)\ [xC(y + z) \iff xCy \text{ or } xCz]$.

RCC$_6$ $(\forall x, y, z \in R)\ [xC(y \cdot z) \iff (\exists w \in R)(wPy \text{ and } wPz \text{ and } xCw)]$.

RCC$_7$ $(\forall x, y \in R)\ [(x \cdot y) \in R \iff xOy]$.

RCC$_8$ $xPy \text{ and } yPx \implies x = y$.

Figure 2.13: **New relations in RCC-11**

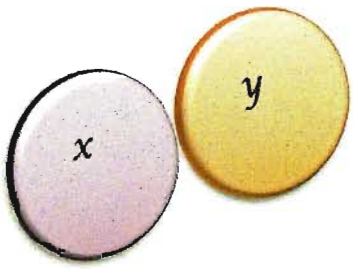


Figure 2.14: x**ECN**y

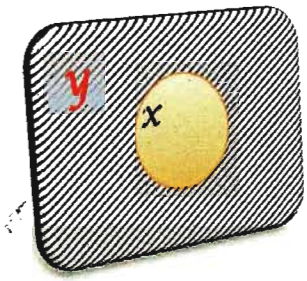


Figure 2.15: x**ECD**y

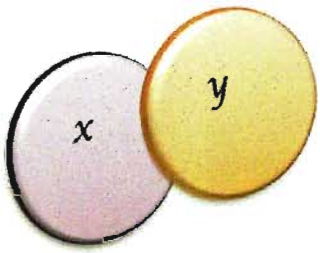


Figure 2.16: x**PON**y

Figure 2.17: x**PODY**y

Figure 2.18: x**PODZ**y

From the set of axioms given above, RCC_1 and RCC_2 imply that the connectedness relation C is a reflexive and symmetric binary relation. Axiom RCC_3 reflects the universal property of region "1". Other axioms such as RCC_{4a} and RCC_{4b} represent

$NTPP = ECD\,;\, NTPP^{\smile}\,;\, ECD$
$P\,;\, NTPP \leq NTPP$
$NTPP\,;\, TPP = NTPP$
$NTPP\,;\, P = NTPP$
$TPP\,;\, NTPP = NTPP$
$ECN = TPP\,;\, ECD$
$xDCz \Longrightarrow xTPP(x+z)$
$xNTPPz$ and $yNTPPz \Longleftrightarrow (x+y)NTPPz$
$xNTPPz \Longrightarrow \overline{x} \cdot zTPPz$
$xNTPPy$ and $xNTPPz \Longleftrightarrow xNTPPy \cdot z$
$ECD\,;\, DC \leq NTPP^{\smile}$
$xECN\,;\, TPPz \Longleftrightarrow xECNx^{*} \cdot xTPPz$
$xTPP^{\smile}\,;\, TPPz \Longleftrightarrow xTPP^{\smile}x \cdot zTPPz$
$xTPP\,;\, TPP^{\smile}z \Longleftrightarrow xTPP(x+z)TPP^{\smile}z$
$yNTPP(x+z)$ and $yDCz \Longrightarrow yNTPPx$
$ECD\,;\, NTPP \leq POD$

Table 2.3: Salient properties of RCC-10 relations.

the idea of complement of a region, RCC_5 denotes the summation of two regions and axioms RCC_6 and RCC_7 represent the product of two regions respectively. As such these axioms show that the structure $\langle R, C, + \rangle$ is a weak model of mereology. However it was proven not to represent a complete model of mereology in [15] since it has an entirely different interpretation of complement. This is attributed to the fact that in RCC models, each proper region x is connected to its complement x^{*}, which happens to be a total contradiction in models of mereology.

If a structure $\langle U, \cdot, +, * \rangle$ represents the algebraic part of an RCC model, then it is a Boolean algebra [16, 44] and every atomless Boolean algebra can be structured into an RCC model by defining an appropriate contact relation [14].

2.4.8 RCC Sequels

The relations between regions as pictured in the Figure 2.4 were considered the base relations in a system commonly referred to as RCC -8. Evidently the universal element 1 is the largest which can be relation algebraically defined from C. More so it was determined in [17] that the investigation of RCC can be restricted to the set $U = R \cap -\{1\}$ and the relations EC and PO split into disjoint nonempty relations definable in terms of P as shown in Figure 2.12. The relation EC i.e., 'x is externally connected to y' splits into two situations depending on whether or not x is equal to

$\overline{y}$ (complement of y), given by ECD and ECN as,

$$ECD = -(P\,;P^{\smile}) \cap -(P^{\smile}\,;P) \qquad xECDy \Longleftrightarrow y = -x \tag{2.15a}$$

$$ECN = EC \cap -ECD \qquad xECNy \Longleftrightarrow x \cdot y = 0, x + y \not\leq 1, xCy \tag{2.15b}$$

Also the relation PO i.e., 'x partially overlaps y' splits into two situations depending on whether or not x is a tangential or non-tangential proper part of $\overline{y}$, given by PON and POD as,

$$POD = PO \cap -(P\,;P^{\smile}) \qquad xPODy \Longleftrightarrow xPOy, x + y = 1 \tag{2.16a}$$

$$PON = PO \cap -POD \qquad xPONy \Longleftrightarrow xPOy, x + y \leq 1 \tag{2.16b}$$

Consequently we get 10 disjoint base relations $\{1', DC, ECD, ECN, POD, PON, TPP, TPP^{\smile}, NTPP, NTPP^{\smile}\}$ from which we can express other relations. These are commonly referred to as RCC -10 relations [15]. We refer to [15] for an extensive elaboration of their relational properties and subsequent interplay with the algebraic operators, some of which are listed in Table 2.3. We may note that the weak composition of these 10 relations is not a relation algebraic composition albeit they constitute the atoms of a semi-associative relation algebra [32].

We observe that the relation POD can further be split into new atoms $PODY$ and $PODZ$ given by,

$$PODZ = ECD\,;NTPP \tag{2.17a}$$

$$PODY = POD \cap -(ECD\,;NTPP) \tag{2.17b}$$

The 11 atoms 1', DC, ECN, ECD, PON, $PODY$, $PODZ$, TPP, $TPP^{\smile}$, $NTPP$, $NTPP^{\smile}$ i.e., 7 symmetric and 4 non-symmetric atoms comprise an RCC -11 model. In addition to the existing relations as in Figure 2.4, the new relations ECN, ECD, PON, $PODY$, $PODZ$ in RCC -11 are depicted in Figure 2.13. A weak composition table for such a model is portrayed by Table 2.4.

In the following chapter we continue to explore the splitting properties of RCC models from a relation algebraic perspective and their implications on mereotopology.

;	DC	ECN	ECD	PON	PODY	PODZ	TPP	TPP$^{\smile}$	NTPP	NTPP$^{\smile}$
DC	1', TPP, TPP$^{\smile}$ NTPP, NTPP$^{\smile}$ PON, ECN, DC	TPP, NTPP PON, ECN DC	NTPP	TPP, NTPP PON, ECN DC	NTPP	NTPP	TPP, NTPP PON, ECN, DC	DC	TPP, NTPP PON, PODY PODZ, ECN ECD, DC	DC
ECN	TPP$^{\smile}$, NTPP$^{\smile}$ PON, ECN, DC	1', TPP, TPP$^{\smile}$ PON, ECN, DC	TPP	TPP, NTPP PON, ECN, DC	TPP, NTPP	NTPP	TPP, NTPP PON, PODY ECN, ECD	ECN, DC	TPP, NTPP PON, PODY PODZ	DC
ECD	NTPP$^{\smile}$	TPP$^{\smile}$	1'	PON	TPP	NTPP	PODY	ECN	PODZ	DC
PON	TPP$^{\smile}$, NTPP$^{\smile}$ PON, ECN DC	TPP$^{\smile}$, NTPP$^{\smile}$ PON, ECN DC	PON	1', TPP, TPP$^{\smile}$ NTPP, NTPP$^{\smile}$ PON, PODY PODZ, ECN ECD, DC	TPP, NTPP PON, PODY PODZ	TPP, NTPP PON, PODY PODZ	TPP, NTPP PON, PODY PODZ	TPP$^{\smile}$, NTPP$^{\smile}$ PON, ECN, DC	TPP, NTPP PON, PODY PODZ	TPP$^{\smile}$, NTPP$^{\smile}$ PON, ECN DC
PODY	NTPP$^{\smile}$	TPP$^{\smile}$, NTPP$^{\smile}$	TPP$^{\smile}$	TPP$^{\smile}$, NTPP$^{\smile}$ PON, PODY PODZ	1', TPP, TPP$^{\smile}$ PON. PODY PODZ	TPP, NTPP PON, PODY PODZ	PODY, PODZ	TPP$^{\smile}$, NTPP$^{\smile}$ PON, PODY ECN, ECD	PODZ	TPP$^{\smile}$, NTPP$^{\smile}$ PON, ECN, DC
PODZ	NTPP$^{\smile}$	NTPP$^{\smile}$	NTPP$^{\smile}$	TPP$^{\smile}$, NTPP$^{\smile}$ PON, PODY PODZ	TPP$^{\smile}$, NTPP$^{\smile}$ PON, PODY PODZ	1', TPP, TPP$^{\smile}$ NTPP, NTPP$^{\smile}$ PON, PODY PODZ	PODZ	TPP$^{\smile}$ NTPP$^{\smile}$, PON PODY, PODZ	PODZ	TPP$^{\smile}$, NTPP$^{\smile}$ PON, PODY PODZ, ECN ECD, DC
TPP	DC	ECN, DC	ECN	TPP, NTPP PON, ECN, DC	TPP, NTPP PON, PODY ECN, ECD	TPP, NTPP PON, PODY PODZ	TPP, NTPP	1', TPP, TPP$^{\smile}$ PON, ECN DC	NTPP	TPP$^{\smile}$, NTPP$^{\smile}$ PON, ECN DC
TPP$^{\smile}$	TPP$^{\smile}$, NTPP$^{\smile}$ PON, ECN, DC	TPP$^{\smile}$, NTPP$^{\smile}$ PON, PODY ECN, ECD	PODY	TPP$^{\smile}$, NTPP$^{\smile}$ PON, PODY PODZ	PODY, PODZ	PODZ	1', TPP, TPP$^{\smile}$ PON,PODY PODZ	TPP$^{\smile}$, NTPP$^{\smile}$	TPP, NTPP PON, PODY PODZ	NTPP$^{\smile}$
NTPP	DC	DC	DC	TPP, NTPP PON, ECN, DC	TPP, NTPP PON, ECN, DC	TPP, NTPP PON, PODY PODZ, ECN ECD, DC	NTPP	TPP, NTPP PON, ECN DC	NTPP	1', TPP, TPP$^{\smile}$ NTPP, NTPP$^{\smile}$ PON, ECN DC
NTPP$^{\smile}$	TPP$^{\smile}$, NTPP$^{\smile}$ PON, PODY PODZ, ECN ECD, DC	TPP$^{\smile}$, NTPP$^{\smile}$ PON, PODY PODZ	PODZ	TPP$^{\smile}$, NTPP$^{\smile}$ PON, PODY PODZ	PODZ	PODZ	TPP$^{\smile}$, NTPP$^{\smile}$ PON, PODY PODZ	NTPP$^{\smile}$	1', TPP, TPP$^{\smile}$ NTPP, NTPP$^{\smile}$ PON, PODZ PODZ	NTPP$^{\smile}$

Table 2.4: RCC-11 Composition Table.

Chapter 3

Splitting & Refinement of Atoms

As we already explored in Chapter 2, contact relation algebra contains more relations than the original RCC -8 relations might suggest; in particular it was refined to RCC -11 and the corresponding weak composition table was also given by Table 2.4. More so it was shown in [15] that each relation algebra generated by the contact relation of an RCC model contains an integral algebra $\mathfrak{A}$ with 25 atoms as a subalgebra. These new atoms were obtained by a two pronged process of splitting certain atoms from the previous algebra into two new relations followed by the removal of certain entries from the composition table for one of the new atoms. Indeed there are more atoms within the current set of 25 atoms that are splittable which can be conveniently verified. Lamentably such a manual task of computing the resulting algebra tends to be impracticable. In this chapter we will develop a mechanism that can be implemented in a computer program in order to support this task. We will capitalize on the various properties discussed earlier in Chapter 2 to formulate such a mechanism. The system implementation for the same will be discussed in Chapter 4.

This chapter is organized as follows. In Section 1 we continue to build on fundamentals of relation algebras discussed in previous chapter. Basically, we recall essential topics of atom structure and complex algebras which act as fundamental tools in algorithm implementation based on relation algebras. In Section 2 we will follow up with existing theory of splitting atoms in relation algebras. We will illustrate its shortcoming in an example by our implementation. In Section 3 we will present our approach to splitting with necessary conditions to ensure that the end result of splitting is certainly a relation algebra and provide few examples. In Section 4 we will discuss the application of splitting atoms in unison with the refinement process from a mereotopological perspective and provide examples.

Atom Structures and Complex Algebras

Boolean algebra forms a reduct of relation algebras with distinguished operations, so every such algebra provides an *atom structure*. A complete and atomic relation algebra can be recovered from its atom structure with the aid of its *complex algebra* construction. Considering that some relation algebras are made for special purposes, e.g., see [1], and it is difficult to store them, these atom structures prove to be very useful. This property is particularly advantageous during storage since atom structure warrants exponentially less space than the entire algebra (see Example 5). For further details on atom structures and complex algebras not mentioned here we refer to [32].

Example 5. *Consider the comparison table given by Table no. 3.1. The first column represents the number of atoms of the relation algebra. The second column denotes the size of the composition table, which arguably contains the largest portion of the data needed to store the relation algebra. Finally, the last column represents the size of the composition table being reduced to atoms.*

Total atoms	Size of composition table	Reduction in atom count
1	4	1
2	16	4
3	64	9
4	256	16
5	1024	25
·	·	·
·	·	·
·	·	·
·	·	·
n	2^n x 2^n	n x n

Table 3.1: Comparitive size analysis - Algebra vs Atom

Definition 17. *An atom structure* $\mathfrak{At}\mathfrak{A} = \langle \mathrm{At}\mathfrak{A}, C(\mathfrak{A}), f, I(\mathfrak{A})\rangle$ *of a NA* $\mathfrak{A}$ *consists of a non-empty set* $\mathrm{At}\mathfrak{A}$ *of atoms, a unary predicate* $I(\mathfrak{A}) = \{x \in \mathrm{At}\mathfrak{A} : x \leq 1'\}$, *a unary function* $f : \mathrm{At}\mathfrak{A} \to \mathrm{At}\mathfrak{A}$ *defined by* $f(x) = x^{\smile}$, *and a ternary relation* $C(\mathfrak{A}) = \{\langle x, y, z\rangle : x, y, z \in \mathrm{At}\mathfrak{A}, x; y \geq z\}$.

Conversely, we start from a relational structure $\mathfrak{S} = \langle U, C, f, I\rangle$, i.e., a set U together with a ternary relation C on U, a unary function $f : U \to U$, and a subset I of U. One may construct an algebra of relational type on $Rel(U)$ of U as follows.

