Relation-Algebraic Verification of Prim's Minimum Spanning Tree Algorithm

Walter Guttmann(✉)

Department of Computer Science and Software Engineering,
University of Canterbury, Christchurch, New Zealand
walter.guttmann@canterbury.ac.nz

Abstract. We formally prove the correctness of Prim's algorithm for computing minimum spanning trees. We introduce new generalisations of relation algebras and Kleene algebras, in which most of the proof can be carried out. Only a small part needs additional operations, for which we introduce a new algebraic structure. We instantiate these algebras by matrices over extended reals, which model the weighted graphs used in the algorithm. Many existing results from relation algebras and Kleene algebras generalise from the relation model to the weighted-graph model with no or small changes. The overall structure of the proof uses Hoare logic. All results are formally verified in Isabelle/HOL heavily using its integrated automated theorem provers.

1 Introduction

A well-known algorithm commonly attributed to Prim [43] – and independently discovered by Jarník [27] and Dijkstra [17] – computes a minimum spanning tree in a weighted undirected graph. It starts with an arbitrary root node, and constructs a tree by repeatedly adding an edge that has minimal weight among the edges connecting a node in the tree with a node not in the tree. The iteration stops when there is no such edge, at which stage the constructed tree is a minimum spanning tree of the component of the graph that contains the root (which is the whole graph if it is connected).

The aim of this paper is to demonstrate the applicability of relation-algebraic methods for verifying the correctness of algorithms on weighted graphs. Accordingly, we will use an implementation of Prim's algorithm close to the above abstraction level. Since its discovery many efficient implementations of this and other spanning tree algorithms have been developed; for example, see the two surveys [21,35]. These implementations typically rely on specific data structures, which can be introduced into a high-level algorithm by means of data refinement; for example, see [4]. We do not pursue this in the present paper.

Relation-algebraic methods have been used to develop algorithms for unweighted graphs; for example, see [4,5,7]. This works well because such a graph can be directly represented as a relation; an adjacency matrix is a Boolean matrix. Weighted graphs do not have a direct representation as a binary relation.

© Springer International Publishing AG 2016
A. Sampaio and F. Wang (Eds.): ICTAC 2016, LNCS 9965, pp. 51–68, 2016.
DOI: 10.1007/978-3-319-46750-4_4

Previous relational approaches to weighted graphs therefore use many-sorted representations such as an incidence matrix and a weight function. In this paper, we directly work with a matrix of weights.

In the context of fuzzy systems, relations have been generalised from Boolean matrices to matrices over the real interval $[0, 1]$ or over arbitrary complete distributive lattices [19]. The underlying idea is to extend qualitative to quantitative methods; see [41] for another instance based on automata. We propose to use matrices over lattices to model weighted graphs, in particular in graph algorithms. Previous work based on semirings and Kleene algebras deals well with path problems in graphs [20]. We combine these algebras with generalisations of relation algebras to tackle the minimum spanning tree problem.

Tarski's relation algebras [46], which capture Boolean matrices, have been generalised to Dedekind categories to algebraically capture fuzzy relations [30]; these categories are also known as locally complete division allegories [18]. In the present paper we introduce a new generalisation – Stone relation algebras – which maintains the signature of relation algebras and weakens the underlying Boolean algebra structure to Stone algebras. We show that matrices over extended reals are instances of Stone relation algebras and of Kleene algebras, and can be used to represent weighted graphs.

Most of the correctness proof of Prim's minimum spanning tree algorithm can be carried out in these general algebras. Therefore, most of our results hold for many instances, not just weighted graphs. A small part of the correctness proof uses operations beyond those available in relation algebras and in Kleene algebras, namely for summing edge weights and identifying minimal edges. We capture essential properties of these operations in a new algebraic structure.

With this approach we can apply well-developed methods and concepts of relation algebras and Kleene algebras to reason about weighted graphs in a new, more direct way. The contributions of this paper are:

- Stone relation algebras, a new algebraic structure that generalises relation algebras but maintains their signature. Many theorems of relation algebras already hold in these weaker algebras. Combined with Kleene algebras, they form a general yet expressive setting for most of the correctness proof of the minimum spanning tree algorithm.
- A new algebra that extends Stone-Kleene relation algebras by dedicated operations and axioms for finding minimal edges and for computing the total weight of a graph.
- Models of the above algebras, including weighted graphs represented by matrices over extended reals. This includes a formalisation of Conway's automata-based construction for the Kleene star of a matrix.
- A Hoare-logic correctness proof of Prim's minimum spanning tree algorithm entirely based on the above algebras.
- Isabelle/HOL theories that formally verify all of the above and all results in and about the algebras stated in the present paper. Proofs are omitted in this paper and can be found in the Isabelle/HOL theory files available at http://www.csse.canterbury.ac.nz/walter.guttmann/algebra/.

Section 2 recalls basic algebraic structures and introduces Stone relation algebras, which we use to represent weighted graphs. They are extended by the Kleene star operation to describe reachability in graphs in Sect. 3. Operations for summing weights and for finding their minimum are added in Sect. 4. In this setting, Sect. 5 presents the minimum spanning tree algorithm, details aspects of its correctness proof and shows how the proof uses the various algebras. Related work is discussed in Sect. 6.

2 Stone Relation Algebras

In this section we introduce Stone relation algebras, which generalise relation algebras so as to model not just Boolean matrices but matrices over arbitrary numbers required to represent weighted graphs. Each entry in such a matrix is taken from the set of real numbers $\mathbb{R}$ extended with a bottom element $\bot$ and a top element $\top$; let $\mathbb{R}' = \mathbb{R} \cup \{\bot, \top\}$. If the entry in row i and column j of the matrix is $\bot$, this means there is no edge from node i to node j. If the entry is a real number, there is an edge with that weight. An entry of $\top$ is used to record the presence of an edge without information about its weight; see below. The order $\leq$ and the operations max and min on $\mathbb{R}'$ are extended from reals so that $\bot$ is the $\leq$-least element and $\top$ is the $\leq$-greatest element. To work with extended reals (weights) and matrices of extended reals (weighted graphs) we use the following well-known algebraic structures [9,12,22].

