Hopscotch—Reaching the Target Hop by Hop

Peter Höfner[a,c,*], Annabelle McIver[b]

[a] *NICTA, Australia*
[b] *Department of Computing, Macquarie University, Australia*
[c] *Computer Science and Engineering, University of New South Wales, Australia*

It is our pleasure to dedicate this paper to Gunther Schmidt at the occasion of his 75th birthday. Gunther has been working in the area of relation algebra for many years, publishing countless papers and two books. His first book particularly addresses the relationship between relations and graphs. Relations are an excellent tool for analysing unweighted graphs; subsequently weighted graphs were "algebraised" using matrices over reals or naturals, such as the min-plus algebra, a long-term interest of Gunther. In this paper we continue this line of research and show how to treat matrices that contain even more information, not only numbers. In this respect we illustrate, as Gunther did on so many occasions, that matrices are a powerful, concise and easy-to-understand tool in computer science. So, all the best Gunther, and many more years of health and success!

Abstract

Concrete and abstract relation algebras have widespread applications in computer science. One of the most famous is graph theory. Classical relations, however, only reason about connectivity, not about the length of a path between nodes. Generalisations of relation algebra, such as the min-plus algebra, use numbers allowing the treatment of weighted graphs. In this way one can for example determine the length of shortest paths (e.g. Dijkstra's algorithm). In this paper we treat matrices that carry even more information, such as the "next hop" on a path towards a destination. In this way we can use algebra not only for determining the length of paths, but also the concrete path. We show how paths can be reconstructed from these matrices, hop by hop. We further sketch an application for message passing in wireless networks.

Keywords: routing, semiring, Kleene algebra, path

1. Introduction

Concrete and abstract relation algebras have wide-spread applications in computer science. Using concrete relation algebra, it is easy and concise to characterise and calculate with concepts such as orderings, equivalence relations, reflexive closures, Church-Rosser theorems, etc. [1, 2, 3, 4, 5]. Concrete relation algebra can be represented either by sets of relations (pairs) or by matrices over the Boolean algebra; it is well-known that both representations are isomorphic.

Abstract relation algebra generalises matrices or sets and uses an axiomatic approach. This abstraction allows inherently algebraic reasoning (often equational), which can be automated [6].

One of the most famous applications of relation algebra is graph theory (e.g. [4]). Characterising (unweighted) graphs by their adjacency matrices allows calculations on graphs to be performed algebraically. For example the reflexive transitive closure can be characterised by the least fixpoint of $x = a \cdot x + 1$, and hence easily calculated by the Knaster-Tarski fixpoint theorem. However, the reflexive transitive closure of

*Corresponding author
Email addresses: peter.hoefner@nicta.com.au (Peter Höfner), annabelle.mciver@mq.edu.au (Annabelle McIver)

a relation only determines whether two nodes are connected via a path or disconnected. Relation algebra cannot be used to determine the length of a path between nodes, nor to handle weighted graphs.

To overcome this deficiency (abstract) relation algebra was generalised and the developed techniques were extended to semirings, Kleene algebras and similar structures (e.g. [7, 8, 9]). For weighted graphs the max-plus algebra [10], the min-plus algebra[1] (e.g. [11]), the min-max algebra and fuzzy relations [12] turned out to be useful. Using these semiring-like structures it is possible to formally derive algorithms, such as Dijkstra's shortest-path algorithm or the algorithm of Floyd and Warshall [13]. Other applications for this type of algebra include refinement [14], knowledge representation [15], social choice theory [16], preference modelling [17, 18] and program analysis [19, 20].

This paper aims to formalise concepts of (wireless) networks algebraically. Based on previous work [21], we show that path finding in a matrix-presentation of a network is not an easy task and that some new theory is needed. As a result, this paper presents new definitions of "hops" and "paths" in an algebraic setting and discuss fundamental properties. To guarantee applicability, we relate our formalism to routing protocols for wireless mesh networks. In this paper we focus on characterising the basic concepts, not on algebraic reasoning.

The paper is organised as follows: in Section 2 we explain the problem of describing paths algebraically in the context of routing protocols for wireless mesh networks. Section 3 recalls the basic algebraic constructions we need, and in Section 4.1 we set out an algebraic development for paths as sequences of nodes. Finally, in Section 5, we illustrate the ideas on a small example taken from routing.

2. A Hidden Path

Let us first present the problem at hand. For this purpose, we assume that the reader is familiar with relations and relation algebra.[2] The remaining necessary algebraic foundations are formally introduced in Section 3.

2.1. The "Classical" Encoding

It is well known that matrices over Boolean algebras can be used to describe (unweighted) graphs by their adjaceny matrices. This (essentially) yields relation algebra. An example is given in Figure 1(b), which encodes the graph presented in Figure 1(a). It is also known that the reflexive transitive closure operation (Kleene star) determines the connectivity between nodes. In the example there is a path between all nodes, i.e., all nodes are connected with each other. Hence the reflexive transitive closure is the all-relation $\top\!\!\top$ ($\top\!\!\top_{ij} = 1$ for all i, j). However, this result neither indicates the distances between nodes nor the actual paths.

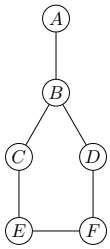
The min-plus semiring can be used to determine path lengths. This algebra uses $\mathbb{R}_{\geq 0} \cup \{\infty\}$ as carrier set, defines addition as min, multiplication as $+$, and Kleene star as $a^* = 0$. As usual, these operations can be lifted to matrices. Figure 1(c) shows the characterisation of the graph of Figure 1(a), using the min-plus algebra; Figure 1(d) the reflexive, transitive closure—it shows the distances between nodes, but still no paths. An entry labelled with ∞ indicates that there is not path between nodes.

2.2. Wireless Mesh Networks and Routing

To encode paths in a matrix-model, one can calculate with language-like models (see e.g. [13]). In this paper we use an algebraic model inspired by *routing tables* as they are used by nodes in a (wireless) network to forward data packets; in particular routing tables used in wireless mesh networks. *Wireless mesh networks* (WMNs) are self-organising ad-hoc networks that support broadband communication without relying on a wired backbone. Instead of having a central component (router), each node in the network acts as an independent router. Route finding and route maintenance are critical for the performance of these networks.

[1] This algebra is sometimes called tropical semiring.

[2] The definitions of abstract and concrete relation algebras are given in the appendix.



(a) 6-nodes graph

	A	B	C	D	E	F
A	0	1	0	0	0	0
B	1	0	1	1	0	0
C	0	1	0	0	1	0
D	0	1	0	0	0	1
E	0	0	1	0	0	1
F	0	0	0	1	1	0

(b) Corresponding Relation

	A	B	C	D	E	F
A	∞	1	∞	∞	∞	∞
B	1	∞	1	1	∞	∞
C	∞	1	∞	∞	1	∞
D	∞	1	∞	∞	∞	1
E	∞	∞	1	∞	∞	1
F	∞	∞	∞	1	1	∞

(c) Corresponding Matrix (in min-plus algebra)

	A	B	C	D	E	F
A	0	1	2	2	3	3
B	1	0	1	1	2	2
C	2	1	0	2	1	2
D	2	1	2	0	2	1
E	3	2	1	2	0	1
F	3	2	2	1	1	0

(d) Reflexive transitive closure (in min-plus algebra)

Figure 1: Graph, relation and reflexive transitive closure (min-plus algebra)

A widely used routing algorithm for WMNs is the Ad hoc On-Demand Distance Vector (AODV) routing protocol [22].