Definition 18. *Given a relational structure* $\mathfrak{S} = \langle U, C, f, I\rangle$ *the complex algebra* $\mathfrak{Cm}\mathfrak{S} = \langle Rel(U), \cup, \cap, {}^{-}, \emptyset, U, ;, {}^{\smile}, 1'\rangle$ *is defined by*

$$X;Y = \{z \in U : \exists x \in X \exists y \in Y \langle x, y, z\rangle \in C\} \text{ and } X^{\smile} = \{f(x) : x \in X\}.$$

The notion of a cycle is very helpful working with atom structures. Cycles basically reflect axiom **R3** or the cycle law introduced earlier. For three elements x, y, z of a relational structure $\mathfrak{S} = \langle U, C, f, I\rangle$ we write $[x, y, z]$ for the following set of up to six triples:

$$\begin{aligned}[x,y,z] =\{&\langle x,y,z\rangle, \langle x^{\smile},z,y\rangle, \langle y,z^{\smile},x^{\smile}\rangle,\\ &\langle y^{\smile},x^{\smile},z^{\smile}\rangle, \langle z^{\smile},x,y^{\smile}\rangle, \langle z,y^{\smile},x\rangle\}\end{aligned} \tag{3.1}$$

$[x, y, z]$ is called a cycle. Its importance can be seen in the next theorem. A proof can be found in [31].

Theorem 3.0.6. *Let* $\mathfrak{S} = \langle U, C, f, I\rangle$ *be a relational structure consisting of a set* U *together with a ternary relation* C *on* U*, a unary function* $f : U \to U$*, and a subset* I *of* U*.*

1. *The following three conditions are equivalent:*

   (i) $\mathfrak{S}$ *is the atom structure of some complete atomic NA.*

   (ii) $\mathfrak{Cm}\mathfrak{S}$ *is a NA.*

   (iii) $\mathfrak{S}$ *satisfies condition (a) and (b)*

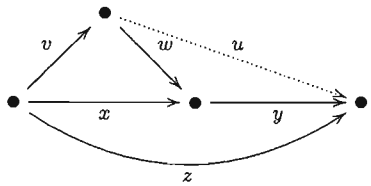
   (a) if $\langle x, y, z\rangle \in C$*, then* $\langle f(x), z, y\rangle \in C$ *and* $\langle z, f(y), x\rangle \in C$*.*

   (b) for all $x, y \in U$*,* $x = y$ *iff there is some* $w \in I$ *such that* $\langle x, w, y\rangle \in C$*.*

2. $\mathfrak{Cm}\mathfrak{S}$ *is a relation algebra iff* $\mathfrak{Cm}\mathfrak{S}$ *is a NA and it satisfies condition (c):*

   (c) for all $x, v, w, x, y, z \in U$*, if* $\langle v, w, x\rangle \in C$ *and* $\langle x, y, z\rangle \in C$*, then there is some* $u \in U$ *such that* $\langle w, y, u\rangle \in C$ *and* $\langle v, u, z\rangle \in C$*.*

From the theorem above one can easily see that condition (a) is already satisfied if C is a union of cycles. Property (c) can nicely be visualized by the following diagram.



Notice that $\mathfrak{Cm}\mathfrak{S}$ is integral iff I is a singleton [32]. In an integral atom structure, i.e., an atom structure of an integral NA, it is possible to remove the cycle including the identity since property (b) determines them uniquely in this case. Therefore, we will normally only list the diversity cycles, i.e., those cycles that do not contain the identity.

Splitting Atoms in Relation Algebra

The method of splitting is well known in cylindric algebra theory, originating with L. Henkin et al. [28]. His work was used to obtain nonrepresentable cylindric algebras from representable ones. The very concept was adopted by H. Andréka et al. [2] to devise splitting atoms in relation algebras. In the following section we recall the theory presented in [2].

Definition 19. *Let $\mathfrak{A}$ and $\mathfrak{B}$ be atomic NA's. We say that $\mathfrak{A}$ is obtained from $\mathfrak{B}$ by splitting if the following conditions are satisfied:*

S1 $\mathfrak{A} \supseteq \mathfrak{B}$.

S2 *Every atom $x \in \mathfrak{A}$ is contained in an atom $c(x) \in \mathfrak{B}$, called the cover of x.*

S3 *For all $x, y \in \mathrm{At}\mathfrak{A}$, if $x, y \leq 0'$, then*

$$x;y = \begin{cases} c(x);c(y) \cdot 0' & \text{iff } x \neq y^{\smile}, \\ c(x);c(y) & \text{iff } x = y^{\smile}. \end{cases}$$

Given an atomic NA $\mathfrak{B}$ one might identify one or more atoms, specify the number of atoms that these identified atoms may be split into symmetric or non-symmetric atoms. This information can summarized by two functions η and θ mapping $\mathrm{At}\mathfrak{B}$ to cardinals. We say that $\mathfrak{A}$ is obtained from $\mathfrak{B}$ by splitting along η and θ if $\mathfrak{A}$ is obtained from $\mathfrak{B}$ by splitting and for all $x \in \mathrm{At}\mathfrak{B}$

$$\begin{aligned} \eta(x) &= \left|\{y \in \mathrm{At}\mathfrak{A} : y \leq x, y \neq y^{\smile}\}\right|, \\ \theta(x) &= \left|\{y \in \mathrm{At}\mathfrak{A} : y \leq x, y = y^{\smile}\}\right|. \end{aligned}$$

In [2] the following two theorems about the existence and uniqueness of a splitting along two functions were shown.

Theorem 3.0.7. *Let $\mathfrak{A}, \mathfrak{A}'$, and $\mathfrak{B}$ be complete atomic NA's. Let η and θ be functions mapping* At$\mathfrak{B}$ *to cardinals. If $\mathfrak{A}$ and $\mathfrak{A}'$ are obtained from $\mathfrak{B}$ by splitting along η and θ, then $\mathfrak{A}$ and $\mathfrak{A}'$ are isomorphic by an isomorphism that leaves $\mathfrak{B}$ fixed.*

Theorem 3.0.8. *Let $\mathfrak{B}$ be an atomic NA. Let η, θ be functions mapping* At$\mathfrak{B}$ *to cardinals, and let $\alpha(x) = \eta(x) + \theta(x)$. Then we have:*

1. *There is an atomic NA $\mathfrak{A}$ obtained from $\mathfrak{B}$ by splitting along η and θ iff the following conditions hold for all $x \in$ At$\mathfrak{B}$:*
   - *(a) $\alpha(x) \geq 1$.*
   - *(b) $\eta(x) = \eta(x^\smile)$.*
   - *(c) $x \leq 1'$ implies $\eta(x) = 0$.*
   - *(d) $x = x^\smile$ implies $\eta(x)$ is even.*
   - *(e) $x \neq x^\smile$ implies $\theta(x) = 0$.*
2. *Suppose $\mathfrak{A}$ is an atomic NA and obtained from $\mathfrak{B}$ by splitting along η and θ. Then $\mathfrak{A}$ is a RA iff $\mathfrak{B}$ is a RA and for all $x, y \in$ At$\mathfrak{B}$ we have; if $\alpha(x) > 1$ and $y; x \neq 0$ and $y \leq 0'$, then $y \leq y; (x; x^\smile \cdot 0')$.*

The situation where a relation algebra and one atom that we want to split into two atoms is given is of particular interest. Therefore, we say that x is splittable in an atomic relation algebra $\mathfrak{B}$ if the following three conditions are satisfied:

A1 $x \leq 0'$.

A2 If $0' \geq y \in$ At$\mathfrak{B}$ and $x; y \neq 0$, then $y \leq (x^\smile; x \cdot 0'); y$.

A3 If $0' \geq y \in$ At$\mathfrak{B}$ and $y; x \neq 0$, then $y \leq y; (x; x^\smile \cdot 0')$.

$\mathfrak{B}$ is called splittable iff $\mathfrak{B}$ has a splittable atom.

By Theorem 3.0.8 x is splittable in $\mathfrak{B}$ iff there is a relation algebra $\mathfrak{A}$ obtained from $\mathfrak{B}$ by splitting such that $x \notin$ At$\mathfrak{A}$. Note that if $\mathfrak{B}$ contains a functional element y below $0'$, i.e., $y^\smile; y \leq 1'$ and $y \leq 0'$, then $\mathfrak{B}$ is not splittable.

Suppose $\mathfrak{A}$ is a relation algebra, $\mathfrak{A}$ is atomic and x is an atom of $\mathfrak{A}$. Then x is called *symmetric* iff $x = x^\smile$ where $x^\smile$ denotes converse of x. Let n be the total number of atoms and s be the number of symmetric atoms in $\mathfrak{A}$. Then $n - s$ denotes the number of non-symmetric atoms in $\mathfrak{A}$. To represent these atoms we shall use natural number sequence $1 \ldots n$. The first atom denoted by 1 is always an identity

since we consider integral relation algebras which implies that the identity is an atom. The Boolean part of complex algebra of an atom structure is defined as usual, and the converse operation on sets is defined componentwise. The atoms $1 \ldots s$ are symmetric and the atoms s+1 ... n are non-symmetric with $m^\smile = m - 1$ if $m - s$ is even and $m^\smile = m + 1$ if $m - s$ is odd. Consequently, $n - s$ must be even. Using these notions we represent integral relation algebras for computation of different splitting methods on the computer.

The next example demonstrates a case where the above mentioned theory of splitting is successful in our program.

Example 6. *Let* $\mathfrak{A}$ *be a* RCC-8 *algebra with* $\mathrm{At}\mathfrak{A} = \{1', DC, EC, PO, TPP, TPP^\smile, NTPP, NTPP^\smile\}$. *We have* $n = 8$ *and* $s = 4$ *since* 1', *DC*, *EC*, *PO are symmetric and rest are non-symmetric atoms respectively. Suppose C is a list of diversity cycles, then any cycle is in fact a representation of upto six triples. Therefore the* RCC-8 *composition table from Table 2.1 condenses to a cycleset C given by,*

C($\mathfrak{A}$) = *[(DC,DC,DC),(DC,DC,EC),(DC,DC,PO),(DC,DC,TPP),*
(DC,EC,EC),(DC,EC,PO),(DC,EC,TPP),(DC,EC,NTPP),(DC,PO,PO),
(DC,PO,TPP),(DC,PO,NTPP),(DC,TPP,TPP),(DC,TPP,NTPP),
(DC,NTPP,NTPP),(EC,EC,EC),(EC,EC,PO),(EC,EC,TPP),(EC,PO,PO),
(EC,PO,TPP),(EC,PO,NTPP),(EC,TPP,TPP),(EC,TPP,NTPP),
(EC,NTPP,NTPP),(PO,PO,PO),(PO,PO,TPP),(PO,PO,NTPP),
(PO,TPP,TPP),(PO,TPP,NTPP),(PO,TPP$^\smile$,TPP$^\smile$),(PO,TPP$^\smile$,NTPP$^\smile$),
(PO,NTPP,NTPP),(PO,NTPP$^\smile$,NTPP$^\smile$),(TPP,TPP,TPP),
(TPP,TPP,NTPP),(TPP,NTPP,NTPP),(TPP,NTPP$^\smile$,NTPP$^\smile$),
(DC,DC,NTPP),(NTPP,NTPP,NTPP)]

Atoms of $\mathfrak{A}$	1'	DC	EC	PO		TPP	TPP$^\smile$	NTPP	NTPP$^\smile$
Atoms of $\mathfrak{B}$	1'	DC	EC	POA	POB	TPP	TPP$^\smile$	NTPP	NTPP$^\smile$

Table 3.2: At$\mathfrak{B}$ vs At$\mathfrak{A}$

It is interesting to note that this particular algebra does not contain any functional elements besides the identity atom. Therefore the splitting method from Theorem 3.0.8 is applicable. We may split the Atom PO into two new atoms. Now we rename the same atom as POA and POB in accordance with the convention of symmetric atoms followed by non-symmetric atoms in pairs, we get a new algebra $\mathfrak{B}$ *with* $n = 9$, $s = 5$ *and the correlation between atoms in* $\mathfrak{B}$ *and* $\mathfrak{A}$ *given by Table 3.2*

The diversity cycles of the new algebra $\mathfrak{B}$ *are as follows:*

$C(\mathfrak{B})$ = [(DC,DC,DC),(DC,DC,EC),(DC,DC,POA),(DC,DC,POB),
(DC,DC,TPP),(DC,DC,NTPP),(DC,EC,EC),(DC,EC,POA),
(DC,EC,TPP),(DC,EC,NTPP),(DC,POA,POA),(DC,POA,POB),
(DC,POB,POB),(DC,POA,TPP),(DC,POB,TPP),(DC,POA,NTPP),
(DC,POB,NTPP),(DC,TPP,TPP),(DC,TPP,NTPP),(DC,NTPP,NTPP),
(EC,EC,EC),(EC,EC,POA),(EC,EC,POB),(EC,EC,TPP),(EC,POA,POA),
(EC,POA,POB),(EC,POB,POB),(EC,POA,TPP),(EC,POB,TPP),
(EC,POA,NTPP),(EC,POB,NTPP),(EC,TPP,TPP),(EC,TPP,NTPP),
(EC,NTPP,NTPP),(POA,POA,POA),(POA,POA,POB),(POA,POB,POB),
(POB,POB,POB),(POA,POA,TPP),(POA,POB,TPP),(POB,POA,TPP),
(POB,POB,TPP),(POA,POA,NTPP),(POA,POB,NTPP),
(POB,POB,NTPP),(POA,TPP,TPP),(POB,TPP,TPP),(POA,TPP,NTPP),
(POB,TPP,NTPP),(POA,$TPP^{\smile}$,$TPP^{\smile}$),(POB,$TPP^{\smile}$,$TPP^{\smile}$),
(POA,$TPP^{\smile}$,$NTPP^{\smile}$),(POB,$TPP^{\smile}$,$NTPP^{\smile}$),(POA,NTPP,NTPP),
(POB,NTPP,NTPP),(POA,$NTPP^{\smile}$,$NTPP^{\smile}$),(POB,$NTPP^{\smile}$,$NTPP^{\smile}$),
(TPP,TPP,TPP),(TPP,TPP,NTPP),(TPP,NTPP,NTPP),
(TPP,$NTPP^{\smile}$,$NTPP^{\smile}$),(DC,EC,POB),(POB,POA,NTPP),
(NTPP,NTPP,NTPP)]

The resulting computation described in Example 6 took a split second to be executed using our implementation. The same task is laborious if computed manually. Incidentally the above methodology cannot be deployed for splitting atoms in algebras containing functional elements like bijections, for example RCC -11 algebra due to the presence of bijective relation namely ECD.

3.0.9 The Extension of a Relation Algebra

In this section we will make use of ordinal arithmetics. Therefore, we want to recall some of their basic properties needed throughout this thesis. The definition of addition can be given inductively:

$$\alpha + 0 = \alpha,$$
$$\alpha + (\beta + 1) = (\alpha + \beta) + 1,$$
and if δ is a limit ordinal, then $\alpha + \delta = \bigcup\{\alpha + \beta : \beta < \delta\}$.

