The Fixed-Point Semantics of Relational Concept Analysis

JÉRÔME EUZENAT, Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LIG, F-38000 Grenoble, France

Background: Relational concept analysis (RCA) is an extension of formal concept analysis dealing with several related formal contexts simultaneously. It can learn description logic theories from data and has been used within various applications. However, RCA returns a single family of concept lattices, though, when the data feature circular dependencies, other solutions may be considered acceptable. The semantics of RCA, provided in an operational way, does not shed light on this issue.
Objectives: This paper aims at defining precisely the semantics of RCA and identifying alternative solutions.
Methods: We first characterise the acceptable solutions as those families of concept lattices which belong to the space determined by the initial contexts (*well-formed*), which cannot scale new attributes (*saturated*), and which refer only to concepts of the family (*self-supported*). We adopt a functional view on the RCA process by defining the space of well-formed solutions and two functions on that space: one expansive and the other contractive. In this context, the acceptable solutions are the common fixed points of both functions.
Results: We show that RCA returns the least element of the set of acceptable solutions. In addition, it is possible to build dually an operation that generates its greatest element. The set of acceptable solutions is a complete sublattice of the interval between these two elements. Its structure, and how the defined functions traverse it, are studied in detail.

JAIR Associate Editor: Ivan José Varzinczak

JAIR Reference Format:
Jérôme Euzenat. 2025. The Fixed-Point Semantics of Relational Concept Analysis. *Journal of Artificial Intelligence Research* 83, Article 1 (June 2025), 38 pages. DOI: 10.1613/jair.1.17882

1 Introduction

Formal concept analysis (FCA) is a well-defined and widely used operation for extracting concept lattices from binary data tables [16]. This means that, from a set of objects described by Boolean attributes (called context), it will generate the set of all descriptions relevant to these objects (called concepts) organised in a lattice following a generalisation order relation between these concepts. For instance, from a description of people through their diploma, major and current job it is possible to generate concepts about those bachelors in literature who hold a teacher position. This concept is a subconcept of bachelors in literature, which is a subconcept of that of people having a bachelor degree. The subconcepts add more constraints (attributes) to the objects they cover and thus cover less objects. These concepts, in turn, may be used to study the diploma-job matching.

Many other techniques exist in AI and elsewhere for analysing data. Features that distinguish FCA is that it is symbolic, i.e. concept descriptions are algebraic combinations of attributes, and complete, i.e. FCA provides the lattice of *all* symbolically described concepts covering the input data. From this set, it is possible to select those which are more relevant to a particular purpose. Some numerical techniques would instead provide the concepts which optimise a criterion or may project the object descriptions in a space that make their separation and grouping easier. This leads to select the concepts to be considered, for which a description remains to be found. FCA and these types of techniques are complementary.

Author's Contact Information: Jérôme Euzenat, ORCID: 0000-0002-5014-4411, Jerome.Euzenat@inria.fr, Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LIG, F-38000 Grenoble, Grenoble, France.

DOI: 10.1613/jair.1.17882

Formal concept analysis has been put to work in a variety of applications [15, 22], but its Boolean descriptions are limited with respect to real data. It has thus received many extensions for using concrete domains to overcome this limitation. For instance, people may be further characterised by their salary, age and hobby. ‘Conceptual scaling’ can group age and salary into intervals so as to be treated by FCA. However, such extensions do not cover relations between objects themselves: the fact that students have been taught by specific teachers or have been registered to specific schools.

Relational concept analysis (RCA) extends FCA by taking into account *relations between objects* [26]. Such relations induce possible ‘relational attributes’ that are added to the initial contexts, through ‘relational scaling’. From a family of related scaled contexts, RCA generates a family of dependent concept lattices which may include concepts that would not exist in FCA. For instance, people may be related to their household, their employer or their properties. Household, themselves may be described by their income, size and related to their members and the neighbourhood in which they are settled. This may lead to classify household depending on theses features and creating new attributes for people based on whether they live in a low-income household and whether they have been employed by a school. These new attributes, in turn, will generate new people concepts such as low-income household teachers and may lead to unveil indirect relations between household and jobs.

Although RCA was initially targeting conceptual description languages such as UML [19], it has been generalised to more varied description logic constructors [26]. Relational concept analysis has been used for instance for analysing the ecological and sanitary quality of water courses [24]. For that purpose, it connects formal contexts corresponding to water courses to data collection points which themselves are connected to measures and to organisms collected in water that can be described by further attributes. These relations between objects help generating richer concept descriptions comprising relations between concepts. It is then possible to connect the abundance or scarcity of some species to the presence of some pollutants, e.g. glyphosate. RCA has also been used for other purposes such as generating link keys used to extract links from RDF data sets [1].

In principle, RCA contributes to the completeness of FCA by providing more concepts to classify objects in. However, some concepts that can be considered acceptable may still not be generated by RCA. Indeed, in presence of circular relationships between objects, RCA may not identify reciprocally supporting concepts. For instance, consider that people are related to the schools they have attended, and schools are related to the students that they graduated. People attended multiple schools and schools had many student. There may be populations who have attended specific groups of schools and graduated from these. Although there may be no attribute distinguishing these populations and groups, they can be described by having attended at least one school of the group of schools having graduated only students of the population. These reciprocally supporting attributes lead to a refined version of the returned concept lattice which contains more concepts. Such concepts may reveal distinct populations based on various hidden factors not directly available from the data, such as wealth or information whose collection is not possible.

These mutually supported concepts may not be returned by RCA. However, they determine families of more complete concept lattices covering the data. Hence, it remains to determine which unique family is returned by RCA. In this respect, this paper may be thought of as a way to restore the completeness of RCA by providing more possible concepts. This completeness may be useful in applications in which the most relevant solution is not the minimal one. It may also contribute to unveil more implications or dependencies between concepts.

This problem of alternative RCA results stemmed out of curiosity. It occurred to us through experimenting with relational concept analysis for extracting link keys. Although RCA was returning acceptable results, it was easy to identify other acceptable results that it did not return. When RCA is used for extracting description logic terminologies, it makes sense to return minimal terminologies that may be extended. But different sets of link keys would return totally different sets of links. The problem also manifests itself in applications in which developers add artificial identifiers in their data in order to constrain the returned solution to include more concepts [8, 6].

To understand why such concepts are not provided, and which concepts are returned by RCA, this paper questions its semantics. The semantics of relational concept analysis has, so far, been provided in a rather operational way [27]. It specifies that RCA returns a family of concept lattices referring to each other that describe the input data and it shows that this result is unique. However, when there exist cycles in the dependencies between data, several families may satisfy these constraints. Hence, relational concept analysis needs a more precise and process-independent semantics that defines what it returns. For that purpose, this paper provides a structured description of the space on which relational concept analysis applies. It then defines acceptable solutions as those families of concept lattices which belong to the space determined by the initial family of formal contexts (*well-formed*), cannot scale new attributes (*saturated*), and refer only to concepts of the family (*self-supported*).

Relational concept analysis is then studied in a functional framework. It characterises the acceptable solutions as the fixed points of two functions, one expansive, which extends concept lattices as long as there are reasons to generate concepts distinguishing objects, and the other contractive, which reduces concept lattices as long as the attributes they are built on are not supported by remaining concepts. These functions can be iterated and the acceptable solutions are those families of concept lattices which are fixed points for both (Proposition 28): there is no reason to either extend or reduce them. The results provided by RCA are then proved to be the smallest acceptable solution, which is the least fixed point of the expansive function (Proposition 29). It also offers an alternative semantics based on the greatest element of this set, which is the greatest fixed point of the contractive function (Proposition 30). The structure of the set of fixed points is further characterised to support algorithmic developments.

This paper extends the results obtained for the RCA^0 restriction of RCA [10], which contains a single formal context, hence a single concept lattice, and no attribute, only relations. In spite of the simplicity of RCA^0, the main arguments of this work were already present and it remains a good introduction to the considered problems. A specific research report [11] builds on the results established in [10] and extends them step-by-step to RCA. It also explains the legacy names used here for functions.

The present paper offers a direct and more synthetic formulation of the fixed-point semantics of RCA and the space of alternative acceptable solutions.

We first present the work on which this one builds (relational concept analysis) and relevant related work (Section 2). Then, simple examples are provided to illustrate that relational concept analysis may accept concept lattices which are not those provided by the RCA operation (Section 3). In order to determine which could be the acceptable solutions of relational concept analysis, Section 4 circumscribes the space of objects that it considers. Two functions, an expansive function, corresponding to the operations of RCA, and a contractive function, aiming at finding self-supported solutions, are defined on this space (Section 5). The fixed points of these functions are discussed leading to a redefinition of acceptable solutions as those objects of the space which are fixed points for both functions (Section 6). In consequence, solutions returned by RCA are precisely characterised as the smallest acceptable solution, a dual operation returning the greatest acceptable solution can be defined, and the structure of the set of acceptable solutions is investigated (Section 7).

2 Preliminaries and Related Work

We mix preliminaries with related work for reasons of space, but also because the paper directly builds on this related work.

2.1 Basics of Formal Concept Analysis

This paper relies only on the most basic results of formal concept analysis expressed as order-preserving functions.

Formal Concept Analysis (FCA) [16] starts with a formal context (hereafter context) $\langle G, M, I\rangle$ such that G denotes a set of objects, M a set of attributes, and $I \subseteq G \times M$ a binary relation between G and M, called the incidence relation. The statement gIm is interpreted as 'object g has attribute m', also noted $m(g)$. Two operators $\cdot^{\uparrow}$ and $\cdot^{\downarrow}$ define a Galois connection between the powersets $\langle 2^G, \subseteq\rangle$ and $\langle 2^M, \subseteq\rangle$, with $A \subseteq G$ and $B \subseteq M$:

$$A^{\uparrow} = \{m \in M \mid gIm \text{ for all } g \in A\},$$
$$B^{\downarrow} = \{g \in G \mid gIm \text{ for all } m \in B\}.$$

The operators $\cdot^{\uparrow}$ and $\cdot^{\downarrow}$ are decreasing, i.e. if $A_1 \subseteq A_2$ then $A_2^{\uparrow} \subseteq A_1^{\uparrow}$ and if $B_1 \subseteq B_2$ then $B_2^{\downarrow} \subseteq B_1^{\downarrow}$. Intuitively, the less objects there are, the more attributes they share, and dually, the less attributes there are, the more objects have these attributes. It can be checked that $A \subseteq A^{\uparrow\downarrow}$ and that $B \subseteq B^{\downarrow\uparrow}$, that $A^{\uparrow} = A^{\uparrow\downarrow\uparrow}$ and that $B^{\downarrow} = B^{\downarrow\uparrow\downarrow}$.

A pair $\langle A, B\rangle \in 2^G \times 2^M$, such that $A^{\uparrow} = B$ and $B^{\downarrow} = A$, is called a formal concept (hereafter concept), where $A = extent(\langle A, B\rangle)$ is the extent and $B = intent(\langle A, B\rangle)$ the intent of $\langle A, B\rangle$. Moreover, for a formal concept $\langle A, B\rangle$, A and B are closed for the closure operations $\cdot^{\uparrow\downarrow}$ and $\cdot^{\downarrow\uparrow}$, respectively, i.e. $A^{\uparrow\downarrow} = A$ and $B^{\downarrow\uparrow} = B$.

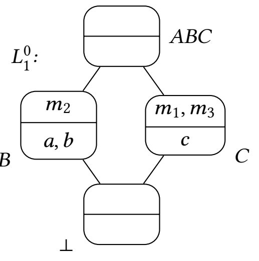
Concepts are partially ordered by $\langle A_1, B_1\rangle \leq \langle A_2, B_2\rangle \Leftrightarrow A_1 \subseteq A_2$ or equivalently $B_2 \subseteq B_1$. With respect to this partial order, the set of all formal concepts is a complete lattice called the concept lattice of $\langle G, M, I\rangle$. It has for supremum the concept $\top = \langle G, G^{\uparrow}\rangle$ and for infimum the concept $\bot = \langle M^{\downarrow}, M\rangle$.

Example 1 (Formal concept analysis). *Starting from a context $K_1^0 = \langle G_1, M_1^0, I_1^0\rangle$ with $G_1 = \{a, b, c\}$, $M_1^0 = \{m_1, m_2, m_3\}$ and I_1^0 as the incidence relation whose table is given below, the application of* FCA *results in the lattice made of the concepts ABC, AB, C and $\bot$ as:*

FCA(

K_1^0	m_1	m_2	m_3
a		×	
b		×	
c	×		×

) =



By convention, concept lattices are represented by their reflexive-transitive reduction (Haase diagram) in which concepts are displayed in two parts: an upper part representing their intent and a lower part representing their extent. They only display the proper part of their intent and extent. Their actual intent is obtained by joining it to the union of the proper intents of their more general concepts. Conversely, their extent is obtained by joining it to the union of the proper extent of their more specific concepts. Hence, $ABC = \langle\{a, b, c\}, \varnothing\rangle$ and $\bot = \langle\varnothing, \{m_1, m_2, m_3\}\rangle$.

Formal concept analysis can be considered as a function that associates to a context $\langle G, M, I\rangle$ its concept lattice $\langle C, \leq\rangle = \text{FCA}(\langle G, M, I\rangle)$ (or $\underline{\mathfrak{B}}(G, M, I)$ [16]). This is illustrated by Example 1. By abuse of language, when a variable L denotes a concept lattice $\langle C, \leq\rangle$, L will also be used to denote C.

The concepts that can be created from a context can be identified by their extent. Hence, $\eta(\langle G, M, I\rangle) = 2^G$ is the set of all concept names that may be used in any such concept lattice[1]. We will identify the concepts by such sets; the extent of a so-named concept will be the set of objects in its name. In any specific concept lattice $L = \text{FCA}(K)$, the subset $\eta(L)$ of $\eta(K)$ is the set of names of concepts in this lattice according to this convention as illustrated in Example 2.

[1]A similar remark is made in [29, §4.1.2].

Example 2 (Concept names). *Consider the context K_1^0 of Example 1. The set of objects of K_1^0 being $G_1 = \{a, b, c\}$, the set of concept names that can be created for them in any concept lattice is $\eta(K_1^0) = \{ABC, AB, AC, BC, A, B, C, \bot\}$. In the specific lattice obtained in Example 1, the set of concept names is $\eta(L_1^0) = \{ABC, AB, C, \bot\}$.*

Throughout the paper, concepts are named after their extent. They will be displayed as uppercase character strings.

In order to discuss algorithms performing formal concept analysis, we will restrict ourselves to finite structures, as it is often the case. In such a case, from finite contexts are generated finite lattices whose concepts have finite extents and intents.

2.2 Extending Formal Concept Analysis with Scaling

Formal concept analysis is defined on relatively simple structures hence many extensions of it have been designed. These may allow formal concept analysis to (*a*) deal with more complex input structures, and/or (*b*) generate more expressive and interpretable knowledge structures.

2.2.1 Scaling: a Generalisation. Scaling is one kind of extension of type (*a*). It is a way to encode a more complex structure Σ into FCA. For that purpose, a scaling operation ς determines a set $D_{\varsigma,\Sigma}$ of Boolean attributes to be added to a context K from a structure Σ. In scaled contexts, these attributes can be interpreted so that the incidence relation I is immediately derived from the attribute $m \in D_{\varsigma,\Sigma}$ following:

$$\Sigma \models gIm \text{ or } \Sigma \models m(g).$$

Hence, adding attributes M' to a context under such a structure Σ consists of adding the attributes and extending the incidence relation according to this interpretation. It may be performed as:

$$\mathrm{K}^{\Sigma}_{+M'}(\langle G, M, I\rangle) = \langle G, M \cup M', I \cup \{\langle g, m\rangle \in G \times M' \mid \Sigma \models m(g)\}\rangle,$$

and suppressing them as:

$$\mathrm{K}^{\Sigma}_{-M'}(\langle G, M, I\rangle) = \langle G, M \setminus M', I \setminus \{\langle g, m\rangle \in G \times M'\}\rangle.$$

Applying a scaling operation ς to a context K following a structure Σ can thus be decomposed into (*i*) determining the set of attributes $D_{\varsigma,\Sigma}$ to add, and (*ii*) extending the context with these attributes:

$$\sigma_{\varsigma}(K, \Sigma) = \mathrm{K}^{\Sigma}_{+D_{\varsigma,\Sigma}}(K).$$

This unified view of scaling may be applied to many available scaling operations, including the initial conceptual scaling [16] and logical scaling [25]. RCA relies on relational scaling.

2.2.2 Relational Scaling. Relational scaling operations (ς) considered in [26] create relational attributes from a binary relation $r \subseteq G \times G'$ between two sets of objects $G = \mathrm{dom}(r)$ and $G' = \mathrm{cod}(r)$ and a concept lattice L on G'. $\varsigma(r, c)$ provides the syntactic form of the attribute. For example, qualified existential scaling ($\varsigma = \exists$) is expressed by $\exists(r, c) = \exists r.c$. For instance, the employer relation may relate people to companies depending on whether one works for the other. If the company lattice contains the school concept, then the relational attribute $\exists$employer.School may be scaled and holds for all people employed by a school.

Thus, the set of qualified existential attributes are:

$$D_{\exists,\langle r,L\rangle} = \{\exists r.c \mid c \in \eta(L)\}.$$

Such attributes are interpreted, according to a closed-world description logic interpretation, by

$$\langle r, L\rangle \models gI\exists r.c \;\text{ iff }\; \exists g'; \langle g, g'\rangle \in r \wedge g' \in extent(c).$$

This is illustrated in Example 3.

Example 3 (Relational scaling). *Consider that the relation q is given by the table:*

q	a	b	c
d	×		
e		×	
f			×

between $G_1 = \{a, b, c\}$ and $G_2 = \{d, e, f\}$ and consider the concept lattice L_1^0 corresponding to the context K_1^0 of Example 1. The concepts in L_1^0 have names in $\eta(L_1^0) = \{ABC, AB, C, \bot\}$ (Example 2) hence scaling by $\sigma_\exists$ will provide the attribute set $\{\exists q.ABC, \exists q.AB, \exists q.C, \exists q.\bot\}$. The description of the relation q allows to uncover the incidence relation for these (the incidence relation for $\bot$ is always empty for $\exists$ so never displayed in the examples):

	$\exists q.ABC$	$\exists q.AB$	$\exists q.C$
d	×	×	
e	×	×	
f	×		×

If, in addition, the context K_2^0 of G_2 had two other attributes n_1 and n_2 whose incidence relation is given as I_2^0 displayed below, then the scaling operation would correspond to:

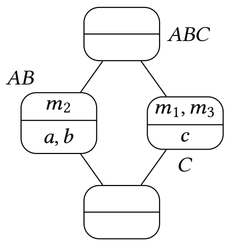
$\sigma_\exists($

K_2^0	n_1	n_2
d	×	
e	×	×
f		

,

q	a	b	c
d	×		
e		×	
f			×

,



$) =$

K_2^1	n_1	n_2	$\exists q.ABC$	$\exists q.AB$	$\exists q.C$
d	×		×	×	
e	×	×	×	×	
f			×		×

When generating relational attributes, scaling only relies on the *names* of the concepts in L. It is thus possible to define the sets of attributes from the set of names. The set of relational attributes that can be scaled from ς and r against a set of concept of names N is $D_{\varsigma,r,N} = \{\varsigma(r, c) \mid c \in N\}$. This can be used with $\eta(L)$, i.e. the names of concepts actually in L, or with $\eta(K)$, i.e. the names of all possible concepts to be generated from a context K. Example 4 illustrates this.

Example 4 (Set of relational attributes). *For the context K_2^0 of Example 3, if there is only one relation q, whose codomain is K_1^0 of Example 1, and the existential scaling operation $\exists$, then the set of possibly scalable relational attributes is:*

$$D_{\exists,q,\eta(K_1^0)} = \{\exists q.ABC, \exists q.AB, \exists q.AC, \exists q.BC, \exists q.A, \exists q.B, \exists q.C, \exists q.\bot\}.$$

But using only those concepts from the lattice L_1^0 obtained in Example 1, this set is reduced to:

$$D_{\exists,q,\eta(L_1^0)} = \{\exists q.ABC, \exists q.AB, \exists q.C, \exists q.\bot\}.$$

Various relational scaling operations exist, such as existential, strict and wide universal, min and max cardinality, which all follow the classical role restriction semantics of description logics [3] under the closed-world assumption (see Table 1). The set of relational attributes obtained from a relation r by relational scaling may be large but remains finite as long as $G' = \text{cod}(r)$ is finite. Cardinality constraints, relying on integers, may entail infinite sets of concepts in theory, but in practice, when G' is finite, the set of meaningful cardinality attributes is bounded by $|G'|$.

In fact, RCA may be considered as a very general way to apply relational scaling across contexts. Various kinds of relational scaling operations have been provided [6, 1, 29].