For the purpose of this paper, we use AODV as motivating example; the presented theory, however, applies to other routing protocols as well. AODV is a reactive protocol, i.e., route discovery processes are only initiated when needed. As for most routing protocols, the basic idea of AODV is inspired by (a variant of) Dijkstra's shortest path algorithm. This shortest path algorithm keeps a record of the paths from a source, extending those paths in each iteration of the algorithm. AODV is based on a distributed version of this idea. Instead of having global knowledge, each node maintains a routing table, which is updated whenever messages are received.

2.3. Next Hops and Routing Tables

Since Dijkstra's shortest path algorithm can be modelled by an algebraic approach [23, 13], it is no surprise that similar techniques can be used for modelling routing tables [21]: entries in the table contain information for each possible destination, including the next hop (not the entire path) together with the total number of hops required to reach the destination. The *next hop* specifies a neighbouring node where the packet has to be sent to reach the destination; the *hop count* specifies the length of the entire path. As before the length of a non-existent path has length ∞; in case no next hop exists—either no path exists or it is the trivial path with hop count 0—we use the symbol ε.

As for relations and the min-plus algebra, any graph can be encoded as a matrix using the introduced pairs (Figure 2(a)). Figure 2(b) shows a matrix indicating *all* shortest paths of the graph of Figure 1(a) (this matrix can be determined by the Kleene star of the matrix of 2(a)). In our model [21], we use pairs (m, x) as matrix entries, where m indicates the next hop and x the hop count. Formally, we assume an ordered set $(\Sigma, \preceq)$ of node names, and calculate using $(\Sigma \times \mathbb{N} \setminus \{0\}) \cup \{(\varepsilon, 0), (\varepsilon, \infty)\}$, where $\varepsilon \notin \Sigma$. The two special entries $(\varepsilon, 0)$ and (ε, ∞)[3] hence no next hop must be given.

A (global) *snapshot* is a $n \times n$ matrix of such pairs. Each line of a snapshot describes a *routing table* of the corresponding node. In the presented example (Figures 1 and 2), the routing table of node F is represented by the matrix' last row. F's entry for destination A (first column) has its next hop registered as D with a hop count 3, whereas the intermediate nodes D and B have the next hops B and A, and hop counts 2 and 1, respectively.

[3]We only use pairs to be consistent with other elements of $\Sigma \times \mathbb{N} \setminus \{0\}$. state that there is a trivial path of length 0 and no path at all (a path of length ∞);

$$\begin{array}{c} \\ A \\ B \\ C \\ D \\ E \\ F \end{array} \begin{array}{c} \begin{array}{cccccc} A & B & C & D & E & F \end{array} \\ \begin{pmatrix} (\varepsilon,\infty) & (B,1) & (\varepsilon,\infty) & (\varepsilon,\infty) & (\varepsilon,\infty) & (\varepsilon,\infty) \\ (A,1) & (\varepsilon,\infty) & (C,1) & (D,1) & (\varepsilon,\infty) & (\varepsilon,\infty) \\ (\varepsilon,\infty) & (B,1) & (\varepsilon,\infty) & (\varepsilon,\infty) & (E,1) & (\varepsilon,\infty) \\ (\varepsilon,\infty) & (B,1) & (\varepsilon,\infty) & (\varepsilon,\infty) & (\varepsilon,\infty) & (F,1) \\ (\varepsilon,\infty) & (\varepsilon,\infty) & (C,1) & (\varepsilon,\infty) & (\varepsilon,\infty) & (F,1) \\ (\varepsilon,\infty) & (\varepsilon,\infty) & (\varepsilon,\infty) & (D,1) & (E,1) & (\varepsilon,\infty) \end{pmatrix} \end{array}$$

(a) Matrix using next hops and hop count

$$\begin{array}{c} \\ A \\ B \\ C \\ D \\ E \\ F \end{array} \begin{array}{c} \begin{array}{cccccc} A & B & C & D & E & F \end{array} \\ \begin{pmatrix} (\varepsilon,0) & (B,1) & (B,2) & (B,2) & (B,3) & (B,3) \\ (A,1) & (\varepsilon,0) & (C,1) & (D,1) & (C,2) & (D,2) \\ (B,2) & (B,1) & (\varepsilon,0) & (B,2) & (E,1) & (E,2) \\ (B,2) & (B,1) & (B,2) & (\varepsilon,0) & (F,2) & (F,1) \\ (C,3) & (C,2) & (C,1) & (F,2) & (\varepsilon,0) & (F,1) \\ (D,3) & (D,2) & (E,2) & (D,1) & (E,1) & (\varepsilon,0) \end{pmatrix} \end{array}$$

(b) Reflexive transitive closure

Figure 2: Matrix representation of Fig 1(a) and Kleene star, using next hops

In [21] it is shown that the set of all snapshots form a Kleene algebra, which allows algebraic modifications.

2.4. The Ad hoc On-demand Distance Vector Protocol

Each routing table (row of a snapshot) represents the local knowledge of a node. Local knowledge is often neither complete nor optimal, i.e., there might be routes in the network, which have not yet been discovered; or the routing protocol might have failed to establish the shortest route—these discovered routes are indicated by the precise matrix entries.

Routing tables (and snapshots) are updated as a result of message passing. In case of AODV, a route discovery process of node A searching for a route to node D (the destination) starts with A broadcasting a route request (RREQ) message, which is received by all nodes within A's transmission range. If an intermediate node (not the destination) receives a RREQ message and does not have a valid entry for the destination in its routing table, the request is forwarded by re-broadcasting the RREQ message. During this forwarding process, the intermediate node updates its routing table and adds a "reverse route" entry with destination A into its routing table, indicating via which next hop A can be reached, and the distance in number of hops.

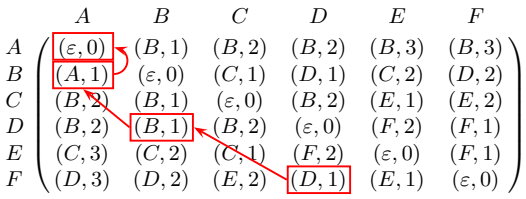
As soon as the RREQ is received by the destination itself or by a node that knows a valid route to the destination, a route reply (RREP) is generated. In contrast to RREQ messages, a RREP message is unicast, i.e., it is only sent to a single node, not to all nodes within transmission range. The RREP message travels from its generator (either D or an intermediate node knowing a route to D) back along the established route towards A, the originator of the RREQ message. All intermediate nodes on the selected route will process the RREP message and, in most cases, forward it towards A. By passing a RREP message towards A, a node adds a "forward route" entry to its routing table.