Zero is an additive identity $\alpha + 0 = 0 + \alpha = \alpha$, addition is associative $(\alpha + \beta) + \gamma = \alpha + (\beta + \gamma)$. Furthermore, ordinal addition is left-cancellative, i.e., if $\alpha + \beta = \alpha + \gamma$,

then $\beta = \gamma$. This property allows to define a left subtraction for ordinals. However, right cancellation is not valid. Similar properties are shared by ordinal multiplication recursively defined by:

$$\alpha * 0 = 0,$$
$$\alpha * (\beta + 1) = (\alpha * \beta) + \alpha,$$
and if δ is a limit ordinal, then $\alpha * \delta = \bigcup\{\alpha * \beta : \beta < \delta\}$.

We have $\alpha * 0 = 0 * \alpha = 0$, multiplication is associative, 1 is an identity (or unit) $\alpha * 1 = 1 * \alpha = \alpha$, and satisfies a cancellation law, namely if $\alpha > 0$ and $\alpha * \beta = \alpha * \gamma$, then $\beta = \gamma$. As for addition right cancellation is not valid.

The notion of a splitting is too restrictive for our purposes because it does not allow the splitting of algebras that contain bijections different from the identity. The super algebra property together with the cover property seems to be sufficient. This guarantees that the composition of elements in the subalgebra can be computed by the elements of the super algebra.

Definition 20. *Let $\mathfrak{A}$ and $\mathfrak{B}$ be atomic integral RA's. We say that $\mathfrak{A}$ is an extension of $\mathfrak{B}$ if the following conditions are satisfied:*

S1 $\mathfrak{A} \supseteq \mathfrak{B}$.

S2 *Every atom $x \in \mathfrak{A}$ is contained in an atom $c(x) \in \mathfrak{B}$, called the cover of x.*

Notice that S2 is redundant if $\mathfrak{A}$ and $\mathfrak{B}$ are finite, but it is needed in the infinite case.

If the atoms in $\mathfrak{A}$ satisfy the condition imposed by to function η and θ, then we say that $\mathfrak{A}$ is an extension of $\mathfrak{B}$ along η and θ. Notice that such an extension does not have to be unique up to isomorphism. The next theorem provides necessary conditions for the existence of an extension. Notice that the proof also provides an explicit construction.

Theorem 3.0.9. *Let $\mathfrak{B}$ be a complete atomic integral RA. Let η, θ be functions mapping* $\mathrm{At}\mathfrak{B}$ *to cardinals, and let $\alpha(x) = \theta(x) + \eta(x)$. Then there is a complete atomic integral RA $\mathfrak{A}$ that is an extension of $\mathfrak{B}$ along η and θ if the following conditions hold for all $x, y \in \mathrm{At}\mathfrak{B}$:*

(a) $\alpha(x) \geq 1$.

(b) $\eta(x) = \eta(\breve{x})$.

(c) $x \in \mathrm{Bij}\mathfrak{B}$ *implies* $\alpha(x) = 1$.

(d) $x = \breve{x}$ *implies* $\eta(x) =$ *even, i.e.,* $\eta(x) = 2 * \beta$ *for some ordinal* β.

(e) $x \neq \breve{x}$ *implies* $\theta(x) = 0$.

(f) $y \in \mathrm{Bij}\mathfrak{B}$ *implies* $\alpha(x; y) = \alpha(x)$.

(g) $y \in \mathrm{Bij}\mathfrak{B}$, $x = \breve{x}$ *and* $\eta(x) > 0$ *implies* $x; y = (x; y)^{\smile}$ *and* $\theta(x) = \theta(x; y)$.

(h) $\alpha(x) > 1$, $y; x \neq 0$ *and* $y \notin \mathrm{Bij}\mathfrak{B}$ *implies* $y \leq y; (x; \breve{x} \cap 0')$.

Proof. First we want to show that $\alpha(\breve{x}) = \alpha(x)$ since we will use this property frequently without mentioning. If $x = \breve{x}$ the assertion is trivial. If $x \neq \breve{x}$, then $\alpha(x) = \eta(x)$ follows from (e), which shows together with (b) that $\alpha(x) = \alpha(\breve{x})$.

We are going to construct a relational structure $\mathfrak{S} = \langle U, C, f, I\rangle$ so that $\mathfrak{Cm}\mathfrak{S}$ is a RA and $\mathfrak{B}$ can be embedded into $\mathfrak{Cm}\mathfrak{S}$. Therefore, notice that $\theta(x), \eta(x)$ as well as $\alpha(x) = \theta(x) + \eta(x)$ are ordinals so that we can assume $\alpha(x) = \{\beta : \beta < \alpha(x)\}, \theta(x) = \{\beta : \beta < \theta(x)\}$ and $\eta(x) = \{\beta : \theta(x) \leq \beta < \alpha(x)\}$. Let be $U = \{(x, \beta) : \beta \in \alpha(x)\}$, and define $f : U \to U$ by

$$f(x,\beta) = \begin{cases} (\breve{x}, \beta) & \text{if } x \neq \breve{x} \text{ or } \beta \in \theta(x), \\ (x, \beta + 1) & \text{if } x = \breve{x} \text{ and } \beta \in \eta(x) \text{ and } \beta = \theta(x) + 2 * \beta', \\ (x, \beta - 1) & \text{if } x = \breve{x} \text{ and } \beta \in \eta(x) \text{ and } \beta = \theta(x) + 2 * \beta' + 1. \end{cases}$$

Obviously, we have $f(f(u)) = u$ for all $u \in U$, and $f(u) = u$ iff $u = (x, \beta)$ with $x = \breve{x}$ and $\beta \in \theta(x)$. We will denote the second component of $f(x, \beta)$ by $f_2(x, \beta)$ so that $f(x, \beta) = (\breve{x}, f_2(x, \beta))$ and $\beta = f_2(\breve{x}, f_2(x, \beta))$ follows. Also notice that $f_2(x, \beta) = f_2(\breve{x}, \beta)$.

We want to show the following property:

$$(*) \text{ If } y \in \mathrm{Bij}\mathfrak{B}, \text{ then } f_2(x, \beta) = f_2(x; y, \beta) \text{ for all } \beta \in \alpha(x) = \alpha(x; y).$$

First, suppose $f_2(x, \beta) \neq \beta$. Then $x = \breve{x}$ and $\beta \in \eta(x)$, and, hence, $\eta(x) > 0$, by the definition of f. From (g) we get $(x; y) = (x; y)^{\smile}$ and $\theta(x) = \theta(x; y)$. This also implies that $\eta(x) = \eta(x; y)$ because of (f) and the left-cancellation property of ordinal addition. Consequently, we have $\beta \in \eta(x; y)$, and, hence, $f_2(x, \beta) = f_2(x; y, \beta)$ follows. Now, assume $f_2(x, \beta) = \beta$. If $f_2(x; y, \beta) \neq \beta$, then we conclude analogously to the previous case that $f_2(x, \beta) \neq \beta$ using that $\breve{y}$ is a bijection and that $x; y; \mathit{brevey} = x$. This is a contradiction so that $f_2(x; y, \beta) = \beta$ follows.

Notice that we also have that $y \in \mathrm{Bij}\mathfrak{B}$ implies $f_2(x,\beta) = f_2(y;x,\beta)$ for all $\beta \in \alpha(x) = \alpha(y;x)$. This follows immediately from $\alpha(x) = \alpha(\breve{x})$ and $f_2(x,\beta) = f_2(\breve{x},\beta)$ for all x and β.

Now let $I := \{(1',0)\}$ and define

$$C = \bigcup\{[(x,\beta),(y,\gamma),(z,\delta)] : z \leq x;y \text{ and } x,y,z \notin \mathrm{Bij}\mathfrak{B}\}$$
$$\cup \bigcup\{[(x,\beta),(y,0),(z,\beta)] : z \leq x;y \text{ and } y \in \mathrm{Bij}\mathfrak{B}\}.$$

In order to show that $\mathfrak{CmS}$ is a NA it is sufficient to show Property (b) of Theorem 3.0.6(1)(iii) since C is defined as a union of cycles.

Suppose $(x,\beta) \in U$. Then we have $[(x,\beta),(1',0),(x,\beta)] \subseteq C$. For the converse implication suppose $\langle(x,\beta),(1',0),(z,\delta)\rangle \in C$. We obtain $[(x,\beta),(1',0),(z,\delta)] = [(z,\delta),(\breve{1'},0),(x,\beta)] \subseteq C$ so that the definition of C implies that $x = z$ and $\beta = \delta$.

In order to prove that $\mathfrak{CmS}$ is a RA suppose $\langle(v,\beta),(w,\gamma),(x,\delta)\rangle \in C$ and $\langle(x,\delta),(y,\epsilon),(z,\rho)\rangle \in C$. We distinguish several cases:

1. $v \in \mathrm{Bij}\mathfrak{B}$: We want to show that $(\breve{v};z,\sigma)$ with $\sigma = f_2(\breve{v};z,f_2(z,\rho))$ is the required element, i.e., that

$$\langle(v,0),(\breve{v};z,\sigma),(z,\rho)\rangle \in C \text{ and } \langle(w,\gamma),(y,\epsilon),(\breve{v};z,\sigma)\rangle \in C.$$

   First, we have $\rho \in \alpha(z)$ which implies $f_2(z,\rho) \in \alpha(\breve{z}) = \alpha(\breve{z};v)$ using (f) so that $\sigma \in \alpha(\breve{v};z)$, and, hence, $(\breve{v};z,\sigma) \in U$ follows. Furthermore, we have $v;\breve{v};z = z$ and $w;y = \breve{v};v;w;y \geq \breve{v};x;y \geq \breve{v};z$ so that the property in the definition of C on the first components of each triple is satisfied. From $[(\breve{z},f_2(z,\rho)),(v,0),(\breve{z};v,f_2(z,\rho))] \subseteq C$ by definition we conclude that the first triple is clearly in C. In order to show that the second triple is also in C we distinguish four cases:

   (a) $w,y,\breve{v};z \notin \mathrm{Bij}\mathfrak{B}$: In this case we have $[(w,\gamma),(y,\epsilon),(\breve{v};z,\sigma)] \subseteq C$ and we are done.

   (b) $w \in \mathrm{Bij}\mathfrak{B}$: Then x is a bijection too because $x \leq v;w$ and v and w are bijections. We conclude from $\langle(x,0),(y,\epsilon),(z,\rho)\rangle \in C$ that we have $[(\breve{y},f_2(y,\epsilon)),(\breve{x},0),(\breve{z},f_2(z,\rho))] = [(\breve{z},f_2(z,\rho)),(x,0),(\breve{y},f_2(y,\epsilon))] \subseteq C$, and, hence, that $f_2(y,\epsilon) = f_2(z,\rho)$. This implies $[(\breve{z};v,f_2(z,\rho)),(w,0),(\breve{y},f_2(y,\epsilon))] \subseteq C$ which shows $\langle(w,0),(y,\epsilon),(\breve{v};z,\sigma)\rangle \in C$.

   (c) $y \in \mathrm{Bij}\mathfrak{B}$: Then we conclude from $\langle(x,\delta),(y,0),(z,\rho)\rangle \in C$, and, hence,

$[(x,\delta),(y,0),(z,\rho)] = [(z,\rho),(\breve{y},0),(x,\delta)] \subseteq C$, that $\delta = \rho$. Similarly, from $((v,0),(w,\gamma),(x,\delta)\rangle \in C$, i.e., $[(\breve{w},f_2(w,\gamma)),(\breve{v},0),(\breve{x},f_2(x,\delta))] = [(\breve{x},f_2(x,\delta)),(v,0),(\breve{w},f_2(w,\gamma))] \subseteq C$, we get $f_2(w,\gamma) = f_2(x,\delta)$ since v is a bijection. We conclude

$$\begin{aligned}
\sigma &= f_2(\breve{v}; z, f_2(z,\rho)) \\
&= f_2(\breve{v}; x; y, f_2(x; y,\delta)) && \delta = \rho \text{ and } z = x; y \\
&= f_2(w; y, f_2(x; y,\delta)) && v; w = x \text{ and } v \text{ bijection} \\
&= f_2(w, f_2(x,\delta)) && (*) \text{ twice} \\
&= f_2(w, f_2(w,\gamma)) && f_2(w,\gamma) = f_2(x,\delta) \\
&= \gamma,
\end{aligned}$$

so that $((w,\gamma),(y,0),(\breve{v}; z,\rho)\rangle \in C$ follows.

(d) $\breve{v}; z \in \mathrm{Bij}\mathfrak{B}$: This implies $\sigma = 0$ and that $z = v; \breve{v}; z$ is also a bijection, and, hence, $\rho = 0$. From $((v,0),(w,\gamma),(x,\delta)\rangle \in C$, and, hence, $[(\breve{x},f_2(x,\delta)),(v,0),(\breve{w},f_2(w,\gamma))] = [(\breve{w},f_2(w,\gamma)),(\breve{v},0),(\breve{x},f_2(x,\delta))] \subseteq C$, we get $f_2(w,\gamma) = f_2(x,\delta)$. In addition $((x,\delta),(y,\epsilon),(z,0)\rangle \in C$, and, hence, $[(\breve{x},f_2(x,\delta)),(z,0),(y,\epsilon)] = [(y,\epsilon),(\breve{z},0),(\breve{x},f_2(x,\delta))] \subseteq C$, implies $f_2(x,\delta) = \epsilon$. Together we obtain $f_2(w,\gamma) = f_2(x,\delta) = \epsilon$ so that $[(\breve{w},f_2(w,\gamma)),(\breve{v}; z,0),(y,\epsilon)] \subseteq C$, i.e., $((w,\gamma),(y,\epsilon),(\breve{v}; z,0)\rangle \in C$, follows.

2. The cases w or $y \in \mathrm{Bij}\mathfrak{B}$ can be shown analogously by using the elements $(w; y, f_2(w; y, f_2(y,\epsilon)))$, and $(w; y,\gamma)$, respectively.