Table 1. Relational scaling operations (inspired from [26, 6]) and additional link key condition scaling operations [1].

name	language (D)	Σ	condition ($m(g)$)
existential	$\exists r$	R	$r(g) \neq \emptyset$
universal (wide)	$\forall r.c$	R, L	$r(g) \subseteq extent(c)$
strict universal	$\forall\exists r.c$	R, L	$r(g) \neq \emptyset \wedge r(g) \subseteq extent(c)$
contains (wide)	$\forall c.r$	R, L	$extent(c) \subseteq r(g)$
strict contains	$\forall\exists c.r$	R, L	$extent(c) \neq \emptyset$ $\wedge extent(c) \subseteq r(g)$
qualified existential	$\exists r.c$	R, L	$r(g) \cap extent(c) \neq \emptyset$
qualified min cardinality	$\leq_n r.c$	R, L	$\|r(g) \cap extent(c)\| \leq n$
qualified max cardinality	$\geq_n r.c$	R, L	$\|r(g) \cap extent(c)\| \geq n$
$\forall$-condition	$\forall\langle r, r'\rangle_k$	$R \times R', L_{C \times C'}$	$r(g) =_k r'(g')$
$\exists$-condition	$\exists\langle r, r'\rangle_k$	$R \times R', L_{C \times C'}$	$r(g) \cap_k r'(g') \neq \emptyset$

2.3 Other Extensions

There are other extensions [7, 13, 14] providing formal concept analysis with more expressiveness in the expression of intents without scaling (type b extension). Instead of scaling, they change the structure of the set of attributes, staying within the scope of Galois lattices. However, these extensions are not directly affected by the problem of context dependencies considered here as the attributes do not refer to concepts.

On the contrary, other approaches [21, 12] aim at extracting conceptual structures from n-ary relations without resorting to scaling. Their concepts have intents that can be thought of as conjunctive queries and extents as tuples of objects, i.e. answers to these queries. Hence, instead of being classes, i.e. monadic predicates, concepts correspond to general polyadic predicates. For that purpose, they rely on more expressive input, e.g. in Graph-FCA [12] the incidence relation is a hypergraph between objects, and produce more expressive representations. A comparison of RCA and Graph-FCA is provided in [20]. Graph-FCA adopts a different approach from RCA but should, in principle, suffer from the same problem as the one considered here as soon as it contains circular dependencies: intents would need to refer to concepts so created, i.e. named subqueries. This remains to be studied.

Finally, description logic base mining [4, 18] and relational concept analysis share the same purpose: inferring a TBox from an ABox (taken as an interpretation). However, RCA does this by introducing new named concepts based on FCA, where description logic base mining does not introduce new names but uses new concept descriptions inspired from Duquenne-Guigues implication bases [17]. Where, in Example 3, relational scaling would use attribute $\exists q.AB$, base mining would use the description $\exists q.\exists m_2.\top$. As soon as cycles occur in context dependencies, this naturally leads to cyclic concept definitions. This has been interpreted with the greatest fixed-point semantics in $\mathcal{EL}_{gfp}$. However, led by complexity considerations, work has focused on extracting minimal bases in $\mathcal{EL}$ through unravelling [4, 18]. The problem raised in this paper is different but applies as well to description logic base mining as soon as it is taken as a knowledge induction task from data: circular dependencies may lead to different, equally well-behaving, bases that would be worth taking in consideration.

2.4 A Very Short Introduction to RCA

Relational Concept Analysis (RCA) [26] extends FCA to the processing of relational datasets and allows inter-object relations to be materialised and incorporated into formal concept intents. It may also be thought of as a general way to deal with circular references using different scaling operations.

RCA applies

- a set of relational scaling operations Ω on
- a family of contexts $K^0 = \{\langle G_x, M^0_x, I^0_x\rangle\}_{x\in X}$ indexed by a finite set X, such that, if $x \neq z$, $G_x \cap G_z = \varnothing$, and
- a finite set of binary relations R, i.e. relations $r \subseteq G_x \times G_z$ (with $x, z \in X$).

$\langle K^0, R\rangle$ is called a relational context[2]. Each context of the family K^t may be abbreviated as $K^t_x = \langle G_x, M^t_x, I^t_x\rangle$. The use of the ordinal superscript t will become clearer in a few lines and can be ignored at that stage. We note $R_x = \{r \in R \mid \mathrm{dom}(r) = G_x\}$ and $R_{x,z} = \{r \in R_x \mid \mathrm{cod}(r) = G_z\}$.

2.4.1 Concept Names and Relational Attributes. The notation used for expressing sets of relational attributes can be generalised. For RCA, $\eta^*(K^0) = \{\eta(K^0_x)\}_{x\in X}$ is the indexed set of all concept names induced from all contexts in K^0. Similarly, for an indexed set of concept lattices $L = \{L_x\}_{x\in X}$, $\eta^*(L) = \{\eta(L_x)\}_{x\in X}$. Then, given a set of relational scaling operations Ω, a set of relations R, the set of scalable relational attributes for a concept with respect to an indexed family of sets of concept names N is:

$$D_{\Omega,R_x,N} = \bigcup_{\varsigma\in\Omega}\bigcup_{z\in X}\bigcup_{r\in R_{x,z}} D_{\varsigma,r,N_z}.$$

This notation is used for identifying the relational attributes that can be scaled for a context K_x from a set of concept lattices:

$$D_{\Omega,R_x,L} = D_{\Omega,R_x,\eta^*(L)},$$

or the set of all the possible relational attributes that can be scaled for a context K_x given a family of formal contexts K^0 as

$$D_{\Omega,R_x,K^0} = D_{\Omega,R_x,\eta^*(K^0)}.$$

Since $\forall z \in X$, $\eta(L_z) \subseteq \eta(K^0_z)$, then $D_{\Omega,R_x,\{L_z\}_{z\in X}} \subseteq D_{\Omega,R_x,K^0}$. This is illustrated by Example 5.

Example 5 (Set of relational attributes (cont'd)). *Following Example 4, if K^0_2, whose objects are $\{d,e,f\}$, is also linked to itself (K^0_2) by the relation s and the current set of concept names is $\eta(L^0_2) = \{DEF, DE, E\}$, then it would additionally scale the relational attributes: $D_{\{\exists\},\{s\},\{L^0_2\}} = D_{\exists,s,\eta(L^0_2)} = \{\exists s.DEF, \exists s.DE, \exists s.E\}$. If, in addition, the strict contains scaling operation ($\forall\exists C.r$, see Table 1) is used, then new relational attributes would be:*

$$\begin{aligned}D_{\{\exists,\forall\exists\},\{q,s\},\{L^0_1,L^0_2\}} = \{&\exists q.ABC, \exists q.AB, \exists q.C, \exists q.\bot, \exists s.DEF, \exists s.DE, \exists s.E,\\ &\forall\exists ABC.q, \forall\exists AB.q, \forall\exists C.q, \forall\exists\bot.q, \forall\exists DEF.s, \forall\exists DE.s, \forall\exists E.s\}.\end{aligned}$$

The semantics of these attributes is provided in the same way as above and noted $\langle R, L\rangle \models gI\varsigma(r,c)$.

2.4.2 Operations and Algorithm. Hereafter, we will consider relational scaling with the structure $\Sigma = \langle R, L\rangle$ made of a set of binary relations R between two sets of objects from K^0, and a family $L^t = \{L^t_x\}_{x\in X}$ of concept lattices obtained from K^t.

RCA applies relational scaling operations from a set Ω to each $K^t_x \in K^t$ and all relations $r \in R_{x,z}$ from the set of concepts in the corresponding $L^t_z = \mathrm{FCA}(K^t_z)$. Such scaling is defined as:

$$\sigma_\Omega(K_x, R, L) = \mathrm{K}^{\langle R,L\rangle}_{+D_{\Omega,R_x,L}}(K_x).$$

The classical RCA algorithm, that is called here <u>RCA</u>, thus relies on FCA and σ_Ω. More precisely, it applies these in parallel on all contexts. Hence, FCA* and σ^*_Ω are defined as:

$$\mathrm{FCA}^*(\{K_x\}_{x\in X}) = \{\mathrm{FCA}(K_x)\}_{x\in X},$$
$$\sigma^*_\Omega(\{K_x\}_{x\in X}, R, L) = \{\sigma_\Omega(K_x, R, L)\}_{x\in X}.$$

[2]We use the term 'relational context' instead of 'relational context family'.

such that L is a family of concept lattices. The whole family of concepts lattices needs to be passed to σ.

RCA starts from the initial family of contexts K^0 and iterates the application of the two operations:

$$K^{t+1} = \sigma^*_\Omega(K^t, R, \text{FCA}^*(K^t))$$

until reaching a fixed point, i.e. reaching n such that $K^{n+1} = K^n$. Then, $\underline{\text{RCA}}_\Omega(K^0, R) = \text{FCA}^*(K^n)$.

Thus, the RCA algorithm proceeds in the following way:

(1) Initial contexts: $t \longleftarrow 0$; $\{\langle G_x, M^t_x, I^t_x\rangle\}_{x\in X} \longleftarrow \{\langle G_x, M_x, I_x\rangle\}_{x\in X}$.
(2) $\{L^t_x\}_{x\in X} \longleftarrow \text{FCA}^*(\{\langle G_x, M^t_x, I^t_x\rangle\}_{x\in X})$ (or, for each context, $\langle G_x, M^t_x, I^t_x\rangle$ the corresponding concept lattice $L^t_x = \text{FCA}(\langle G_x, M^t_x, I^t_x\rangle)$ is created using FCA).
(3) $\{\langle G_x, M^{t+1}_x, I^{t+1}_x\rangle\}_{x\in X} \longleftarrow \sigma^*_\Omega(\{\langle G_x, M^t_x, I^t_x\rangle\}_{x\in X}, R, \{L^t_x\}_{x\in X})$ (i.e. relational scaling is applied, for each relation r whose codomain lattice has new concepts, generating new contexts $\langle G_x, M^{t+1}_x, I^{t+1}_x\rangle$ including both plain and relational attributes in M^{t+1}_x).
(4) If $\exists x \in X$ such that $M^{t+1}_x \neq M^t_x$ (scaling has occurred), then $t \longleftarrow t + 1$; go to Step 2.
(5) Return $\{L^t_x\}_{x\in X}$.

This is illustrated by Example 6.

Example 6 (Relational concept analysis). *Consider two relations p and q defined as:*

p	d	e	f
a	×		
b		×	
c			×

q	a	b	c
d	×		
e		×	
f			×

and applying on the contexts K^0_1 of Example 1 and K^0_2 of Example 3 (Figure 1).

Applying FCA* *to the two contexts K^0_1 and K^0_2, provides the very simple lattices L^0_1 and L^0_2 of Figure 1 with concepts ABC, AB, C and $\bot$, and DEF, DE and E, respectively. Applying scaling, as seen partially in Example 3, provides the context K^1_1 with new attributes $\exists p$.DEF, $\exists p$.DE, $\exists p$.E and K^1_2 with $\exists q$.ABC, $\exists q$.AB, $\exists q$.C (Example 4). Applying* FCA* *to these contexts provides the lattices L^1_1 and L^1_2 with additional concepts B and F. These can in turn go through scaling and unveil new attributes $\exists p$.F and $\exists q$.B added to K^1_1 and K^1_2 to give K^2_1 and K^2_2.* FCA* *introduces the new attributes in the intent of relevant concepts but does not introduce any new concept. Hence the process stops and* RCA *returns the family of concept lattices $\{L^2_1, L^2_2\}$.*

The result is thus quite different from the $\{L^0_1, L^0_2\}$ that would have been returned by FCA alone.

2.4.3 Properties and Semantics. A context $K = \langle G, M, I\rangle$ is a subcontext of another $K' = \langle G', M', I'\rangle$ whenever $G \subseteq G'$, $M \subseteq M'$ and $I = I' \cap (G \times M)$ [16, §3.1]. By abuse of notation, this is noted $K \subseteq K'$. This is generalised to families of contexts $\{K_x\}_{x\in X} \subseteq \{K'_x\}_{x\in X}$ whenever $\forall x \in X, K_x \subseteq K'_x$.

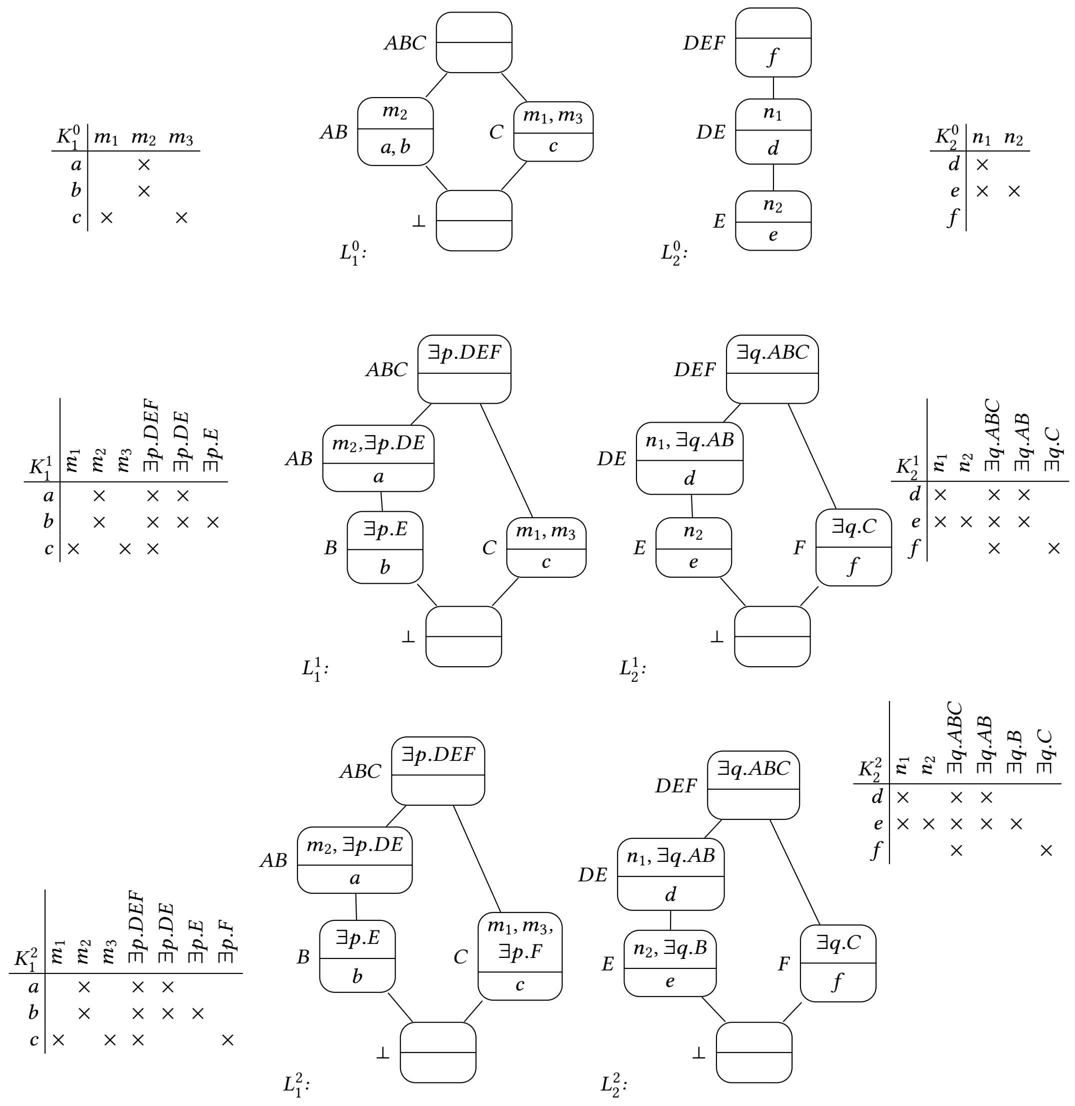
RCA always reaches a closed family of contexts for reason of finiteness [26] and the sequence $(K^t)^n_{t=0}$ is non-(intent-)contracting, i.e. $\forall t \geq 0, K^t \subseteq K^{t+1}$ [27].

The RCA semantics characterises the set of concepts in resulting RCA lattices as all and only those concepts grounded on the initial family (K^0) based on relations (R) [27]. This can thus be considered as a well-grounded semantics: an attribute is scaled and applied to an object at iteration $t + 1$ only if its condition applies at stage t. Hence, everything is ultimately relying on K^0.

[27] established that RCA indeed finds *the* K^n satisfying these constraints through correctness (the concepts of FCA$^*(K^n)$ are grounded in K^0 through R) and completeness (all so-grounded concepts are in K^n).

2.5 Dependencies and Cycles

As can be seen, relations in RCA define a dependency graph between objects (of different or the same context). In turn, this graph of objects induces a dependency graph between concepts through the scaled attributes that refer



K_1^0	m_1	m_2	m_3
a		×	
b		×	
c	×		×

K_2^0	n_1	n_2
d	×	
e	×	×
f		

K_1^1	m_1	m_2	m_3	$\exists p.DEF$	$\exists p.DE$	$\exists p.E$
a		×		×	×	
b		×		×	×	×
c	×		×	×		

K_2^1	n_1	n_2	$\exists q.ABC$	$\exists q.AB$	$\exists q.C$
d	×		×	×	
e	×	×	×	×	
f			×		×

K_1^2	m_1	m_2	m_3	$\exists p.DEF$	$\exists p.DE$	$\exists p.E$	$\exists p.F$
a		×		×	×		
b		×		×	×	×	
c	×		×	×			×

K_2^2	n_1	n_2	$\exists q.ABC$	$\exists q.AB$	$\exists q.B$	$\exists q.C$
d	×		×	×		
e	×	×	×	×	×	
f			×			×

Fig. 1. The three iterations of RCA from the initial contexts K_1^0 and K_2^0.

to other concepts. It also induces a dependency graph between contexts: an edge exists between two contexts if an object of the former is related to an object of the latter.

This paper is related to the circular dependencies, i.e. the circuits, that may exist within these graphs. Circular dependencies create a problem when one wants to define the family of concept lattices that should be returned by relational concept analysis. As will be seen in Section 3, there may exists several such families.

3 Motivating Examples

In order to illustrate the weakness of the RCA semantics, we first carry on the introductory Examples 1–6 (§3.1). We then display it on a more minimal example that will be carried over the paper (§3.2).

From such a simple basis, it is possible to consider more complex settings:

- By using more than two contexts;
- By using more than two relations between these contexts;
- By using more than two objects in each context;
- By using more than zero attributes in the contexts.

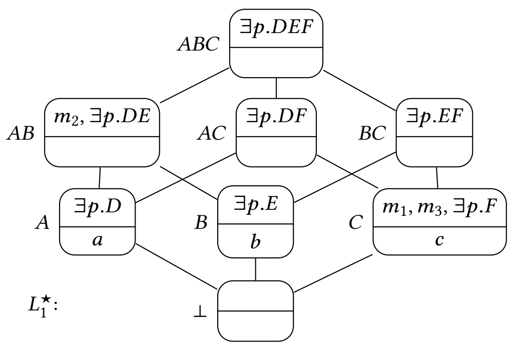
3.1 RCA May Accept Different Families of Concept Lattices

The simple Example 6 (Figure 1) does not present a result in which each object is identified by a single class. Indeed, a has not more attributes than b. This result could also be obtained with far more objects a', a'', etc. sharing the attributes of a and b, or duplicating other objects.

However, the lattices $L_1^\star$ and $L_2^\star$ displayed in Figure 2 seem another good way to describe the data given as input to RCA. There is in fact an objective difference between AC and BC: AC denotes all objects connected by p to DF and BC all objects connected by p to EF. Reciprocally, DF denotes all objects connected by q to AC and EF those connected by q to BC.

$K_1^\star$	m_1	m_2	m_3	$\exists p.DEF$	$\exists p.DE$	$\exists p.DF$	$\exists p.EF$	$\exists p.D$	$\exists p.E$	$\exists p.F$
a		×		×	×	×		×		
b		×		×	×		×		×	
c	×		×	×		×	×			×

$K_2^\star$	n_1	n_2	$\exists q.ABC$	$\exists q.AB$	$\exists q.AC$	$\exists q.BC$	$\exists q.A$	$\exists q.B$	$\exists q.C$
d	×		×	×	×		×		
e	×	×	×	×		×		×	
f			×		×	×			×



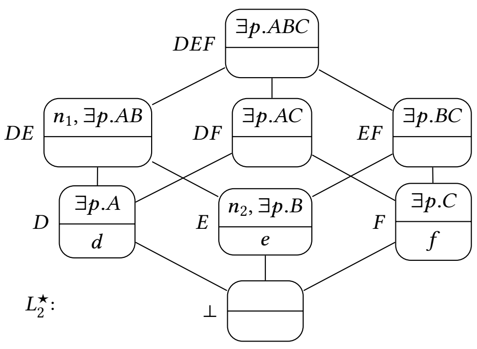


Fig. 2. Alternative concept lattices for the example of Section 3.1 (★ is simply a way to identify these objects).

These lattices share many common points with those returned by RCA: they are also (*i*) valid concept lattices, (*ii*) whose contexts extend K_1^0 and K_2^0 with attributes of $D_{\{\exists\},\{p\},K^0}$ and $D_{\{\exists\},\{q\},K^0}$ (Example 4), (*iii*) stable for scaling, and (*iv*) such that each attribute refers only to concepts in the lattices. The only difference with L_1^2 and L_2^2 is that they are not those returned by RCA. We will temporarily informally consider lattices sharing these features as *acceptable*.