Definition 1. A *bounded semilattice* is an algebraic structure $(S, \sqcup, \bot)$ where $\sqcup$ is associative, commutative and idempotent and has unit $\bot$:

$$x \sqcup (y \sqcup z) = (x \sqcup y) \sqcup z \qquad x \sqcup y = y \sqcup x \qquad x \sqcup x = x \qquad x \sqcup \bot = x$$

A *bounded distributive lattice* is an algebraic structure $(S, \sqcup, \sqcap, \bot, \top)$ where $(S, \sqcup, \bot)$ and $(S, \sqcap, \top)$ are bounded semilattices and the following distributivity and absorption axioms hold:

$$\begin{aligned} x \sqcup (y \sqcap z) &= (x \sqcup y) \sqcap (x \sqcup z) \qquad & x \sqcup (x \sqcap y) &= x \\ x \sqcap (y \sqcup z) &= (x \sqcap y) \sqcup (x \sqcap z) \qquad & x \sqcap (x \sqcup y) &= x \end{aligned}$$

The *lattice order* is given by

$$x \leq y \;\Leftrightarrow\; x \sqcup y = y$$

A *distributive p-algebra* $(S, \sqcup, \sqcap, \overline{\ }, \bot, \top)$ expands a bounded distributive lattice $(S, \sqcup, \sqcap, \bot, \top)$ with a pseudocomplement operation $\overline{\ }$ satisfying the equivalence

$$x \sqcap y = \bot \;\Leftrightarrow\; x \leq \overline{y}$$

This means that $\overline{y}$ is the $\leq$-greatest element whose meet with y given by $\sqcap$ is $\bot$. A *Stone algebra* is a distributive p-algebra satisfying the equation

$$\overline{x} \sqcup \overline{\overline{x}} = \top$$

An element $x \in S$ is *regular* if $\overline{\overline{x}} = x$. A *Boolean algebra* is a Stone algebra whose elements are all regular.

Note that there is no obvious way to introduce a Boolean complement on $\mathbb{R}'$, which is why we use the weaker Stone algebras. We obtain the following consequences for Stone algebras; in particular, extended reals form a Stone algebra and so do matrices over extended reals. See [20] for similar matrix semirings and the max-min semiring of extended reals. The set of square matrices with indices from a set A and entries from a set S is denoted by $S^{A\times A}$. It represents a graph with node set A and edge weights taken from S.

Theorem 1. *Let $(S, \sqcup, \sqcap, \overline{\ }, \bot, \top)$ be a Stone algebra and let A be a set.*

1. *The regular elements of S form a Boolean algebra that is a subalgebra of S [22].*
2. *$(S^{A\times A}, \sqcup, \sqcap, \overline{\ }, \bot, \top)$ is a Stone algebra, where the operations $\sqcup$, $\sqcap$, $\overline{\ }$, $\bot$, $\top$ and the lattice order $\leq$ are lifted componentwise.*
3. *$(\mathbb{R}', \max, \min, \overline{\ }, \bot, \top)$ is a Stone algebra with*

$$\overline{x} = \begin{cases} \top \text{ if } x = \bot \\ \bot \text{ if } x \neq \bot \end{cases}$$

and the order $\leq$ on $\mathbb{R}'$ as the lattice order.

The regular elements of the Stone algebra $\mathbb{R}'$ are $\bot$ and $\top$. In particular, applying the pseudocomplement operation $\overline{\ }$ twice maps $\bot$ to itself and every other element to $\top$. Applying $\overline{\ }$ twice to a matrix over $\mathbb{R}'$, which represents a weighted graph, yields a matrix over $\{\bot, \top\}$ that represents the structure of the graph forgetting the weights. A related operation called the 'support' of a matrix is discussed in [33]; it works on matrices over natural numbers and maps 0 to 0 and each non-zero entry to 1. Relations are used to describe the 'shape' of a matrix of complex numbers in [16]; a shape represents a superset of the non-zero entries of a matrix, but an operator to obtain the non-zero entries is not discussed there.

The matrices over $\{\bot, \top\}$ are the regular elements of the matrix algebra and form a subalgebra of it. This situation, shown in Fig. 1, is analogous to that of partial identities – subsets of the identity relation used to represent

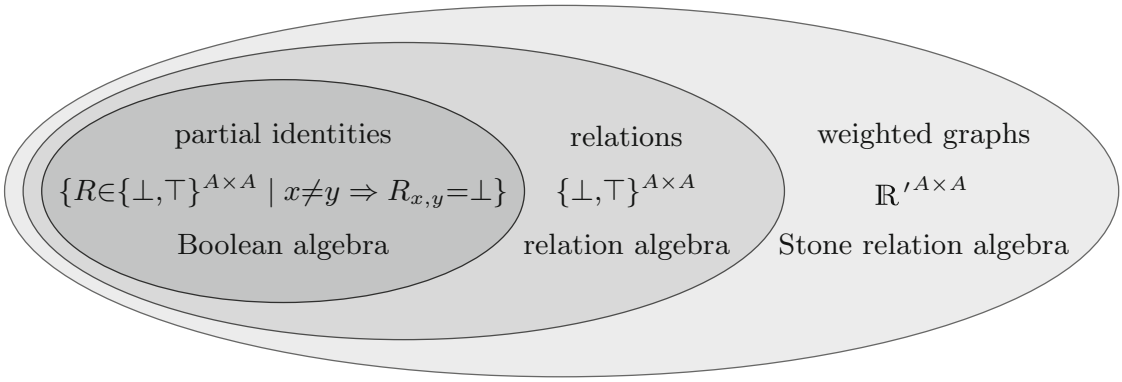

Fig. 1. Relations form a substructure of weighted graphs as partial identities form a substructure of relations

conditions in computations – which form a substructure of the encompassing relation algebra. In both cases, the substructure can be obtained as the image of a closure operation. The regular matrices are the image of the closure operation $\lambda x.\overline{\overline{x}}$ that is used in the correctness proof whenever only the structure of the graph is important, not the weights. The graph structure can be represented as a (Boolean) relation; in the context of fuzzy systems these are also called 'crisp' relations to distinguish them from fuzzy relations [19]. An operation to obtain the least crisp relation containing a given fuzzy relation is discussed in [47].

The order $\leq$ of Stone algebras allows us to compare edge weights. For matrices the comparison and all operations of Stone algebras work componentwise. These operations cannot be used to propagate information about edges through a graph. To combine information from edges between different pairs of nodes we add a relational structure with the operations of composition and converse. In unweighted graphs, they would be provided by relation algebras. To handle weighted graphs, we introduce the following generalisation.

Definition 2. A *Stone relation algebra* $(S, \sqcup, \sqcap, \cdot, \overline{\ }, {}^{\mathsf{T}}, \bot, \top, 1)$ is a Stone algebra $(S, \sqcup, \sqcap, \overline{\ }, \bot, \top)$ with a composition $\cdot$ and a converse ${}^{\mathsf{T}}$ and a constant 1 satisfying Eqs. (1)–(10). We abbreviate $x \cdot y$ as xy and let composition have higher precedence than the operators $\sqcup$ and $\sqcap$. The axioms are:

$$(xy)z = x(yz) \tag{1}$$
$$1x = x \tag{2}$$
$$(x \sqcup y)z = xz \sqcup yz \tag{3}$$
$$(xy)^{\mathsf{T}} = y^{\mathsf{T}}x^{\mathsf{T}} \tag{4}$$
$$(x \sqcup y)^{\mathsf{T}} = x^{\mathsf{T}} \sqcup y^{\mathsf{T}} \tag{5}$$
$$x^{\mathsf{T}^{\mathsf{T}}} = x \tag{6}$$
$$\bot x = \bot \tag{7}$$
$$xy \sqcap z \leq x(y \sqcap x^{\mathsf{T}}z) \tag{8}$$
$$\overline{\overline{xy}} = \overline{\overline{x}}\,\overline{\overline{y}} \tag{9}$$
$$\overline{\overline{1}} = 1 \tag{10}$$

An element $x \in S$ is a *vector* if $x\top = x$, *symmetric* if $x = x^{\mathsf{T}}$, *injective* if $xx^{\mathsf{T}} \leq 1$, *surjective* if $1 \leq x^{\mathsf{T}}x$ and *bijective* if x is injective and surjective. An element $x \in S$ is an *atom* if both $x\top$ and $x^{\mathsf{T}}\top$ are bijective. A *relation algebra* $(S, \sqcup, \sqcap, \cdot, \overline{\ }, {}^{\mathsf{T}}, \bot, \top, 1)$ is a Stone relation algebra whose reduct $(S, \sqcup, \sqcap, \overline{\ }, \bot, \top)$ is a Boolean algebra.