The route discovery process is completed when the RREP reaches node A; an end-to-end route from A to D has been established, and data packets can start to flow. Full details can be found in RFC 3561 [22], the de facto standard of AODV.

2.5. Message Passing

We observe that messages are either *broadcast* or *unicast*: broadcast messages are intended for any node in transmission range, whereas unicast messages are addressed to a single node. Thus unicast messages are only received by the addressee, and any other node which picks up the message simply ignores it. Message passing can be modelled by algebraic matrix operations [21]. At the algebraic level message sending is expressed by $a + b + b \cdot c$, for some algebraic elements a, b, c (details are given later).

It has been shown that broadcasting a message can "easily" modelled in algebra, as long as the current topology b is given. Unicasting, however, is a much harder task since knowledge about the next hops needs to be distilled from the snapshot. To explain this desire we revisit the example an consider again the matrix of Figure 2(b).



The first entry in the last row only indicates the existence of a path from F to A of length 3, with given next hop D. A message needs to be sent to D first (via the link $(D,1)$). Node D, after it has received the packet from F, inspects its own routing table (4th row of the matrix) and determines that its own path to A has B as next hop. Hence the packet is forwarded via the link $(B,1)$. This forwarding continues until the destination A is reached. This "hopping from node to node" is one of the fundamental ideas of routing in (wireless) networks. It allows flexible changes in the path information without informing every node.

Assuming that one can distil the marked entries from the matrix, then this matrix can be used as topology-matrix b in the above equation, which then models unicast algebraically. The main contribution of this paper is to determine an algebraic presentation of paths, which paves the way for algebraic modelling and algebraic reasoning for networks in general and routing in particular. As an example it should be possible to give algebraic characterisations of properties for routing protocols, that should be guaranteed:

- *packet delivery*: when a node has information about a destination D, then the intermediate nodes do so as well;
- *loop freedom*: at no time should there be a non-trivial loop in the snapshot.

These properties might be obvious for algorithms such as Dijkstra's shortest path. However, we have shown that in AODV snapshots only reflect local knowledge about the networks and hence theses properties have to be guaranteed. Both properties can be characterised over paths; hence it is necessary to distil paths out of a given matrix. Of course one can just use domain-theoretic reasoning, but a purely algebraic approach brings several advantages, such equational reasoning and proof automation with off-the-shelf automated theorem provers. [6]

3. Kleene Algebras, Products and Special Elements

In this section we briefly recapitulate the algebraic theory needed. It is well known that semirings and Kleene algebras model sequential composition, (non-deterministic) choice and, in case of Kleene algebra, finite iteration in a first-order equational calculus.

Formally, an *idempotent semiring* (*i-semiring*) is a quintuple $(S,+,0,\cdot,1)$ such that $(S,+,0)$ is an idempotent commutative monoid, $(S,\cdot,1)$ is a monoid, multiplication distributes over addition and 0 is an annihilator w.r.t. multiplication $\cdot$. The *natural order* $\leq$ on S is given by $a \leq b \Leftrightarrow a+b = b$. An i-semiring induces an *upper semilattice* in which $a+b$ is the supremum of a and b, and 0 is the least element.

A *Kleene algebra* is an idempotent semiring S extended by an operation $^* : S \to S$ for iterating an element an arbitrary but finite number of times. Such an operation has to satisfy the *star unfold* and the *star induction* axioms

$$1 + a \cdot a^* = a^* \,, \qquad b + a \cdot c \leq c \Rightarrow a^* \cdot b \leq c \,,$$
$$1 + a^* \cdot a = a^* \,, \qquad b + c \cdot a \leq c \Rightarrow b \cdot a^* \leq c \,.$$

The (direct) product of two i-semirings (Kleene algebras) forms again an i-semiring (a Kleene algebra) if all operations are defined component-wise, i.e., $(a_1,a_2)+(b_1,b_2) = (a_1+a_2, b_1+b_2)$, $(a_1,a_2)\cdot(b_1,b_2) = (a_1\cdot a_2, b_1\cdot b_2)$, and $(a_1,a_2)^* = (a_1^*, a_2^*)$. In this case, the natural order is also component-wise. If the natural order should coincide with a lexicographical order, the product becomes more intricate.

Theorem 3.1 (see [21]). Assume two totally ordered sets $(N, \preceq)$ and $(H, \sqsubseteq)$ with an isotone binary operation $\circ$ on N and a binary operation $\bullet$ on H that is strictly isotone, i.e., $a \sqsubset b \Rightarrow (c \bullet a \sqsubset c \bullet b) \wedge (a \bullet c \sqsubset b \bullet c)$. On the set $\mathbb{M} = (N \times H) \cup \{\bot, \top\}$ addition and multiplication is defined as

$$\begin{array}{rcl} (m,x) + (n,y) & = & \begin{cases} (m,x) & \text{if } (x \sqsubset y) \vee (x = y \wedge m \preceq n) \\ (n,y) & \text{otherwise} \end{cases} \\ (m,x) \cdot (n,y) & = & (m \circ n, x \bullet y) \ , \end{array}$$

for $(m,x),(n,y) \in N \times H$. For the special elements $\bot$ and $\top$ choice is defined by $\bot + r = r + \bot = r$ and $\top + r = r + \top = \top$, and multiplication by $\bot \cdot r = r \cdot \bot = \bot$ and $\top \cdot r = r \cdot \top = r$, for all $r \in \mathbb{M}$.

(a) The structure $S = (\mathbb{M}, +, \top, \cdot, \bot)$ is an i-semiring, if $a \sqsubset a \bullet b$ for all $a, b \in H$; the natural order coincides with the lexicographical order $(n,y) \leq (m,x) \Leftrightarrow (x \sqsubset y) \vee (x = y \wedge m \preceq n)$; $\bot$ is the least and $\top$ the greatest element.

(b) Under the conditions of Part (a), setting $r^* = \top$ for all $r \in \mathbb{M}$ turns S into a Kleene algebra.

If we assume an ordered set $(\Sigma, \preceq)$ of node identifiers and the hop count to be an element of $\mathbb{N} \setminus \{0\}$, this theorem shows that there is an underlying Kleene algebra behind the entries of snapshots (see Section 2). In this algebra $(\varepsilon, 0)$ relates to $\top$ and (ε, ∞) to $\bot$. Note that addition chooses the better route. Therefore the natural order reads as "worse than". In particular, $(n,y) \leq (m,x)$ means that the entry (n,y) is worse than (m,x). Most often this implies that (m,x) has a smaller hop count than (n,y). In the remainder we refer to this algebra as *next-hop semiring*.

If elements of i-semirings/Kleene algebras characterise single elements or transitions, whole transition systems are usually encoded by matrices.

Theorem 3.2 (e.g. [24]). Standard matrix addition and multiplication turns the family $M(n,K)$ of $n \times n$ matrices over a Kleene algebra K into a Kleene algebra; the zero matrix is neutral w.r.t. $+$, and the identity matrix I w.r.t. multiplication. In case K has a greatest element $\top$, the matrix containing $\top$ at all positions is the greatest element in the matrix model.