But, if there exists several acceptable solutions for a given Ω and R, why does RCA only returns one of these, and which one? To help answering this question, we illustrate the problem with a minimal running example below.

3.2 Minimal RCA Example

As another example, consider the following two empty contexts K_3^0 and K_4^0 of Figure 4 and the two relations p and q of Figure 3.

p	c	d
a	×	
b		×

q	a	b
c	×	
d		×

Fig. 3. Relations p and q for RCA.

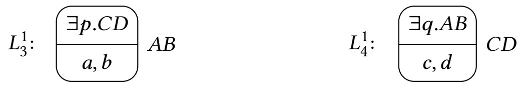
Applying FCA to the two contexts K_3^0 and K_4^0 provides the very simple lattices L_3^0 and L_4^0 of Figure 4. From this, RCA generates new context K_3^1 and K_4^1 through scaling which provides new lattices L_3^1 and L_4^1 (Figure 4).

K_3^0
a
b

L_3^0:
a, b
AB
L_4^0:
c, d
CD

K_4^0
c
d

K_3^1	$\exists p.CD$
a	×
b	×



K_4^1	$\exists q.AB$
c	×
d	×

Fig. 4. The two iterations of RCA from the initial contexts K_3^0 and K_4^0.

The lattices L_3^1 and L_4^1 of Figure 4 are those returned by RCA as applying scaling from them returns the same contexts K_3^1 and K_4^1.

However, there could be other acceptable solutions such as those displayed in Figure 5. They are all acceptable solutions for $\{K_3^0, K_4^0\}$ (Figure 4) as they satisfy the four conditions of Section 3.1.

On the contrary, Figure 6 displays a family of concept lattices $\{L_3^\#, L_4^\#\}$ which is not an acceptable solution. Although they contain all concepts of $\{L_3^0, L_4^0\}$ and no concept not in $\{L_3^\star, L_4^\star\}$, they would generate more attributes through scaling and applying RCA to their contexts $\{K_3^\#, K_4^\#\}$ would lead to $\{L_3^\star, L_4^\star\}$

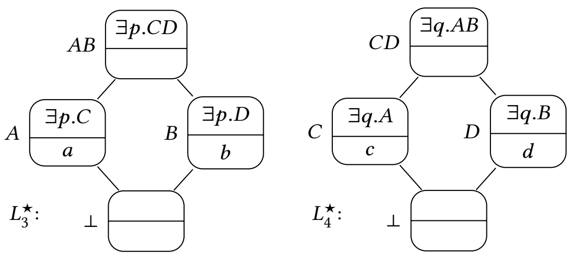
Hence the question raised in the previous section: Why does RCA return only one solution, and which one? Answering it requires to reconsider the RCA semantics. More precisely, it requires to define formally which families of concept lattices could be considered as acceptable solutions and which of them is returned by the RCA operation.

We will thus redefine the objects on which RCA operates (§4), precising well-formedness, and introduce two functions on this space (§5) from whose fixed points acceptability can be defined (§6), offering a fixed-point semantics for relational concept analysis (§7).

4 Dual Context-Lattice Space

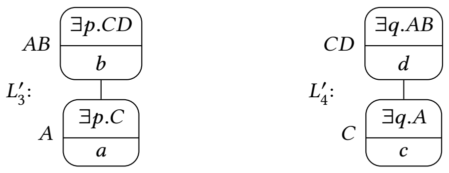
In order to investigate the semantics of relational concept analysis, we need to define the objects on which it applies. They are determined by three elements given once and for all: $K^0 = \{\langle G_x, M_x^0, I_x^0\rangle\}_{x\in X}$, R, and Ω. Through the application of *RCA*, only M_x^t and I_x^t change, hence the other may remain implicit.

$K_3^\star$	$\exists p.CD$	$\exists p.C$	$\exists p.D$
a	×	×	
b	×		×



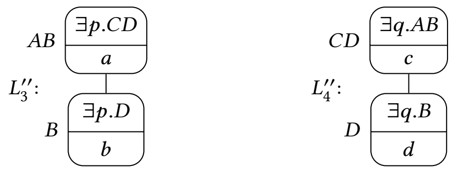
$K_4^\star$	$\exists q.AB$	$\exists q.A$	$\exists q.B$
c	×	×	
d	×		×

K_3'	$\exists p.CD$	$\exists p.C$
a	×	×
b	×	



K_4'	$\exists q.AB$	$\exists q.A$
c	×	×
d	×	

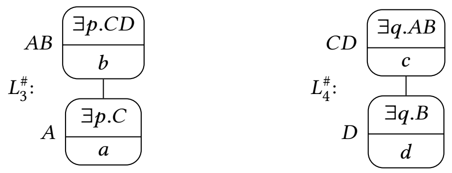
K_3''	$\exists p.CD$	$\exists p.D$
a	×	
b	×	×



K_4''	$\exists q.AB$	$\exists q.B$
c	×	
d	×	×

Fig. 5. Alternative pairs of concept lattices covering the contexts of Figure 4.

$K_3^\#$	$\exists p.CD$	$\exists p.C$
a	×	×
b	×	



$K_4^\#$	$\exists q.AB$	$\exists q.B$
c	×	
d	×	×

Fig. 6. A family of concept lattices $\{L_3^\#, L_4^\#\}$ which is not an acceptable solution.

These objects are introduced progressively by considering in a row: contexts (§4.1), lattices (§4.2), context-lattice pairs (§4.3) and indexed families of context-lattice pairs (§4.4). The first three subsections will only consider one context K_x, one concept lattice L_x and one context-lattice pair T_x at a time; the fourth section will consider them together.

Hereafter, we use a simple and regular notation: K denotes contexts, L lattices, T context-lattice pairs and O families of context-lattice pairs. In addition, R is used for relations, N for names and D for attributes. Greek letters are devoted to some auxiliary functions (η, κ, σ, π). E, F, P and Q are used for functions names and * is used for distinguishing the parallel application of such functions.

4.1 The Space of Contexts $\mathscr{K}$

We first study the semantics of RCA from the standpoint of the contexts. The contexts considered by RCA are scaled from an initial context using the scaling operations. Property 1 highlights that the incidence of the scaled attributes does only depend on the relation and not on the concept lattices.

Property 1 (The incidence relation depends only on the attributes). *Given a set Ω of relational scaling operations, a family $K = \{\langle G_x, M_x, I_x\rangle\}_{x\in X}$ of formal contexts and a set R of relations on K. Given L_z and L'_z two concept lattices on G_z, for all scaled attributes $m \in D_{\Omega,R_{x,z},L_z} \cap D_{\Omega,R_{x,z},L'_z}$, $\forall g \in G_x$, $\langle R, L_z\rangle \models m(g)$ if and only if $\langle R, L'_z\rangle \models m(g)$.*

Proof. $m = \varsigma(r, c)$ is scaled from a scaling operation ς, a relation r and a concept c (possibly a cardinal n). From Table 1, $\langle R, L_z\rangle \models gIm$ only depends on ς, r and the extent of c. However, ς and r are independent from L_z and L'_z. The concept c is identified by a name which denotes its extent. Hence, its extent is the same in L_z and L'_z. So whether an object of $g \in G_x$ satisfies the attribute $\varsigma(r, c)$ or not depends solely on the attribute $\varsigma(r, c)$ and not on the specific lattice considered. □

This means that the interpretation of an attribute never changes: it is given by its syntactic form $\varsigma(r, c)$ and, in particular, the concept name. Hence, when adding or suppressing attributes, through K^{Σ}_{+M} or K^{Σ}_{-M}, Σ can be $\langle R, N\rangle$ with N an indexed set of names, and in particular $\eta^*(K^0)$.

In RCA, the set of objects G_x does not change and for any object and attribute, either the object satisfies the attribute or not. Property 1 entails that, if $M_x \subseteq M'_x$, then $I_x \subseteq I'_x$. Thus we are justified in using, for RCA, the definition of subcontexts introduced in Section 2.4.3. For comparing two contexts, it suffices to compare their sets of attributes: if $M_x \subseteq M'_x$, then $K_x \subseteq K'_x$.

The attribute language $D_{\Omega,R_x,N}$ that can be generated by scaling depends on the finite set of relations R, the scaling operations Ω and the set of possible concepts identified by their standardised names (§2.2.2 and 2.4.1). Given $N \subseteq \eta^*(K^0)$ an indexed family of names of concepts that can be in the codomain of relations in R, the set of contexts that can be obtained by scaling is

$$\mathscr{K}^N_{K^0_x,R,\Omega} = \{\mathrm{K}^{\langle R,\eta^*(K^0)\rangle}_{+M}(K^0_x) \mid M \subseteq D_{\Omega,R_x,N}\}$$

with $\mathrm{K}^{\langle R,\eta^*(K^0)\rangle}_{+M}(.)$ the operation defined in §2.2. Passing $\eta^*(K^0)$ to K allows to interpret the generated attributes and to determine I.

Below, when we write $\mathscr{K}^N$, the property applies for any $\{N_x\}_{x\in X}$ such that $\forall x \in X$, $N_x \subseteq \eta(K^0_x)$, and in particular for $\eta^*(K^0)$.

We can define two operations, $\wedge$ and $\vee$ on $\mathscr{K}^N_{K^0_x,R,\Omega}$.

Definition 1 (Meet and join of contexts). *Given $K, K' \in \mathscr{K}^N_{\langle G,M^0,I^0\rangle,R,\Omega}$, such that $K = \langle G, M^0 \cup M, I^0 \cup I\rangle$ and $K' = \langle G, M^0 \cup M', I^0 \cup I'\rangle$, $K \vee K'$ and $K \wedge K'$ are defined as:*

$$K \vee K' = \langle G, M^0 \cup (M \cup M'), I^0 \cup (I \cup I')\rangle, \quad \text{(join)}$$
$$K \wedge K' = \langle G, M^0 \cup (M \cap M'), I^0 \cup (I \cap I')\rangle. \quad \text{(meet)}$$

The set of contexts is closed by meet and join.

Property 2 ([10]). *$\forall K, K' \in \mathscr{K}^N_{K^0_x,R,\Omega}$, $K \wedge K' \in \mathscr{K}^N_{K^0_x,R,\Omega}$ and $K \vee K' \in \mathscr{K}^N_{K^0_x,R,\Omega}$.*

Proof. Meet and join are defined from the union and intersection of subsets of D_{Ω,R_x,K^0} (Definition 1). But $\mathscr{K}^N_{K^0_x,R,\Omega}$ is closed by union and intersection of the sets of attributes to add to M^0 and the incidence relation is fully determined by the set of attributes (Property 1). Hence, meet and join of contexts in $\mathscr{K}^N_{K^0_x,R,\Omega}$ belong to $\mathscr{K}^N_{K^0_x,R,\Omega}$. □

4.2 The Space of Lattices $\mathscr{L}$

From $\mathscr{K}^N_{K^0_x,R,\Omega}$, one can define $\mathscr{L}^N_{K^0_x,R,\Omega}$ as the set of images of $\mathscr{K}^N_{K^0_x,R,\Omega}$ by FCA. These are concept lattices obtained by applying FCA on K^0_x extended with a subset of $D_{\Omega,R_x,N}$:

$$\mathscr{L}^N_{K^0_x,R,\Omega} = \{\mathrm{FCA}(K^{\langle R,\eta^*(K^0)\rangle}_{+M}(K^0_x)) \mid M \subseteq D_{\Omega,R_x,N}\}.$$

For each subset of attributes, the lattice obtained by FCA is necessarily syntactically different as its concepts refer to different attributes in their intents (at least one of them).

There is in fact a bijective correspondence between $\mathscr{L}_{K^0_x,R,\Omega}$ and $\mathscr{K}_{K^0_x,R,\Omega}$ [16, §1.2]. On the one hand, to any context in $\mathscr{K}_{K^0_x,R,\Omega}$ corresponds only one lattice by FCA. On the other hand, to any concept lattice $\langle C, \leq\rangle \in \mathscr{L}_{K^0_x,R,\Omega}$ corresponds a formal context:

$$\kappa(\langle C,\leq\rangle) = \langle \bigcup_{c\in C} extent(c), \bigcup_{c\in C} intent(c), \bigcup_{c\in C} extent(c) \times intent(c)\rangle.$$

$\kappa(L)$ collects the attributes and objects present in L intents to build the unique M and G, from which the corresponding I is obtained.

It is such that $\mathrm{FCA} \circ \kappa = id_{\mathscr{K}}$ and $\kappa \circ \mathrm{FCA} = id_{\mathscr{L}}$ ($id_{\mathscr{K}}$ and $id_{\mathscr{L}}$ being the respective identity functions). This may be stated as:

Property 3. *$K = \kappa(L)$ iff $L = \mathrm{FCA}(K)$.*

Proof. $\Rightarrow$) $K = \langle G, M, I\rangle = \langle \bigcup_{c\in C} extent(c), \bigcup_{c\in C} intent(c), \bigcup_{c\in C} extent(c) \times intent(c)\rangle = \kappa(\langle C, \leq\rangle) = \kappa(L)$. This means that $\langle C, \leq\rangle$ is the concept lattice of a context $\langle G', M', I'\rangle$. But since $G = \bigcup_{c\in C} extent(c)$ and $M = \bigcup_{c\in C} intent(c)$, then $G = G'$ and $M = M'$. We need to prove that $I = I'$. Consider $I \neq I'$, this could be because there exists at least one $g \in G$ and $m \in M$ such that $\langle g, m\rangle \in I \setminus I'$ (or, but not exclusively, $\langle g, m\rangle \in I' \setminus I$). In this condition, there could not exists $c \in C$ such that $g \in extent(c)$ and $m \in intent(c)$ (resp. it exists). Then $\langle g, m\rangle \notin \bigcup_{c\in C} extent(c) \times intent(c)$ (resp. on the contrary it is there) and thus $I \neq \bigcup_{c\in C} extent(c) \times intent(c)$ which contradicts $\langle G, M, I\rangle = \kappa(\langle C, \leq\rangle)$. Hence, $I = I'$ and $L = \langle C, \leq\rangle = \mathrm{FCA}(\langle G, M, I\rangle) = \mathrm{FCA}(K)$

$\Leftarrow$) $L = \langle C, \leq\rangle = \mathrm{FCA}(\langle G, M, I\rangle) = \mathrm{FCA}(K)$ entails $\forall c \in C, \forall g \in extent(c), \forall m \in intent(c), gIm$, i.e. $extent(c) \times intent(c) \subseteq I$. In addition, if gIm, then there exists $c \in \mathrm{FCA}(\langle G, M, I\rangle) = \langle C, \leq\rangle$ such that $g \in extent(c)$ and $m \in intent(c)$, thus $I \subseteq \bigcup_{c\in C} extent(c) \times intent(c)$. Moreover, $\top = \langle G, G^{\uparrow}\rangle \in C$ and $\bot = \langle M^{\downarrow}, M\rangle \in C$, hence $\bigcup_{c\in C} extent(c) = G$ and $\bigcup_{c\in C} intent(c) = M$. Thus $K = \langle G, M, I\rangle = \langle \bigcup_{c\in C} extent(c), \bigcup_{c\in C} intent(c), \bigcup_{c\in C} extent(c) \times intent(c)\rangle = \kappa(L)$. □

We define a specific type of homomorphisms between two concept lattices when concepts are simply mapped into concepts with the same extent and a possibly increased intent[3].

Definition 2 (Lattice homomorphism [10]). *A concept lattice homomorphism $h : \langle C, \leq\rangle \to \langle C', \leq'\rangle$ is a function which maps each concept $c \in C$ into a corresponding concept $h(c) \in C'$ such that:*

- *$\forall c \in C$, $intent(c) \subseteq intent(h(c))$,*
- *$\forall c \in C$, $extent(c) = extent(h(c))$, and*
- *$\forall c, d \in C$, $c \leq d \Rightarrow h(c) \leq' h(c)$.*

[3]The results in the remainder of this section are specific to RCA: $\preceq$ is defined with the equality of extents and $\subseteq$ depends only on M because G is always the same.

We note $L \preceq L'$ if there exists a homomorphism from L to L'. In principle, $L \simeq L'$ if $L \preceq L'$ and $L' \preceq L$, but here, $\simeq$ is simply $=$. This owes to the fact that the homomorphism maps concepts of equal extent, hence, if they hold in both ways, there should be as many concepts in each lattice and these concepts will also have the same intent.

Property 4 (FCA is monotone). *$\forall K = \langle G, M, I\rangle$ and $K' = \langle G, M', I'\rangle$,*

$$K \subseteq K' \Rightarrow \text{FCA}(K) \preceq \text{FCA}(K').$$

Proof. Any concept $c \in \text{FCA}(K)$ characterises a set of objects $extent(c)$ by the set of attributes $intent(c)$ that these objects are the only ones to satisfy. Since both contexts have the same set of objects G, there are not more objects satisfying these in K' and since $M \subseteq M'$ these attributes are still in K', thus $c \in \text{FCA}(K')$. Hence, it is always possible to define h such as $h(c) = c$. Then $extent(h(c)) = extent(c)$ and $intent(h(c)) \supseteq intent(c)$. Moreover, if $c \leq d$, then $h(c) \leq h(d)$ because $extent(c) \subseteq extent(d)$ entails $extent(h(c)) \subseteq extent(h(d))$. Hence, $\text{FCA}(K) \preceq \text{FCA}(K')$ (Definition 2). □

If $K \subseteq K'$, then each concept that can be built from K can be built from K'. The additional attributes in M' can only be used to separate further objects of existing concepts, introducing additional concepts. All concepts are preserved, possibly with a larger intent, which preserves the homomorphism.

We can define $\wedge$ and $\vee$ on $\mathscr{L}^N_{K^0_x,R,\Omega}$ from the corresponding operators in $\mathscr{K}^N_{K^0_x,R,\Omega}$.

Definition 3 (Meet and join of lattices). *Given $L, L' \in \mathscr{L}^N_{K^0_x,R,\Omega}$,*

$$L \vee L' = \text{FCA}(\kappa(L) \vee \kappa(L')), \quad \text{(join)}$$

$$L \wedge L' = \text{FCA}(\kappa(L) \wedge \kappa(L')). \quad \text{(meet)}$$

The set of lattices is also closed by meet and join:

Property 5. *$\forall L, L' \in \mathscr{L}^N_{K^0_x,R,\Omega}$, $L \wedge L' \in \mathscr{L}^N_{K^0_x,R,\Omega}$ and $L \vee L' \in \mathscr{L}^N_{K^0_x,R,\Omega}$.*

Proof. $\mathscr{L}^N_{K^0_x,R,\Omega}$ is closed by meet and join since $\mathscr{K}^N_{K^0_x,R,\Omega}$ is closed by meet and join (Property 2) and $\mathscr{L}^N_{K^0_x,R,\Omega}$ is the image of $\mathscr{K}^N_{K^0_x,R,\Omega}$ by FCA. □

4.3 The Lattice $\mathscr{T}$ of Context-Lattice Pairs

Although $\mathscr{K}^N$ and $\mathscr{L}^N$ have been presented independently, it is useful to consider the two sets together as, in RCA, lattices in $\mathscr{L}^N$ are an intermediate result of the process which is used for computing the next contexts. Instead of dealing with two interrelated spaces independently, we tightly connect them. Doing so, we will consider objects which are pairs of contexts and associated concept lattices through FCA. They are called context-lattice pairs.

From any context in $\mathscr{K}$, it is possible to generate a context-lattice pair using FCA. The T constructor does this.

Definition 4 (T constructor). *Given a context $K \in \mathscr{K}^N$, $\text{T} : \mathscr{K}^N \longrightarrow \mathscr{K}^N \times \mathscr{L}^N$ generates a context-lattice pair, such that:*

$$\text{T}(K) = \langle K, \text{FCA}(K)\rangle.$$

We consider the set $\mathscr{T}^N_{K^0_x,R,\Omega}$ of pairs in $\mathscr{K}^N_{K^0_x,R,\Omega} \times \mathscr{L}^N_{K^0_x,R,\Omega}$ such that:

$$\mathscr{T}^N_{K^0_x,R,\Omega} = \{\langle K, L\rangle \in \mathscr{K}^N_{K^0_x,R,\Omega} \times \mathscr{L}^N_{K^0_x,R,\Omega} | L = \text{FCA}(K)\}.$$

This set is well defined because $\mathscr{K}^N_{K^0_x,R,\Omega}$ has already been defined and $\mathscr{L}^N_{K^0_x,R,\Omega}$ are precisely those lattices obtained by FCA from an element of $\mathscr{K}^N_{K^0_x,R,\Omega}$.