We reuse the concise characterisations of vectors, atoms, symmetry, injectivity, surjectivity and bijectivity known from relation algebras [44]. In the instance of relations over a set A, a vector represents a subset of A and an atom represents a relation containing a single pair. Hence, in the graph model a vector describes a set of nodes – such as the ones visited in Prim's algorithm – and an atom

describes an edge of the graph. Injectivity then means that two nodes cannot have the same successor, which is a property of trees.

Observe that relation algebras and Stone relation algebras have the same signature. The main difference between them is the weakening of the lattice structure from Boolean algebras to Stone algebras. In particular, the property

$$x^\mathsf{T}\overline{xy} \leq \overline{y} \tag{11}$$

holds in Stone relation algebras. Tarski's relation algebras require a Boolean algebra, axioms (1)–(6), and property (11) [34]. Axioms (7)–(10) follow in relation algebras.

Axiom (8) has been called 'Dedekind formula' or 'modular law' [8,30]. Besides being typed, Dedekind categories require that composition has a left residual and that each Hom-set is a complete distributive lattice [29] and therefore a Heyting algebra, which is more restrictive than a Stone algebra. Rough relation algebras [13] weaken the lattice structure of relation algebras to double Stone algebras, which capture properties of rough sets. Axioms (9) and (10) state that regular elements are closed under composition and its unit.

Many results of relation algebras hold in Stone relation algebras directly or with small modifications. For example, $x \leq xx^\mathsf{T}x$, the complement of a vector is a vector, and composition with an injective element distributes over $\sqcap$ from the right. We also obtain the following variant of the so-called Schröder equivalence:

$$xy \leq \overline{z} \;\Leftrightarrow\; x^\mathsf{T}z \leq \overline{y}$$

The following result shows further consequences for Stone relation algebras. In particular, every Stone algebra can be extended to a Stone relation algebra, and the Stone relation algebra structure can be lifted to matrices by using the usual matrix composition (taking $\sqcup$ and $\cdot$ from the underlying Stone relation algebra as addition and multiplication, respectively).

Theorem 2. *1. The regular elements of a Stone relation algebra S form a relation algebra that is a subalgebra of S.*

2. Let $(S, \sqcup, \sqcap, \overline{\ }, \bot, \top)$ be a Stone algebra. Then $(S, \sqcup, \sqcap, \sqcap, \overline{\ }, \lambda x.x, \bot, \top, \top)$ is a Stone relation algebra with the identity function as converse.

3. Let $(S, \sqcup, \sqcap, \cdot, \overline{\ }, \mathsf{T}, \bot, \top, 1)$ be a Stone relation algebra and let A be a finite set. Then $(S^{A\times A}, \sqcup, \sqcap, \cdot, \overline{\ }, \mathsf{T}, \bot, \top, 1)$ is a Stone relation algebra, where the operations $\cdot$, T and 1 are defined by

$$(M \cdot N)_{i,j} = \bigsqcup_{k\in A} M_{i,k} \cdot N_{k,j}$$
$$(M^\mathsf{T})_{i,j} = (M_{j,i})^\mathsf{T}$$
$$1_{i,j} = \begin{cases} 1 & \text{if } i = j \\ \bot & \text{if } i \neq j \end{cases}$$

Hence weighted graphs form a Stone relation algebra as follows: for weights the operations are $x \cdot y = \min\{x, y\}$ and $x^\mathsf{T} = x$ according to Theorem 2.2, and these operations are lifted to matrices as shown in Theorem 2.3. Because in

this instance the converse operation of the underlying Stone relation algebra is the identity, the lifted converse operation only transposes the matrix. Thus for a finite set A, the set of matrices $\mathbb{R}'^{A\times A}$ is a Stone relation algebra with the following operations:

$$\begin{aligned}
(M \sqcup N)_{i,j} &= \max(M_{i,j}, N_{i,j}) \\
(M \sqcap N)_{i,j} &= \min(M_{i,j}, N_{i,j}) \\
(M \cdot N)_{i,j} &= \max_{k\in A} \min(M_{i,k}, N_{k,j}) \\
\overline{M}_{i,j} &= \overline{M_{i,j}} \\
M^{\mathsf{T}}{}_{i,j} &= M_{j,i} \\
\bot_{i,j} &= \bot \\
\top_{i,j} &= \top \\
1_{i,j} &= \begin{cases} \top \text{ if } i = j \\ \bot \text{ if } i \neq j \end{cases}
\end{aligned}$$

The order in this structure is $M \leq N \;\Leftrightarrow\; \forall i, j \in A : M_{i,j} \leq N_{i,j}$.

3 Stone-Kleene Relation Algebras

In this section, we combine Stone relation algebras with Kleene algebras [32] in order to obtain information about reachability in graphs. Kleene algebras are used to model finite iteration for regular languages and relations. In particular, they expand semirings by a unary operation – the Kleene star – which instantiates to the reflexive-transitive closure of relations. The properties of the Kleene star have been studied in [14] and we use the axiomatisation given in [32].

Definition 3. An *idempotent semiring* is an algebraic structure $(S, \sqcup, \cdot, \bot, 1)$ where $(S, \sqcup, \bot)$ is a bounded semilattice and $\cdot$ is associative, distributes over $\sqcup$ and has unit 1 and zero $\bot$:

$$\begin{array}{llll}
x(y \sqcup z) = xy \sqcup xz & x\bot = \bot & x1 = x & x(yz) = (xy)z \\
(x \sqcup y)z = xz \sqcup yz & \bot x = \bot & 1x = x &
\end{array}$$

A *Kleene algebra* $(S, \sqcup, \cdot, {}^*, \bot, 1)$ is an idempotent semiring $(S, \sqcup, \cdot, \bot, 1)$ with an operation * satisfying the unfold and induction axioms

$$\begin{array}{ll}
1 \sqcup yy^* \leq y^* & z \sqcup yx \leq x \Rightarrow y^*z \leq x \\
1 \sqcup y^*y \leq y^* & z \sqcup xy \leq x \Rightarrow zy^* \leq x
\end{array}$$