The matrix algebra over the next-hop semiring is called *routing algebra*.

For our definition of paths, we also need the concept of irreducibility. Assume an upper semilattice $(S, \leq)$ with the least element 0 and $+$ as join operation. An element $a \in S$ is *join-irreducible* if

$$\forall\, b, c : a = b + c \Rightarrow (a = b \vee a = c) \ .$$

The next-hop semiring is selective, i.e., $a + b \in \{a, b\}$, hence every element is join-irreducible. Since addition on matrices is defined point-wise, the property lifts nicely to matrices: a matrix is join-irreducible if all but one entries are equal to 0 and if the single entry is join-irreducible w.r.t. the underlying algebra. In the routing algebra this means that there is at most one entry different to (ε, ∞).

Assume an i-semiring with greatest element. In any matrix model, if an element a is join-irreducible, the equation $\top \cdot a \cdot \top$ fills the entire matrix with the non-trivial entry occurring in a. An example, using routing algebra, reads as follows

$$\top \cdot \begin{pmatrix} (A,2) & (\varepsilon,0) & (\varepsilon,0) \\ (\varepsilon,0) & (\varepsilon,0) & (\varepsilon,0) \\ (\varepsilon,0) & (\varepsilon,0) & (\varepsilon,0) \end{pmatrix} \cdot \top = \begin{pmatrix} (A,2) & (A,2) & (A,2) \\ (A,2) & (A,2) & (A,2) \\ (A,2) & (A,2) & (A,2) \end{pmatrix} .$$

In case multiplication is defined, an element a is *multiplicative-irreducible* if

$$\forall\, b, c : a = b \cdot c \Rightarrow (a = b \vee a = c) \ .$$

In the next-hop semiring a is multiplicative-irreducible iff $a = (*, 1)$ (for some $* \in \Sigma$), $a = (\varepsilon, 0)$ or $a = (\varepsilon, \infty)$, meaning that a route cannot be split up into two "proper subroutes". Since multiplication is not point-wise on matrices, the property does not lift nicely. In the next section we will discuss alternatives to this definition that can be successfully lifted to matrices.

4. Building Paths using Bricks

4.1. Nodes and Tests

A path should always end up in a node. The simplest path is a node with a connection to itself. In the routing algebra a node has exactly one entry different to (ε, ∞). Moreover, this value needs to be the multiplicative unit $(\varepsilon, 0)$ and on the diagonal. An example is given in Figure 3.

$$\begin{pmatrix} \boldsymbol{(\varepsilon,0)} & (\varepsilon,\infty) & (\varepsilon,\infty) \\ (\varepsilon,\infty) & (\varepsilon,\infty) & (\varepsilon,\infty) \\ (\varepsilon,\infty) & (\varepsilon,\infty) & (\varepsilon,\infty) \end{pmatrix}$$

Figure 3: A pre-node

Definition 4.1. Assume an i-semiring S with greatest element $\top$. An element $a \in S$ is a *pre-node* if it is a subidentity ($a \leq 1$), join-irreducible, and satisfies $\top \cdot a \cdot \top = \top$.

In models based on matrices, the subidentity-property forces the non-trivial value to sit on the diagonal.

In the setting of relation algebra, Schmidt and Ströhlein defined the concept of points [25]. They were inspired by the "general theory of graphs". The intuition behind both definitions is the same—modelling single nodes in a graph. A point in a concrete relation algebra is modelled as a matrix that has exactly one row filled with 1's. Although pre-nodes and points are not equivalent, they are closely related; see Appendix Appendix A for more details.

Pre-nodes and points are also related to the concept of tests [26]. A *test semiring* [26] is a pair $(S, \mathsf{test}(S))$, where S is an i-semiring and $\mathsf{test}(S) \subseteq [0,1]$ is a Boolean subalgebra of the set $[0,1]$ of S such that $0, 1 \in \mathsf{test}(S)$ and join and meet in $\mathsf{test}(S)$ coincide with adddition $(+)$ and multiplication $(\cdot)$. By $\neg$ we denote complementation in $\mathsf{test}(S)$. Since the test algebra forms a Boolean algebra, the following shunting rule holds for $p, q, r \in \mathsf{test}(S)$

$$p \cdot q \leq r \;\Leftrightarrow\; p \leq \neg q + r \;.$$

Idempotent semirings admit at least the test algebra $\{0,1\}$ and can have different test algebras. A semiring with test algebra $\{0,1\}$ is called *discrete*.

In the matrix model, the maximal test algebra contains all diagonal matrices with tests of the underlying algebra on its diagonal. Hence Theorem 3.2 can be extended to hold for Kleene algebras with tests.

Lemma 4.2. Assume a matrix-semiring $M(n,S)$ of $n \times n$ matrices over an i-semiring S. The following properties hold:

(1) The identity matrix I equals the sum of all pre-nodes.

(2) For two pre-nodes $a, b$with $a \neq b$, $a \cdot b = 0$.

(3) Every node is a (join-irreducible) element of the maximal test algebra, but

(4) not every join-irreducible test element is a pre-node.

Proof: Parts (1) and (2) are straightforward, using arguments on the structure of pre-nodes. Part (3) follows by setting $\neg a = \sum_{b \in \{c \mid c \text{ is pre-node, } c \neq a\}} b$ for a node a. Basically, the idea is to create a diagonal matrix, which entries are 0, where the pre-nodes a had a non-zero entry, and vice versa. Note, that the complement of a is in general not a pre-node. For Part (4) we give a counter example. Assume $S = \mathsf{test}(S) = \{0, p, \neg p, 1\}$; all operations are given by the Boolean algebra of tests. We now consider 2×2-matrices. The matrix $\left(\begin{smallmatrix} p & 0 \\ 0 & 0 \end{smallmatrix}\right)$ is a test in the matrix-semiring, but not a pre-node, since

$$\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} p & 0 \\ 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} p & p \\ p & p \end{pmatrix} \neq \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} . \qquad \square$$

This lemma shows that the rule $\top \cdot a \cdot \top = \top$ is crucial for our definition of pre-nodes. However, it also shows that pre-nodes often behave similar to tests.

Proposition 4.3. *Assume a matrix-algebra over an i-semiring (a Kleene algebra) that has a discrete test algebra that is a maximal test algebra. Then all tests are sums of pre-nodes; in case of* 0*, it is the empty sum.*

Proof: Tests in a matrix-algebra that has a discrete test algebra that is a maximal test algebra are diagonal matrices. The entries on the diagonal are either 1 or 0. Since matrices with exactly one 1 on the diagonal are pre-nodes, the claim is trivial. □

In the general setting of semirings Lemma 4.2 does not hold; counter examples can easily generated by automated tools, such as Mace4 [27]. For example, the following generated example presents a counter example of Lemma 4.2(2).