Alternatively, using Property 3, it can be defined from κ:

$$\mathscr{T}^N_{K^0_x,R,\Omega} = \{\langle K, L\rangle \in \mathscr{K}^N_{K^0_x,R,\Omega} \times \mathscr{L}^N_{K^0_x,R,\Omega} | K = \kappa(L)\}.$$

As before, we use $\mathscr{T}_{K^0_x,R,\Omega} = \mathscr{T}^{\eta^*(K^0)}_{K^0_x,R,\Omega}$ and, for any $\langle K, L\rangle \in \mathscr{T}^N_{K^0_x,R,\Omega}$ we note:

$$k(\langle K, L\rangle) = K,$$
$$l(\langle K, L\rangle) = L.$$

It is possible to define the meet and join:

Definition 5 (Meet and join of context-lattice pairs). *Given $T, T' \in \mathscr{T}^N_{K^0_x,R,\Omega}$ $T \vee T'$ and $T \wedge T'$ are defined as:*

$$T \vee T' = \mathrm{T}(k(T) \vee k(T')), \tag{join}$$
$$T \wedge T' = \mathrm{T}(k(T) \wedge k(T')). \tag{meet}$$

As this definition makes clear, the operations of $\mathscr{T}^N$ only depend on the context part. But the usual relations with the meet and join on contexts and lattices are preserved:

Property 6.

$$T \vee T' = \langle k(T) \vee k(T'), l(T) \vee l(T')\rangle,$$
$$T \wedge T' = \langle k(T) \wedge k(T'), l(T) \wedge l(T')\rangle.$$

Proof. This is a simple consequence on the definition of conjunction and disjunction on context-lattice pairs (Definition 4) and lattices (Definition 3) as:

$$L \vee L' = \mathrm{FCA}(K \vee K'), \tag{join}$$
$$L \wedge L' = \mathrm{FCA}(K \wedge K'). \tag{meet}$$

□

The set of context-lattice pairs is closed by meet and join:

Property 7. $\forall T, T' \in \mathscr{T}^N_{K^0_x,R,\Omega}$, $T \wedge T' \in \mathscr{T}^N_{K^0_x,R,\Omega}$ *and* $T \vee T' \in \mathscr{T}^N_{K^0_x,R,\Omega}$.

Proof. $\mathscr{T}^N_{K^0_x,R,\Omega}$ is closed by meet and join because it is based on T (Definition 5), T builds a context-lattice pair in $\mathscr{T}^N_{K^0_x,R,\Omega}$ from contexts in $\mathscr{K}^N_{K^0_x,R,\Omega}$ (Definition 4), and $\mathscr{K}^N_{K^0_x,R,\Omega}$ itself is closed by meet and join (Property 2). □

We also define the order between two context-lattice pairs by combining the orders on contexts and lattices:

Definition 6 (Order). *Given $T, T' \in \mathscr{T}^N_{K^0_x,R,\Omega}$,*

$$T \preceq T' \textit{ if } k(T) \subseteq k(T') \textit{ and } l(T) \preceq l(T').$$

Figure 7 presents the relations between $\mathscr{K}^N$, $\mathscr{L}^N$ and $\mathscr{T}^N$ and their respective orders. Since FCA is monotone (Property 4), $T \preceq T'$ iff $k(T) \preceq k(T')$. Like before, we note $T \simeq T'$ if $T \preceq T'$ and $T' \preceq T$, and again $\simeq$ is $=$.

This can be applied to the T constructor.

Property 8. $\forall K, K' \in \mathscr{K}^N_{K^0_x,R,\Omega}$, $K \subseteq K'$ *iff* $\mathrm{T}(K) \preceq \mathrm{T}(K')$.

Proof. ($\Rightarrow$) This is due to monotony of FCA (Property 4). $k(\mathrm{T}(K)) = K \subseteq K' = k(\mathrm{T}(K'))$ means that $l(\mathrm{T}(K)) = \mathrm{FCA}(K) \preceq \mathrm{FCA}(K') = l(\mathrm{T}(K'))$. Thus, $\mathrm{T}(K) \preceq \mathrm{T}(K')$. ($\Leftarrow$) $\mathrm{T}(K) \preceq \mathrm{T}(K')$ entails, by Definition 6, that $K \subseteq K'$. □

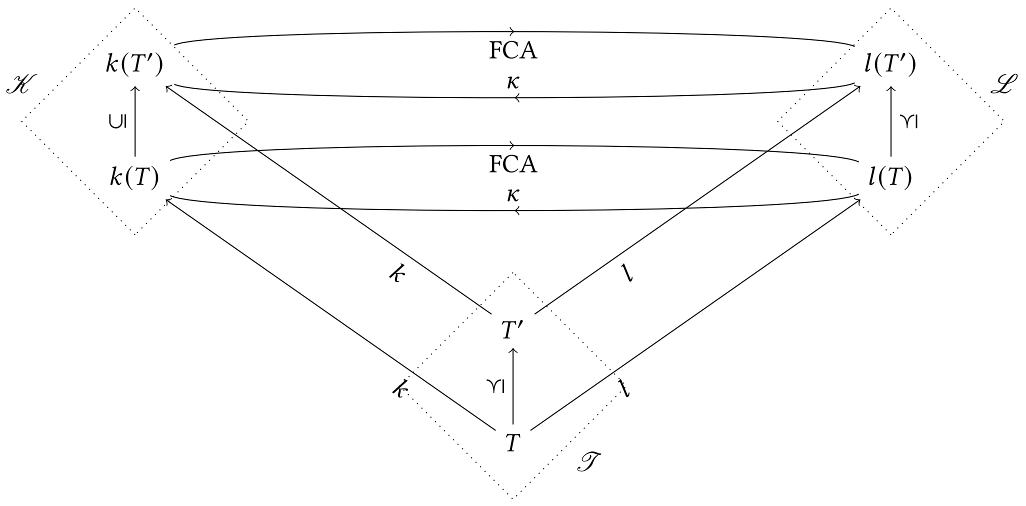


Fig. 7. Relations between $\mathscr{T}$, $\mathscr{K}$ and $\mathscr{L}$.

This has the consequence that $T = T'$ if and only if $k(T) = k(T')$.

Joining together contexts and lattices was a preliminary step to consider families of such pairs to represent the behaviour of relational concept analysis as a whole. This is done hereafter.

4.4 The Lattice $\mathscr{O}$ of Families of Context-Lattice Pairs

So far, we have only considered one context independently from the others. We now consider relational concept analysis in its entirety.

RCA deals with families of contexts. Its elements are thus simple vectors of the pairs generated by each context. These vectors will be considered as sets indexed by X. All provided definitions can be applied to indexed families of context-lattice pairs, the order between them will be the product of the piece-wise orders. The only important change is that N will be frozen to the set of concept names in all the different contexts in K^0.

The input of RCA is given by a family of contexts: $K^0 = \{K^0_x\}_{x \in X}$, a set R of relations between the objects of these contexts, and a set Ω of relational scaling operations. From this, it is possible to characterise the space $\mathscr{O}_{K^0,R,\Omega}$ associated with RCA by the direct product of the sets of context-lattice pairs associated with each context.

Definition 7 ($\mathscr{O}_{K^0,R,\Omega}$). *Given an indexed family of contexts $K^0 = \{\langle G_x, M^0_x, I^0_x \rangle\}_{x \in X}$, a set R of relations between the objects of these contexts, and a set Ω of relational scaling operations, the space $\mathscr{O}_{K^0,R,\Omega}$ of indexed families of context-lattice pairs is:*

$$\mathscr{O}_{K^0,R,\Omega} = \prod_{x \in X} \mathscr{T}^{\eta^*(K^0)}_{K^0_x,R,\Omega}.$$

As usual, $\mathscr{O}_{K^0,R,\Omega}$ will simply be referred to as $\mathscr{O}$.

This is well defined because the set of all possible concept extents across all contexts is determined by the set of objects in the context. This permits us to name unambiguously all the concepts in the family of concept lattices. In turn, since $\eta^*(K^0)$, R and Ω do not change and I is determined by $\{M_x\}_{x \in X}$ (Property 1), this determines all attributes that can occur in a scaled RCA context.

Contrary to RCA^0 [10], the scaled attributes depend on R that makes the connection from one context to another, e.g. from $\mathscr{T}_x$ to $\mathscr{T}_z$. But since it is possible to name concepts in the lattices generated by the scaled attributes according to their elements in G_z, then, as soon as G_z is finite, the set of scalable attributes in M_x is finite and can be established as D_{Ω,R_x,K^0} from the beginning.

The previous notations can be extended:

$$\mathrm{T}^*(\{K_x\}_{x\in X}) = \{\mathrm{T}(K_x)\}_{x\in X} = \{\langle K_x, \mathrm{FCA}(K_x)\rangle\}_{x\in X}.$$

For any $\{T_x\}_{x\in X} \in \mathscr{O}_{K^0,R,\Omega}$:

$$\begin{aligned} k(\{T_x\}_{x\in X}) &= \{k(T_x)\}_{x\in X},\\ l(\{T_x\}_{x\in X}) &= \{l(T_x)\}_{x\in X},\\ k_z(\{T_x\}_{x\in X}) &= k(T_z),\\ l_z(\{T_x\}_{x\in X}) &= l(T_z). \end{aligned}$$

Finally, for any indexed family of context-lattice pairs $O \in \mathscr{O}$, the family of lattices $l(O)$ is determined directly from $k(O)$: $l(O) = \mathrm{FCA}^*(k(O))$.

We can define $\wedge$ and $\vee$ on $\mathscr{O}_{K^0,R,\Omega}$.

Definition 8 (Meet and join of families of context-lattice pairs). *Given* $O = \{T_x\}_{x\in X}$, $O' = \{T'_x\}_{x\in X} \in \mathscr{O}_{K^0,R,\Omega}$, $O \vee O'$ *and* $O \wedge O'$ *are defined as:*

$$O \vee O' = \{T_x \vee T'_x\}_{x\in X}, \quad \text{(join)}$$

$$O \wedge O' = \{T_x \wedge T'_x\}_{x\in X}. \quad \text{(meet)}$$

The set of families of context-lattice pairs is once again closed by meet and join:

Property 9. $\forall O, O' \in \mathscr{O}^N_{K^0,R,\Omega}$, $O \wedge O' \in \mathscr{O}^N_{K^0,R,\Omega}$ *and* $O \vee O' \in \mathscr{O}^N_{K^0,R,\Omega}$.

Proof. $\mathscr{O}_{K^0,R,\Omega}$ is closed by meet and join because meet and join are the piecewise meet and join of context-lattice pairs (Definition 8) and for each $x \in X$, $\mathscr{T}^{\eta^*(K^0)}_{K^0_x,R,\Omega}$ is closed by meet and join (Property 7). □

We also define the order between two objects by combining the previous definitions.

Definition 9 (Order). *Given* $O = \{T_x\}_{x\in X}$, $O' = \{T'_x\}_{x\in X} \in \mathscr{O}_{K^0,R,\Omega}$,

$$O \preceq O' \textit{ if } \forall x \in X, T_x \preceq T'_x.$$

Like before, we note $O \simeq O'$ if $O \preceq O'$ and $O' \preceq O$, and again $\simeq$ is $=$.

Property 8 can be generalised: the order between families of context-lattice pairs may be reduced to the order between contexts (and ultimately the order between their sets of attributes).

Property 10. $\forall O, O' \in \mathscr{O}_{K^0,R,\Omega}$, *if* $k(O) \subseteq k(O')$ *then* $O \preceq O'$.

Proof. $k(O) \subseteq k(O')$ means that $\forall x \in X, k_x(O) \subseteq k_x(O')$ which is equivalent, by Property 8, to $\langle k_x(O), l_x(O)\rangle \preceq \langle k_x(O'), l_x(O')\rangle$ and hence $O \preceq O'$ (Definition 9). □

Finally, the set $\mathscr{O}_{K^0,R,\Omega}$ of families of context-lattice pairs is a complete lattice.

Proposition 11. $\langle \mathscr{O}_{K^0,R,\Omega}, \vee, \wedge\rangle$ *is a complete lattice.*

Proof. $\mathscr{O}_{K^0,R,\Omega}$ is closed by meet and join (Property 9). The commutativity, associativity and the absorption law of $\vee$ and $\wedge$ ultimately rely on these properties holding for the same operators on contexts. They are satisfied because these are properties of union and intersection on sets. Hence, $\langle \mathscr{O}_{K^0,R,\Omega}, \vee, \wedge\rangle$ is a lattice. It is complete because $\vee$ and $\wedge$ are well-defined (Definition 8) and $\forall S \subseteq \mathscr{O}$, $\forall O \in S$, $\bigwedge_{O'\in S} O' \preceq O \preceq \bigvee_{O'\in S} O'$ (due to the completeness of powerset lattices which apply to those of attributes). □

The set $\mathscr{O}$ of families of context-lattice pairs arguably contains all possible solutions that can be returned by relational concept analysis. However, they have been defined on independent contexts, taking into account the scalable attributes but not the coherence between the considered concepts. This will now be achieved through specific functions.

5 A Functional Standpoint on RCA

Here we adopt a functional standpoint on operations on the space of families of context-lattice pairs. We first define an expansion function EF^* which corresponds to the internal operations of RCA (§5.1). We then consider the notion of support which eliminates potential solutions referring to non-available concepts (§5.2). This allows us to define a contraction function reducing the non-supported part of a family of context-lattice pairs (§5.3).

5.1 The Expansion Function EF^*

The solutions are now characterised as elements of $\mathscr{O}_{K^0,R,\Omega}$. In the following, a function on this set is defined to reformulate RCA. This is $EF^*_{K^0,R,\Omega}$, the expansion function attached to a relational context $\langle K^0, R\rangle$ and a set Ω of scaling operations.

Definition 10 (Expansion function). *Given a relational context $\langle K^0, R\rangle$ and a set Ω of relational scaling operations, the expansion function $EF^*_{K^0,R,\Omega} : \mathscr{O}_{K^0,R,\Omega} \longrightarrow \mathscr{O}_{K^0,R,\Omega}$ is defined by:*

$$EF^*_{K^0,R,\Omega}(O) = \mathrm{T}^*(\sigma^*_\Omega(k(O), R, l(O))).$$

As previously, we will abbreviate $EF^*_{K^0,R,\Omega}$ as EF^*.

This function is the basis of RCA: it covers scaling and the application of FCA* embedded in the function T*. Thus, the two steps (3) and (2) of the RCA algorithm (Section 2.4.2) have been merged into one.

A family of context-lattice pairs is called saturated if it is not possible to scale new relational attributes in any of its contexts.

Definition 11 (Saturated family of context-lattice pairs). *A family of context-lattice pairs $O \in \mathscr{O}$ is* saturated *if $\forall x \in X$, $k_x(O) = \sigma_\Omega(k_x(O), R, l(O))$.*

EF^* is an extensive and monotone internal operation for $\mathscr{O}$:

Property 12 (EF^* is internal to $\mathscr{O}$). *$\forall O \in \mathscr{O}$, $EF^*(O) \in \mathscr{O}$.*

Proof. $\forall x \in X$, $\sigma_\Omega(k_x(O), R, l(O)) \in \mathscr{K}^{\eta^*(K^0)}_{K^0_x,R,\Omega}$ because σ_Ω only scales attributes in D_{Ω,R_x,K^0}. Therefore, $\mathrm{T}(\sigma_\Omega(k_x(O), R, l(O))) \in \mathscr{T}^{\eta^*(K^0)}_{K^0_x,R,\Omega}$. Hence, $EF^*(O) = \{\mathrm{T}(\sigma_\Omega(k_x(O), R, l(O)))\}_{x\in X} \in \mathscr{O}_{K^0_x,R,\Omega}$ (Definition 7). □

Property 13 (EF^* is extensive and monotone). *The function EF^* attached to a relational context and a set of scaling operations satisfies:*

$$O \preceq EF^*(O), \qquad \text{(Extensivity)}$$
$$O \preceq O' \Rightarrow EF^*(O) \preceq EF^*(O'). \qquad \text{(Monotony)}$$

Proof. Extensivity holds because EF^* can only add to $k(O)$ attributes scaled from $l(O)$, hence $\forall x \in X, k_x(O) \subseteq k_x(EF^*(O))$. Thus, by Property 10, $O \preceq EF^*(O)$. Monotony holds because $O \preceq O'$ means that $\forall x \in X$, $l_x(O) \preceq l_x(O')$ and $k_x(O) \subseteq k_x(O')$. The former entails that $\forall x \in X$, $\eta(l_x(O)) \subseteq \eta(l_x(O'))$ and consequently, that $D_{\Omega,R_x,l(O)} \subseteq D_{\Omega,R_x,l(O')}$. A smaller context ($k_x(O)$) is extended by a smaller set of attributes ($D_{\Omega,R_x,l(O)}$), thus $k_x(EF^*(O)) \subseteq k_x(EF^*(O'))$. Hence, by Property 10, $EF^*(O) \preceq EF^*(O')$. □

5.2 Self-Supported Lattices

In a family of context-lattice pairs O, there may be a context $k(O_x)$ containing attributes which refer to concepts non existing in $l(O)$. Such an attribute, e.g. $\exists r.c$, belonging to $k(O_x)$ may belong to D_{Ω,R_x,K^0} and be well-defined by the incidence relation, but c may not be a concept of $l(O)$. This is illustrated by Example 7.

Example 7 (Non self-supported families of context-lattice pairs). *Figure 6 (p.13) shows a familly of contexts $\{K_3^\#, K_4^\#\}$ and the associated family of concept lattices $\{L_3^\#, L_4^\#\}$ that could be a solution for the example of Section 3.2 as it belongs to $\mathscr{O}_{\{\exists\},\{p,q\},\{K_3^0,K_4^0\}}$. However, this family is not self-supported because the context $K_3^\#$ (and thus concept A) uses the attribute $\exists p.C$ which refers to a concept (C) not present in $L_4^\#$ and similarly for $\exists q.B$ in context $K_4^\#$.*

A family of context-lattice pairs in $\mathscr{O}$ containing such attributes will see them preserved by EF^* which only extends the contexts. This is not the expected result: concepts referred to by attributes are expected to exist in the corresponding lattice.

One may consider identifying such attributes and forbidding them. However, support is contextual: a supported attribute in a large family of context-lattice pairs may not be supported in a smaller one.

In order to define acceptable solutions for RCA, we introduce the notion of context supported by a family of concept lattices, i.e. those contexts whose relational attributes only refer to concepts in the lattices.

Definition 12 (Supported context). *A context $\langle G_x, M_x, I_x\rangle$ is* supported *by a family of indexed lattices $\{L_z\}_{z\in X}$, with respect to a set R of relations, if $\forall\varsigma(r,c) \in M_x$, $r \in R_{x,z}$ and $c \in L_z$.*

By extension, an indexed family of context-lattice pairs is said *self-supported* if each context of the family is supported by the family lattices. The support of a single context may use several lattices of the family ($l(O)$).

Definition 13 (Self-supported families of context-lattice pairs). *A family of context-lattice pairs $O \in \mathscr{O}$ is* self-supported*, with respect to a set R of relations, if $\forall x \in X$, $k_x(O)$ is supported by $l(O)$, with respect to R.*

The definition of self-supported families of context-lattice pairs does not provide a direct way to transform a non self-supported family into a self-supported one. A possible way to solve this problem consists of extracting only the attributes currently in the lattice and to apply FCA to the resulting context.

For that purpose, we introduce a filtering or purging function π^* which suppresses *from the contexts* ($\kappa(L_x)$) induced by the lattices L_x in L those attributes non supported by L:

Definition 14 (Purging function). *The function $\pi^*_{K^0,R,\Omega} : \prod_{x\in X} \mathscr{L}_{K_x^0,R,\Omega} \longrightarrow \prod_{x\in X} \mathscr{K}_{K_x^0,R,\Omega}$ returns the family of contexts reduced of those attributes not present in a family of lattices:*

$$\pi^*_{K^0,R,\Omega}(L) = \{\pi_{K^0,R,\Omega}(L_x, L)\}_{x\in X}$$

with

$$\pi_{K^0,R,\Omega}(L_x, L) = \mathrm{K}^{\langle R,L\rangle}_{-(D_{\Omega,R_x,K^0})\setminus D_{\Omega,R_x,L})}(\kappa(L_x)).$$

When unambiguous, we refer to $\pi_{K^0,R,\Omega}$ (resp. $\pi^*_{K^0,R,\Omega}$) as π (resp. π^*). This extends the purging function of [10]. The purging function only restricts the sets of possible contexts and does not expand them.

π and σ are not inverse functions: in particular, σ greatly depends on Ω and R to decide which attributes to scale, through π simply suppresses attributes non supported by the lattices, independently from Ω, which however determines the attribute language.

π^* can be used to determine if a family of context-lattice pairs is self-supported:

Property 14. *O is self-supported if and only if $k(O) = \pi^*(l(O))$.*

Proof. O is self-supported means that $\forall x \in X$, $k_x(O)$ is supported by $l(O)$ (Definition 13) which means that, in $k_x(O)$, there is no attribute built from concepts out of $l(O)$, i.e. not belonging to $D_{\Omega,R_x,l(O)}$. By Definition 14, this is equivalent to having $\forall x \in X$, $\pi(l_x(O), l(O)) = \kappa(l_x(O))$. However, by Property 3, $k_x(O) = \kappa(l_x(O))$, thus $k_x(O) = \pi(l_x(O), l(O))$ and then $k(O) = \pi^*(l(O))$. □

Like the scaling function, the purging function, is only one step: it suppresses currently unsupported attributes, but this may lead to less concepts to be generated by FCA, and thus to other non-supported attributes. Hence, it has to be iterated.