3. $v,w,y \notin \mathrm{Bij}\mathfrak{B}$: Since $x \leq v; w$, $z \leq x; y$ and $\mathfrak{B}$ is a RA there is an element $u \in \mathrm{At}\mathfrak{B}$ with $z \leq v; u$ and $u \leq w; y$. If $u \notin \mathrm{Bij}\mathfrak{B}$, then we choose $\sigma = f_2(v,\beta)$ if $z \in \mathrm{Bij}\mathfrak{B}$ and an arbitrary $\sigma \in \alpha(u)$ otherwise. This choice gives $[(\breve{v},f_2(v,\beta)),(z,\rho),(u,\sigma)] \subseteq C$ which immediately implies

$$\langle(v,\beta),(u,\sigma)),(z,\rho)\rangle \in C \text{ and } \langle(w,\gamma),(y,\epsilon),(u,\sigma)\rangle \in C.$$

Now, suppose $u \in \mathrm{Bij}\mathfrak{B}$. If $\beta = \rho$ and $f_2(w,\gamma) = \epsilon$, then $[(v,\beta),(u,0),(z,\rho)] \subseteq C$ and $[f(w,\gamma),(u,0),(y,\epsilon)] \subseteq C$ implies the assertion. The remaining two cases are shown as follows:

(a) $\beta \neq \rho$: Then $\beta \in \alpha(v) = \alpha(v; u) = \alpha(z) \ni \rho$ because of (f) and $z = v; u$. Consequently $\alpha(\breve{z}) = \alpha(z) > 1$. In addition $\breve{x} \leq \breve{w}; \breve{v} = \breve{w}; u; \breve{u}; \breve{v} = y; \breve{z}$

shows

$$\begin{aligned}
\breve{w} &= \breve{w};u;\breve{u} \\
&= y;\breve{u} && y = \breve{w};u \text{ since } u \leq w;y \text{ and } u \text{ bijection} \\
&\leq y;(\breve{z};z\cdot 0');\breve{u} && \text{(h) since } y \notin \mathrm{Bij}\mathfrak{B},\ \alpha(\breve{z}) > 1 \text{ and } y;\breve{z} > \breve{x} \\
&= \breve{w};u;(\breve{u};\breve{v};v;u\cdot 0');\breve{u} && y = \breve{w};u, z = v;u \\
&= \breve{w};(\breve{v};v\cdot u;0';\breve{u}) && \text{Lemma 2.3.2} \\
&= \breve{w};(\breve{v};v\cdot 0') && \text{Lemma 2.3.2.}
\end{aligned}$$

This implies that there is an atom $u' \in \mathfrak{B}$ with $\breve{w} \leq \breve{w};u'$ and $u' \leq \breve{v};v\cdot 0'$ since otherwise $\breve{w} = 0$ would follow. In particular, $u' \notin \mathrm{Bij}\mathfrak{B}$ because otherwise $\breve{w};u' = \breve{w}$, and, hence, $u' = 1'$ would follow, a contradiction to $u' \leq 0'$. Therefore, $u';u$ is an atom and not a bijection. Furthermore, we have $y = \breve{w};u \leq \breve{w};u';u = \breve{w};(u';u)$ and $u';u \leq (\breve{v};v\cdot 0');u \leq \breve{v};v;u = \breve{v};z$ so that $u';u \leq w;y$ and $z \leq v;(u';u)$ follows. Consequently, we can use $u';u$ instead of the bijection u.

(b) $f_2(w,\gamma) = \epsilon$: This case is shown analogously to the previous case.

The algebra $\mathfrak{B}$ can be embedded into $\mathfrak{CmS}$ using the function $h(b) = \{(x,\beta) \mid x \in \mathrm{At}\mathfrak{B}, x \leq b, \beta \in \alpha(x)\}$ which is easy to verify. The obvious definition $c : \mathrm{At}\mathfrak{CmS} \to \mathrm{At}h(\mathfrak{B})$ by $c(x,\beta) = h(x)$, i.e., $c(x,\beta) = \{(x,\beta) : \beta \in \alpha(x)\}$, shows that $\mathfrak{CmS}$ is an extension of the image $h(\mathfrak{B})$ along η and θ. Finally, we obtain $\mathfrak{A}$ by replacing the image $h(\mathfrak{B})$ of $\mathfrak{B}$ in $\mathfrak{CmS}$ by $\mathfrak{B}$ itself. □ □

We follow-up with different cases exemplifying situations whether the splitting mechanism based on Theorem 3.0.9 is applicable or not.

Example 7. *Let $\mathfrak{A}$ be a relation algebra with $\mathrm{At}\mathfrak{A} = \{1', a, b\}$. We have $n = 3$, $s = 3$ as all the atoms are symmetric and the following diversity cycles,*

$$C(\mathfrak{A}) = [(a,a,b),(a,b,b)].$$

Atoms of $\mathfrak{A}$	1'	a		b
Atoms of $\mathfrak{B}$	1'	a1	a2	b

Table 3.3: $\mathrm{At}\mathfrak{B}$ vs $\mathrm{At}\mathfrak{A}$

The relation algebra above is the so-called pentagonal relation algebra [32]. By the application of splitting algorithm induced by Theorem 3.0.9 on Atom a, we obtain the following new structure $\mathfrak{B}$ with $n = 4$, $s = 4$ with correlation Table 3.3 between the atoms in $\mathfrak{B}$ and $\mathfrak{A}$ and its full composition Table 3.4.

;	a1	a2	b
a1	b	b	a1,a2,b
a2	b	b	a1,a2,b
b	a1,a2,b	a2,b	a1,a2

Table 3.4: Composition table for $\mathfrak{B}$

This structure is not a relation algebra since $(\{a1\};\{a2\});\{b\} = \{b\};\{b\} = \{a1, a2\} \neq \{a1, a2, b\} = \{a1\};\{a1, a2, b\} = \{a1\};(\{a2\};\{b\})$. *In fact, Property (h) of Theorem 3.0.9 is violated since*

$$a;(b;b \cdot (a+b)) = a;((1'+a) \cdot (a+b)) = a;a = 1'+b \not\geq a.$$

We did some further investigation on this very interesting example. Actually there is no integral extension of the relation algebra $\mathfrak{A}$ along any possible pair of functions so that Atom a or Atom b (or both) splits into two atoms. However, the algebra is representable on a set of five elements. All three atoms are represented by non-atomic relations which shows that all atoms can be split resulting in a simple (but not integral) relation algebra. Notice that in [25] it was shown that there are even algebras that cannot be properly embedded in any simple relation algebra.

The next example illustrates a case where extensional splitting mechanism from Theorem 3.0.9 succeeds while the regular method using Theorem 3.0.8 is futile.

Example 8. *Let $\mathfrak{A}$ be a* RCC-11 *algebra with* $\mathrm{At}\mathfrak{A} = \{1', DC, ECN, ECD, PON, PODY, PODZ, TPP, TPP^{\smile}, NTPP, NTPP^{\smile}\}$. *We have $n = 11$ and $s = 7$ since 1', DC, ECN, ECD, PON, PODY, PODZ are symmetric and rest are non-symmetric atoms respectively. Suppose C is a list of diversity cycles, then* RCC-11 *composition table from Table 2.4 condenses to a cycleset C given by,*

$$\begin{aligned}\mathrm{C}(\mathfrak{A}) = [\ &(TPP,TPP,TPP),(TPP,TPP,NTPP),(TPP,TPP^{\smile},DC),\\ &(TPP,TPP^{\smile},PON),(TPP,TPP^{\smile},ECN),(TPP,NTPP,NTPP),\\ &(TPP,NTPP^{\smile},NTPP^{\smile}),(TPP,NTPP^{\smile},PON),(TPP,NTPP^{\smile},DC),\\ &(TPP,NTPP^{\smile},ECN),(TPP,PON,TPP),(TPP,PON,NTPP),(TPP,PON,PON),\end{aligned}$$

(TPP,PON,ECN),(TPP,PON,DC),(TPP,PODY,TPP),(TPP,PODY,PON),
(TPP,PODY,NTPP),(TPP,PODY,PODY),(TPP,PODY,ECN),
(TPP,PODY,ECD),(TPP,PODZ,TPP),(TPP,PODZ,NTPP),
(TPP,PODZ,PON),(TPP,PODZ,PODY),(TPP,PODZ,PODZ),
(TPP,ECN,ECN),(TPP,ECN,DC),(TPP,ECD,ECN),
(TPP,DC,DC),(NTPP,NTPP,NTPP),(NTPP,NTPP$^{\smile}$,PON),
(NTPP,NTPP$^{\smile}$,ECN),(NTPP,NTPP$^{\smile}$,DC),(NTPP,PON,NTPP),
(NTPP,PON,PON),(NTPP,PON,ECN),(NTPP,PON,DC),
(NTPP,PODY,NTPP),(NTPP,PODY,PON),(NTPP,PODY,ECN),
(NTPP,PODY,DC),(NTPP,PODZ,NTPP),(NTPP,PODZ,PON),
(NTPP,PODZ,PODY),(NTPP,PODZ,PODZ),(NTPP,PODZ,ECN),
(NTPP,PODZ,ECD),(NTPP,PODZ,DC),(NTPP,ECN,DC),
(NTPP,ECD,DC),(NTPP,DC,DC),(PON,PON,PON),(PON,PON,PODY),
(PON,PON,PODZ),(PON,PON,DC),(PON,PON,ECN),(PON,PON,ECD),
(PON,PODY,PODY),(PON,PODY,PODZ),(PON,PODZ,PODZ),
(PON,ECN,ECN),(PON,ECN,DC),(PON,DC,DC),(PODY,PODY,PODY),
(PODY,PODY,PODZ),(PODY,PODZ,PODZ),(PODZ,PODZ,PODZ),
(ECN,ECN,ECN),(ECN,ECN,DC),(ECN,DC,DC),(DC,DC,DC),
]

Atoms of $\mathfrak{A}$	**Atoms of $\mathfrak{B}$**
DC	DC
ECN	ECNA ECNB
ECD	ECD
PON	PON
PODY	PODYA PODYB
PODZ	PODZ
TPP	TPPA TPPB
TPP$^{\smile}$	TPPA$^{\smile}$ TPPB$^{\smile}$
NTPP	NTPP
NTPP$^{\smile}$	NTPP$^{\smile}$

Table 3.5: At$\mathfrak{B}$ vs At$\mathfrak{A}$

As mentioned before, apart from the ubiquitous identity relation, RCC-11 algebra

contains an additional bijection -the relation ECD. None of the atoms of this particular algebra can be split using the mechanism based on Theorem 3.0.8. However Theorem 3.0.9 can be aptly applied to RCC-11 and we can split the relation TPP into two new atoms called TPPA and TPPB. As TPP is a non-symmetric atom, we must split its converse too i.e., $TPP^{\smile}$ into two new relations namely $TPPA^{\smile}$ and $TPPB^{\smile}$. Also, given that the algebra contains the bijection atom ECD, we need to split atoms ECN and PODY as well, each into two new relations ECNA, ECNB and PODYA, PODYB respectively (see properties (b) & (f) - Theorem 3.0.9). This results in an algebra RCC-15 denoted by $\mathfrak{B}$ with $n = 15$, $s = 9$. The correlation between the old and new atoms is as shown in Table 3.5.

The diversity cycles of the new algebra $\mathfrak{B}$ are as follows:

$C(\mathfrak{B})$ = [(TPPA,TPPA,TPPA),(TPPA,TPPA,TPPB),(TPPA,TPPB,TPPA),
(TPPA,TPPB,TPPB),(TPPB,TPPA,TPPA),(TPPB,TPPA,TPPB),
(TPPB,TPPB,TPPA),(TPPB,TPPB,TPPB),(TPPA,TPPA,NTPP),
(TPPA,TPPB,NTPP),(TPPB,TPPA,NTPP),(TPPB,TPPB,NTPP),
(TPPA,$TPPA^{\smile}$,DC),(TPPA,$TPPB^{\smile}$,DC),(TPPB,$TPPB^{\smile}$,DC),
(TPPA,$TPPA^{\smile}$,PON),(TPPA,$TPPB^{\smile}$,PON),(TPPB,$TPPB^{\smile}$,PON),
(TPPA,$TPPA^{\smile}$,ECNA),(TPPA,$TPPA^{\smile}$,ECNB),(TPPA,$TPPB^{\smile}$,ECNA),
(TPPA,$TPPB^{\smile}$,ECNB),(TPPB,$TPPB^{\smile}$,ECNA),(TPPB,$TPPB^{\smile}$,ECNB),
(TPPA,NTPP,NTPP),(TPPB,NTPP,NTPP),(TPPA,$NTPP^{\smile}$,$NTPP^{\smile}$),
(TPPB,$NTPP^{\smile}$,$NTPP^{\smile}$),(TPPA,$NTPP^{\smile}$,PON),(TPPB,$NTPP^{\smile}$,PON),
(TPPA,$NTPP^{\smile}$,DC),(TPPB,$NTPP^{\smile}$,DC),(TPPA,$NTPP^{\smile}$,ECNA),
(TPPA,$NTPP^{\smile}$,ECNB),(TPPB,$NTPP^{\smile}$,ECNA),(TPPB,$NTPP^{\smile}$,ECNB),
(TPPA,PON,TPPA),(TPPA,PON,TPPB),(TPPB,PON,TPPB),
(TPPA,PON,NTPP),(TPPB,PON,NTPP),(TPPA,PON,PON),
(TPPB,PON,PON),(TPPA,PON,ECNA),(TPPA,PON,ECNB),
(TPPB,PON,ECNA),(TPPB,PON,ECNB),(TPPA,PON,DC),
(TPPB,PON,DC),(TPPA,PODYA,TPPA),(TPPA,PODYA,TPPB),
(TPPA,PODYB,TPPA),(TPPA,PODYB,TPPB),(TPPB,PODYA,TPPB),
(TPPB,PODYB,TPPB),(TPPA,PODYA,PON),(TPPA,PODYB,PON),
(TPPB,PODYA,PON),(TPPB,PODYB,PON),(TPPA,PODYA,NTPP),
(TPPA,PODYB,NTPP),(TPPB,PODYA,NTPP),(TPPB,PODYB,NTPP),
(TPPA,PODYA,PODYA),(TPPA,PODYA,PODYB),(TPPA,PODYB,PODYA),
(TPPA,PODYB,PODYB),(TPPB,PODYA,PODYA),(TPPB,PODYA,PODYB),
(TPPB,PODYB,PODYA),(TPPB,PODYB,PODYB),(TPPA,PODYA,ECNA),
(TPPA,PODYA,ECNB),(TPPA,PODYB,ECNA),(TPPA,PODYB,ECNB),