$+$	0	a	1	$\top$
0	0	a	1	$\top$
a	a	a	1	$\top$
1	1	1	1	$\top$
$\top$	$\top$	$\top$	$\top$	$\top$

$\cdot$	0	a	1	$\top$
0	0	0	0	0
a	0	a	a	$\top$
1	0	a	1	$\top$
$\top$	0	$\top$	$\top$	$\top$

All elements are join-irreducible, a and 1 are the only pre-nodes; but obviously $a \cdot 1 = a$. This counter example also illustrates that 1 cannot be a pre-node, except it is the only one.

It has been shown in several situations that tests are a useful and powerful (e.g. [9, 28]). Since we want to make use of both concepts pre-nodes and tests, we combine them.

Definition 4.4. A *node* of an i-semiring is an element which is both a pre-node and a test.

For an i-semiring S, the set of all nodes is denoted by $\mathcal{N}_S$.

Premultiplying an arbitrary matrix in the routing algebra by a pre-node, a node, or a test selects certain rows; hence the routing table information about particular nodes can be selected. Postmultiplying selects columns, i.e., the matrix is "restricted" to information about a particular node or a set of nodes.

It has been shown that i-semirings and Kleene algebras can be equipped with forward and backward modal operators [29], which can be defined via tests. These operators provide a concise and convenient algebraic framework for calculi such as the wp calculus and predicate transformer semantics, but they have also been used in the context of routing [21]. In this paper we will use this concept to model paths. The (modal) forward box-operator $|\_] : S \to (\mathsf{test}(S) \to \mathsf{test}(S))$ is defined by

$$p \leq |a]q \;\Leftrightarrow\; p \cdot a \cdot \neg q \leq 0 \;,\text{ and} \qquad |a \cdot b]p \;=\; |a](|b]p) \;.$$

The forward diamond-operator $|\_\rangle$ is the de Morgan dual of this operation, i.e., $|a\rangle p \;=\; \neg|a]\neg p$. This operator will be used to determine whether there is a route from p to q in a (checking $p \leq |a\rangle q$). The backward operators are defined symmetrically by $p \leq [a|q \;\Leftrightarrow\; \neg q \cdot a \cdot p \leq 0$, $[a \cdot b|p = [a|([b|p)$ and $\langle a|p = \neg[a|\neg p$. Others have derived many properties for modal Kleene algebra (e.g., [29, 30]); here, we only list properties used in the remainder. For an i-semiring S with $a \in S$, $p, q \in \mathsf{test}(S)$, we have

$$p \cdot a = 0 \;\Leftrightarrow\; \langle a|p = 0 \;, \qquad |a\rangle p \leq q \;\Leftrightarrow\; p \leq [a|q \;,\text{ and} \qquad |p \cdot a\rangle q = p \cdot |a\rangle q \;.^{4}$$

Note that diamond- and box-operators bind more tightly than addition and multiplication. Test semirings equipped with $|\_]$ and $|\_\rangle$ are called *modal*.

It is straight forward to see that both the next-hop semiring and the routing algebra are modal. Modal operators can be used to check whether a route from p to q is known at a snapshot a, checking $p \leq |a\rangle q$. The dual inequality $p \leq \langle a|q$ checks whether p is a known destination of q.

4.2. Bricks

Based on the definition of nodes we can now turn to paths. A path should list all intermediate nodes as 1-hop neighbours and then finally reach the destination, where it "loops" forever. Remember Figure 2(b): the path from F to D is given by 1-hop neighbours—a packet travelling along that path needs to be sent

[4] Symmetric rules for the other type of box- and diamond operator hold, but are not listed explicitly.

via the 1-hop connection to D, then to B and finally to A. That means that only entries of the form $(*, 1)$, $(\varepsilon, 0)$ and (ε, ∞) should be allowed.

If we are able to characterise matrices that have at most one entry of the form $(*, 1)$ or $(\varepsilon, 0)$ (all other entries being (ε, ∞)), then paths can be built from these elements. A matrix with one entry of the form $(*, 1)$ would plead for something like multiplicative-irreducibility on matrix level. However, as mentioned before the definition of multiplicative-irreducibility does not lift to matrices. A counter example for this property, using elements of the routing algebra, is the following

$$\begin{pmatrix} (\varepsilon,\infty) & (B,1) \\ (\varepsilon,\infty) & (\varepsilon,\infty) \end{pmatrix} = \begin{pmatrix} (B,1) & (\varepsilon,\infty) \\ (\varepsilon,\infty) & (A,3) \end{pmatrix} \cdot \begin{pmatrix} (\varepsilon,\infty) & (\varepsilon,0) \\ (\varepsilon,\infty) & (\varepsilon,\infty) \end{pmatrix}.$$

Intuitively, the matrix on the left-hand side should be irreducible since the path $(B, 1)$ cannot be split into two non-trivial paths. However it can be split into two matrices, both different from the original one. The example also shows that the splitting matrices can contain arbitrary entries; in particular entries with a hop count larger than 1. To exclude matrices with multiple non-trivial entries, we can make use of join-irreducibility; to allow different positioning of an element in a matrix, we will fill up the matrix using the "$\top \cdot x \cdot \top$-construction" and compare the results.

Definition 4.5. An element a of an i-semiring S is called a *brick* if

(1) a is join-irreducible, and

(2) $\forall b, c : a = b \cdot c \Rightarrow (\top \cdot a \cdot \top = \top \cdot b \cdot \top \ \vee \ \top \cdot a \cdot \top = \top \cdot c \cdot \top)$.

The element $\left(\begin{smallmatrix} (\varepsilon,\infty) & (B,1) \\ (\varepsilon,\infty) & (\varepsilon,\infty) \end{smallmatrix}\right)$ is a brick. For an i-semiring, we denote the set of all bricks by $\mathcal{B}_S$, i.e., $\mathcal{B}_S = \{a \in S \mid a \text{ is a brick}\}$.

Lemma 4.6. In the routing algebra, an element a is a brick if and only if it contains at most one entry of the form $(*, 1)$ or $(\varepsilon, 0)$; all other entries are (ε, ∞).

Proof: Join-irreducibility does not allow multiple non-(ε, ∞) entries. Assume a matrix M that has an entry with hop count larger than 1, say $(B, 3)$, at position i, j, and that satisfies (2). In the next-hop semiring such a single entry can be split into two non-trivial entries: for example, $(B, 3) = (B, 1) \cdot (A, 2)$. It is easy to construct matrices N and O such that $M = N \cdot O$. In detail, all entries of N and O are equal to (ε, ∞), except N_{ij} and O_{jk}, which are $(B, 1)$ and $(A, 2)$ respectively (for some k). Neither $\top \cdot N \cdot \top = \top \cdot M \cdot \top$ nor $\top \cdot N \cdot \top = \top \cdot O \cdot \top$ hold, hence (2) is not satisfied, a contradiction. □

Corollary 4.7. *In any algebra that satisfies the Tarski-rule $a \neq 0 \Rightarrow \top \cdot a \cdot \top = \top$ the second item of Definition 4.5 is a tautology. Therefore, in such algebras, a brick is equivalent to a join-irreducible element.*

However, most i-semirings do not satisfy the Tarski-rule, e.g., the next-hop semiring and the routing algebra. Hence the concepts of join-irreducibility and bricks are different.