We introduce a contraction function, $PQ^*_{K^0,R,\Omega}$, attached to a relational context $\langle K^0, R\rangle$ and a set Ω of scaling operations, which suppresses non-supported attributes and whose closure yields self-supported families of context-lattice pairs.

5.3 The Contraction Function PQ^*

Similarly to $EF^*_{K^0,R,\Omega}$, it is possible to define $PQ^*_{K^0,R,\Omega}$ the contraction function attached to a relational context $\langle K^0, R\rangle$ and a set Ω of scaling operations.

Definition 15 (Contraction function). *Given a relational context $\langle K^0, R\rangle$ and a set Ω of relational scaling operations, the contraction function $PQ^*_{K^0,R,\Omega} : \mathscr{O}_{K^0,R,\Omega} \longrightarrow \mathscr{O}_{K^0,R,\Omega}$ is defined by:*

$$PQ^*_{K^0,R,\Omega}(O) = \mathrm{T}^*(\pi^*_{K^0,R,\Omega}(l(O))).$$

As previously, we will abbreviate $PQ^*_{K^0,R,\Omega}$ as PQ^*.

PQ^* is an anti-extensive and monotone internal operation for $\mathscr{O}$:

Property 15 (PQ^* is internal to $\mathscr{O}$). $\forall O \in \mathscr{O}, PQ^*(O) \in \mathscr{O}$.

Proof. $PQ^*(O) = \mathrm{T}^*(\pi^*(l(O))) = \mathrm{T}^*(\{\pi(l_x(O), l(O))\}_{x\in X}) = \{\mathrm{T}(\pi(l_x(O), l(O)))\}_{x\in X}$. So, $PQ^*(O) \in \mathscr{O}$ if $\pi(l_x(O), l(O)) \in \mathscr{K}^{\eta^*(K^0)}_{K^0_x,R,\Omega}$ (Definition 7). This is the case because (*i*) $\mathscr{K}^{\eta^*(K^0)}_{K^0_x,R,\Omega}$ contains all contexts extending K^0_x with attributes from D_{Ω,R_x,K^0}, (*ii*) $k_x(O) \in \mathscr{K}^{\eta^*(K^0)}_{K^0_x,R,\Omega}$, and (*iii*) π only suppresses attributes from $k_x(O)$ preserving those of K^0_x. □

Property 16 (PQ^* is anti-extensive and monotone). *The function PQ^* attached to a relational context and a set of scaling operations satisfies:*

$$PQ^*(O) \preceq O, \qquad \text{(Anti-extensivity)}$$
$$O \preceq O' \Rightarrow PQ^*(O) \preceq PQ^*(O'). \qquad \text{(Monotony)}$$

Proof. Anti-extensivity holds because PQ^* can only suppress from $k(O)$ attributes not supported by $l(O)$, hence $\forall x \in X, k_x(PQ^*(O)) \subseteq k_x(O)$. Thus, by Property 10, $PQ^*(O) \preceq O$. Monotony holds because $O \preceq O'$ means that $\forall x \in X, k_x(O) \subseteq k_x(O')$ and $l_x(O) \preceq l_x(O')$. This entails that $\eta^*(l(O)) \subseteq \eta^*(l(O'))$ and thus, $\forall x \in X$, $D_{\Omega,R_x,l(O)} \subseteq D_{\Omega,R_x,l(O')}$. Because $PQ^*(O)$ suppresses from $k_x(O)$ attributes not in $M^0_x \cup D_{\Omega,R_x,l(O)}$, this entails that $k_x(PQ^*(O)) \subseteq k_x(PQ^*(O'))$. Hence, by Property 10, $PQ^*(O) \preceq PQ^*(O')$. □

6 The Fixed Points of EF^* and PQ^*

We end up with two functions, EF^* and PQ^*, over families of context-lattice pairs, the former extensive and the latter anti-extensive. We consider the fixed points of these functions as saturated and self-supported families of context-lattice pairs (§6.1). Closure functions are defined which return the smallest subsuming and greatest subsumed fixed points of a family (§6.2). This allows us to precisely define acceptable solutions as those families of context-lattice pairs which are fixed points for both functions (§6.3).

6.1 Fixed Points

RCA is a world of fixed points, hence it is easy to get lost among the various fixed points involved:

- In description logics, which RCA targets, the semantics of concepts is given by (least) fixed points when circularities occur [23];
- FCA's goal is to compute fixed points: concepts are the result of a closure operation which is also a fixed point [5];
- finally, the RCA result is the fixed point of the function that grows a family of concept lattices from the previous one through scaling.

The present work is concerned with the fixed points of the latter function taking the others into account.

Given EF^* and PQ^*, it is possible to define their sets of fixed points, i.e. the sets of families of context-lattice pairs closed for EF^* and PQ^*, as:

Definition 16 (Fixed points). *A family of context-lattice pairs $O \in \mathscr{O}$ is a fixed point for a function ϕ, if $\phi(O) \simeq O$. We call $\text{fp}(\phi)$ the set of fixed points for ϕ.*

This characterises $\text{fp}(EF^*)$ and $\text{fp}(PQ^*)$.
This may be directly expressed

Property 17. *$O \in \text{fp}(EF^*)$ iff $\sigma^*_\Omega(k(O), R, l(O)) = k(O)$.*

Proof. $O = \text{T}^*(k(O))$ and $EF^*(O) = \text{T}^*(\sigma^*_\Omega(k(O), R, l(O)))$. ($\Leftarrow$) If $\sigma^*_\Omega(k(O), R, l(O)) = k(O)$, then $EF^*(O) = O$ and thus O is a fixed point of EF^*. ($\Rightarrow$) If $\sigma^*_\Omega(k(O), R, l(O)) \neq k(O)$, then $EF^*(O) \neq O$, so it is not a fixed point. □

Property 18. *$O \in \text{fp}(PQ^*)$ iff $\pi^*(l(O)) = k(O)$.*

Proof. $O = \text{T}^*(k(O))$ and $PQ^*(O) = \text{T}^*(\pi^*(l(O)))$. ($\Leftarrow$) If $\pi^*(l(O)) = k(O)$, then $PQ^*(O) = O$ and thus is a fixed point of PQ^*. ($\Rightarrow$) If $\pi^*(l(O)) \neq k(O)$, then $PQ^*(O) \neq O$, so it is not a fixed point. □

Since $\mathscr{O}$ is a complete lattice (Proposition 11) and EF^* and PQ^* are order-preserving (or monotone) on $\mathscr{O}$ (Properties 13 and 16), then we can apply the Knaster-Tarski theorem.

Theorem (Knaster-Tarski theorem [28]). *Let $\mathscr{O}$ be a complete lattice and let $\phi : \mathscr{O} \longrightarrow \mathscr{O}$ be an order-preserving function. Then the set of fixed points of ϕ in $\mathscr{O}$ is also a complete lattice.*

Thus, $\langle \text{fp}(EF^*), \preceq \rangle$ and $\langle \text{fp}(PQ^*), \preceq \rangle$ are complete lattices. This warrants that there exists least and greatest fixed points of EF^* and PQ^* in $\mathscr{O}$. For such a function ϕ, operating on the set $\mathscr{O}$, their least and greatest fixed points are:

$$\text{lfp}(\phi) = \bigwedge_{O \in \text{fp}(\phi)} O \text{ and } \text{gfp}(\phi) = \bigvee_{O \in \text{fp}(\phi)} O.$$

The fixed points of these two functions may be further characterised. The smallest fixed point of PQ^* is the smallest element of $\mathscr{O}$ which cannot be further reduced:

Property 19 (Least fixed point of PQ^*).

$$\text{lfp}(PQ^*) = \text{T}^*(K^0).$$

Proof. $\text{T}^*(\pi^*(\text{FCA}^*(K^0))) = \text{T}^*(K^0)$ because (*a*) $\forall x \in X$, $\kappa(\text{FCA}(K^0_x)) = K^0_x$ (Property 3), and (*b*) $\pi(\text{FCA}(K^0_x), \text{FCA}^*(K^0)) = K^0_x$ as it is not possible to suppress attributes from K^0_x which, being an initial (unscaled) context, does not comprise any attribute referring to concepts. Thus, $PQ^*(\text{T}^*(K^0)) = \text{T}^*(\pi^*(\text{FCA}^*(K^0))) = \text{T}^*(K^0)$. Moreover, $\forall O \in \mathscr{O}$, $\text{T}^*(K^0) \preceq O$. Hence, $\text{T}^*(K^0)$ is a fixed point of PQ^* and all other fixed points are greater. □

The greatest fixed point of EF^* is the family that cannot be further extended (generalising Proposition 2 of [10]):

Property 20 (Greatest fixed point of EF^*).

$$\mathrm{gfp}(EF^*_{K^0,R,\Omega}) = \mathrm{T}^*(\{\mathrm{K}^{\langle R,\eta^*(K^0)\rangle}_{+D_{\Omega,R_x,K^0}}(K^0_x)\}_{x\in X}).$$

Proof. This family of context-lattice pairs is the greatest element of $\mathscr{O}$ as $\forall x \in X$, the context $k_x(O)$ contains all attributes of $M^0_x \cup D_{\Omega,R_x,K^0}$ and due to Property 10. It is also a fixed point because EF^* is extensive (Property 13) and internal (Property 12). □

6.2 Closure Functions ($EF^{*\infty}$ and $PQ^{*\infty}$)

EF^* and PQ^* are not closure operations as they are not idempotent. However, with the same arguments as [26], and in particular the finiteness of contexts (see Section 2.4.3), it can be argued that their repeated application converges to a fixed point.

Property 21 (Stability of EF^*). *$\forall O \in \mathscr{O}, \exists n; EF^{*n}(O) = EF^{*n+1}(O)$.*

Proof. EF^* can only increase the contexts when there are new concepts in lattices and increase the lattices when contexts grow. However, the set of attributes that can increase contexts, and the set of concepts that can be in lattices, is finite. Hence, at each step either an attribute is added or n has been reached such that the family of context-lattice pairs is the same. This is the same argument as that of [26]. □

This below is an extension of Proposition 5 of [10]:

Property 22 (Stability of PQ^*). *$\forall O \in \mathscr{O}, \exists n; PQ^{*n}(O) = PQ^{*n+1}(O)$.*

Proof. PQ^* can only decrease the contexts and reduce lattices. Since these are finite (and the decrease does not affect the attributes of K^0), there exists a n at which the decrease stops. □

The finite application of EF^* and PQ^* as many times as necessary, i.e. to the first n satisfying Properties 21 and 22, are closure operations denoted by $EF^{*\infty}$ and $PQ^{*\infty}$, respectively.

Property 23. *$EF^{*\infty}$ and $PQ^{*\infty}$ are closures.*

Proof. Since EF^* is extensive and monotone (Property 13), $EF^{*\infty}$ is also extensive and monotone by transitivity of $\preceq$. In order to be a closure operation it has to be idempotent. This is the case, because $\forall O \in \mathscr{O}$, $EF^{*\infty}(O) = EF^{*n}(O) = EF^{*n+1}(O) = EF^*(EF^{*n}(O))$. Since $EF^{*n}(O) = EF^*(EF^{*n}(O))$, EF^* can be applied n times, leading to $EF^{*\infty}(O) = EF^{*n}(O) = EF^{*n}(EF^{*n}(O)) = EF^{*\infty}(EF^{*\infty}(O))$.
The same can be obtained from PQ^*, albeit anti-extensive (Property 16). □

In addition, they are extrema of the sets of fixed points of their respective functions.

Property 24 (EF^* and PQ^* return the smallest subsuming and greatest subsumed fixed points). *$\forall O \in \mathscr{O}$,*

$$EF^{*\infty}(O) = \min_{\preceq}(\mathrm{fp}(EF^*) \cap \{O'|O \preceq O'\}),$$
$$PQ^{*\infty}(O) = \max_{\preceq}(\mathrm{fp}(PQ^*) \cap \{O'|O' \preceq O\}).$$

Proof. $EF^{*\infty}(O) \in \mathrm{fp}(EF^*)$ and $PQ^{*\infty}(O) \in \mathrm{fp}(PQ^*)$ as they satisfy Definition 16. Moreover, $EF^{*\infty}(O) \in \{O'|O \preceq O'\}$ and $PQ^{*\infty}(O) \in \{O'|O' \preceq O\}$ as EF^* and PQ^* are respectively extensive and anti-extensive and monotone (Property 13 and 16). There cannot be $O' \in \mathrm{fp}(EF^*) \cap \{O'|O \preceq O'\}$ such that $O' \prec EF^{*\infty}(O)$ because otherwise $k(O') \subset k(EF^{*\infty}(O))$ and $k(O) \subseteq k(O')$. In other terms, O' contains all attributes of O but not all attributes of $EF^{*\infty}(O)$. But, $EF^{*\infty}$ only adds scalable attributes and $k(EF^{*\infty}(O))$ contains only attributes scalable from O. Hence, O' is not closed for EF^* ($O' \notin \text{fp}(EF^*)$).

The same holds for $PQ^{*\infty}(O)$, there cannot be $O' \in \text{fp}(PQ^*) \cap \{O'|O' \preceq O\}$ such that $PQ^{*\infty}(O) \prec O'$ because otherwise $k(O') \subseteq k(O)$. In other terms, all attributes of O' are in O but O' contains all attributes of $PQ^{*\infty}(O)$. However, $PQ^{*\infty}$ only suppress attributes not supported by those of O. Hence, O' is not closed for PQ^* ($O' \notin \text{fp}(PQ^*)$), as it would contain non-supported attributes. □

The respective relations of these various objects can be summarised by the following property:

Property 25. *$\forall O \in \mathscr{O}$,*

$$\text{lfp}(PQ^*) \preceq PQ^{*\infty}(O) \preceq PQ^*(O) \preceq O \preceq EF^*(O) \preceq EF^{*\infty}(O) \preceq \text{gfp}(EF^*).$$

Proof. All the inner equations are consequences of the extensivity of EF^* (Property 13) and anti-extensivity of PQ^* (Property 16). The outer ones owe to the fact that the two closure operations are fixed points (Property 24), thus they are subsumed by, resp. subsuming, their greatest, resp. least, fixed point. □

6.3 Acceptable Solutions

What is called acceptable solutions in Section 3 is now rephrased in Definition 17.

Definition 17 (Acceptable family of context-lattice pairs). *Given a family K^0 of contexts, a set Ω of scaling operations and a set R of relations, a family of context-lattice pairs O is acceptable if*

- *$O \in \mathscr{O}_{K^0,R,\Omega}$,* *(well-formedness)*
- *O is saturated,* *(saturation)*
- *O is self-supported.* *(self-support)*

This can be characterised as those families of context-lattice pairs which are fixed points of both EF^* and PQ^*. The fixed points of EF^* are exactly those saturated elements of $\mathscr{O}$:

Property 26 (Fixed points of EF^* are saturated). *$\forall O \in \mathscr{O}$, O is saturated iff $O \in \text{fp}(EF^*)$.*

Proof. $O \in \text{fp}(EF^*)$ means that $k(O) = \sigma^*_\Omega(k(O), R, l(O))$ (Property 17) which is equivalent to $\forall x \in X, k_x(O) = \sigma_\Omega(k_x(O), R, l(O))$ which, by Definition 11, means that O is saturated. □

The fixed points of PQ^* are exactly those self-supported objects in $\mathscr{O}$:

Property 27 (Fixed points of PQ^* are self-supported). *$\forall O \in \mathscr{O}$, O is self-supported iff $O \in \text{fp}(PQ^*)$.*

Proof. O is self-supported iff $k(O) = \pi^*(l(O))$ (Property 14) which is equivalent to $O = PQ^*(O)$ (Property 18), i.e. $O \in \text{fp}(PQ^*)$. □

Hence, the set of acceptable solutions is $\text{fp}(EF^*) \cap \text{fp}(PQ^*)$.

Proposition 28 (Acceptable solutions are fixed points of both EF^* and PQ^*). *Given a family K^0 of contexts, a set Ω of scaling operations and a set R of relations, a family of context-lattice pairs O is acceptable iff $O \in \mathscr{O}_{K^0,R,\Omega}$ and $O \in \text{fp}(EF^*) \cap \text{fp}(PQ^*)$.*

Proof. O is well-formed as it belongs to $\mathscr{O}_{K^0,R,\Omega}$. It is saturated if and only if it belongs to $\text{fp}(EF^*)$ (Property 26) and it is self-supported if and only if it belongs to $\text{fp}(PQ^*)$ (Property 27). Hence, O is acceptable (Definition 17). □

Example 8 illustrates this:

Example 8 (Acceptable solutions). *In the example of Section 3.2, it can be checked that the given solutions belong to the expected fixed points:*

$$EF^*(\{\langle K_3^1, L_3^1\rangle, \langle K_4^1, L_4^1\rangle\}) = \{\langle K_3^1, L_3^1\rangle, \langle K_4^1, L_4^1\rangle\} = PQ^*(\{\langle K_3^1, L_3^1\rangle, \langle K_4^1, L_4^1\rangle\}),$$
$$EF^*(\{\langle K_3^\star, L_3^\star\rangle, \langle K_4^\star, L_4^\star\rangle\}) = \{\langle K_3^\star, L_3^\star\rangle, \langle K_4^\star, L_4^\star\rangle\} = PQ^*(\{\langle K_3^\star, L_3^\star\rangle, \langle K_4^\star, L_4^\star\rangle\}),$$
$$EF^*(\{\langle K_3', L_3'\rangle, \langle K_4', L_4'\rangle\}) = \{\langle K_3', L_3'\rangle, \langle K_4', L_4'\rangle\} = PQ^*(\{\langle K_3', L_3'\rangle, \langle K_4', L_4'\rangle\}),$$

and

$$EF^*(\{\langle K_3'', L_3''\rangle, \langle K_4'', L_4''\rangle\}) = \{\langle K_3'', L_3''\rangle, \langle K_4'', L_4''\rangle\} = PQ^*(\{\langle K_3'', L_3''\rangle, \langle K_4'', L_4''\rangle\}).$$

and none of the other elements of $\mathscr{O}$ as displayed in Figure 9 (p.32).

In lattice theory, saturation and self-support would have been easily called closedness. The terms saturation and self-support have been chosen in order to differentiate them.

7 The Fixed-Point Semantics of RCA

Now that the acceptable solutions have been characterised structurally and functionally, we can answer our initial question and define the semantics of RCA. $\underline{\text{RCA}}$ returns the smallest acceptable solution. It is also the least fixed point of the EF^* function (§7.1). Another interesting operation is the one that generates the greatest acceptable solution, which is also the greatest fixed point of PQ^* (§7.2). It is also worth considering obtaining the whole set $\text{fp}(EF^*) \cap \text{fp}(PQ^*)$. Section 7.3 investigates the structure of $[\text{lfp}(EF^*),\ \text{gfp}(PQ^*)]$ and its relation with $\text{fp}(EF^*) \cap \text{fp}(PQ^*)$ towards that goal. It provides various results that may be exploited to develop efficient algorithms.

7.1 Classical RCA Computes EF^*'s Least Fixed Point

$\underline{\text{RCA}}$ as it has been defined in Section 2.4.2 (p.8) may be redefined as

$$\underline{\text{RCA}}_\Omega(K^0, R) = l(EF^{*\infty}_{K^0,R,\Omega}(\text{T}^*(K^0)))$$

i.e. $\underline{\text{RCA}}$ iterates EF^* from $\text{T}^*(K^0)$ until reaching a fixed point, and ultimately the corresponding lattices are returned.

It thus seems that $\underline{\text{RCA}}$ returns a fixed point of EF^*. Hence the question: which fixed point is returned by RCA's well-grounded semantics? This is the least fixed point.

Proposition 29 (The RCA algorithm computes the least fixed point of EF^*). *Given EF^* the expansion function associated to K^0, R and Ω,*

$$\underline{\text{RCA}}_\Omega(K^0, R) = l(\text{lfp}(EF^*_{K^0,R,\Omega})).$$

Proof. $\text{T}^*(K^0) \in \mathscr{O}$, hence $EF^{*\infty}(\text{T}^*(K^0)) \in \mathscr{O}$ (by Property 12). Moreover, $EF^{*\infty}(\text{T}^*(K^0)) = \min_\preceq(\text{fp}(EF^*) \cap \{O' \mid \text{T}^*(K^0) \preceq O'\})$ (Property 24). But $\forall O' \in \mathscr{O}$, $\text{T}^*(K^0) \preceq O'$, hence $EF^{*\infty}(\text{T}^*(K^0)) = \min_\preceq(\text{fp}(EF^*))$. Thus, $EF^{*\infty}_{K^0,R,\Omega}(\text{T}^*(K^0))$ is a fixed point more specific than all fixed points: it is the least fixed point.