A *Stone-Kleene relation algebra* is a structure $(S, \sqcup, \sqcap, \cdot, {}^{\overline{\ }}, {}^{\mathsf{T}}, {}^*, \bot, \top, 1)$ such that the reduct $(S, \sqcup, \sqcap, \cdot, {}^{\overline{\ }}, {}^{\mathsf{T}}, \bot, \top, 1)$ is a Stone relation algebra, the reduct $(S, \sqcup, \cdot, {}^*, \bot, 1)$ is a Kleene algebra and the following equation holds:

$$\overline{\overline{x^*}} = (\overline{\overline{x}})^* \tag{12}$$

An element $x \in S$ is *acyclic* if $xx^* \leq \overline{1}$ and x is a *forest* if x is injective and acyclic. A *Kleene relation algebra* $(S, \sqcup, \sqcap, \cdot, ^{-}, ^{\mathsf{T}}, ^*, \bot, \top, 1)$ is a Stone-Kleene relation algebra whose reduct $(S, \sqcup, \sqcap, \cdot, ^{-}, ^{\mathsf{T}}, \bot, \top, 1)$ is a relation algebra.

Axiom (12) states that regular elements are closed under the operation *. Many results of Kleene relation algebras hold in Stone-Kleene relation algebras directly or with small modifications. For example, $(xx^{\mathsf{T}})^* = 1 \sqcup xx^{\mathsf{T}}$ for each vector x, the operations converse and Kleene star commute, and

$$x^* x^{\mathsf{T}*} \sqcap x^{\mathsf{T}} x \leq 1$$

for each forest x. The latter follows using the cancellation property

$$xy \leq 1 \;\Rightarrow\; x^* y^* \leq x^* \sqcup y^*$$

which we have proved in Kleene algebras as part of the present verification work; such properties can also be interpreted in rewrite systems [45]. Proofs of the above properties – and other algebraic results and consequences stated in this paper – can be found in the Isabelle/HOL theory files mentioned in the introduction. The following result shows further consequences for Kleene algebras, Stone-Kleene relation algebras and Kleene relation algebras.

Theorem 3. *1. The regular elements of a Stone-Kleene relation algebra S form a Kleene relation algebra that is a subalgebra of S.*

2. Let $(S, \sqcup, \sqcap, \bot, \top)$ be a bounded distributive lattice. Then $(S, \sqcup, \sqcap, \lambda x.\top, \bot, \top)$ is a Kleene algebra with the constant $\top$ function as the star operation.

3. Let $(S, \sqcup, \sqcap, ^{-}, \bot, \top)$ be a Stone algebra.
Then $(S, \sqcup, \sqcap, \sqcap, ^{-}, \lambda x.x, \lambda x.\top, \bot, \top, \top)$ is a Stone-Kleene relation algebra.

4. Let $(S, \sqcup, \sqcap, \cdot, ^{-}, ^{\mathsf{T}}, ^*, \bot, \top, 1)$ be a Stone-Kleene relation algebra and let A be a finite set. Then $(S^{A \times A}, \sqcup, \sqcap, \cdot, ^{-}, ^{\mathsf{T}}, ^*, \bot, \top, 1)$ is a Stone-Kleene relation algebra, where the operation * is defined recursively using Conway's automata-based construction [14]:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}^* = \begin{pmatrix} e^* & a^* b f^* \\ d^* c e^* & f^* \end{pmatrix} \qquad \textit{where} \qquad \begin{pmatrix} e \\ f \end{pmatrix} = \begin{pmatrix} a \sqcup b d^* c \\ d \sqcup c a^* b \end{pmatrix}$$

This shows the recursive case, which splits a matrix into smaller matrices. At termination, the Kleene star is applied to the entry of a one-element matrix.

In particular, this provides a formally verified proof of Conway's construction for the Kleene star of matrices, which is missing in existing Isabelle/HOL theories of Kleene algebras [3, Sect. 5.7].

As a consequence, weighted graphs form a Stone-Kleene relation algebra as follows: for weights the max-min lattice is extended with the Kleene star operation $x^* = \top$ according to Theorem 3.3, and the Kleene star is defined for matrices by Conway's construction shown in Theorem 3.4.

4 An Algebra for Minimising Weights

In this section we extend Stone-Kleene relation algebras by dedicated operations for the minimum spanning tree application. First, the algorithm needs to select an edge with minimal weight; this is done by the operation m. Second, the sum of edge weights needs to be minimised according to the specification; the sum is obtained by the operation s. Third, the axioms of s use the operation $+$ to add the weights of corresponding edges of two graphs. These operations are captured in the following algebraic structure.

Definition 4. An *M-algebra* $(S, \sqcup, \sqcap, \cdot, +, \overline{\ }, {}^{\mathsf{T}}, {}^*, s, m, \bot, \top, 1)$ is a Stone-Kleene relation algebra $(S, \sqcup, \sqcap, \cdot, \overline{\ }, {}^{\mathsf{T}}, {}^*, \bot, \top, 1)$ with an addition $+$, a summation s and a minimum selection m satisfying the following properties:

$$\overline{\overline{x}} = \overline{\overline{y}} \ \wedge \ x \leq y \Rightarrow z + x \leq z + y \tag{13}$$

$$x + s(\bot) = x \tag{14}$$

$$s(x) + s(y) = s(x \sqcup y) + s(x \sqcap y) \tag{15}$$

$$s(x^{\mathsf{T}}) = s(x) \tag{16}$$

$$x \neq \bot \Rightarrow s(y) \leq \overline{\overline{s(x)}} \tag{17}$$

$$m(x) \leq \overline{\overline{x}} \tag{18}$$

$$\overline{\overline{m(x)}} = m(x) \tag{19}$$

$$x \neq \bot \Rightarrow m(x) \text{ is an atom} \tag{20}$$

$$y \text{ is an atom} \ \wedge \ \overline{\overline{y}} = y \ \wedge \ y \sqcap x \neq \bot \Rightarrow s(m(x) \sqcap x) \leq s(y \sqcap x) \tag{21}$$

Among the new operations, only m is used in the algorithm. The axioms have the following meaning:

(13) The operation $+$ is $\leq$-isotone in its second argument as long as no new edges are introduced (this is required because edges may have negative weights).
(14) The empty graph adds no weight; the given axiom is weaker than the conjunction of $s(\bot) = \bot$ and $x + \bot = x$.
(15) This generalises the inclusion-exclusion principle to sets of numbers.
(16) Reversing edges does not change the sum of weights.
(17) The result of s is represented by a graph with one fixed edge.
(18) The minimal edge is contained in the graph.
(19) The result of m is represented as a relation.
(20) The result of m is just one edge, if the graph is not empty.
(21) Any edge y in the graph x weighs at least as much as $m(x)$; the operation s is used to compare the weights of edges between different nodes.

A precise definition of the operations $+$, s and m on weighted graphs is given in the following result, which shows that weighted graphs form an M-algebra.