4.3. Paths

Finally, we can define paths as a sequence of bricks (between nodes) leading to a single node, the destination. Before we give a formal definition, we look at some examples of paths and non-paths. The first example, depicted in Figure 4, shows a simple path, leading from C via B to A. We notice that a matrix, modelling a path, should only contain entries of the form $(*, 1)$, (ε, ∞) and $(\varepsilon, 0)$ (as mentioned before). We further notice that every row has exactly one entry; and every column has exactly one entry as well, except the column containing $(\varepsilon, 0)$, which has two entries. All paths we want to characterise should be infinite. That means a path either loops at a single node at its end or forms a loop (2nd example of Figure 4); the third example shows an example, which is not a path. All nodes on a path have at most one successor and at most one predecessor. The only exception is the final node: since at this node the path can loop forever, it has up to two predecessors.

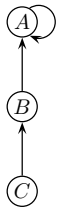
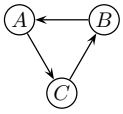
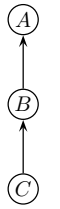
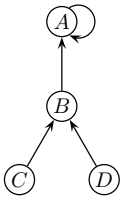
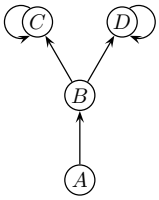
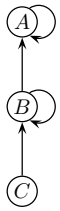
graph[5]	matrix representation	description
	$\begin{array}{c} \\ A \\ B \\ C \end{array} \begin{array}{c} \begin{array}{ccc} A & B & C \end{array} \\ \begin{pmatrix} (\varepsilon,0) & (\varepsilon,\infty) & (\varepsilon,\infty) \\ (A,1) & (\varepsilon,\infty) & (\varepsilon,\infty) \\ (\varepsilon,\infty) & (B,1) & (\varepsilon,\infty) \end{pmatrix} \end{array}$	One of the most simple paths; it starts at C, takes B as intermediate node and reaches node A. Paths should only contain entries of the form $(*,1)$, (ε,∞) and $(\varepsilon,0)$.
	$\begin{array}{c} \\ A \\ B \\ C \end{array} \begin{array}{c} \begin{array}{ccc} A & B & C \end{array} \\ \begin{pmatrix} (\varepsilon,\infty) & (\varepsilon,\infty) & (C,1) \\ (A,1) & (\varepsilon,\infty) & (\varepsilon,\infty) \\ (\varepsilon,\infty) & (B,1) & (\varepsilon,\infty) \end{pmatrix} \end{array}$	Paths are allowed to have non-trivial loops.
	$\begin{array}{c} \\ A \\ B \\ C \end{array} \begin{array}{c} \begin{array}{ccc} A & B & C \end{array} \\ \begin{pmatrix} (\varepsilon,\infty) & (\varepsilon,\infty) & (\varepsilon,\infty) \\ (A,1) & (\varepsilon,\infty) & (\varepsilon,\infty) \\ (\varepsilon,\infty) & (B,1) & (\varepsilon,\infty) \end{pmatrix} \end{array}$	We assume that every node has at least one successor node. Since A does not have a successor, this is not a path (by our understanding).
	$\begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{pmatrix} (\varepsilon,0) & (\varepsilon,\infty) & (\varepsilon,\infty) & (\varepsilon,\infty) \\ (A,1) & (\varepsilon,\infty) & (\varepsilon,\infty) & (\varepsilon,\infty) \\ (\varepsilon,\infty) & (B,1) & (\varepsilon,\infty) & (\varepsilon,\infty) \\ (\varepsilon,\infty) & (B,1) & (\varepsilon,\infty) & (\varepsilon,\infty) \end{pmatrix} \end{array}$	No node can have more than one predecessor (except the destination); hence no path.
	$\begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{pmatrix} (\varepsilon,\infty) & (B,1) & (\varepsilon,\infty) & (\varepsilon,\infty) \\ (\varepsilon,\infty) & (\varepsilon,\infty) & (C,1) & (D,1) \\ (\varepsilon,\infty) & (\varepsilon,\infty) & (\varepsilon,0) & (\varepsilon,\infty) \\ (\varepsilon,\infty) & (\varepsilon,\infty) & (\varepsilon,\infty) & (\varepsilon,0) \end{pmatrix} \end{array}$	Every node has exactly one successor; hence no path.
	$\begin{array}{c} \\ A \\ B \\ C \end{array} \begin{array}{c} \begin{array}{ccc} A & B & C \end{array} \\ \begin{pmatrix} (\varepsilon,0) & (\varepsilon,\infty) & (\varepsilon,\infty) \\ (A,1) & (\varepsilon,0) & (\varepsilon,\infty) \\ (\varepsilon,\infty) & (B,1) & (\varepsilon,\infty) \end{pmatrix} \end{array}$	Paths are not allowed to have loops at intermediate nodes. (This coincides with the multiple successor and multiple predecessor rules.)

Figure 4: Two paths and four non-paths

[5]The graphical representation does not show the hop count. In this figure we assume self-loops to have hop count 0 and all other connections to have hop count 1.

We now present the main definition of this paper.

Definition 4.8. A *path* of an i-semiring S is an element $a \in S$ satisfying the following three properties.

(1) $\exists\, B \subseteq \mathcal{B}_S : a = \sum_{b \in B} b$;

(2) $\forall\, p, q \in \mathcal{N}_S : q \cdot a \cdot p \neq 0 \;\Rightarrow\; p \cdot a \neq 0$;

(3) $\forall\, p \in \mathcal{N}_S : \neg p \cdot |a\rangle p \in \mathcal{N}_S \;\wedge\; \langle a| p \in \mathcal{N}_S$.

In the routing algebra (1) means that the matrix consists of $(\varepsilon, 0)$, (ε, ∞) and $(*, 1)$-entries only; (2) guarantees a successor of a path (every node must have (at least one)): if node p can be reached from q via a then the path a must have a continuation from p. (3) restricts paths to have at most one successor and at most one predecessor; with the one exception mentioned before. In the routing algebra this definition allows degenerate paths, meaning that there are snapshots that are paths and that have $(\varepsilon, 0)$ somewhere outside the diagonal. However, in routing tables this never occurs, since all path have at least length 1.

It is not convenient to work with this definition due to the inequations. However, Part (2) can be read as $p \cdot a \leq 0 \;\Rightarrow\; q \cdot a \cdot p \leq 0$. Hence, by using the formulas of Page 8, we get the following result.

Proposition 4.9. *Def. 4.8(2) is equivalent to* $\langle a| p \leq 0 \;\Rightarrow\; p \cdot \langle a| q \leq 0$, $\forall\, p, q \in \mathcal{N}_S$.

In the routing algebra, a snapshot contains a path, if both matrices coincide at all positions where the path is not (ε, ∞).