$\underline{\text{RCA}}_\Omega(K^0, R) = l(EF^{*\infty}_{K^0,R,\Omega}(\text{T}^*(K^0)))$ returns the family of lattices associated with the least fixed point of $EF^*_{K^0,R,\Omega}$. □

7.2 Greatest Fixed-Point (of PQ^*) Semantics

It is possible to define $\overline{\text{RCA}}$ as returning the greatest acceptable solution. The greatest fixed point of EF^* (Property 20) is not necessarily an acceptable solution because it may not be self-supported. Said otherwise, it does not belong to $\text{fp}(EF^*) \cap \text{fp}(PQ^*)$ because it is not a fixed point for PQ^*.

Alternatively, a dual procedure $\overline{\text{RCA}}$ may be defined as:

$$\overline{\text{RCA}}_\Omega(K^0, R) = l(PQ^{*\infty}_{K^0,R,\Omega}(\text{T}^*(\{\text{K}^{\langle R,\eta^*(K^0)\rangle}_{+D_{\Omega,R_x,K^0}}(K^0_x)\}_{x\in X})))$$

and it can be characterised analogously as the greatest fixed point of $PQ^*_{K^0,R,\Omega}$.

Proposition 30 ($\overline{\text{RCA}}$ determines the greatest fixed point of PQ^*). *Given PQ^* the contraction function associated to K^0, R and Ω,*

$$\overline{\text{RCA}}_\Omega(K^0, R) = l(\text{gfp}(PQ^*_{K^0,R,\Omega})).$$

Proof. $O^\infty = \text{T}^*(\{\text{K}^{\langle R,\eta^*(K^0)\rangle}_{+D_{\Omega,R_x,K^0}}(K^0_x)\}_{x\in X}) \in \mathscr{O}$, hence $PQ^{*\infty}_{K^0,R,\Omega}(O^\infty) \in \mathscr{O}$ (by Property 15). Moreover, $PQ^{*\infty}(O^\infty)$ $= \max_\preceq(\text{fp}(PQ^*) \cap \{O'|O' \preceq O^\infty\})$ (Property 24). But $\forall O' \in \mathscr{O}$, $O' \preceq O^\infty$, hence $PQ^{*\infty}(O^\infty) = \max_\preceq(\text{fp}(PQ^*))$. Thus, $PQ^{*\infty}_{K^0,R,\Omega}(O^\infty)$ is a fixed point more general than all fixed points: it is the greatest fixed point.

$\overline{\text{RCA}}_\Omega(K^0, R) = l(PQ^{*\infty}_{K^0,R,\Omega}(O^\infty))$ returns the family of lattices associated with the greatest fixed point of $PQ^*_{K^0,R,\Omega}$. □

In order to find $\text{gfp}(PQ^*)$, the process starts with $\text{T}^*(\{\text{K}^{\langle R,\eta^*(K^0)\rangle}_{+D_{\Omega,R_x,K^0}}(K^0_x)\}_{x\in X})$, the largest family of context-lattice pairs, and iterates the application of PQ^*, i.e. the two operations π^* and FCA^*, until reaching a fixed point, i.e. reaching n such that $O^{n+1} = O^n$.

Thus, the $\overline{\text{RCA}}$ algorithm proceeds in the following way:

(1) Initial contexts: $t \longleftarrow 0$; $\{\langle G_x, M^t_x, I^t_x\rangle\}_{x\in X} \longleftarrow \{\text{K}^{\langle R,\eta^*(K^0)\rangle}_{+D_{\Omega,R_x,K^0}}(K^0_x)\}_{x\in X}$
(2) $\{L^t_x\}_{x\in X} \longleftarrow \text{FCA}^*(\{\langle G_x, M^t_x, I^t_x\rangle\}_{x\in X})$ (or, for each context, $\langle G_x, M^t_x, I^t_x\rangle$ the corresponding concept lattice $L^t_x = \text{FCA}(\langle G_x, M^t_x, I^t_x\rangle)$ is created using FCA).
(3) $\{\langle G_x, M^{t+1}_x, I^{t+1}_x\rangle\}_{x\in X} \longleftarrow \pi^*(\{L^t_x\}_{x\in X})$ (i.e. suppressing from K^t_x each attribute in L^t_x referring through a relation $r \in R_{x,z}$ to a concept c_z not appearing in L^t_z).
(4) If $\exists x \in X; M^{t+1}_x \neq M^t_x$ (purging has occurred), then $t \longleftarrow t + 1$; go to Step 2.
(5) Return: $\{L^t_x\}_{x\in X}$.

This algorithm is the dual of the $\underline{\text{RCA}}$ procedure.

Example 9 shows how this is processed.

Example 9 (Greatest fixed-point semantics). *In the cases presented in Section 3.1 and 3.2, the greatest (or maximum) elements $\{\langle K^\star_1, L^\star_1\rangle, \langle K^\star_2, L^\star_2\rangle\}$ and $\{\langle K^\star_3, L^\star_3\rangle, \langle K^\star_4, L^\star_4\rangle\}$ of $\mathscr{O}$ are fixed points for both EF^* and PQ^*. Instead, consider the example of Section 3.1 in which the relation p is changed to:*

p	d	e	f
a	×	×	
b	×	×	
c			×

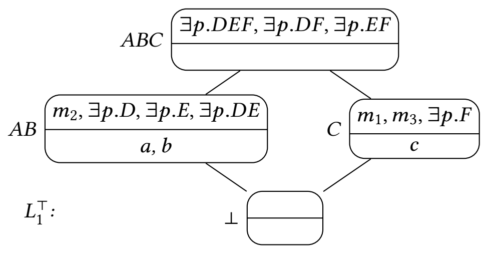
q	a	b	c
d	×		
e		×	
f			×

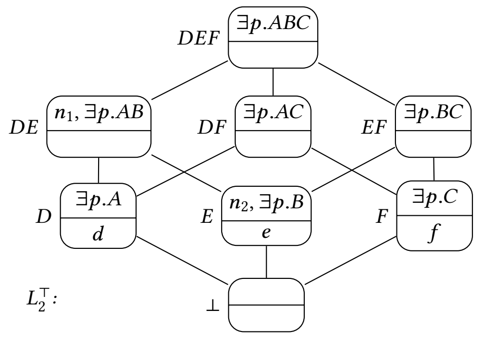
q remaining the same. The effect of changing p is to make a and b non distinguishable and reduce the sets of supported concepts.

Then (Property 20), $\text{gfp}(EF^*) = \text{T}^*(\{\text{K}^{\langle R,\eta^*(K^0)\rangle}_{+D_{\Omega,R_x,K^0}}(K^0_x)\}_{x\in\{1,2\}}) = \{\langle K^\top_1, L^\top_1\rangle, \langle K^\top_2, L^\top_2\rangle\}$ *is presented below:*

$K_1^\top$	m_1	m_2	m_3	$\exists p.DEF$	$\exists p.DE$	$\exists p.DF$	$\exists p.EF$	$\exists p.D$	$\exists p.E$	$\exists p.F$
a		×		×	×	×	×	×	×	
b		×		×	×	×	×	×	×	
c	×		×	×		×	×			×

$K_2^\top$	n_1	n_2	$\exists q.ABC$	$\exists q.AB$	$\exists q.AC$	$\exists q.BC$	$\exists q.A$	$\exists q.B$	$\exists q.C$
d	×		×	×	×		×		
e	×	×	×	×		×		×	
f			×		×	×			×

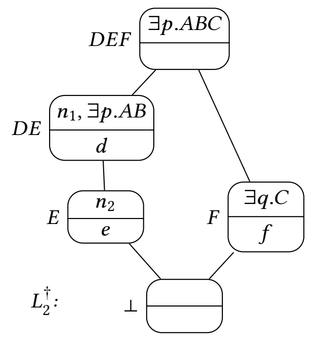

ABC
$\exists p.DEF, \exists p.DF, \exists p.EF$
AB
$m_2, \exists p.D, \exists p.E, \exists p.DE$
a, b
C
$m_1, m_3, \exists p.F$
c
$L_1^\top$:
$\perp$



It can be checked that $\{\langle K_1^\top, L_1^\top\rangle, \langle K_2^\top, L_2^\top\rangle\}$ is a fixed point for EF^*: no additional attribute can be scaled. On the contrary, it is not a fixed point for PQ^* which can be applied to it.

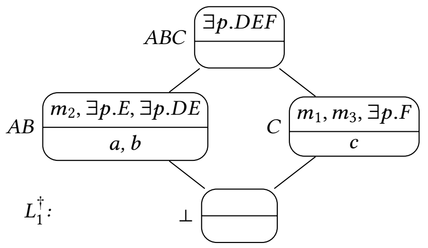
In a first application, it will purge $\langle K_2^\top, L_2^\top\rangle$ to:

$K_2^\dagger$	n_1	n_2	$\exists q.ABC$	$\exists q.AB$	$\exists q.C$
d	×		×	×	
e	×	×	×	×	
f			×		×



A second application will purge $\langle K_1^\top, L_1^\top\rangle$ with respect to $\langle K_2^\dagger, L_2^\dagger\rangle$:

$K_1^\dagger$	m_1	m_2	m_3	$\exists p.DEF$	$\exists p.DE$	$\exists p.E$	$\exists p.F$
a		×		×	×	×	
b		×		×	×	×	
c	×		×	×			×



It can be checked that $\{\langle K_1^\dagger, L_1^\dagger\rangle, \langle K_2^\dagger, L_2^\dagger\rangle\}$ is a fixed point for PQ^*, but also for EF^*.

It may be interesting, for some applications to check if there is only one acceptable solution. This can easily be characterised by:

Proposition 31. $\text{lfp}(EF^*_{K^0,R,\Omega}) = \text{gfp}(PQ^*_{K^0,R,\Omega})$ *iff* $|\text{fp}(EF^*_{K^0,R,\Omega}) \cap \text{fp}(PQ^*_{K^0,R,\Omega})| = 1$.

The proof of this proposition is given in the next section (7.3) as it relies on further results.

This can be tested using $\underline{\text{RCA}}$ and $\overline{\text{RCA}}$.

FCA can be described as RCA with $R = \varnothing$. In this case, $\forall x \in X$, $D_{\Omega,R_x,K^0} = \varnothing$. Thus, $\mathcal{O} = \{\text{T}^*(K^0)\} = \{\langle K^0, \text{FCA}(K^0)\rangle\}$ and $\text{fp}(EF^*) = \text{fp}(PQ^*) = \{\text{T}^*(K^0)\}$. Hence,

$$\underline{\text{RCA}}_\Omega(K^0, \varnothing) = \overline{\text{RCA}}_\Omega(K^0, \varnothing) = \text{FCA}(K^0).$$

7.3 The Structure of Fixed Points

Besides obtaining the least fixed point of EF^* ($\underline{\text{RCA}}_\Omega$) or the greatest fixed point of PQ^* ($\overline{\text{RCA}}_\Omega$), an interesting problem is to obtain all acceptable solutions, i.e. those families of context-lattice pairs belonging to the fixed points of both functions ($\text{fp}(EF^*) \cap \text{fp}(PQ^*)$).

A naive algorithm for this consists in enumerating all elements of $\mathcal{O}$ and testing if they are fixed points. This would not be very efficient. One way to try to improve on this situation is to understand the structure of the set of fixed points and its relation with the two functions and their closures. Figure 8 illustrates the structure of $\mathcal{O}$ and how $EF^{*\infty}$ and $PQ^{*\infty}$ and their composition traverse this structure.

An interesting property of the functions EF^* and PQ^* is that they preserve each other stability:

Property 32 (EF^* is internal to $\text{fp}(PQ^*)$). *$\forall O \in \text{fp}(PQ^*)$, $EF^*(O) \in \text{fp}(PQ^*)$.*

Proof. If $O \in \text{fp}(PQ^*)$, all attributes in intents of $l(O)$ are supported by concepts in $l(O)$ (Property 27 and Definition 13). By Property 13, $O \preceq EF^*(O)$, so these concepts are still in $l(EF^*(O))$. Moreover, EF^* only adds to $k(O)$ attributes which are supported by $l(O)$ (they only refer to concepts in $l(O)$). Hence, the attributes in $k(EF^*(O))$ and those scaled by σ_Ω are still supported by $l(EF^*(O))$. □

Property 33 (PQ^* is internal to $\text{fp}(EF^*)$). *$\forall O \in \text{fp}(EF^*)$, $PQ^*(O) \in \text{fp}(EF^*)$.*

Proof. If $O \in \text{fp}(EF^*)$, this means that $EF^*(O) = O$ and, in particular, that σ^*_Ω does not scale new attributes based on the concepts in $l(O)$. By Property 16, $PQ^*(O) \preceq O$, so that $l(PQ^*(O))$ does not contain more concepts than $l(O)$, then σ^*_Ω cannot scale new attributes ($\sigma^*_\Omega(k(PQ^*(O)), R, l(PQ^*(O))) \subseteq \sigma^*_\Omega(k(O), R, l(O)) = \varnothing$). Hence, $PQ^*(O) \in \text{fp}(EF^*)$. □

This shows that $\text{fp}(EF^*) \cap \text{fp}(PQ^*) \neq \varnothing$: acceptable solutions always exist. In addition, the closure operations associated with the two functions preserve their extrema.

Property 34. *$PQ^{*\infty}(\text{gfp}(EF^*)) = \text{gfp}(PQ^*)$ and $EF^{*\infty}(\text{lfp}(PQ^*)) = \text{lfp}(EF^*)$.*

Proof. $\forall O \in \mathcal{O}, O \preceq \text{gfp}(EF^*)$ (from Property 25), and $PQ^{*\infty}$ is order preserving (Property 23), thus $PQ^{*\infty}(O) \preceq PQ^{*\infty}(\text{gfp}(EF^*))$. Hence, $\forall O \in \text{fp}(PQ^*)$, $O \preceq PQ^{*\infty}(\text{gfp}(EF^*))$. Moreover, $PQ^{*\infty}(\text{gfp}(EF^*)) \in \text{fp}(PQ^*)$, thus $PQ^{*\infty}(\text{gfp}(EF^*)) = \text{gfp}(PQ^*)$.

Similarly, $\forall O \in \mathcal{O}, \text{lfp}(PQ^*) \preceq O$ (Property 25), and $EF^{*\infty}$ is order preserving (Property 23), thus $EF^{*\infty}(\text{lfp}(PQ^*)) \preceq EF^{*\infty}(O)$. Hence, $\forall O \in \text{fp}(EF^*)$, $EF^{*\infty}(\text{lfp}(PQ^*)) \preceq O$. Moreover, $EF^{*\infty}(\text{lfp}(PQ^*)) \in \text{fp}(EF^*)$, therefore $EF^{*\infty}(\text{lfp}(PQ^*)) = \text{lfp}(EF^*)$. □

Proposition 35 complements Property 25 for elements of $\text{fp}(EF^*) \cap \text{fp}(PQ^*)$. The elements of $\text{fp}(EF^*) \cap \text{fp}(PQ^*)$ thus belong to the interval $[\text{lfp}(EF^*)\ \text{gfp}(PQ^*)]$ (which is more restricted than $[\text{lfp}(PQ^*)\ \text{gfp}(EF^*)]$, see Figure 8).

Proposition 35. *$\forall O \in \text{fp}(EF^*) \cap \text{fp}(PQ^*)$,*

$$\text{lfp}(PQ^*) \preceq \text{lfp}(EF^*) \preceq O \preceq \text{gfp}(PQ^*) \preceq \text{gfp}(EF^*).$$

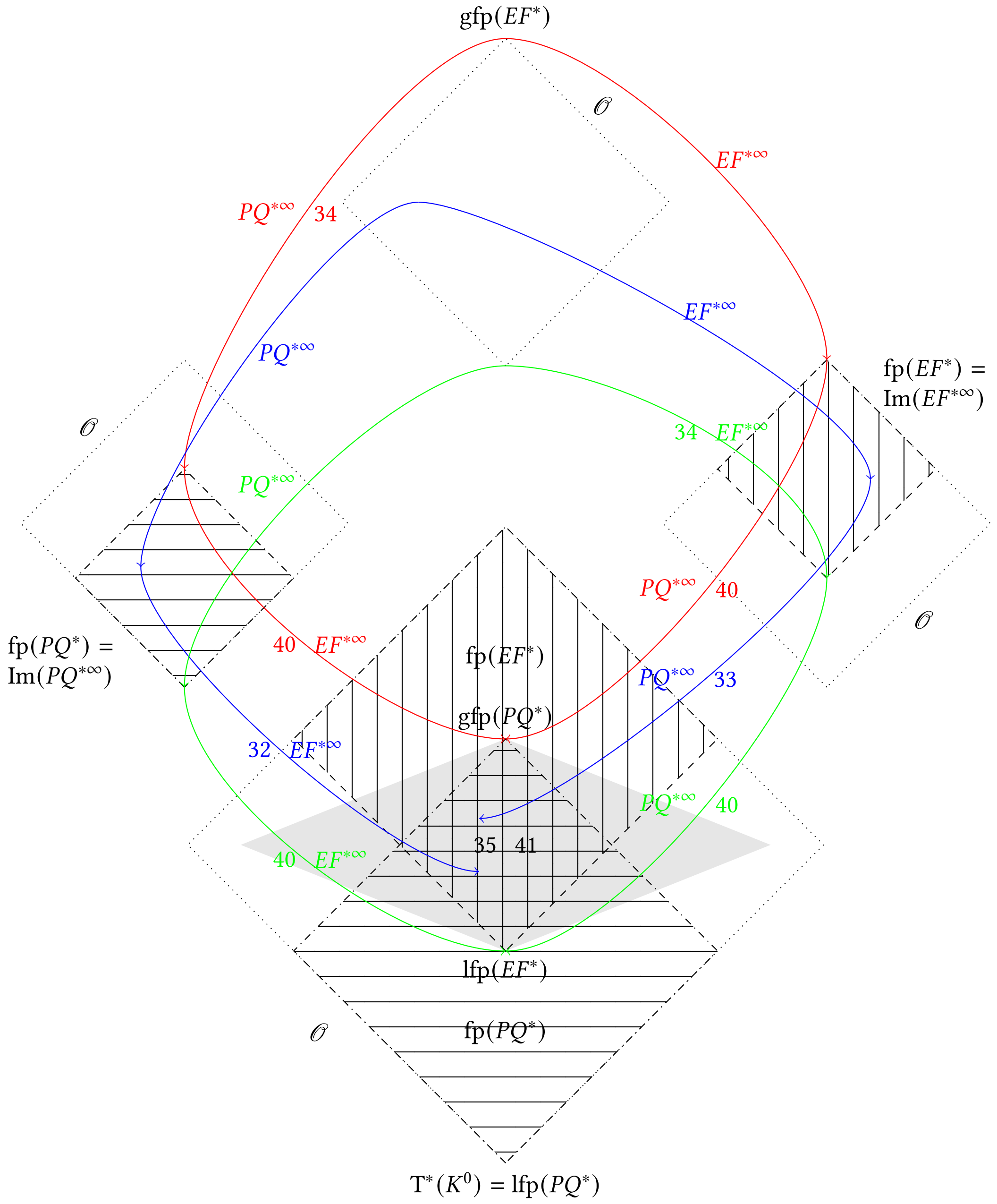


Fig. 8. Illustration of Properties 32, 33, 35, 34, 36, 40 and 41. The figure displays four times $\mathscr{O}$ and the images of gfp(EF^*) (red), a random family of context-lattice pairs (blue) and lfp(PQ^*) = T$^*(K^0)$ (green) through $PQ^{*\infty}$ (left) and $EF^{*\infty}$ (right). fp(EF^*) is drawn in vertical lines; fp(PQ^*) in horizontal lines and the grey area depicts the interval [lfp(EF^*), gfp(PQ^*)].