Theorem 4. *Let A be a finite set. Let $\prec$ be a strict total order on A with least element h. Then the set of matrices $\mathbb{R}'^{A\times A}$ is an M-algebra with the following operations:*

$$(M+N)_{i,j} = M_{i,j} + N_{i,j} \tag{22}$$

$$s(M)_{i,j} = \begin{cases} \sum_{k,l\in A} M_{k,l} & \text{if } i=j=h \\ \bot & \text{if } i\neq h \vee j\neq h \end{cases} \tag{23}$$

$$m(M)_{i,j} = \begin{cases} \top \text{ if } M_{i,j}\neq\bot \wedge \\ \quad \forall k,l\in A : (M_{k,l}\neq\bot \Rightarrow M_{i,j}\leq M_{k,l}) \wedge \\ \quad\quad ((k\prec i \vee (k=i\wedge l\prec j)) \Rightarrow M_{i,j}\neq M_{k,l}) \\ \bot \text{ otherwise} \end{cases} \tag{24}$$

The addition $+$ on $\mathbb{R}'$ used in (22) is defined by

$x+y=$	$y=\bot$	$y\in\mathbb{R}$	$y=\top$
$x=\bot$	$\bot$	y	$\top$
$x\in\mathbb{R}$	x	$x+_{\mathbb{R}} y$	$\top$
$x=\top$	$\top$	$\top$	$\top$

The finite summation $\sum$ on $\mathbb{R}'$ used in (23) is defined recursively using this binary addition, which is associative and commutative.

Equation (24) means that $m(M)_{i,j}=\top$ if (i,j) is the smallest pair (according to the lexicographic order based on $\prec$) such that $M_{i,j}$ is minimal among the weights different from $\bot$. The function s uses the entry in row h and column h to store the sum of the weights different from $\bot$.

5 Correctness of the Minimum Spanning Tree Algorithm

In this section we present a minimum spanning tree algorithm and prove its correctness. In particular, we show how the algebras introduced in the previous sections are used to reason about graph properties. The algorithm is shown in Fig. 2. It is a while-program with variables whose values range over an M-algebra S.

The input of the algorithm is a weighted graph $g\in S$ and a root node $r\in S$. The algorithm constructs a minimum spanning tree $t\in S$ and maintains a set of visited nodes v. Both r and v are represented as vectors. The algorithm starts with an empty tree t and the single visited node r. The expression $v\overline{v}^{\mathsf{T}}\sqcap g$ restricts g to the edges starting in v and ending outside of v. In each iteration an edge e is chosen with minimal weight among these edges; then e is added to t and the end node of e is added to v. When there are no edges from v to its complement set, the while-loop finishes and the output of the algorithm is t.

We show correctness of the algorithm relative to two assumptions:

1. The while-loop terminates. This follows since a new edge is added to the spanning tree in each iteration and the graph is finite. Such termination proofs can also be done algebraically [23], but this is not part of the present paper.

```
input g, r
t ← ⊥
v ← r
while v v̄ᵀ ⊓ g ≠ ⊥ do
   e ← m(v v̄ᵀ ⊓ g)
   t ← t ⊔ e
   v ← v ⊔ eᵀ ⊤
end
output t
```

Fig. 2. A relational minimum spanning tree algorithm

2. There exists a minimum spanning tree. This follows since the number of spanning trees of a finite graph is finite. A proof of this is not part of the present paper, but could be based on cardinalities of relations [28].

We do not assume that the graph g is connected. As a consequence, the above algorithm will produce a minimum spanning tree of the component of g that contains r. In M-algebras, the nodes in this component are given by

$$c(g,r) = r^\mathsf{T} \overline{\overline{g}}^*$$

which is the converse of a vector that represents the set of nodes reachable from r in the graph g ignoring edge weights. It follows that for connected g the result is a minimum spanning tree of the whole graph. The correctness proof uses the following predicates.

Definition 5. Let S be an M-algebra and let $g, r, t, v \in S$. Then t is a *spanning tree* of g with root r if t is a forest, t is regular and

$$t \leq c(g,r)^\mathsf{T} c(g,r) \sqcap \overline{\overline{g}} \tag{25}$$

$$c(g,r) \leq r^\mathsf{T} t^* \tag{26}$$

Such a t is a *minimum spanning tree* of g with root r if, additionally,

$$s(t \sqcap g) \leq s(u \sqcap g)$$

for each spanning tree u of g with root r. Next, the *precondition* requires that g is symmetric, that r is regular, injective and a vector, and that a minimum spanning tree of g with root r exists. Next, the *loop invariant* requires the precondition and $v^\mathsf{T} = r^\mathsf{T} t^*$ and that t is a spanning tree of $v v^\mathsf{T} \sqcap g$ with root r, and that $t \leq w$ for some minimum spanning tree w of g with root r. Finally, the *postcondition* requires that t is a minimum spanning tree of g with root r.

By lattice properties and since $c(g,r)$ is the converse of a vector, inequality (25) is equivalent to the conjunction of $t \leq \overline{\overline{g}}$ and $t \leq c(g,r)^\mathsf{T}$ and $t \leq c(g,r)$. The first of these inequalities states that all edges of t are contained in g (ignoring

the weights). The second inequality states that each edge of t starts in a node in the component of g that contains r. The third inequality expresses the same for the end nodes of the edges of t.

Also inequality (26) is concerned with the component of g that contains r. It states that all nodes in this component are reachable from r using edges in t. Observe that $r^\mathsf{T} t^* = c(t,r)$ since t is regular, so together with $t \leq \overline{\overline{g}}$ we obtain $c(g,r) = c(t,r)$ as a consequence.

Symmetry of g specifies that the graph is undirected. The properties of r in the precondition state that r represents a single node. Assumption 2 amounts to the existence of a minimum spanning tree in the precondition.

The verification conditions to establish the postcondition are automatically generated from the precondition and the loop invariant using Hoare logic. We use an implementation of Hoare logic that comes with Isabelle/HOL; see [36,37]. The generated conditions are predicates whose variables range over an M-algebra; all calculations take place in this algebra or its reducts. The high-level structure of the proof is standard; the difference here is that the whole argument is carried out in new algebraic structures that directly model weighted graphs.

Theorem 5. *Assume the precondition stated in Definition 5 holds. Then the postcondition stated there holds after the algorithm in Fig. 2 finishes.*

In the following we discuss several parts of the proof, which are carried out in different algebraic structures. Our aim is not completeness, but to show that many results used in the proof actually hold in more general settings. We focus on the preservation of the loop invariant for the current tree t and the current set of visited nodes v. Let $t' = t \sqcup e$ and $v' = v \sqcup e^\mathsf{T} \top$ be the values of these variables at the end of the body of the while-loop.

First, the proof involves showing that t' is a spanning tree of $v'v'^\mathsf{T} \sqcap g$ with root r, that is, of the subgraph of g restricted to nodes in v'. In particular, this requires that t' is injective. To this end, we use the following property given in [39] that also holds in Stone relation algebras.