(TPPB,PODYA,ECNA),(TPPB,PODYA,ECNB),(TPPB,PODYB,ECNA),
(TPPB,PODYB,ECNB),(TPPA,PODZ,TPPA),(TPPA,PODZ,TPPB),
(TPPB,PODZ,TPPB),(TPPA,PODZ,NTPP),(TPPB,PODZ,NTPP),
(TPPA,PODZ,PON),(TPPB,PODZ,PON),(TPPA,PODZ,PODYA),
(TPPA,PODZ,PODYB),(TPPB,PODZ,PODYA),(TPPB,PODZ,PODYB),
(TPPA,PODZ,PODZ),(TPPB,PODZ,PODZ),(TPPA,ECNA,ECNA),
(TPPA,ECNA,ECNB),(TPPA,ECNB,ECNA),(TPPA,ECNB,ECNB),
(TPPB,ECNA,ECNA),(TPPB,ECNA,ECNB),(TPPB,ECNB,ECNA),
(TPPB,ECNB,ECNB),(TPPA,ECNA,DC),(TPPA,ECNB,DC),
(TPPB,ECNA,DC),(TPPB,ECNB,DC),(TPPA,DC,DC),(TPPB,DC,DC),
(NTPP,NTPP,NTPP),(NTPP,NTPP$^{\smile}$,PON),(NTPP,NTPP$^{\smile}$,ECNA),
(NTPP,NTPP$^{\smile}$,ECNB),(NTPP,NTPP$^{\smile}$,DC),(NTPP,PON,NTPP),
(NTPP,PON,PON),(NTPP,PON,ECNA),(NTPP,PON,ECNB),
(NTPP,PON,DC),(NTPP,PODYA,NTPP),(NTPP,PODYB,NTPP),
(NTPP,PODYA,PON),(NTPP,PODYB,PON),(NTPP,PODYA,ECNA),
(NTPP,PODYA,ECNB),(NTPP,PODYB,ECNA),(NTPP,PODYB,ECNB),
(NTPP,PODYA,DC),(NTPP,PODYB,DC),(NTPP,PODZ,NTPP),
(NTPP,PODZ,PON),(NTPP,PODZ,PODYA),(NTPP,PODZ,PODYB),
(NTPP,PODZ,PODZ),(NTPP,PODZ,ECNA),(NTPP,PODZ,ECNB),
(NTPP,PODZ,DC),(NTPP,ECNA,DC),(NTPP,ECNB,DC),
(NTPP,DC,DC),(PON,PON,PON),(PON,PON,PODYA),
(PON,PON,PODYB),(PON,PON,PODZ),(PON,PON,DC),
(PON,PON,ECNA),(PON,PON,ECNB),(PON,PODYA,PODYA),
(PON,PODYA,PODYB),(PON,PODYB,PODYB),(PON,PODYA,PODZ),
(PON,PODYB,PODZ),(PON,PODZ,PODZ),(PON,ECNA,ECNA),
(PON,ECNA,ECNB),(PON,ECNB,ECNB),(PON,ECNA,DC),
(PON,ECNB,DC),(PON,DC,DC),(PODYA,PODYA,PODYA),
(PODYA,PODYA,PODYB),(PODYA,PODYB,PODYB),
(PODYB,PODYB,PODYB),(PODYA,PODYA,PODZ),
(PODYA,PODYB,PODZ),(PODYB,PODYB,PODZ),(PODYA,PODZ,PODZ),
(PODYB,PODZ,PODZ),(PODZ,PODZ,PODZ),(ECNA,ECNA,ECNA),
(ECNA,ECNA,ECNB),(ECNA,ECNB,ECNB),(ECNB,ECNB,ECNB),
(ECNA,ECNA,DC),(ECNA,ECNB,DC),(ECNB,ECNB,DC),(ECNA,DC,DC),
(ECNB,DC,DC),(DC,DC,DC),(TPPA$^{\smile}$,ECD,PODYA),
(TPPA$^{\smile}$,ECD,PODYB),(TPPB$^{\smile}$,ECD,PODYA),(TPPB$^{\smile}$,ECD,PODYB),
(TPPA,ECD,ECNA),(TPPA,ECD,ECNB),(TPPB,ECD,ECNA),

(TPPB,ECD,ECNB),(NTPP$^{\smile}$,ECD,PODZ),(NTPP,ECD,DC),
(PON,ECD,PON)]

The splitting (and some further splittings) underlined in Example 8 was already accomplished in [15]. The distinction being that in the aforementioned paper, the resulting algebra was computed manually whereas the result in Example 8 was arrived using a Haskell based system implementing the approach of Theorem 3.0.9. Further details on the system functioning will be discussed in Chapter 4.

Refinement of Algebras

In the previous section we discussed ways of formulating extensional relation algebras. The deduction of cycles from such algebras forms another important process in generating new algebras from old ones. This is usually termed as the *refinement of algebras.* Such a refinement is achieved in a two-step manner. The first step involves procurement of further extensions of an algebra via Theorem 3.0.9 application followed by the removal of specific cycles. If Theorem 3.0.9 is in effect the net result of a maximal extension, all possible extensions can be obtained using this process and similar approaches. Incidentally this process plays a predominant role in applications of mereotopological domain. The refinement process can be explained using Example 9 and Example 10 as follows.

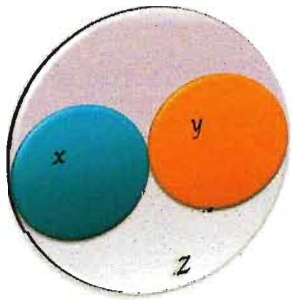


Figure 3.1: $ECN \circ TPP \cap TPP \neq \varnothing$

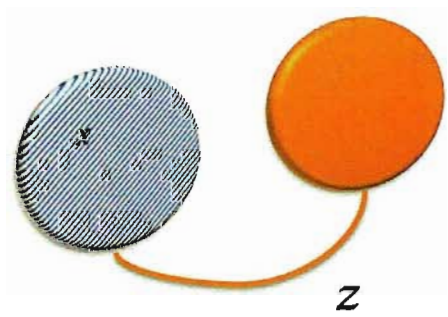


Figure 3.2: $\overline{ECN \circ TPP} \cap TPP \neq \varnothing$

Example 9. *Referring the composition Table 2.4 of RCC-11 consider the composition of relations ECN and TPP. We have weak compositional property* $ECN ; TPP \cap TPP \neq \varnothing$ *i.e., whenever one region is inside another region but their borders intersect, then there is a region that is externally connected to the first and itself a*

tangential proper part of the second. This is depicted in Figure 3.1 whereby $xTPPz$ implies the existence of a y with $xECNy$ and $yTPPz$. But this condition may not always hold true as depicted in Figure 3.2 where $xTPPz$ does not necessarily imply the existence of such a y. As such we construct examples as well as counterexamples for this particular statement by splitting atom TPP in RCC-11. This is the starting point for splitting TPP in RCC-11 (see Example 8 about splitting TPP in RCC-11). This process not only involves splitting TPP but also warrants removal of the triple (ECN, TPP, TPP) for one of the two copies of TPP in the new algebra RCC-15. In addition such a removal requires discarding all related triples from the newly split algebra. These related triples are the product of relative multiplication by isomorphism of a given algebra. For example in case of cycle (ECN,TPP,TPP), copies of related triples such as $(TPP, TPP^{\smile}, ECN)$, $(TPP^{\smile}, ECN, TPP^{\smile})$, (ECN, ECN, ECN), $(TPP^{\smile}, TPP, PODY)$, $(TPP, PODY, TPP)$, $(PODY, TPP^{\smile}, TPP^{\smile})$, $(PODY, PODY, PODY)$ must be discarded from the new algebra RCC-15. These related triples were obtained multiplying (ECN, TPP, TPP) by isomorphisms $[ID, ECD]$ in RCC-11 algebra. Furthermore when we remove a triple from a cycle, we may also have to remove other triples in a step by step manner to satisfy the associative property (see Theorem 3.0.6 (c)). This removal process continues until the associative property is satisfied and we get a valid integral relation algebra.

Example 10. *Let $\mathfrak{A}$ be a three atom algebra with $\mathrm{At}\mathfrak{A} = \{1', a, b\}$. We have $n = 3$ and $s = 3$ since all atoms are symmetric in $\mathfrak{A}$ and has the following diversity cycles,*

$$C(\mathfrak{A}) = [(a, a, b), (a, b, b)].$$

Atoms of $\mathfrak{A}$	1'	a		b
Atoms of $\mathfrak{B}$	1'	a1	a2	b

Table 3.6: At$\mathfrak{B}$ vs At$\mathfrak{A}$

The composition table for $\mathrm{At}\mathfrak{A}$ is given by Table 3.7. By the application of splitting algorithm induced by Theorem 3.0.9 on Atom a, we obtain the following new structure $\mathfrak{B}$ with $n = 4$, $s = 4$ with correlation Table 3.6 between the atoms in $\mathfrak{B}$ and $\mathfrak{A}$ and its full composition Table 3.8.

Splitting Atom a in the algebra $\mathfrak{A}$ gives rise to atoms $\{ a1, a2 \}$ in the newer algebra $\mathfrak{B}$. Suppose our motivation is to remove the cycle $(a2, b, b)$ from $\mathfrak{B}$ since its clone $(a1, b, b)$ is already existing in the newer algebra. The resulting algebra after this deduction process has the diversity cycles: $[(a1, a1, a1), (a1, a1, a2), (a1, a2,$

;	a	b
a	a	b
b	b	a

Table 3.7: Composition table for $\mathfrak{A}$

Atom $a \xrightarrow{split} \{a1, a2\}$

;	a1	a2	b
a1	a1, a2	a1, a2	b
a2	a1, a2	a1, a2	b
b	b	b	a1, a2

Table 3.8: Composition table for $\mathfrak{B}$

$a2)$, $(a2, a2, a2)$, $(a1, b, b)]$ *which are not associative and therefore not a relation algebra. We deploy the refinement process of cycle elimination until the property of associativity is satisfied i.e., in the above case the resultant diversity cycleset is,* $[(a1, a1, a1), (a2, a2, a2), (a1, b, b)]$. *This was deduced by eliminating the cycles* $(a1, a1, a2)$ *and* $(a1, a2, a2)$ *as illustrated in Tables 3.9 to 3.11.*

;	a1	a2	b
a1	a1, a2	a1, a2	b
a2	a1, a2	a1, a2	***b***
b	b	***b***	a1, ***a2***

Table 3.9: Refinement step 1 in $\mathfrak{B}$

deduct cycle $(a2, b, b)$

;	a1	a2	b
a1	a1, ***a2***	***a1, a2***	b
a2	***a1, a2***	***a1***, a2	
b	b		a1

Table 3.10: Refinement step 2 in $\mathfrak{B}$

deduct cycles $(a1, a1, a2), (a1, a2, a2)$

;	a1	a2	b
a1	a1		b
a2		a2	
b	b		a1

Table 3.11: Final composition table of $\mathfrak{B}$

Figure. The transitions of refinement process in algebra $\mathfrak{B}$

The previous two examples consolidates the concept of refinement in relation algebras and thereby concludes this chapter.

Chapter 4

The System

This chapter deals with implementation aspect of the concepts and techniques discussed in the previous chapter. It will be accomplished with the aid of functional structures.

4.1 Overview

The general design features of the system can be broadly outlined as follows,

I The algebras will be computed with a total of n atoms of which s atoms are symmetric. The data pertaining to such algebras will be organized and stored in a pre-formatted data structure. Each element of this data structure will hold all the information such as name of the algebra, n, s, atom names and its diversity cycles.

II The widgets used in the design of the user-interface are specifically selected to support the necessary actions of manipulating the stored algebras. In addition the core functionality and the user-interface are fully decoupled to ensure testability and maintainability of the system.

III Parsers are designed to implicitly convert the data in storable format and also retrieve them for display purposes and data manipulation. Storage of data is possible for individual algebras or the entire list of algebras.

The system is developed using Haskell, a functional programming language. Haskell is based on lambda calculus, a mathematical theory of functions, and not on the Turing machine like imperative programming languages do. Shortly described, Haskell is a strongly typed, lazy, pure functional programming language. Haskell has a static

type system so that all type errors are detected at compile time. This makes Haskell programs very type safe. Its type class system is complex but powerful. Haskell's evaluation model is called lazy or non-strict, because arguments of functions are only evaluated if they are needed for computations. This leads to demand-driven evaluation. As such expressions are minimally evaluated to get the impending result so parts of them may not be evaluated at all. In contrast to imperative languages, program order is not needed in Haskell. A Haskell program can be viewed as a collection of module components and each module defines its own set of values, data types, type synonyms, classes, etc. The implementation of this system constitutes a collection of such modules. Figure 4.1 portrays dependencies among various modules of the system. GTK+ toolkit along with Haskell library Gtk2Hs has been adopted to design user interface for this system. The knowledge of Haskell language and GTK+ libraries are key requisites for good understanding of this chapter and therefore it is imperative that the reader be familiar with them.

In the following sections we will be discussing only research relevant functions of the modules RAGenerator, RAParser, SplitWindow, CropCycle, SplitnCrop, CompTable and FileInterface. The rest of the functions including those in the module AppMethods are helper functions which are self explanatory. We will only mention their type signature wherever necessary. Also the functions of module TreeController is just an extension of the standard GTK+ TreeView library incorporating additional graphical features such as colored columns and checkbox widgets. It will not be discussed in this chapter as it out of scope of our research.

4.2 RAGenerator

Recall from Chapter 3 that every atomic relation algebra induces a particular structure on its atoms, its `Atom Structure`. In particular, composition is represented by a set of triples of atoms, i.e., by a ternary relation. Conversely, given an atom structure one may form the complex algebra thereof. This module defines the atom structure for an algebra and packages its core set of operations . Some of these operations and the atom structure are exported for the use of other modules. The module treats atoms as integers ranging from 1 to N, where 1 is the identity. The atoms 1 ... S are symmetric, the rest of the atoms S + 1 ... N being non-symmetric. For more details regarding the conventions used here we refer to Section 2 of Chapter 3.