Proof. By Property 34, $PQ^{*\infty}(\mathrm{gfp}(EF^*)) = \mathrm{gfp}(PQ^*)$, but $PQ^*(O) \preceq O$ (Property 16) and $PQ^{*\infty}(O) \preceq PQ^*(O)$ (Property 25), thus $\mathrm{gfp}(PQ^*) = PQ^{*\infty}(\mathrm{gfp}(EF^*)) \preceq PQ^*(\mathrm{gfp}(EF^*)) \preceq \mathrm{gfp}(EF^*)$. Moreover, by Property 34, $EF^{*\infty}(\mathrm{lfp}(PQ^*)) = \mathrm{lfp}(EF^*)$, but $O \preceq EF^*(O)$ (Property 13) and $EF^*(O) \preceq EF^{*\infty}(O)$ (Property 25), thus $\mathrm{lfp}(PQ^*) \preceq EF^*(\mathrm{lfp}(PQ^*)) \preceq EF^{*\infty}(\mathrm{lfp}(PQ^*)) = \mathrm{lfp}(EF^*)$. Finally, $O \in \mathrm{fp}(EF^*) \cap \mathrm{fp}(PQ^*)$ entails that $\mathrm{lfp}(EF^*) \preceq O$ and $O \preceq \mathrm{gfp}(PQ^*)$. □

It is now possible to prove Proposition 31:

Proof of Proposition 31. First, both $\mathrm{lfp}(EF^*_{K^0,R,\Omega})$ and $\mathrm{gfp}(PQ^*_{K^0,R,\Omega})$ are among the acceptable solutions. Indeed, $\mathrm{lfp}(EF^*) = EF^{*\infty}(\mathrm{lfp}(PQ^*))$ (Property 34), but $\mathrm{lfp}(PQ^*) \in \mathrm{fp}(PQ^*)$ thus $EF^{*\infty}(\mathrm{lfp}(PQ^*)) \in \mathrm{fp}(PQ^*)$ (Property 32), hence $\mathrm{lfp}(EF^*) \in \mathrm{fp}(PQ^*)$. The same reasoning can be applied to $\mathrm{gfp}(PQ^*_{K^0,R,\Omega})$ with Property 33.
$\Rightarrow$) Since all acceptable solutions are within the interval between both fixed points (Proposition 35), if these are equal then the interval contains only one object which is the only acceptable solution.
$\Leftarrow$) If there is only one acceptable solution, then $\mathrm{lfp}(EF^*_{K^0,R,\Omega}) = \mathrm{gfp}(PQ^*_{K^0,R,\Omega})$. □

Proposition 35 together with the preamble of the proof of Proposition 31 determine that $[\mathrm{lfp}(EF^*)\ \mathrm{gfp}(PQ^*)]$ is the smallest interval in which $\mathrm{fp}(EF^*) \cap \mathrm{fp}(PQ^*)$ is included since its bounds are acceptable solutions. However, acceptable solutions do not cover the whole interval: the converse of Proposition 35 does not hold in general as shown by the counter-example 10.

Example 10 (Non coverage in RCA). *In the example of Section 3.2,* $\mathrm{lfp}(EF^*) = \{\langle K_1^1, L_1^1\rangle, \langle K_2^1, L_2^1\rangle\}$ *and* $\mathrm{gfp}(PQ^*) = \{\langle K_1^\star, L_1^\star\rangle, \langle K_2^\star, L_2^\star\rangle\}$*. The family* $\{\langle K_1^\#, L_1^\#\rangle, \langle K_2^\#, L_2^\#\rangle\}$ *of Figure 6 belongs to* $[\mathrm{lfp}(EF^*)\ \mathrm{gfp}(PQ^*)]$*, but not to* $\mathrm{fp}(EF^*) \cap \mathrm{fp}(PQ^*)$ *as mentioned in Example 7. Figure 9 shows that 12 out of 16 elements of the interval are in this situation: only 4 belong to* $\mathrm{fp}(EF^*) \cap \mathrm{fp}(PQ^*)$*.*

This interval may be thought of as an approximation of the situation described by the initial context K^0. For some purposes, this may be sufficient. However, it may also be interesting to navigate within the set $\mathrm{fp}(EF^*) \cap \mathrm{fp}(PQ^*)$ of fixed points or to compute it.

In order to find the elements of $\mathrm{fp}(EF^*) \cap \mathrm{fp}(PQ^*)$, the closure of EF^* and PQ^*, $EF^{*\infty}$ and $PQ^{*\infty}$, can be used as functions which maps elements of $\mathscr{O}$ into families of context-lattice pairs in $\mathrm{fp}(EF^*)$ and $\mathrm{fp}(PQ^*)$, respectively. Moreover, Properties 33 and 32 entail that $PQ^{*\infty} \circ EF^{*\infty}$ and $EF^{*\infty} \circ PQ^{*\infty}$ map any element of $\mathscr{O}$ into an acceptable family of context-lattice pairs in $\mathrm{fp}(EF^*) \cap \mathrm{fp}(PQ^*)$. Hence, the set of acceptable solutions are those elements in the image of $\mathscr{O}$ by the composition of these two closure operations, in any order.

Property 36. $\mathrm{Im}(PQ^{*\infty} \circ EF^{*\infty}) = \mathrm{fp}(EF^*) \cap \mathrm{fp}(PQ^*) = \mathrm{Im}(EF^{*\infty} \circ PQ^{*\infty})$.

Proof. We show it for $PQ^{*\infty} \circ EF^{*\infty}$, the other part is dual:

$\subseteq$ By definition, $\mathrm{Im}(PQ^{*\infty} \circ EF^{*\infty}) \subseteq \mathrm{Im}(PQ^{*\infty}) = \mathrm{fp}(PQ^*)$. Moreover, $\mathrm{Im}(EF^{*\infty}) = \mathrm{fp}(EF^*)$, but by Property 33, if $O \in \mathrm{fp}(EF^*)$, then $PQ^{*\infty}(O) \in \mathrm{fp}(EF^*)$. Hence, $\mathrm{Im}(PQ^{*\infty} \circ EF^{*\infty}) \subseteq \mathrm{fp}(EF^*) \cap \mathrm{fp}(PQ^*)$.

$\supseteq$ $\forall O \in \mathrm{fp}(PQ^*) \cap \mathrm{fp}(EF^*)$, $O \in \mathrm{fp}(EF^*)$, thus $EF^{*\infty}(O) = O$ and $O \in \mathrm{fp}(PQ^*)$, thus $PQ^{*\infty}(O) = O$. Hence, $O = PQ^{*\infty}(EF^{*\infty}(O)) = PQ^{*\infty} \circ EF^{*\infty}(O) \in \mathrm{Im}(PQ^{*\infty} \circ EF^{*\infty})$ and consequently $\mathrm{fp}(EF^*) \cap \mathrm{fp}(PQ^*) \subseteq \mathrm{Im}(PQ^{*\infty} \circ EF^{*\infty})$. □

In addition, these functions are monotone and idempotent.

Property 37. $PQ^{*\infty} \circ EF^{*\infty}$ *(resp.* $EF^{*\infty} \circ PQ^{*\infty}$*) is order-preserving and idempotent:*

$$\forall O, O' \in \mathscr{O}, O \preceq O' \Rightarrow (PQ^{*\infty} \circ EF^{*\infty})(O) \preceq (PQ^{*\infty} \circ EF^{*\infty})(O'), \qquad \text{(Monotony)}$$

$$(PQ^{*\infty} \circ EF^{*\infty}) \circ (PQ^{*\infty} \circ EF^{*\infty})(O) = PQ^{*\infty} \circ EF^{*\infty}(O). \qquad \text{(Idempotence)}$$

Proof. We prove it for $PQ^{*\infty} \circ EF^{*\infty}$, the $EF^{*\infty} \circ PQ^{*\infty}$ case is strictly dual.

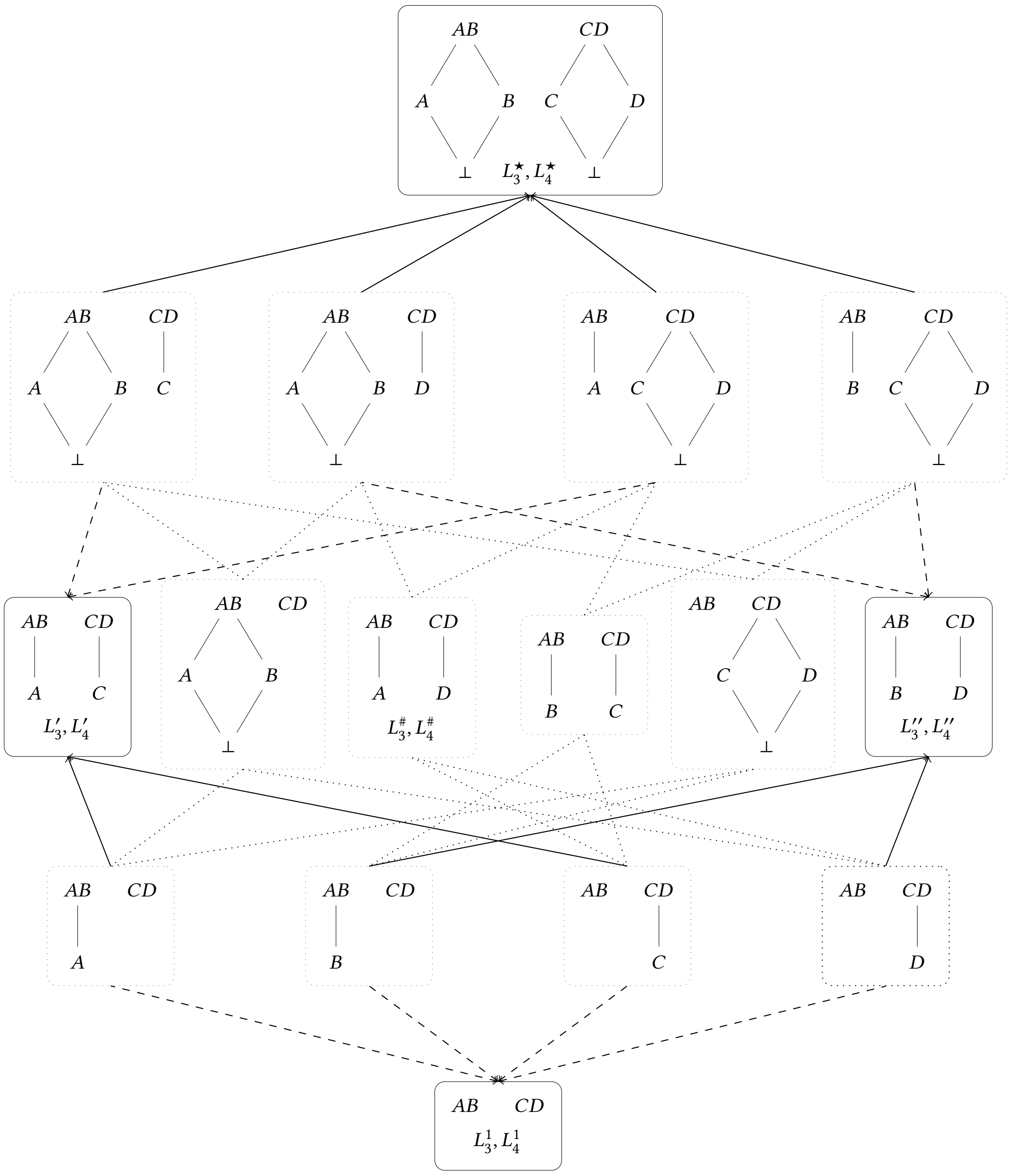


Fig. 9. All the families of lattices belonging to $[\mathrm{lfp}(EF^*)\ \mathrm{gfp}(PQ^*)]$ in the example of Section 3.2. Those in $\mathrm{fp}(EF^*) \cap \mathrm{fp}(PQ^*)$ are within solid boxes. As usual, only direct edges are displayed. Solid arrows show direct application of EF^* and dashed arrows show direct application of PQ^*. Dotted arrows are order relations not corresponding to EF^* or PQ^* applications.

Monotony is obtained as the combination of order-preservation of the two functions: $O \preceq O'$, hence $EF^{*\infty}(O) \preceq EF^{*\infty}(O')$, and thus $PQ^{*\infty} \circ EF^{*\infty}(O) \preceq PQ^{*\infty} \circ EF^{*\infty}(O')$ (applying Property 23 twice).

Idempotence is obtained from Property 36: $\forall O \in \mathscr{O}$, $PQ^{*\infty} \circ EF^{*\infty}(O) \in \text{fp}(EF^*) \cap \text{fp}(PQ^*)$, hence $PQ^{*\infty} \circ EF^{*\infty}(O) = O$ and $PQ^{*\infty} \circ EF^{*\infty} \circ PQ^{*\infty} \circ EF^{*\infty}(O) = O$, thus $PQ^{*\infty} \circ EF^{*\infty} \circ PQ^{*\infty} \circ EF^{*\infty}(O) = PQ^{*\infty} \circ EF^{*\infty}(O)$. □

The monotony of these functions entails that $\text{fp}(EF^*) \cap \text{fp}(PQ^*)$ is a complete lattice:

Proposition 38. $\langle \text{fp}(EF^*) \cap \text{fp}(PQ^*), \preceq \rangle$ *is a complete sublattice of* $\langle \mathscr{O}, \preceq \rangle$.

Proof. $\text{fp}(EF^*) \cap \text{fp}(PQ^*) = \text{Im}(PQ^{*\infty} \circ EF^{*\infty})$ (Property 36) and $\text{Im}(PQ^{*\infty} \circ EF^{*\infty}) = \text{fp}(PQ^{*\infty} \circ EF^{*\infty})$ due to idempotence (Property 37), hence the Knaster-Tarski theorem can be applied based on Property 37 (monotony), concluding that it is a complete lattice. It is included in $\mathscr{O}$, thus this is a sublattice of $\langle \mathscr{O}, \preceq \rangle$. □

This is illustrated by Example 11.

Example 11 (Interval lattice). *Figure 9 shows all elements of* $[\text{lfp}(EF^*)\ \text{gfp}(PQ^*)]$ *for the example of Section 3.2. It can be observed that* $\langle \text{fp}(EF^*) \cap \text{fp}(PQ^*), \preceq \rangle$ *is a proper sublattice of* $\mathscr{O}$. *Actually only 4 out of 16 possible objects in the interval are acceptable.*

In the figure, direct edges corresponding to EF^* *or* PQ^**, from lattice pairs of level 2 and 4, are drawn in solid or dashed, respectively. All the objects of level 3 that are not comparable with the two intermediate fixed points map to the extrema of the interval and thus* EF^* *are* PQ^* *are not displayed.*

However, the functions $PQ^{*\infty} \circ EF^{*\infty}$ and $PQ^{*\infty} \circ EF^{*\infty}$ are not necessarily extensive, nor anti-extensive (see Figure 10 and Example 12, p. 34). Hence, they would not be closure operations.

For any family of context-lattice pairs within the fixed points, i.e. $\text{fp}(EF^*)$ *or* $\text{fp}(PQ^*)$ (the vertically or horizontally stripped area of Figure 10), the two functions are equal.

Property 39. $\forall O \in \text{fp}(EF^*) \cup \text{fp}(PQ^*), PQ^{*\infty} \circ EF^{*\infty}(O) = EF^{*\infty} \circ PQ^{*\infty}(O)$.

Proof. For any lattice O belonging to $\text{fp}(EF^*) \cap \text{fp}(PQ^*)$, $PQ^*(O) = EF^*(O) = O$, hence $PQ^{*\infty} \circ EF^{*\infty}(O) = EF^{*\infty} \circ PQ^{*\infty}(O) = O$. Similarly, for any lattice O belonging to $\text{fp}(EF^*)$, then $EF^{*\infty}(O) = O$, so $PQ^{*\infty} \circ EF^{*\infty}(O) = PQ^{*\infty}(O)$. However, by Property 32, since $O \in \text{fp}(EF^*)$, $PQ^{*\infty}(O) \in \text{fp}(EF^*)$. This means that $EF^{*\infty} \circ PQ^{*\infty}(O) = PQ^{*\infty}(O)$ as well. The same can be proved for $O \in \text{fp}(PQ^*)$ with Property 33. □

What is actually shown by the proof of Property 39 is that:

$$\text{if } O \in \text{fp}(EF^*) \text{ then } PQ^{*\infty} \circ EF^{*\infty}(O) = EF^{*\infty} \circ PQ^{*\infty}(O) = PQ^{*\infty}(O),$$
$$\text{if } O \in \text{fp}(PQ^*) \text{ then } PQ^{*\infty} \circ EF^{*\infty}(O) = EF^{*\infty} \circ PQ^{*\infty}(O) = EF^{*\infty}(O).$$

In particular, this applies to the bounds of $\text{fp}(EF^*) \cup \text{fp}(PQ^*)$:

Property 40.

$$EF^{*\infty} \circ PQ^{*\infty}(\text{gfp}(EF^*)) = PQ^{*\infty} \circ EF^{*\infty}(\text{gfp}(EF^*)) = \text{gfp}(PQ^*),$$

and

$$PQ^{*\infty} \circ EF^{*\infty}(\text{lfp}(PQ^*)) = EF^{*\infty} \circ PQ^{*\infty}(\text{lfp}(PQ^*)) = \text{lfp}(EF^*).$$

Proof. The first part of these equations are consequences of Property 39, since $\text{gfp}(EF^*)$ and $\text{lfp}(PQ^*)$ belong to $\text{fp}(EF^*)$ and $\text{fp}(PQ^*)$, respectively. The second part is due to $PQ^{*\infty} \circ EF^{*\infty}(\text{gfp}(EF^*)) = PQ^{*\infty}(\text{gfp}(EF^*))$ and $EF^{*\infty} \circ PQ^{*\infty}(\text{lfp}(PQ^*)) = EF^{*\infty}(\text{lfp}(PQ^*))$ for the same reason that $\text{gfp}(EF^*) \in \text{fp}(EF^*)$ and $\text{lfp}(PQ^*) \in \text{fp}(PQ^*)$, respectively. Property 34 shows that the second terms correspond to $\text{gfp}(PQ^*)$ and $\text{lfp}(EF^*)$, respectively. □

Example 12 shows that $EF^{*\infty} \circ PQ^{*\infty}(O^{\#}) \prec O^{\#} \prec PQ^{*\infty} \circ EF^{*\infty}(O^{\#})$ hence that the equality does not hold in general.

Example 12 (Counterexample to equality in RCA). *Consider Example 3.2 (p. 12),* $\text{lfp}(EF^*) = O^1_{12} = \{\langle K^1_1, L^1_1\rangle, \langle K^1_2, L^1_2\rangle\}$ *and* $\text{gfp}(PQ^*) = O^{\star}_{12} = \{\langle K^{\star}_1, L^{\star}_1\rangle, \langle K^{\star}_2, L^{\star}_2\rangle\}$. $O^{\#}_{12} = \{\langle K^{\#}_1, L^{\#}_1\rangle, \langle K^{\#}_2, L^{\#}_2\rangle\}$ *belongs to* $[\text{lfp}(EF^*)\ \text{gfp}(PQ^*)]$ *but not to* $\text{fp}(EF^*) \cap \text{fp}(PQ^*)$. *It happens that* $EF^*(O^{\#}_{12}) = O^{\star}_{12}$ *and* $PQ^*(O^{\#}_{12}) = O^1_{12}$*, hence* $PQ^{*\infty} \circ EF^{*\infty}(O^{\#}_{12}) = PQ^* \circ EF^*(O^{\#}_{12}) = O^{\star}_{12}$ *and* $EF^{*\infty} \circ PQ^{*\infty}(O^{\#}_{12}) = EF^* \circ PQ^*(O^{\#}_{12}) = O^1_{12}$. *These two objects are not isomorphic. What can be said, in this case, is that* $EF^{*\infty} \circ PQ^{*\infty}(O^{\#}_{12}) \prec PQ^{*\infty} \circ EF^{*\infty}(O^{\#}_{12})$. *This is the result of σ which may add needed support (for C and B from A and D) and π which may suppress unsupported concepts (A missing C and D missing B).*

It is not necessary that the results of the closure be the bounds of the interval as is shown for any object of the second and fourth lines of the lattice of Figure 9.

It may be that, as illustrated by Example 12, when $PQ^{*\infty}$ is first applied, it suppresses non-supported attributes which cannot be recovered by scaling. Conversely, $EF^{*\infty}$ applied first may scale attributes ($\exists p.D$ in Example 12) which generate new concepts (B) supporting previously non-supported concept (D). These will not be suppressed any more.

Property 41 shows that, in addition, there is still a homomorphism between the two resulting objects.

Property 41. $\forall O \in \mathscr{O}, EF^{*\infty} \circ PQ^{*\infty}(O) \preceq PQ^{*\infty} \circ EF^{*\infty}(O)$.

Proof. $PQ^{*\infty}(O) \preceq O$ by Property 25. But $EF^{*\infty}$ is monotone (Property 23), hence $EF^{*\infty} \circ PQ^{*\infty}(O) \preceq EF^{*\infty}(O)$. $PQ^{*\infty}$ is also monotone (Property 23), thus $PQ^{*\infty} \circ EF^{*\infty} \circ PQ^{*\infty}(O) \preceq PQ^{*\infty} \circ EF^{*\infty}(O)$. However, $PQ^{*\infty} \circ EF^{*\infty}(O) \in \text{fp}(EF^*) \cup \text{fp}(PQ^*)$ so $PQ^{*\infty} \circ EF^{*\infty}(O) = EF^{*\infty} \circ PQ^{*\infty}(O)$ (Property 39). Thus, $PQ^{*\infty} \circ EF^{*\infty} \circ PQ^{*\infty}(O) = EF^{*\infty} \circ PQ^{*\infty} \circ PQ^{*\infty}(O) = EF^{*\infty} \circ PQ^{*\infty}(O)$. This means that $EF^{*\infty} \circ PQ^{*\infty}(O) \preceq PQ^{*\infty} \circ EF^{*\infty}(O)$. □

Alternative proof. The same reasoning can be held from $O \preceq EF^{*\infty}(O)$ (Property 25) and $EF^{*\infty}$ and $PQ^{*\infty}$ being monotone (Property 23). Hence, $PQ^{*\infty}(O) \preceq PQ^{*\infty} \circ EF^{*\infty}(O)$ and $EF^{*\infty} \circ PQ^{*\infty}(O) \preceq EF^{*\infty} \circ PQ^{*\infty} \circ EF^{*\infty}(O)$. But, $PQ^{*\infty} \circ EF^{*\infty}(O) \in \text{fp}(EF^*) \cup \text{fp}(PQ^*)$, so $PQ^{*\infty} \circ EF^{*\infty}(O) = EF^{*\infty} \circ PQ^{*\infty}(O)$ (Property 39). Thus $EF^{*\infty} \circ PQ^{*\infty} \circ EF^{*\infty}(O) = PQ^{*\infty} \circ EF^{*\infty} \circ EF^{*\infty}(O) = PQ^{*\infty} \circ EF^{*\infty}(O)$. Hence, $EF^{*\infty} \circ PQ^{*\infty}(O) \preceq PQ^{*\infty} \circ EF^{*\infty}(O)$. □

The alternative proof is given here to show that starting from the EF^* or PQ^* give the same result.