Lemma 1. *Let S be a Stone relation algebra. Let $t, e \in S$ such that t and e are injective and $et^\mathsf{T} \leq 1$. Then $t \sqcup e$ is injective.*

The assumptions of Lemma 1 are established as follows:

- Injectivity of t follows from the invariant.
- e is an atom by axiom (20), so $e\top$ is injective, whence e is injective.
- $et^\mathsf{T} = \bot \leq 1$ follows by another general result of Stone relation algebras from $e \leq v\overline{v}^\mathsf{T}$ and $t \leq vv^\mathsf{T}$ and that v is a vector.

We also require that t' is contained in the subgraph of g restricted to the nodes in v'. For this we use the following result of Stone relation algebras.

Lemma 2. *Let S be a Stone relation algebra. Let $t, e, v, g \in S$ such that $t \leq vv^\mathsf{T} \sqcap g$ and $e \leq v\overline{v}^\mathsf{T} \sqcap g$. Then $t' \leq v'v'^\mathsf{T} \sqcap g$ where $t' = t \sqcup e$ and $v' = v \sqcup e^\mathsf{T} \top$.*

Next, we also require that t' is acyclic. To show this, we use the following result of Stone-Kleene relation algebras.

Lemma 3. *Let S be a Stone-Kleene relation algebra. Let $t, e, v \in S$ such that t is acyclic, v is a vector and $e \leq v\overline{v}^\mathsf{T}$ and $t \leq vv^\mathsf{T}$. Then $t \sqcup e$ is acyclic.*

Note that this lemma does not require that t is a tree or that e contains just one edge. It is a much more general statement that can be used in reasoning about graphs in other contexts than the minimum spanning tree algorithm – in fact, it holds not only for weighted graphs but for any other instance of Stone-Kleene relation algebras. The same observation applies to the previous lemmas and many others used in the correctness proof.

Next, the invariant maintains that v is the set of nodes reachable from r in t, which is formulated as $v^\mathsf{T} = r^\mathsf{T}t^*$. To preserve this property, we use the following result of Stone-Kleene relation algebras.

Lemma 4. *Let S be a Stone-Kleene relation algebra. Let $t, e, r, v \in S$ such that v is a vector, $e \leq v\overline{v}^\mathsf{T}$ and $et = \bot$ and $v^\mathsf{T} = r^\mathsf{T}t^*$. Then ${v'}^\mathsf{T} = r^\mathsf{T}t'^*$ where $t' = t \sqcup e$ and $v' = v \sqcup e^\mathsf{T}\top$.*

The assumption $et = \bot$ follows similarly to $et^\mathsf{T} = \bot$ for Lemma 1.

Finally, we discuss how to preserve the property that the currently constructed spanning tree t can be extended to a minimum spanning tree. The situation is shown in Fig. 3. Assuming that there is a minimum spanning tree w of g such that $t \leq w$, we have to show that there is a minimum spanning tree w' of g such that $t' = t \sqcup e \leq w'$ where $e = m(v\overline{v}^\mathsf{T} \sqcap g)$ is an edge of g with minimal weight going from a node in v to a node not in v. We do this by explicitly constructing the new minimum spanning tree w'. To this end, we need to find the edge f in w that crosses the cut from v to $\overline{v}$, and replace it with the edge e – this does not increase the weight due to minimality of e. An algebraic expression for the edge f is

$$f = w \sqcap v\overline{v}^\mathsf{T} \sqcap \top e w^{\mathsf{T}*}$$

The three terms on the right hand side enforce that f is in w, that f starts in v and ends in $\overline{v}$, and that there is a path in w from the end node of f to the end node of e. It can be shown algebraically that f is an atom, that is, that f represents the unique edge satisfying these conditions. An algebraic expression for the path p from the end of f to the end of e is

$$p = w \sqcap \overline{v}\,\overline{v}^\mathsf{T} \sqcap \top e w^{\mathsf{T}*}$$

The three terms on the right hand side enforce that the edges in p are in w, that they start and end in $\overline{v}$, and that there is a path in w from each of their end nodes to the end node of e. The required tree w' is then obtained by removing the edge f from w, turning around the path p, and inserting the edge e. An algebraic expression for w' is

$$w' = (w \sqcap \overline{f \sqcup p}) \sqcup p^\mathsf{T} \sqcup e$$

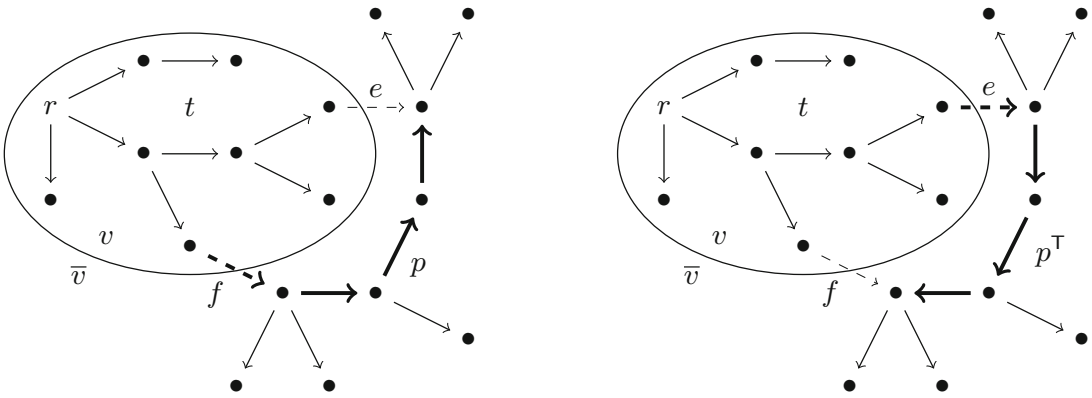

Fig. 3. Replacing the edge f in w (left) with the minimal edge e in w' (right) where t is the tree in the oval and v is the set of nodes in t

We then show that w' so defined is a minimum spanning tree of g with root r and that $t \sqcup e \leq w'$. In the following we focus on the part of this proof that shows $s(w' \sqcap g) \leq s(u \sqcap g)$ for each spanning tree u of g with root r. This follows by the calculation

$$\begin{aligned} s(w' \sqcap g) &= s(w \sqcap \overline{f \sqcup p} \sqcap g) + s(p^\mathsf{T} \sqcap g) + s(e \sqcap g) && (27)\\ &\leq s(w \sqcap \overline{f \sqcup p} \sqcap g) + s(p \sqcap g) + s(f \sqcap g) && (28)\\ &= s(((w \sqcap \overline{f \sqcup p}) \sqcup p \sqcup f) \sqcap g) && (29)\\ &= s(w \sqcap g) && (30)\\ &\leq s(u \sqcap g) && (31) \end{aligned}$$

Equation (27) holds by axioms (14) and (15) since $w \sqcap \overline{f \sqcup p}$ and p^T and e are pairwise disjoint (that is, their pairwise meet given by $\sqcap$ is $\bot$). A similar argument justifies Eq. (29). Axiom (21) is used to show $s(e \sqcap g) \leq s(f \sqcap g)$ in inequality (28). Axiom (16) is used to show that replacing p with p^T does not change the weight there. Equation (30) follows by a simple calculation, most of which takes place in Stone algebras. Finally, Eq. (31) holds since w is a minimum spanning tree of g with root r.