The data types are defined as follows:

```
type Ternary = (Int,Int,Int)
```

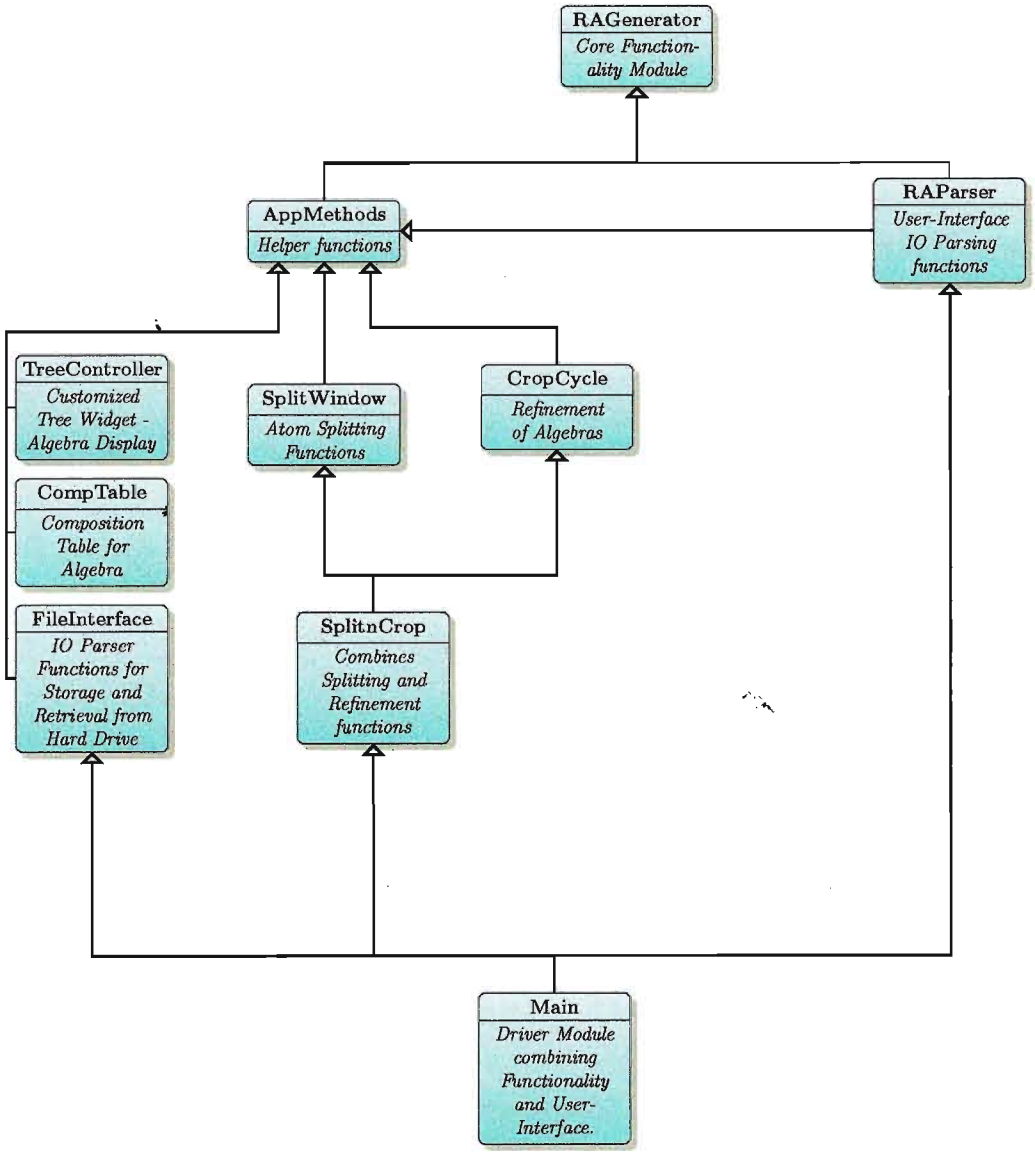


Figure 4.1: Module Dependencies of the System.

```
type Cycle = [ Ternary ]

data RelAlg = RelAlg { name :: String, n :: Int,

    s :: Int, absTable :: Cycle, atomNames

    :: [String] } deriving (Eq,Show)
```

`Ternary` is a triple of `Int` since we use integer numbers as a representation of atoms. In Chapter 3 we define the cycle which is actually an array of such triples, therefore we use the list of `Ternary` to represent such a `Cycle`. The atom structure denoted by `RelAlg` may be better explained by means of Haskell-pseudocode using a record constructor as follows,

Definition 21. *The structure* `RelAlg(name,n,s,absTable,atomNames)` *is an atomic relational structure where*

- `n` *is the number of atoms,*
- `s` *is the number of symmetric atoms, i.e.* $\mathtt{s} \leq \mathtt{n}$,
- `absTable` *is the ternary relation for composition given by cycles,*
- `atomNames` *is list of names of the atoms in the given order.*

The whole concept can be illustrated by taking RCC-11 algebra as an example. Its atomic structure will be displayed by the application as follows: The left side of Figure 4.2 represents all the respective attributes of a RelAlg structure for RCC-11, i.e., `RelAlg(Name,N,S,Cycles,Atoms)` with `Name` = RCC11, `N`=11, `S`=7, `Cycles` represent all the diversity cycles and `Atoms` indicate list of names of the atoms for RCC-11.

Given a converse operation `f` and the type `Cycle`, the cycle structure of Equation 3.1 from Chapter 3 can be defined as,

```
cycleSet :: Eq a => (a->a) -> (a,a,a) -> [(a,a,a)]

cycleSet f (x,y,z) = nub [(x,y,z),(f x,z,y),(y,f z,f x),

                          (f y,f x,f z),(f z,x,f y),(z,f y,x)]
```

Observe that the control panel layout of Figure 4.2 has button widgets `IsRA` and `IsIntegral` which determine whether the given atom structure leads to a relation algebra and if it is integral respectively. To determine this we make use of the two properties mentioned in Chapter 3 as follows.

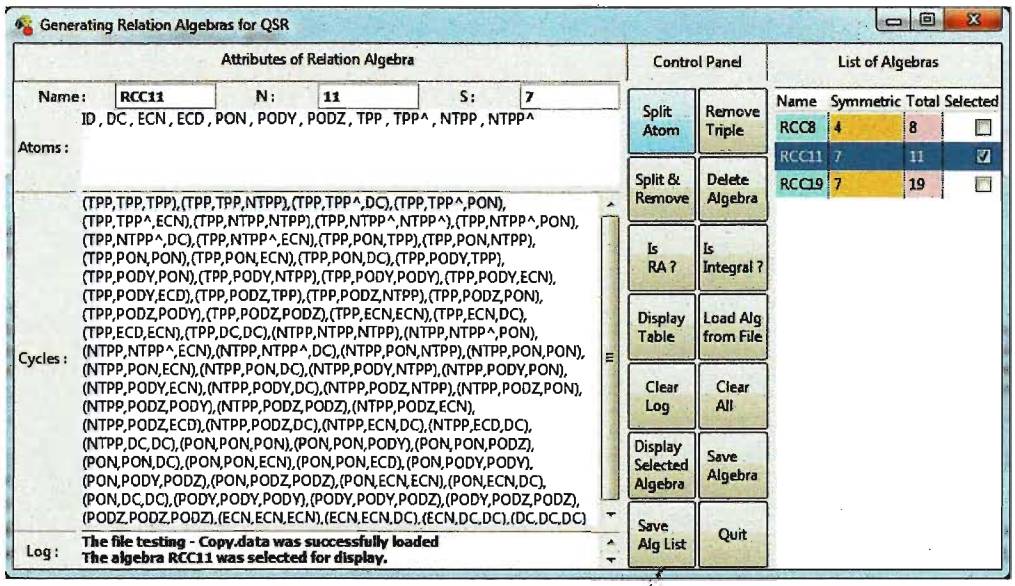


Figure 4.2: Structure for RCC-11

```
isRA ::  (Num a, Enum a) ⇒ (a->a) -> a -> [(a,a,a)] -> Bool

isRA f n na = (isIntegral f n na) && ( and [ all (existU f

                n na ) (match f t1 t2) | t1 <- na, t2 <- na ]
```

The `existU` call in the above function implements the property (c) of Theorem 3.0.6.

```
existU ::  (Num a, Enum a) ⇒ (a->a) -> a -> [(a,a,a)]

          -> (a,a,a,a,a) -> Bool

existU f n c (v,w,x,y,z) = any ( \u -> (isIn (w,y,u) c)

                          && (isIn (v,u,z) c)) [2..n]

                where isIn = cycElem f
```

The `IsIntegral` button widget validates the property from Definition 12 of Chapter 2.

```
isIntegral ::  (Num a, Enum a) ⇒ (a->a) -> a -> [(a,a,a)] -> Bool

isIntegral f n c = and [ any ( \w -> cycElem f (x,w,y) c)

                    [2..n] | x <- [2..n], y <- [2..n], x/=y]
```

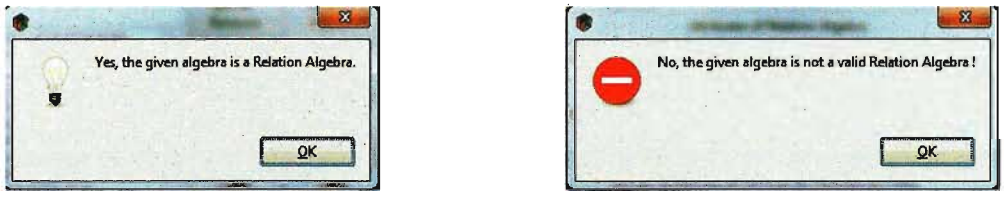


Figure 4.3: Typical response from IsRA button action.

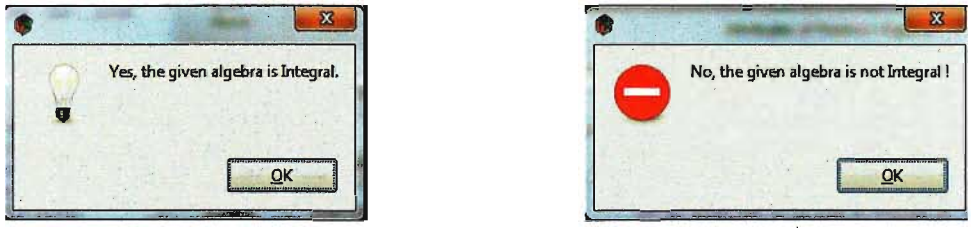


Figure 4.4: Typical response from IsIntegral button action.

The corresponding outputs for `IsRA` and `IsIntegral` button actions are depicted by Figures 4.3 and 4.4 respectively.

The `getBijections` function retrieves all bijective atoms except the identity for a given relation algebra. Bijections follow the properties mentioned in Lemma 2.3.2 of Chapter 2.

```
getBijections ::  RelAlg -> [Int]

getBijections RelAlg { n, s, absTable } = [ x | x <- [2..n],

                    all ( \y -> not $ cycElem (converse n s)

                      (x,y,x) absTable) [2..n]]
```

The `compTable` function generates individual cells of the composition table for a given relation algebra. For more details we refer to Definition 14 of Chapter 2.

```
compTable ::  RelAlg -> Int -> Int -> [Int]

compTable ra@RelAlg { n } x y = [c | (a,b,c) <-

                          (genAllCycles ra),x==a && y==b]
```

Given an atom structure `RelAlg` and the atom to be split, the `canAtomSplit` function checks the property (h) of Theorem 3.0.9 from Chapter 3 and outputs a corresponding Boolean value.

```
canAtomSplit ::  RelAlg -> Int -> Bool
```

```
canAtomSplit ra@RelAlg { n,absTable } atom =
        let f = converseRA ra
            y = [2..n]
            tableTriples = getAllCycles f absTable
            compVal first second = [c | (a,b,c)
              <- tableTriples,a==first&&b==second]
            cmpX [] _ = True
            cmpX (y':ys') xs = elem y' (concat [(compVal y'
              x')|x'<-xs]) && cmpX ys' xs
        in cmpX y (compVal atom (f atom))
```

The function relatedTriples follows up with one of the steps in the refinement process of relation algebras as mentioned in last section of Chapter 3. Suppose the intention is to remove the triple (R, S, T) from an algebra then it also warrants the elimination of triples (f ; R, S ; g, f ; T ; g), (f ; R ; f, $f^{\smile}$; S ; g, f ; T ; g), (f ; R ; $f^{\smile}$, f ; S ; g, f ; T ; g), (f ; R ; g, $g^{\smile}$; S ; g, f ; T ; g), (f ; R ; $g^{\smile}$, g ; S ; g, f ; T ; g) for every pair of isomorphisms inclusive of the identity relation (id). For example we consider the pairs (id,id), (id,h), (h,id), (id,k), (k,id), (h,h), (h,k),(k,h),(k,k) where (id, h, k) are isomorphisms in the given algebra.

```
relatedTriples ::  Ternary -> RelAlg -> Cycle
relatedTriples (r,s,t) ra = let bj = getBijections ra
    isoPairs = [(x,y)|x<-(1:bj),y<-(1:bj)]
    conv = converseRA ra
    makeTriples f g = let bcomp x y = head $ compTable ra x y
                          f_r param = bcomp (bcomp f r) param
                          s_g param = (bcomp param (bcomp s g))
                          f_t_g = (bcomp (bcomp f t) g)
```

```
                in nub [((f_r 1),(s_g 1),f_t_g),((f_r f),
                    (s_g (conv f)),f_t_g),
                    ((f_r (conv f)),(s_g f),f_t_g),
                    ((f_r g),(s_g (conv g)),f_t_g),
                    ((f_r (conv g)),(s_g g), f_t_g)]
            in let allTriples = concatMap
                    ( \(x',y')-> makeTriples x' y') isoPairs
              in nub $ getAllCycles conv allTriples
```

The function `triples2Remove` also contributes to the refinement process of relation algebras. It generates a list of triples that have to be removed in the newly split algebra. It takes the following input parameters: `newAtomList` - contains the mapping of the old list of atoms to the newly created atom list, `RelAlg` - atom structure of the old relation algebra, a triple (r', s', t') - where r' ; s' represents the spatial condition and t' is the atom to be split using the spatial condition. Recall from Example 9 of Chapter 3, suppose our motivation is to split the relation TPP of RCC -11 using the spatial relation ECN ; TPP. In this case we have the input triple as (ECN, TPP, TPP) and since TPP is non symmetric relation we have to remove its converse as well i.e., (ECN,TPP,TPP)$^{\smile}$ $\Rightarrow$ ($TPP^{\smile}$, $ECN^{\smile}$, $TPP^{\smile}$) $\Rightarrow$ ($TPP^{\smile}$, ECN, $TPP^{\smile}$) needs to be removed. The `newAtomList` will be generated by applying splitting mechanism of Theorem 3.0.9 in Chapter 3 for atom TPP in RCC -11. Also the process needs to be repeated for a combination of triples in the new algebra RCC -15 i.e., ($ECNA$, $TPPA$, $TPPB$),($ECNA$, $TPPB$, $TPPB$), ($ECNB$, $TPPA$, $TPPB$) and ($ECNB$, $TPPB$, $TPPB$). Notice that only the greater atom TPPB of the split pair ($TPPA$,$TPPB$) is being removed in the given triples. This is done in conjunction with the sub-function `largerAtom` of `triples2Remove`. For the mentioned input parameters the function `triples2Remove` generates a list of triples as given in Example 9 of Chapter 3 that needs to be eliminated from RCC -15.

```
triples2Remove ::  [[Int]] -> RelAlg -> Ternary -> Cycle

triples2Remove newAtomList oldRa@RelAlg{s} (r',s',t') =

    let relTriples = relatedTriples (r',s',t') oldRa
```