Definition 4.10. Assume an i-semiring S. A path $b \in \mathcal{B}_S$ is *contained* in an element a, written as $b \sqsubseteq a$, if

$$\forall\, p, q \in \mathcal{N}_S : p \cdot b \cdot q = p \cdot a \cdot q \;\vee\; p \cdot b \cdot q = 0 \;.$$

Proposition 4.11. *Let S be an i-semiring.*

(1) The relation $\sqsubseteq$ is reflexive.

(2) The relation $\sqsubseteq$ is transitive, i.e., for all $a \in S$ and $b, c \in \mathcal{B}_S$

$$b \sqsubseteq c \wedge c \sqsubseteq a \;\Rightarrow\; b \sqsubseteq a \;.$$

The proof is straight forward, using the definition.

5. A Brief Application in Routing

After setting up the definitions, we now briefly turn to an application in routing. Here, we sketch modelling aspects, i.e., we show how certain routing properties can be characterised by algebra. Using the presented characterisation to reason about concrete protocols is part of future work.

5.1. Properties of Routing Protocols

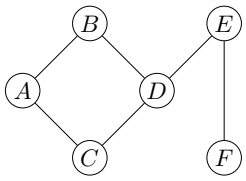
We first model the packet delivery property: if a node has information about a destination, then there is a path in the snapshot leading to the destination.

In the routing algebra, a node A has information about destination D (with respect to a given snapshot) if the corresponding entry in the snapshot is not equal to (ε, ∞). In Figure 2(b), node F has information about A, since the corresponding entry is $(D, 3)$. This can be checked by restricting the snapshot to the corresponding nodes.

Algebraically, a node p has information about node d w.r.t. a if $p \cdot a \cdot a \neq 0$. By this we can define packet delivery.

Definition 5.1. Let K be a Kleene algebra and $a \in K$. A node p satisfies the *packet delivery* property for node q if a contains a path that is consistent with the corresponding position:

$$p \cdot a \cdot q \neq 0 \;\Rightarrow\; \exists\, b \in \mathcal{B}_K :\; b \sqsubseteq a \wedge p \cdot b^* \cdot q \;=\; p \cdot a \cdot q \;.$$



$$\begin{array}{c} \\ A \\ B \\ C \\ D \\ E \\ F \end{array} \begin{array}{c} \begin{array}{cccccc} A & B & C & D & E & F \end{array} \\ \begin{pmatrix} (\varepsilon,0) & (B,1) & (\varepsilon,\infty) & (\varepsilon,\infty) & (\varepsilon,\infty) & (\varepsilon,\infty) \\ (A,1) & (\varepsilon,0) & (\varepsilon,\infty) & (\varepsilon,\infty) & (\varepsilon,\infty) & (\varepsilon,\infty) \\ (A,1) & (\varepsilon,\infty) & (\varepsilon,0) & (\varepsilon,\infty) & (\varepsilon,\infty) & (\varepsilon,\infty) \\ (\mathbf{B},\mathbf{2}) & (B,1) & (\varepsilon,\infty) & (\varepsilon,0) & (\varepsilon,\infty) & (\varepsilon,\infty) \\ (\varepsilon,\infty) & (\varepsilon,\infty) & (\varepsilon,\infty) & (\varepsilon,\infty) & (\varepsilon,0) & (\varepsilon,\infty) \\ (\varepsilon,\infty) & (\varepsilon,\infty) & (\varepsilon,\infty) & (\varepsilon,\infty) & (\varepsilon,\infty) & (\varepsilon,0) \end{pmatrix} \end{array}$$

Figure 5: Graph and snapshot

Example 5.2. Assume the snapshot depicted in Figure 5. This snapshot, which we denote M, does not contain many paths; it is a typical example of a network running AODV where not many messages have been sent. In fact, only A broadcast a message in the past to its neighbours B and C, and B forwarded this message.

Let us look at D's routing table. It is easy to see that D satisfies the packet delivery property for A. However, if the entry would be $(C,2)$—instead of $(B,2)$—the property would not hold anymore, since M contains no path from D via C to A. □

In the same vein as packet delivery, we can characterise loop freedom. Informally, loop freedom states that at no time there should be a non-trivial loop in a snapshot. We can use the transitive closure to check for non-trivial paths. As usual, the *transitive closure* of an element a of a Kleene algebra is defined as $a^+ = a \cdot a^*$.

Definition 5.3. An element a of a Kleene algebra K contains a loop if

$$\exists\, b \in \mathcal{B}_K,\, p \in \mathcal{N}_K : b \sqsubseteq a \wedge p \cdot b^+ \cdot p \neq 0 \wedge p \cdot b^+ \cdot p \neq 1 \ .$$

An element is *loop free* if it does not contain a loop.

5.2. Broadcast, Unicast and Forward

As mentioned earlier message sending can be expressed by

$$a + b \cdot (1 + c) \ ,$$

where a, b, c are elements of an i-semiring S. By distributivity, this expression consists of three parts: a, b and $b \cdot c$. Informally, a describes the system before the message is sent (hence it is not part of the message itself); it is a snapshot which then, after the message has been delivered, is updated by the two other summands. The node b represents the current topology[6] and establishes a 1-hop connection between nodes. The third term $(b \cdot c)$ transmits knowledge c via the topology; during sending the knowledge is adapted to the topology. We refer to [21] for details.

From this algebraic perspective broadcast and unicast is the same. But, the topology must be adapted in a way that it only offers the links used.

Example 5.4. Let us look at the topology that corresponds to the graph given in Figure 5:

$$\begin{array}{c} \\ A \\ B \\ C \\ D \\ E \\ F \end{array} \begin{array}{c} \begin{array}{cccccc} A & B & C & D & E & F \end{array} \\ \begin{pmatrix} (\varepsilon,0) & (B,1) & (C,1) & (\varepsilon,\infty) & (\varepsilon,\infty) & (\varepsilon,\infty) \\ (A,1) & (\varepsilon,0) & (\varepsilon,\infty) & (D,1) & (\varepsilon,\infty) & (\varepsilon,\infty) \\ (A,1) & (\varepsilon,\infty) & (\varepsilon,0) & (D,1) & (\varepsilon,\infty) & (\varepsilon,\infty) \\ (\varepsilon,\infty) & (B,1) & (C,1) & (\varepsilon,0) & (E,1) & (\varepsilon,\infty) \\ (\varepsilon,\infty) & (\varepsilon,\infty) & (\varepsilon,\infty) & (D,1) & (\varepsilon,0) & (F,1) \\ (\varepsilon,\infty) & (\varepsilon,\infty) & (\varepsilon,\infty) & (\varepsilon,\infty) & (E,1) & (\varepsilon,0) \end{pmatrix} \end{array}$$

[6] By a topology we understand a matrix which characterises a particular graph, i.e., it only contains entries of the form $(*,1)$, (ε,∞) and $(\varepsilon,0)$.

If node D wants to broadcast a message then the topology should be restricted to the 4th row; this is easily achieved by premultiplying the topology by node D.