This is the main part of the overall proof where the operations and axioms of M-algebras are used. Most of the proof, however, can already be carried out in Stone-Kleene relation algebras or weaker structures as discussed above. We expect such results to be useful for reasoning about other graph algorithms.

6 Related Work

In this section we compare the present paper with related work on algorithms for minimum spanning trees. Often the correctness of such algorithms is argued informally with varying amounts of mathematical rigour and details; for example, see [15,31]. Our results are fully verified in Isabelle/HOL [38] based on formal definitions and models.

A formal derivation of Prim's minimum spanning tree algorithm in the B event-based framework using Atelier B is presented in [1]. The paper also discusses the role of refinement in this process, which is not part of the present paper. The B specification is based on sets and relations, uses an inductive definition of trees, and represents weights by functions, whence objects of several different sorts are involved.

Our formalisation is based on Stone-Kleene relation algebras, which generalise relation algebras and Kleene algebras, and can be instantiated directly by weight matrices. The generalisation is crucial as weight matrices do not support a Boolean complement; accordingly we do not use implementations of relation algebras such as [2,24]. Nevertheless we can build on well-developed relational concepts and methods for our new algebras – such as algebraic properties of trees – which are useful also in other contexts.

We mostly apply equational reasoning based on a single-sorted algebra. This is well supported by automated theorem provers and SMT solvers such as those integrated in Isabelle/HOL via the Sledgehammer tool [11,42] that we heavily use in the verification. Typically the tool can automatically find proofs of steps at a granularity comparable to manual equational reasoning found in papers. Automation works less well in some cases, for example, chains of inequalities, applications of isotone operations, and steps that introduce intermediate terms that occur on neither side of an equation. On the other hand, in some cases the tool can automatically prove a result that would take several manual steps.

A distributed algorithm for computing minimum spanning trees is verified using the theorem prover Nqthm in [25]. The specification is again based on sets and a weight function. The main focus of the paper is on the distributed aspects of the algorithm, which uses asynchronous messages and differs essentially from Prim's minimum spanning tree algorithm. The distributed algorithm is the topic of a number of other papers using a variety of formalisms including Petri nets and modal logic.

Relation algebras are used to derive spanning tree algorithms in [6]. The given proof is created manually and not verified using a theorem prover. It uses relations and, in absence of weighted matrices, an incidence matrix representation and a weight function in a setting with several different sorts.

Constraint-based semirings are used to formulate minimum spanning tree algorithms in [10]. These semirings abstract from the edge weights and represent graphs by sets of edges. The semiring structure is not lifted to the graph level, whereas we lift Stone algebras to Stone relation algebras – and similarly for Kleene algebras – and can therefore exploit the algebraic structure of graphs. Detailed proofs are not presented and there is no formal verification of results. The paper is mainly concerned with extending the algorithms to partially ordered edge weights, which is not part of the present paper.

Semirings with a pre-order, so-called dioids, are used to formulate various shortest-path problems in [20]. The corresponding algorithms are generalisations of methods for solving linear equations over these structures. Other approaches to path problems are based on Kleene algebras; for example, see [26], which also

discusses many previous works in this tradition. Semirings and Kleene algebras are suitable for path problems as they capture the essential operations of choosing between alternatives, composing edges and building paths. It is not clear how to model the minimum spanning tree problem using Kleene algebras only.

Relational methods based on allegories are used for algorithm development in [8], but there relations mostly represent computations rather than the involved data. An extension to quantitative analysis is discussed in [40].

7 Conclusion

The generalisation of Boolean algebras to Stone algebras gives a promising way to extend correctness reasoning from unweighted to weighted graphs. When applied to relation algebras, many results continue to hold with no changes or small changes. In combination with Kleene algebras, we could carry out most of the correctness proof of Prim's minimum spanning tree algorithm.

A small part of the proof needed some additional operations; we captured a few key properties in the present paper, but the underlying structure should be studied further. To this end, we will look at variants of the minimum spanning tree algorithm and other graph algorithms. We will also consider the integration of termination proofs, complexity reasoning and combinatorial arguments using cardinalities of relations.

Using algebras for proving the correctness of programs is well supported by Isabelle/HOL. We have benefited from the existing verification condition generator for Hoare logic, from the structuring mechanisms that allow the development of hierarchies of algebras and their models, and heavily from the integrated automated theorem provers, SMT solvers and counterexample generators.

Acknowledgements. I thank the anonymous referees for helpful feedback including the suggestion to generalise Lemma 1 to its present form. I thank Rudolf Berghammer for discussions about the cardinality of relations and ways to generalise it. I thank Peter Höfner and Bernhard Möller for discussing alternative approaches to minimum spanning trees in Kleene algebras. I thank the participants of the 73rd meeting of IFIP WG 2.1, the 14th Logic and Computation Seminar of Kyushu University and the 2016 Workshop on Universal Structures in Mathematics and Computing for the opportunity to talk about this work and for their valuable feedback. The presentation at Kyushu University was part of a JSPS Invitation Fellowship for Research in Japan.

References

1. Abrial, J.-R., Cansell, D., Méry, D.: Formal derivation of spanning trees algorithms. In: Bert, D., Bowen, J.P., King, S., Waldén, M. (eds.) ZB 2003. LNCS, vol. 2651, pp. 457–476. Springer, Heidelberg (2003). doi:10.1007/3-540-44880-2_27
2. Armstrong, A., Foster, S., Struth, G., Weber, T.: Relation algebra. Archive of Formal Proofs (2016). First version (2014)
3. Armstrong, A., Gomes, V.B.F., Struth, G., Weber, T.: Kleene algebra. Archive of Formal Proofs (2016). First version (2013)