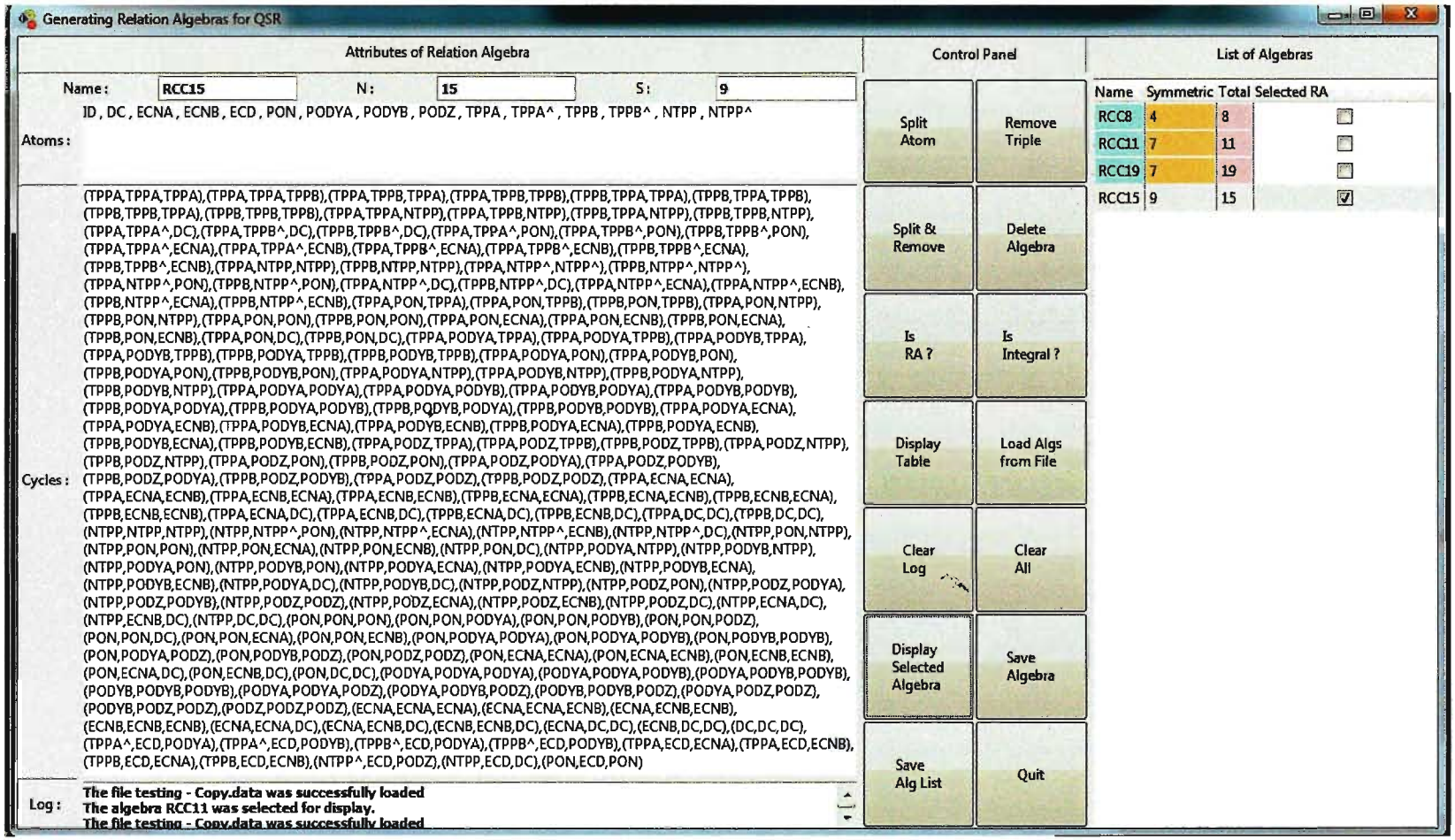
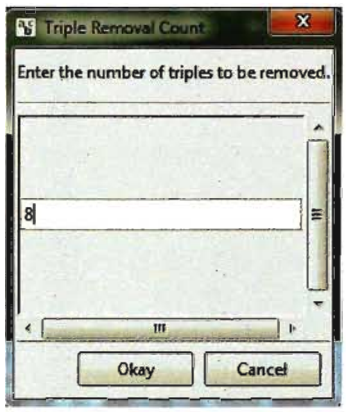


Figure 4.5: Atom Structure for RCC-15

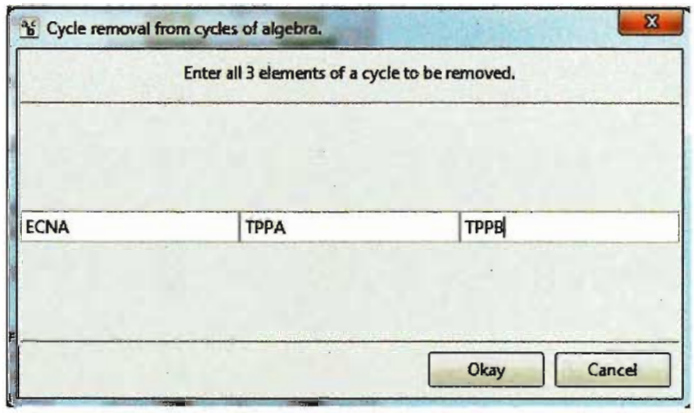
```
        newList oldAtom = (newAtomList !!  (oldAtom-1))
        largerAtom index = last $ newList index
        conv = converseRA oldRa
        isSymm atom = if atom > s then False else True
        combinations (x,y,z) = [(x',y',(largerAtom z))
          |x'<-(newList x),y'<-(newList y)]
        combinations' (x,y,z) = if isSymm z then []
          else combinations ((conv y),(conv x),(conv z))
    in concatMap ( \t-> (combinations t) ++
        (combinations' t)) relTriples
```

4.3 CropCycle

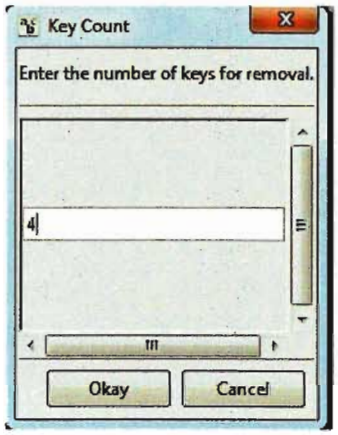
This module provides a graphical interface to enact the refinement process of relation algebras as mentioned in last section of Chapter 3. The module functions will be invoked when the user activates the button "Remove Triple" on the control panel. In particular this module defines the following user interface functions: input the number of triples to be removed including the triples themselves for a given algebra, number of key atoms to be considered for refinement and its corresponding key list for inputting the refinable atoms and a name for the newly refined algebra. Furthermore when we remove the triple from a cycle, we may also have to remove other triples in a step by step manner to satisfy the associative property (c) of Theorem 3.0.6. If there are multiple candidates for removal, the user will be prompted to choose an appropriate triple for removal as evident from Figure 4.6(e). This process continues until it reaches a point where the associative property is satisfied and we get a valid integral relation algebra. These series of steps for a RCC -15 algebra (see Figure 4.5) can be visualized using the snapshots as depicted in Figure 4.6. In effect it carries out tasks that are similar to the description of function `triples2Remove`, the difference being that the latter function is used for automating the entire operation of related triple removal in `SplitnRemove` module.



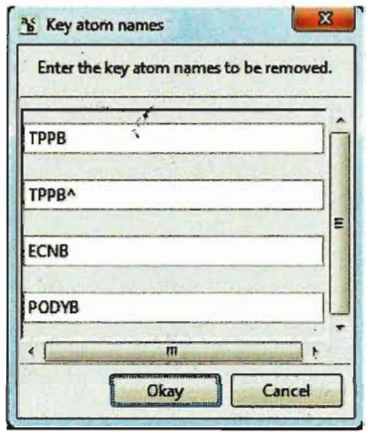
(a) Number of triples.



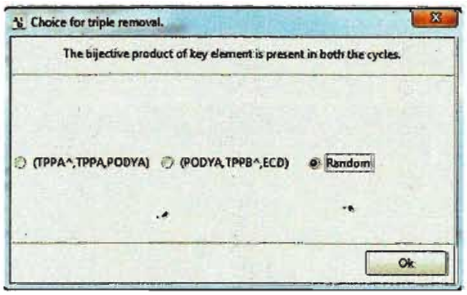
(b) Enter triple names for removal.



(c) Number of key atoms.

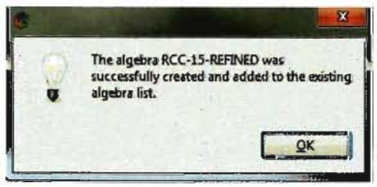


(d) Enter the names for key atoms.



(e) Choice of triple removal to satisfy associativity.

(f) Prompt for name of refined algebra.



(g) Newly created algebra added to the list.

Figure 4.6: In progress - Various stages of the Remove Triple operation.

4.4 SplitWindow

This module provides a graphical interface to implement the splitting mechanism of Theorem 3.0.9 from Chapter 2. The module functions will be instigated when the user activates the button "Split Atom" on the control panel. The conditions (a)-(f) as well as the secondary condition in (h) of Theorem 3.0.9 are pre-determined before the actual splitting action is initiated. Thereafter the condition (g) and part of the condition (h) are evaluated for the data entered by the user. This is done by clicking the button "Split" in the split window as shown in Figure no. 4.7.

For example we may split the atom TPP into two new atoms $TPPA$ and $TPPB$ using RCC-11 example from Figure 4.2 by setting its corresponding η value to 2. We obtain a new algebra `RelAlg(Name,N,S,Cycles,Atoms)` with `Name` = RCC15, `N`=15, `S`=9, `Cycles` representing diversity cycles and `Atoms` indicating list of names of the atoms for RCC-15 as shown in Figure 4.5. Notice that the atoms ECN, $PODY$ and $TPP^{\smile}$ also have to be split into 2 atoms following Theorem 3.0.9.

4.5 SplitnRemove

This module provides a graphical interface to implement the splitting mechanism of Theorem 3.0.9 from Chapter 2 followed by the refinement steps mentioned in Chapter 3. In effect this module imports functionalities from the previous two modules and performs a unified operation on given algebra (see module dependency Figure 4.1). The module functions will be instigated when the user activates the button "SplitnRemove" on the control panel.

Specifically this module provides the following user interface functions: input the atom to be split including the spatial relation used for its splitting; as a triple, a name for the newly created algebra and the names for newly formed atoms due to splitting. Furthermore when we remove the triple from a cycle, we may also have to remove other triples in a step by step manner to satisfy the associative property (c) of Theorem 3.0.6. If there are multiple candidates for removal, the user will be prompted to choose an appropriate triple for removal as evident from Figure 4.6(e). This process continues until it reaches a point where the associative property is satisfied and we get a valid integral relation algebra. These series of steps for a RCC -11 algebra (see Figure 4.2) can be visualized using the snapshots as depicted in Figure 4.8 whose final outcome is a refined RCC -15 algebra. Internally this module makes use of the function `triples2Remove` for automating the entire operation of related triple removal

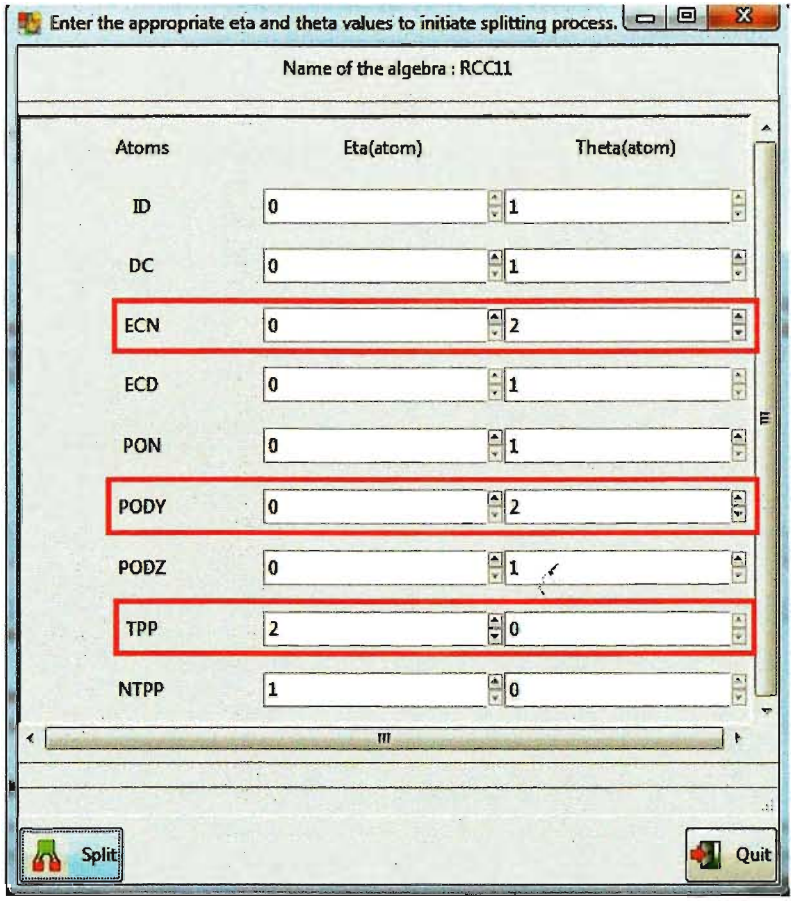
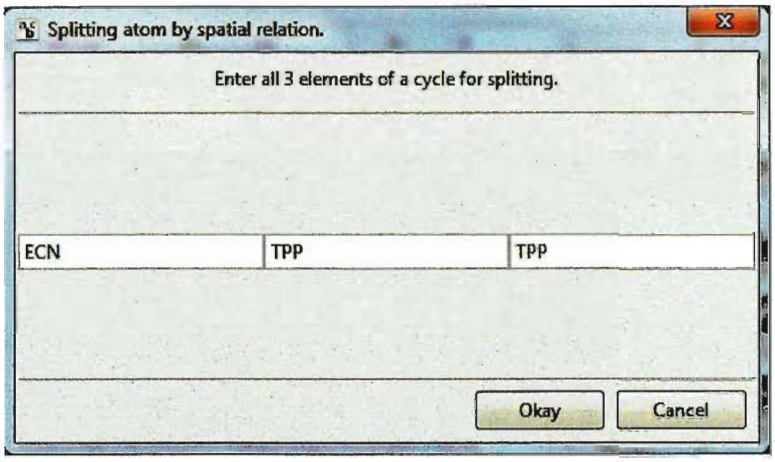


Figure 4.7: Splitting atom TPP in RCC-11

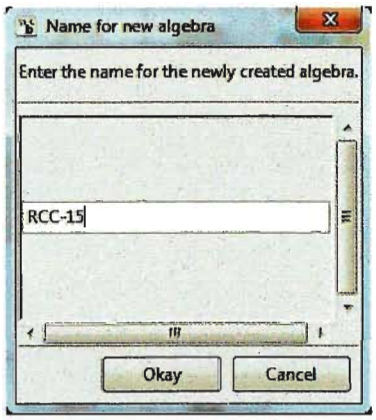
unlike the manual input of the triples (for removal) in the operation of `CropCycle` module.

4.6 RAParser

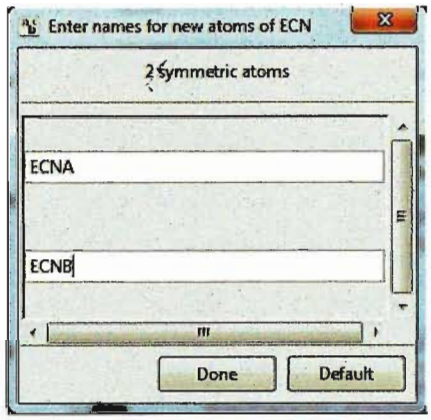
This module provides the background parser functions to display the `Atom Structure` of an algebra in human-readable format i.e., it displays the types `Ternary`, `Cycle` and `absTable` of an algebra in terms of its atom names rather than just machine-storable integers. It caters to back and forth internal conversions of algebras from the tree controller listing all algebras and algebraic attributes display (refer to Figure 4.5). The module functions are more commonly invoked when the user activates the button "Display Selected Algebra" on the control panel and also when the user instigates an operation on the currently viewed algebra. The latter part is done to ensure manipulation of machine-readable data by other module operations which is infeasible



(a) Triple - Spatial relation and the atom to be split using it.