It is thus unclear what to do with $EF^{*\infty} \circ PQ^{*\infty}$ and $PQ^{*\infty} \circ EF^{*\infty}$ in general. For instance, if one needs an operation to map elements of $\mathscr{O}$ to $\text{fp}(EF^*) \cap \text{fp}(PQ^*)$, which one is preferable? There may be an interest in studying the interval $[EF^{*\infty} \circ PQ^{*\infty}(O)\ PQ^{*\infty} \circ EF^{*\infty}(O)]$. Does it contain only fixed points or no fixed points? Are these the image of other lattices? This question can be answered if O can be compared to these bounds (Proposition 42): the intermediate families are *not* fixed points.

Proposition 42. $\forall O \in \mathscr{O} \setminus (\text{fp}(EF^*) \cap \text{fp}(PQ^*))$*:*

- *if* $O \preceq PQ^{*\infty} \circ EF^{*\infty}(O)$*, then* $\forall O' \in [O\ PQ^{*\infty} \circ EF^{*\infty}(O)[, O' \notin \text{fp}(EF^*) \cap \text{fp}(PQ^*)$,
- *if* $EF^{*\infty} \circ PQ^{*\infty}(O) \preceq O$*, then* $\forall O' \in]EF^{*\infty} \circ PQ^{*\infty}(O), O], O' \notin \text{fp}(EF^*) \cap \text{fp}(PQ^*)$.

Proof. Considering the first item of the proposition, $O \preceq PQ^{*\infty} \circ EF^{*\infty}(O)$ can only occur if $EF^{*\infty}(O) \in \text{fp}(PQ^*)$, i.e. $PQ^{*\infty} \circ EF^{*\infty}(O) = EF^{*\infty}(O)$. Indeed, if this were not the case, then $PQ^{*\infty}$ would suppress attributes from $EF^{*\infty}(O)$. However, since $O \preceq PQ^{*\infty} \circ EF^{*\infty}(O)$, these could not be attributes from O, but only attributes added by $EF^{*\infty}$. But since $EF^{*\infty}$ only adds attributes if they are supported and it starts with attributes from O, this is not possible. Thus, if $O' \in [O\ PQ^{*\infty} \circ EF^{*\infty}(O)[$ then $O' \in [O\ EF^{*\infty}(O)[$. However, O' cannot be a fixed point for EF^* because it contains all attributes of O which would scale to generate all those of $EF^{*\infty}(O)$. Hence $O' \notin \text{fp}(EF^*) \cap \text{fp}(PQ^*)$.

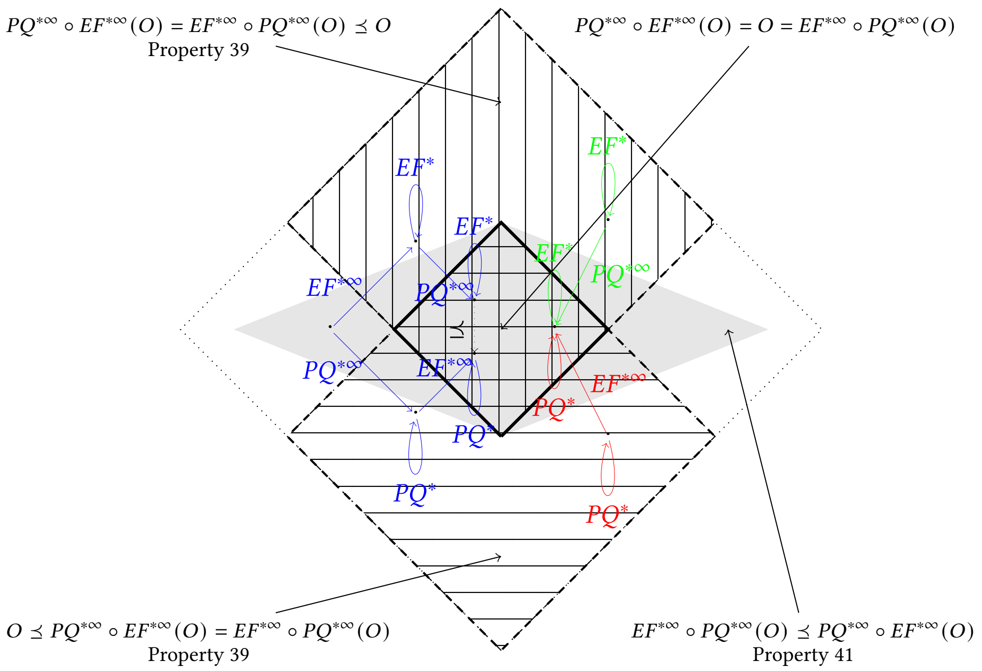

$PQ^{*\infty} \circ EF^{*\infty}(O) = EF^{*\infty} \circ PQ^{*\infty}(O) \preceq O$
Property 39
$PQ^{*\infty} \circ EF^{*\infty}(O) = O = EF^{*\infty} \circ PQ^{*\infty}(O)$
EF^*
$EF^{*\infty}$
$PQ^{*\infty}$
$\preceq$
PQ^*
$EF^{*\infty}$
$O \preceq PQ^{*\infty} \circ EF^{*\infty}(O) = EF^{*\infty} \circ PQ^{*\infty}(O)$
Property 39
$EF^{*\infty} \circ PQ^{*\infty}(O) \preceq PQ^{*\infty} \circ EF^{*\infty}(O)$
Property 41

Fig. 10. Illustration of the position of $PQ^{*\infty} \circ EF^{*\infty}(O)$ and $EF^{*\infty} \circ PQ^{*\infty}(O)$ depending on O's origin (in dotted, $\mathscr{O}$, in dashed $\text{fp}(EF^*) \cup \text{fp}(PQ^*)$, in plain $\text{fp}(EF^*) \cap \text{fp}(PQ^*)$). Colours correspond to that of Figure 8: green starting from $\text{fp}(EF^*)$, red starting from $\text{fp}(PQ^*)$, blue starting outside of them.

The second item has a similar proof: $EF^{*\infty} \circ PQ^{*\infty}(O) \preceq O$ can only occur if $PQ^{*\infty}(O) \in \text{fp}(EF^*)$, i.e. $EF^{*\infty} \circ PQ^{*\infty}(O) = PQ^{*\infty}(O)$. Indeed, if this were not the case, then $EF^{*\infty}$ would generate attributes from $PQ^{*\infty}(O)$. However, since $EF^{*\infty} \circ PQ^{*\infty}(O) \preceq O$, these could only be attributes of O which were suppressed by $PQ^{*\infty}$ due to lack of support. But this is not possible because if they lacked support in O, there is not more support for them in $PQ^{*\infty}(O)$, which only reduces O. Thus, if $O' \in]EF^{*\infty} \circ PQ^{*\infty}(O)\ O]$, then $O' \in]PQ^{*\infty}(O)\ O]$. However, O' cannot be a fixed point for PQ^* because it contains less attributes than O: if these attributes lacked supports in O, they would still lack it in O'. Hence, $O' \notin \text{fp}(EF^*) \cap \text{fp}(PQ^*)$. □

This is illustrated by Example 13:

Example 13. *In Example 12 (p. 34), $O^\#_{34} = \{\langle K^\#_3, L^\#_3\rangle, \langle K^\#_4, L^\#_4\rangle\}$ is a fixed point for neither EF^* nor PQ^*. $PQ^{*\infty} \circ EF^{*\infty}(O^\#_{34}) = EF^{*\infty}(O^\#_{34}) = O^\star_{34}$ and $EF^{*\infty} \circ PQ^{*\infty}(O^\#_{34}) = PQ^{*\infty}(O^\#_{34}) = O^1_{34}$. None of the objects in the interval between $O^\#_{34}$ and either $O^\star_{34}$ or O^1_{34} belongs to $\text{fp}(EF^*) \cap \text{fp}(PQ^*)$ as can be seen in Figure 9.*

This result cannot be generalised to the interval $]EF^{*\infty} \circ PQ^{*\infty}(O)\ PQ^{*\infty} \circ EF^{*\infty}(O)[$ as shown by Example 14.

Example 14 (The subinterval may contain fixed points). *Following Example 13, $O'_{34} = \{\langle K'_3, L'_3\rangle, \langle K'_4, L'_4\rangle\}$ belongs to $]EF^{*\infty} \circ PQ^{*\infty}(O^\#_{34})\ PQ^{*\infty} \circ EF^{*\infty}(O^\#_{34})[\ =]O^1_{34},\ O^\star_{34}[$ as can be observed in Figure 9. However, $O'_{34} \in \text{fp}(EF^*) \cap \text{fp}(PQ^*)$.*

Proposition 42 can however be useful algorithmically. Indeed, if one considers the non-acceptable objects of Figure 9 on the same line as $O^\#_{34}$, then this result identifies as non acceptable two objects on the second and fourth line without testing them.

8 Conclusion

We addressed the questions of which family of concept lattices was returned by relational concept analysis and, more generally, which such families could be considered acceptable.

This paper provides an answer to these questions by characterising the acceptable families of context-lattice pairs that describe a particular initial family of contexts as those families which are well-formed, saturated and self-supported. It identifies the results returned by relational concept analysis as the smallest element of this set. It also defines an alternative operation providing its greatest elements. The structure of the set of acceptable solutions has been further characterised.

To that extent the paper defines the set of well-formed objects $\mathscr{O}$, a function EF^*, generalising RCA, expanding a family, and a function PQ^* contracting a family. The fixed points of these functions characterise the saturated families and the self-supported families respectively. Hence, the acceptable solutions are those elements of the intersection of the fixed points of both functions ($\text{fp}(EF^*) \cap \text{fp}(PQ^*)$).

These results rely fundamentally on the finiteness of the structure and monotony of the operations. Dealing with infinite structures would jeopardise the construction of the sets of scalable relational attributes (Section 2.4.1), however as soon as the termination of the application of the operations is preserved, this should not be a problem. Non-monotonic operations could be induced by non-monotonic scaling operations. Such operations would prevent relational concept analysis to work properly and would require fully different mechanisms.

In FCA, conceptual scaling is considered as a human-driven analysis tool: a knowledgeable person could provide attributes to be scaled for describing better the data to be analysed. In RCA, scaling is used as an extraction tool, with the drawback to potentially generate many attributes. By only extracting the least fixed point, RCA avoids generating too many of them. This is useful when generating a description logic TBox because all concepts are well-defined and necessary, but other contexts may benefit from exploiting other solutions.

Beyond the minimal common acceptable lattices returned by $\underline{\text{RCA}}$ and the most detailed ones that $\overline{\text{RCA}}$ returns, algorithms may be developed for returning all acceptable solutions [2]. The characterisation of the structure of the space of acceptable solutions aims at contributing to this goal. However, our work does not provide an 'efficient' way to obtain all elements of this set.

This work also opens perspectives for helping users to identify the acceptable solution that they prefer. Beyond generating all solutions, another option is to offer users the opportunity to guide the navigation among them. The structure of admissible solutions and the associated functions may be fruitfully exploited in order to help users finding an acceptable solution featuring the concepts and attributes that they want and not unnecessary ones.

An anonymous reviewer remarked that variations of RCA, such as those based on AOC-posets [9], may receive the same treatment. This is a perspective worth pursuing, that may lead to generalise the results presented here.

Finally, the position of relational concept analysis with respect to formal concept analysis and Galois connections would be worth investigating. On the one hand, this work shows that, contrary to other extensions that use scaling to encode a problem within FCA, RCA cannot be encoded in FCA. Indeed, RCA admits various fixed points contrary to FCA. RCA is not just the application of a product or sequence of FCA, but relations between contexts introduce constraints between them leading to the possibility of alternative fixed points. Hence, an encoding would not be direct, so that it provides RCA solutions directly. On the other hand, other generalisations of FCA get closer to general Galois connections by extending the structure of attributes. The open question is whether RCA is another instance of a Galois connection extending FCA or if these two need a common generalisation.

Acknowledgments

This work has been partially funded by the ANR Elker project (ANR-17-CE23-0007-01). The author thanks Petko Valtchev for comments and suggestions on an earlier version of this work, Marianne Huchard for taking the pain to describe the trick used for forcing discrimination, Amedeo Napoli for explaining me RCA (and so many other things), Jérôme David for reviewing the text, and Philippe Besnard for pointing to the Knaster-Tarski theorem. Anonymous reviews helped clarifying the paper.

References

[1] M. Atencia, J. David, J. Euzenat, A. Napoli, and J. Vizzini. 2020. Link key candidate extraction with relational concept analysis. *Discrete applied mathematics*, 273, 2–20. DOI: 10.1016/J.DAM.2019.02.012.

[2] M. Atencia, J. David, J. Euzenat, A. Napoli, and J. Vizzini. 2021. Relational concept analysis for circular link key extraction. Deliverable 1.2. Elker. 57 pp. https://moex.inria.fr/files/reports/elker-1.2.pdf.

[3] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, (Eds.) 2007. *The description logic handbook: theory, implementations and applications*. (2nd ed.). Cambridge University Press, Cambridge (UK). DOI: 10.1017/cbo9780511711787.

[4] F. Baader and F. Distel. 2008. A finite basis for the set of $\mathscr{EL}$-implications holding in a finite model. In *Proc. 6th International Conference on Formal Concept Analysis (ICFCA), Montréal (CA)* (Lecture notes in computer science). Vol. 4933, 46–61. DOI: 10.1007/978-3-540-78137-0_4.

[5] R. Belohlávek. 2008. Introduction to formal concept analysis. Tech. rep. Univerzita Palackého, Olomouc (CZ). http://belohlavek.inf.upol.cz/vyuka/IntroFCA.pdf.

[6] A. Braud, X. Dolques, M. Huchard, and F. Le Ber. 2018. Generalization effect of quantifiers in a classification based on relational concept analysis. *Knowledge-based systems*, 160, 119–135. DOI: 10.1016/J.KNOSYS.2018.06.011.

[7] L. Chaudron and N. Maille. 2000. Generalized formal concept analysis. In *Proc. 8th International Conference on Conceptual Structures (ICCS), Darmstadt (DE)* (Lecture notes in computer science). Vol. 1867, 357–370. DOI: 10.1007/10722280_25.

[8] X. Dolques, M. Huchard, C. Nebut, and P. Reitz. 2012. Fixing generalization defects in UML use case diagrams. *Fundamenta informaticae*, 115, 4, 327–356. DOI: 10.3233/FI-2012-658.

[9] X. Dolques, F. Le Ber, and M. Huchard. 2013. AOC-posets: a scalable alternative to concept lattices for relational concept analysis. In *Proc. of the 10th international conference on concept lattices and their applications (CLA), La Rochelle (FR)*, 129–140. https://ceur-ws.org/Vol-1062/paper11.pdf.

[10] J. Euzenat. 2021. Fixed-point semantics for barebone relational concept analysis. In *Proc. 16th international conference on formal concept analysis (ICFCA), Strasbourg (FR)* (Lecture notes in computer science). Vol. 12773, 20–37. DOI: 10.1007/978-3-030-77867-5_2.

[11] J. Euzenat. 2023. Stepwise functional refoundation of relational concept analysis. Research report 9518. INRIA. DOI: 10.48550/arXiv.2310.06441.

[12] S. Ferré and P. Cellier. 2020. Graph-FCA: an extension of formal concept analysis to knowledge graphs. *Discrete applied mathematics*, 273, 81–102. DOI: 10.1016/J.DAM.2019.03.003.

[13] S. Ferré and O. Ridoux. 2000. A logical generalization of formal concept analysis. In *Proc. 8th International Conference on Conceptual Structures (ICCS), Darmstadt (DE)* (Lecture notes in computer science). Vol. 1867, 371–384. DOI: 10.1007/10722280_26.

[14] B. Ganter and S. Kuznetsov. 2001. Pattern structures and their projections. In *Proc. 9th International conference on conceptual structures (ICCS), Stanford (CA US)* (Lecture Notes in Computer Science). Vol. 2120, 129–142. DOI: 10.1007/3-540-44583-8_10.

[15] B. Ganter, G. Stumme, and R. Wille, (Eds.) 2005. *Formal concept analysis: foundations and applications*. Springer, Heildelberg (DE). DOI: 10.1007/978-3-540-31881-1.

[16] B. Ganter and R. Wille. 1999. *Formal Concept Analysis: mathematical foundations*. Springer, Berlin (DE). DOI: 10.1007/978-3-642-59830-2.

[17] J.-L. Guigues and V. Duquenne. 1986. Familles minimales d'implications informatives résultant d'un tableau de données binaires. *Mathématiques et sciences humaines*, 95, 5–18. https://www.numdam.org/item/MSH_1986__95__5_0/.

[18] R. Guimarães, A. Ozaki, C. Persia, and B. Sertkaya. 2023. Mining $\mathscr{EL}^{\perp}$ bases with adaptable role depth. *Journal of artificial intelligence research*, 76, 119–135. DOI: 10.1613/jair.1.13777.

[19] M. Huchard, M. Rouane Hacène, C. Roume, and P. Valtchev. 2007. Relational concept discovery in structured datasets. *Annals of Mathematics and Artificial Intelligence*, 49, 1, 39–76. DOI: 10.1007/s10472-007-9056-3.

[20] P. Keip, S. Ferré, A. Gutierrez, M. Huchard, P. Silvie, and P. Martin. 2020. Practical comparison of FCA extensions to model indeterminate value of ternary data. In *Proc. 15th International Conference on Concept Lattices and Their Applications (CLA), Tallinn (EE)* (CEUR Workshop Proceedings). Vol. 2668, 197–208. https://ceur-ws.org/Vol-2668/paper15.pdf.

[21] J. Kötters. 2013. Concept lattices of a relational structure. In *Proc. 20th International Conference on Conceptual Structures (ICCS), Mumbay (IN)* (Lecture Notes in Computer Science). Vol. 7735, 301–310. DOI: 10.1007/978-3-642-35786-2_23.

[22] R. Missaoui, L. Kwuida, and T. Abdessalem, (Eds.) 2022. *Complex data analytics with formal concept analysis*. Springer, Cham (CH). DOI: 10.1007/978-3-030-93278-7.

[23] B. Nebel. 1990. *Reasoning and revision in hybrid representation systems. Lecture Notes in Artificial Intelligence*. Vol. 422. Springer, Berlin (DE). DOI: 10.1007/BFB0016445.

[24] A. Ouzerdine, A. Braud, X. Dolques, M. Huchard, and F. Le Ber. 2019. Adjusting the exploration flow in relational concept analysis – an experience on a watercourse quality dataset. In *Advances in Knowledge Discovery and Management* (Studies in Computational

Intelligence). R. Jaziri, A. Martin, M.-C. Rousset, L. Boudjeloud-Assala, and F. Guillet, (Eds.) Vol. 1004. Springer, 175–198. DOI: 10.1007/978-3-030-90287-2_9.

[25] S. Prediger. 1997. Logical scaling in formal concept analysis. In *Proc. 5th International Conference on Conceptual Structures (ICCS), Seattle (WA US)* (Lecture Notes in Computer Science). Vol. 1257, 332–341. DOI: 10.1007/BFB0027881.

[26] M. Rouane Hacène, M. Huchard, A. Napoli, and P. Valtchev. 2013. Relational concept analysis: mining concept lattices from multi-relational data. *Annals of Mathematics and Artificial Intelligence*, 67, 1, 81–108. DOI: 10.1007/S10472-012-9329-3.

[27] M. Rouane Hacène, M. Huchard, A. Napoli, and P. Valtchev. 2013. Soundness and completeness of relational concept analysis. In *Proc. 11th International Conference on Formal Concept Analysis (ICFCA), Dresden (DE)* (Lecture notes in computer science). Vol. 7880, 228–243. DOI: 10.1007/978-3-642-38317-5_15.

[28] A. Tarski. 1955. A lattice-theoretical fixpoint theorem and its applications. *Pacific journal of mathematics*, 5, 2, 285–309. DOI: 10.2140/pjm.1955.5.285.

[29] M. Wajnberg. 2020. *Analyse relationnelle de concepts: une méthode polyvalente pour l'extraction de connaissance*. Ph.D. Dissertation. Université du Québec à Montréal & Université de Lorraine. https://hal.science/tel-03042085.

Received 18 December 2023; revised 18 December 2024; accepted 26 April 2025