4. Berghammer, R., Fischer, S.: Combining relation algebra and data refinement to develop rectangle-based functional programs for reflexive-transitive closures. J. Log. Algebr. Methods Program. **84**(3), 341–358 (2015)
5. Berghammer, R., von Karger, B.: Relational semantics of functional programs. In: Brink, C., Kahl, W., Schmidt, G. (eds.) Relational Methods in Computer Science, chap. 8, pp. 115–130. Springer, Wien (1997)
6. Berghammer, R., von Karger, B., Wolf, A.: Relation-algebraic derivation of spanning tree algorithms. In: Jeuring, J. (ed.) MPC 1998. LNCS, vol. 1422, pp. 23–43. Springer, Heidelberg (1998). doi:10.1007/BFb0054283
7. Berghammer, R., Rusinowska, A., de Swart, H.: Computing tournament solutions using relation algebra and RelView. Eur. J. Oper. Res. **226**(3), 636–645 (2013)
8. Bird, R., de Moor, O.: Algebra of Programming. Prentice Hall, Englewood Cliffs (1997)
9. Birkhoff, G.: Lattice Theory. Colloquium Publications, vol. XXV, 3rd edn. American Mathematical Society, Providence (1967)
10. Bistarelli, S., Santini, F.: C-semiring frameworks for minimum spanning tree problems. In: Corradini, A., Montanari, U. (eds.) WADT 2008. LNCS, vol. 5486, pp. 56–70. Springer, Heidelberg (2009). doi:10.1007/978-3-642-03429-9_5
11. Blanchette, J.C., Böhme, S., Paulson, L.C.: Extending Sledgehammer with SMT solvers. In: Bjørner, N., Sofronie-Stokkermans, V. (eds.) CADE 2011. LNCS (LNAI), vol. 6803, pp. 116–130. Springer, Heidelberg (2011). doi:10.1007/978-3-642-22438-6_11
12. Blyth, T.S.: Lattices and Ordered Algebraic Structures. Springer, Berlin (2005)
13. Comer, S.D.: On connections between information systems, rough sets and algebraic logic. In: Rauszer, C. (ed.) Algebraic Methods in Logic and in Computer Science. Banach Center Publications, vol. 28, pp. 117–124. Institute of Mathematics, Polish Academy of Sciences (1993)
14. Conway, J.H.: Regular Algebra and Finite Machines. Chapman and Hall, London (1971)
15. Cormen, T.H., Leiserson, C.E., Rivest, R.L.: Introduction to Algorithms. MIT Press, Cambridge (1990)
16. Desharnais, J., Grinenko, A., Möller, B.: Relational style laws and constructs of linear algebra. J. Log. Algebr. Methods Program. **83**(2), 154–168 (2014)
17. Dijkstra, E.W.: A note on two problems in connexion with graphs. Numerische Mathematik **1**(1), 269–271 (1959)
18. Freyd, P.J., Ščedrov, A.: Categories, Allegories. North-Holland Mathematical Library, vol. 39. Elsevier Science Publishers (1990)
19. Goguen, J.A.: L-fuzzy sets. J. Math. Anal. Appl. **18**(1), 145–174 (1967)
20. Gondran, M., Minoux, M.: Graphs, Dioids and Semirings. Springer, Heidelberg (2008)
21. Graham, R.L., Hell, P.: On the history of the minimum spanning tree problem. Ann. Hist. Comput. **7**(1), 43–57 (1985)
22. Grätzer, G.: Lattice Theory: First Concepts and Distributive Lattices. W. H. Freeman and Co., San Francisco (1971)
23. Guttmann, W.: Algebras for correctness of sequential computations. Sci. Comput. Program. **85**(Part B), 224–240 (2014)
24. Guttmann, W., Struth, G., Weber, T.: A repository for Tarski-Kleene algebras. In: Höfner, P., McIver, A., Struth, G. (eds.) Automated Theory Engineering. CEUR Workshop Proceedings, vol. 760, pp. 30–39 (2011)
25. Hesselink, W.H.: The verified incremental design of a distributed spanning tree algorithm: extended abstract. Formal Aspects Comput. **11**(1), 45–55 (1999)

26. Höfner, P., Möller, B.: Dijkstra, Floyd and Warshall meet Kleene. Formal Aspects Comput. **24**(4), 459–476 (2012)
27. Jarník, V.: O jistém problému minimálním (Z dopisu panu O. Borůvkovi). Práce moravské přírodovědecké společnosti **6**(4), 57–63 (1930)
28. Kawahara, Y.: On the cardinality of relations. In: Schmidt, R.A. (ed.) RelMiCS/AKA 2006. LNCS, vol. 4136, pp. 251–265. Springer, Heidelberg (2006). doi:10.1007/11828563_17
29. Kawahara, Y., Furusawa, H.: Crispness in Dedekind categories. Bull. Inf. Cybern. **33**(1–2), 1–18 (2001)
30. Kawahara, Y., Furusawa, H., Mori, M.: Categorical representation theorems of fuzzy relations. Inf. Sci. **119**(3–4), 235–251 (1999)
31. Knuth, D.E.: The Art of Computer Programming. Fundamental Algorithms, vol. 1, 3rd edn. Addison-Wesley, Reading (1997)
32. Kozen, D.: A completeness theorem for Kleene algebras and the algebra of regular events. Inf. Comput. **110**(2), 366–390 (1994)
33. Macedo, H.D., Oliveira, J.N.: A linear algebra approach to OLAP. Formal Aspects Comput. **27**(2), 283–307 (2015)
34. Maddux, R.D.: Relation-algebraic semantics. Theor. Comput. Sci. **160**(1–2), 1–85 (1996)
35. Mareš, M.: The saga of minimum spanning trees. Comput. Sci. Rev. **2**(3), 165–221 (2008)
36. Nipkow, T.: Winskel is (almost) right: Towards a mechanized semantics textbook. Formal Aspects Comput. **10**(2), 171–186 (1998)
37. Nipkow, T.: Hoare logics in Isabelle/HOL. In: Schwichtenberg, H., Steinbrüggen, R. (eds.) Proof and System-Reliability, pp. 341–367. Kluwer Academic Publishers (2002)
38. Nipkow, T., Paulson, L.C., Wenzel, M.: Isabelle/HOL: A Proof Assistant for Higher-Order Logic. LNCS, vol. 2283. Springer, Heidelberg (2002)
39. Oliveira, J.N.: Extended static checking by calculation using the pointfree transform. In: Bove, A., Barbosa, L.S., Pardo, A., Pinto, J.S. (eds.) LerNet 2008. LNCS, vol. 5520, pp. 195–251. Springer, Heidelberg (2009). doi:10.1007/978-3-642-03153-3_5
40. Oliveira, J.N.: Towards a linear algebra of programming. Formal Aspects Comput. **24**(4), 433–458 (2012)
41. Oliveira, J.N.: Weighted automata as coalgebras in categories of matrices. Int. J. Found. Comput. Sci. **24**(6), 709–728 (2013)
42. Paulson, L.C., Blanchette, J.C.: Three years of experience with Sledgehammer, a practical link between automatic and interactive theorem provers. In: Sutcliffe, G., Ternovska, E., Schulz, S. (eds.) Proceedings of the 8th International Workshop on the Implementation of Logics, pp. 3–13 (2010)
43. Prim, R.C.: Shortest connection networks and some generalizations. Bell Syst. Tech. J. **36**(6), 1389–1401 (1957)
44. Schmidt, G., Ströhlein, T.: Relations and Graphs. Springer, Berlin (1993)
45. Struth, G.: Abstract abstract reduction. J. Log. Algebr. Program. **66**(2), 239–270 (2006)
46. Tarski, A.: On the calculus of relations. J. Symbol. Log. **6**(3), 73–89 (1941)
47. Winter, M.: A new algebraic approach to L-fuzzy relations convenient to study crispness. Inf. Sci. **139**(3–4), 233–252 (2001)