(b) Enter name for new algebra.



(c) Specify names for atoms in the new algebra.

Figure 4.8: In progress - Various stages of the Split and Remove operation.

using the displayed text format. Apart from a host of other helper functions, the crux of this conversion is implemented by the functions `convertCycles`, `tripleAsNames`, `parseTernary`, `parseCycle`, `parseRelAlg`.

We begin with functions `tripleAsNames` and `convertCycles` which convert internally stored algebras to text format for display purpose.

```
tripleAsNames ::  [String] -> Ternary -> String

tripleAsNames atms (a1,a2,a3) =

        let trans t = fromJust $ intToName1 atms t

            (t1,t2,t3) = (trans a1,trans a2,trans a3)

        in "(" ++ t1 ++ "," ++ t2 ++ "," ++ t3 ++ ")"
```

```
convertCycles ::  [String] -> Cycle -> String

convertCycles atms cyc=let allTriples = map (tripleAsNames atms) cyc

                       withCommas = intersperse "," allTriples

                   in concat withCommas
```

The functions `parseTernary`, `parseCycle` and `parseRelAlg` convert the displayed algebras to machine-readable data for use by other modules.

```
parseTernary ::  Parser RAGenerator.Ternary

parseTernary = do

                        char '('

                        a <- number

                        commaspace

                        b <- number

                        commaspace

                        c <- number

                        char ')'

                        return (a,b,c)

                      <?> "ternary parse fail"
```

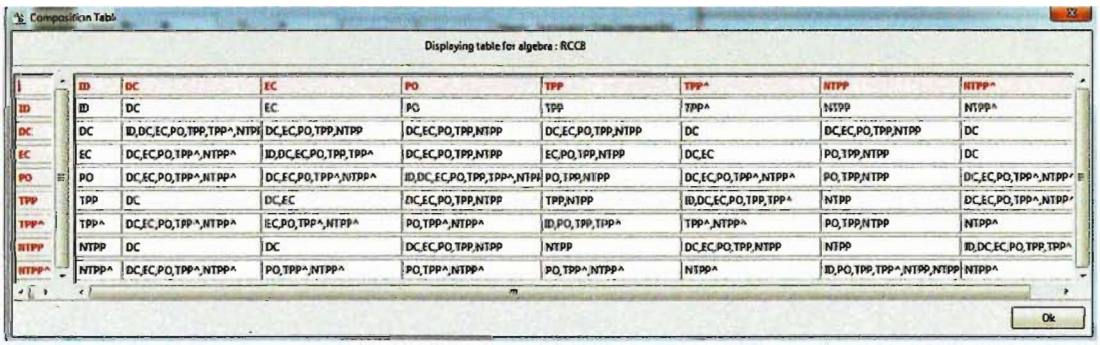


Figure 4.9: Composition Table for RCC-8 displayed by the System.

```
parseCycle ::  Parser RAGenerator.Cycle

parseCycle = do

                    char '['

                    a <- number

                    list <|>(char ']' >> return [])

               <?> "cycle"

               where list = do

                         triple <- parseTernary

                         rest <- (commaspace >> list) <|> (char ']'

                              >> return []);

                         return (triple:rest)

                         <?> "ternary list parse fail"
```

Using the above two parser functions we will be able to reconstruct the `Atom Structure` for an algebra that is displayed in the text format.

```
parseRelAlg ::  Parser RAGenerator.RelAlg

parseRelAlg = do

                    string "RelAlg { name = "

                    name' <- RAParser.name

                    string ", n = "

                    n' <- number

                    string ", s = "

                    s' <- number

                    string ", absTable = "

                    absTable' <- parseCycle
```

```
        string ", atomNames = "
        atomNames' <- namelist
        char ' } '
        return (RAGenerator.RelAlg { RAGenerator.name
                = name', RAGenerator.n = n',
                RAGenerator.s = s',
                RAGenerator.absTable = absTable',
                RAGenerator.atomNames = atomNames' } )
<?> "Error in RelAlg parsing"
```

4.7 FileInterface

The module `FileInterface` encapsulates useful file routines such as loading a file containing formatted algebras, saving all the algebras appearing in the tree controller list and also an option to save an individual algebra in a data file. The routines in the module utilize standard file manipulation libraries of Haskell language to implement these operations. The user activation of the buttons "Load Algs from File", "Save Alg List" and "Save Algebra" shown in the control panel layout of Figure 4.5 results in the invokation of these routines.

4.8 CompTable

This module has functions that display the composition table (refer to Definition 14 of Chapter 2.) for the currently viewed algebra. The functions in this module get invoked when the user instigates the button "Display Table" on the control panel of the system resulting in graphic tabular interpretation of composition for the given algebra. Figure 4.9 shows a typical output of this operation for RCC -8 algebra.

Chapter 5

Conclusions

This chapter rehashes the main line of argument of the thesis. We begin with the summarization of the research work described in previous chapters leading to its eventual resolution. We conclude by having an insight into related areas that require further investigation and has potential for advancing our research work.

5.1 Synopsis

The agenda of the thesis is to focus chiefly in developing a mechanism of generating new atom structures of relation algebras from old ones. This task was achieved by devising a new splitting mechanism in Chapter 3. Although the impetus for such a splitting was primarily given in [15], this thesis presents a comprehensive framework for automating this process. The splitting strategy stipulated in this thesis acts as a starting point of a variety of methods for splitting atoms in relation algebras. The very general definition of an extension provides the opportunity for this study. It will be interesting to characterize the different methods by additional properties. For example, Theorem 3.0.8 along with the definition of splitting succinctly characterizes this construction. But we have not provided such a full characterization of our method yet.

As described in the previous chapter the background functionality of the entire system is based on Haskell platform whilst the front end interface libraries are provided by GTK+ toolkit. This front end user interface caters to the creation, visualization and archival of atom structures for reusing data.

5.2 Future Work

One of the drawbacks of the system is its inability to perform faster operations on relation algebras with large N values especially in cases where $N > 15$. This could be overcome by introducing the concept of concurrency in our system whereby many of the subtasks may be designated as multiple threads. However this concurrency field is still in its nascent stage in Haskell platform. Moreover it warranted a lot more investment of time than the critical timelines permitted for our research. As a result the conversion process could not be persisted and it leaves scope for future incorporation.

There are a host of other tools developed at Brock University on the similar lines of research for managing relation algebras. For example, the tool developed by Si Zhang in [52]. These tools can be integrated with our system for better manageability and eliminate potential redundancies.

We may also look into development of certain core functions using a multithreaded approach in different languages like C or C++. This is due to the fact that efficiency of these tasks is better in C or C++ in comparison to concurrency in Haskell. The functionality of these external functions may then be tapped by Haskell based system using the foreign function interfaces.

Bibliography

[1] Andréka, H. , Düntsch, I. , and Németi, I. , *A non permutational integral relation algebra.*, Michigan Math **39** (1992), 371–384.

[2] Andréka, H. , Maddux, R. D. , and Németi, I. , *Splitting in relation algebras*, Proceedings of The American Mathematical Society **111** (1991).

[3] Asher, N. and Vieu, L. , *Toward a geometry of common sense: A semantics and a complete axiomatization of mereotopology*, in Proceedings of IJCAI95, 1995.

[4] Bowman, C. L. , *A calculus of individuals based on 'connection'.*, Notre Dame J. Formal Logic **22** (1981), 204–218 (English).

[5] Brink, C. , Kahl, W. , and Schmidt, G. , Relational Methods in Computer Science, Advances in Computing 1997 (Vienna), Springer-Verlag, 1997.

[6] Clementini, E. , Sharma, J. , and Egenhofer, M. J. , *Modelling topological spatial relations: Strategies for query processing*, Computers and Graphics **18** (1994), 815–822.

[7] Cohn, A. , Bennett, B. , Gooday, J. , and Gotts, N. , *Representing and reasoning with qualitative spatial relations about regions*, Spatial and Temporal Reasoning (Stock, O. , ed.), Kluwer, IRST, 1997, pp. 97–134.

[8] Cohn, A. G. , Bennett, B. , Gooday, J. , and Gotts, N. M. , Qualitative Spatial Representation and Reasoning with the Region Connection Calculus, GeoInformatica 1 (1997), 275–316.

[9] Cohn, A. G. and Gooday, J. , *Defining the syntax and the semantics of a visual programming language in a spatial logic*, AAAI-94 Workshop (Seattle, Washington, USA) (Anger, F. and Loganantharaj, R. , eds.), July 1994.

[10] Cui, Z. , Cohn, A. G. , and Randell, D. A. , *Qualitative and topological relationships in spatial databases*, Symposium on Large Spatial Databases, 1993, pp. 296–315.

[11] De Laguna, T. , *Point, line and surface as sets of solids*, The journal of philosophy **19** (1922), 449–461.

[12] Dimov, G. and Vakarelov, D. , *Contact algebras and region-based theory of space: Proximity approach - ii*, Fundamenta Informaticae **74** (2006), 251–282.

[13] Düntsch, I. , *Relation algebras and their application in temporal and spatial reasoning*, Artificial Intelligence Review **23** (2005), 315–357.

[14] Düntsch, I. and Orlowska, E. , *A proof system for contact relation algebras*, Journal of Philosophical Logic (2000).

[15] Düntsch, I. , Schmidt, G. , and Winter, M. , *A necessary relation algebra for mereotopology*, Studia Logica - An International Journal for Symbolic Logic **69** (2001), 381–409.

[16] Düntsch, I. , Wang, H. , and Mccloskey, S. , *Relations algebras in qualitative spatial reasoning*, Fundamenta Informaticae **39** (1999), 229–248.

[17] ———, *A relation - algebraic approach to the region connection calculus*, Theoretical Computer Science **255** (2001), 63–83.

[18] Düntsch, I. and Winter, M. , *A representation theorem for boolean contact algebras*, Theoretical Computer Science **347** (2003), 498–512.

[19] ———, *Algebraization and representation of mereotopological structures*, Relational Methods in Computer Science, 2004.

[20] ———, *Weak contact structures*, Relational Methods in Computer Science, 2005, pp. 73–82.

[21] ———, *The lattice of contact relations on a boolean algebra*, Relational Methods in Computer Science, 2008, pp. 99–109.

[22] Egenhofer, M. J. , *Reasoning about binary topological relations*, Symposium on Large Spatial Databases, 1991, pp. 143–160.

[23] Egenhofer, M. J. and Herring, J. R. , Categorizing topological relationships between regions, lines, and points in geographic database, Tech. report, University of Maine, 1991.

[24] Egenhofer, M. J. and Sharma, J. , *Topological consistency*, University of South Carolina, Columbia, SC, 1992.

[25] Frias, M. F. and Maddux, R. D. , *Non-embeddable simple relation algebras*, Algebra Universalis **38** (1997), 115–135.

[26] Givant, S. , *The calculus of relations as a foundation for mathematics*, Journal of Automated Reasoning **37** (2006), 277–322.

[27] Gotts, N. M. , *Research report series- university of leeds school of computer studies lu scs rr*, Journal of Automated Reasoning (1996).

[28] Henkin, L. , Monk, J. D. , and Tarski, A. , *Cylindric algebras*, Studies in logic and the foundations of mathematics, no. v. 2, North-Holland Pub. Co., 1985.

[29] Jónsson, B. and Tarski, A. , *Boolean algebras with operators part ii*, Amer. Journal of Mathematics **73** (1952), 127–162.

[30] Maddux, R. D. , *Some varieties containing relation algebras*, Transactions of The American Mathematical Society **272** (1982), 501–501.

[31] ———, *Finite integral relation algebras*, Proceedings of a Conference held at Charleston **1149** (1985), 175–197.

[32] ———, *: Studies in logic and the foundations of mathematics*, vol. 150, Elsevier Science, 2006.

[33] Morris, S. A. , *Topology without tears*, University of New England, 1989.

[34] Orlowska, E. and Szalas, A. , Relational Methods for Computer Science Applications, RelMics 2001 (Heidelberg), Springer-Physica Verlag, 2001.

[35] Pratt-Hartmann, I. , *Empiricism and rationalism in region-based theories of space*, Fundam. Inf. **46** (2001), 159–186.

[36] Pratt-Hartmann, I. and Schoop, D. , *A complete axiom system for polygonal mereotopology of the real plane*, Journal of Philosophical Logic **27** (1998), 621–658, 10.1023/A:1004361501703.

[37] Randell, D. A. and Cohn, A. G. , *Modelling topological and metrical properties in physical processes*, Principles of Knowledge Representation and Reasoning, 1989, pp. 357–368.

[38] Randell, D. A. , Cohn, A. G. , and Cui, Z. , *Computing transivity tables: A challenge for automated theorem provers*, Conference on Automated Deduction, 1992, pp. 786–790.

[39] Randell, D. A. , Cui, Z. , and Cohn, A. G. , *A spatial logic based on regions and connection*, Principles of Knowledge Representation and Reasoning, 1992, pp. 165–176.

[40] Schlieder, C. , *Geometrische aspekte räumlichen schließens*, KI **7** (1993), no. 4, 29–34.

[41] Schmidt, G. and Ströhlein, T. , *Relations and graphs: discrete mathematics for computer scientists*, EATCS monographs on theoretical computer science, Springer-Verlag, 1993.

[42] Siddavaatam, P. and Winter, M. , *Refining relational algebras for qualitative spatial reasoning*, *Ph.D. Student Programme of 12th International Conference on Relational and Algebraic Methods in Computer Science (RAMiCS 12)*, June 2011.

[43] ———, *Splitting atoms in relational algebras*, Relational and Algebraic Methods in Computer Science (Rotterdam) (de Swart, H. , ed.), Lecture Notes in Computer Science, vol. 6663, June 2011, pp. 331–346.

[44] Stell, J. G. , *Boolean connection algebras: A new approach to the region-connection calculus*, Artificial Intelligence **122** (2000), 111–136.

[45] Tarski, A. , *On the calculus of relations*, Journal of Symbolic Logic **6** (1941), 73–89.

[46] Tarski, A. and Givant, S. , *A formalization of set theory without variables*, American Mathematical Society (1996).

[47] Vakarelov, D. , Dimov, G. , Düntsch, I. , and Bennett, B. , *A proximity approach to some region-based theories of space*, Journal of Applied Non-classical Logics **12** (2002), 527–559.

[48] Vieu, L. , *Sémantique des relations spatiales et inférences spatio-temporelles : une contribution à l'étude des structures formelles de l'espace en langage naturel*, Ph.D. thesis, Université Paul Sabatier, IRIT, 1991.

[49] Whitehead, A. N. , *Process and reality*, MacMillan, New York, 1929.

[50] Winter, M. , *Relation algebras are matrix algebras over a suitable basis*, Tech. Report 1998-05, UBwM, November 1998.

[51] ———, *A pseudo representation theorem for various categories of relations*, Theory and Applications of Categories (TAC) **7** (2000), 23–37.

[52] Zhang, S. , *Generating finite integral relation algebras*, Master's thesis, Brock University, St.Catharines, Canada, 2010.