In AODV, route replies are unicast back to the originator of the route discovery process. We have to restrict the topology or more precisely the snapshot in such a way that it contains only the single path back to the originator. If we consider the snapshot given in Figure 5, and D needs to unicast a packet to A, we have to pick a path starting with the entry $(D, 1)$. This can be done by the path selected for the packet delivery property. If this path is restricted to the sender, the packet is sent to the next hop. □

Propagating and forwarding the message through the whole network can be expressed by using Kleene star.

$$a + b \cdot |b^*\rangle p + b^* \cdot p \ .$$

This equation, which was derived in [21], is now true for both, broadcast and unicast, depending on the topology chosen. Remember that for unicast a path is chosen.

6. Conclusion and Outlook

In this paper we have treated matrices carrying information useful for routing algorithms. For modelling routing tables we have used pairs as entries of the matrices. These pairs consist of the length of a path and the next hop on the path. Using examples of wireless mesh networks we have argued that a path is a concept needed to reason about routing. We have then shown how to treat paths algebraically and have established a relationship to routing algorithms. Having the concept of paths solved one problem that occurred when modelling a unicast-machanism, and which was mentioned in [21].

The approach we follow in [21] and in this paper is based on ideas of Sobrinho and Griffin [31, 32, 33] and of Carré [34] Sobrinho and Griffin were the first to bring algebraic reasoning into the realm of hop-by-hop routing. Sobrinho [31] states that he follows an algorithmic approach while others use a matrix approach [34]. One of our aim is to combine these two approaches. A recent contribution, also based on algebraic principles, is NetKAT [35]. It aims to provide a solid mathematical foundation for new network programming language based on Kleene algebras with tests [26]. This approach is orthogonal to ours: whereas our goal is to be able to derive and verify routing algorithms, the aim of NetKAT to use the algebraic foundation to ensure that languages built on top of it satisfy some universal properties.

So far we concentrated on modelling crucial properties of routing protocols algebraically; formal reasoning using the given definitions is part of future work. Moreover we hope to find even simpler characterisations for bricks, paths and the properties of packet delivery and loop freedom: this would be extremely useful since inequations, as they appear in the definitions, are inconvenient for algebraic reasoning.

So far the routing algebra is based on pairs storing the next hop and the hop count. Routing protocols used in industry save additional information inside the routing tables. An example are sequence numbers which indicate the freshness of routes. However, when including this into the routing algebra, we cannot use Theorem 3.1 any longer and the resulting matrix algebra does not satisfy distributivity nor associativity laws. A solution to this is to use module-like structures such as Kleene modules [36]; a detailed analysis, however, on the relationship between our treatment of paths, the modelling of AODV (as presented in [21]) and Kleene modules is part of future work.

> *'Copilowish [11] "enjoyed the full benefits of the matrix approach" and regretted that this "elegant machinery is apparently too little known". We think these feelings can be shared today.'*[7]
>
> Gunther Schmidt [25]

Acknowledgement. NICTA is funded by the Australian Government through the Department of Communications and the Australian Research Council through the ICT Centre of Excellence Program.

[7] The reference [11] in the quote is the same as [37] in our references.

References

[1] B. Carré, Graphs and Networks, Oxford applied mathematics and computing science series, Clarendon Press, 1979.
[2] G. Schmidt, T. Ströhlein, Relationen und Graphen, Mathematik für Informatiker, Springer, 1989.
[3] E. Mayr, G. Schmidt, G. Tinhofer (Eds.), Graph-Theoretic Concepts in Computer Science, volume 903 of *Lecture Notes in Computer Science*, Springer, 1995.
[4] G. Schmidt, T. Ströhlein, Relations and Graphs: Discrete Mathematics for Computer Scientists, Springer, 1993.
[5] G. Schmidt, Relational Mathematics, Encyclopedia of Mathematics and its Applications, Cambridge University Press, 2011.
[6] P. Höfner, G. Struth, On automating the calculus of relations, in: A. Armando, P. Baumgartner, G. Dowek (Eds.), Automated Reasoning, volume 5195 of *Lecture Notes in Artificial Intelligence*, Springer, 2008, pp. 50–66. doi:10.1007/978-3-540-71070-7_5.
[7] J. H. Conway, Regular Algebra and Finite Machines, Chapman & Hall, 1971.
[8] D. Kozen, On Kleene algebras and closed semirings, in: MFCS '90: Proceedings on Mathematical foundations of computer science 1990, Lecture Notes in Computer Science, Springer, 1990, pp. 26–47.
[9] D. Kozen, On Hoare logic and Kleene algebra with tests, ACM Transactions on Computational Logic 1 (2000) 60–76.
[10] B. Heidergott, G. J. Olsder, J. W. van der Woude, Max Plus at Work : Modeling and Analysis of Synchronized Systems : A Course on Max-Plus algebra and its Applications, Princeton series in Applied Mathematics, Princeton University Press, 2006.
[11] W. Kuich, Semirings and formal power series: Their relevance to formal languages and automata, in: Handbook of formal languages, Vol. 1, Springer, 1997, pp. 609–677.
[12] Y. Kawahara, On the cardinality of relations, in: R. Schmidt (Ed.), Relations and Kleene Algebra in Computer Science, volume 4136 of *Lecture Notes in Computer Science*, Springer, 2006, pp. 251–265.
[13] P. Höfner, B. Möller, Dijkstra, Floyd and Warshall meet Kleene, Formal Aspects of Computing 24 (2012) 459–476.
[14] J. von Wright, Towards a refinement algebra, Science of Computer Programming 51 (2004) 23–45.
[15] B. Möller, Modal knowledge and game semirings, The Computer Journal 56 (2013) 53–69.
[16] G. Schmidt, Relational concepts in social choice, in: W. Kahl, T. Griffin (Eds.), Relational and Algebraic Methods in Computer Science, volume 7560 of *Lecture Notes in Computer Science*, Springer, 2012, pp. 278–293.
[17] G. Schmidt, R. Berghammer, Relational measures and integration in preference modeling, Journal of Logic and Algebraic Programming 76 (2008) 112–129.
[18] B. Möller, P. Roocks, M. Endres, An algebraic calculus of database preferences, in: J. Gibbons, P. Nogueira (Eds.), Mathematics of Program Construction, volume 7342 of *Lecture Notes in Computer Science*, Springer, 2012, pp. 241–262.
[19] J. Desharnais, B. Möller, G. Struth, Algebraic notions of termination, Logical Methods in Computer Science 7 (2011).
[20] W. Guttmann, Unifying lazy and strict computations, in: W. Kahl, T. Griffin (Eds.), Relational and Algebraic Methods in Computer Science, volume 7560 of *Lecture Notes in Computer Science*, Springer, 2012, pp. 17–32.
[21] P. Höfner, A. McIver, Towards an algebra of routing tables, in: H. de Swart (Ed.), Relational and Algebraic Methods in Computer Science, volume 6663 of *Lecture Notes in Computer Science*, Springer, 2011, pp. 212–229. doi:10.1007/978-3-642-21070-9_17.
[22] C. Perkins, E. Belding-Royer, S. Das, Ad hoc on-demand distance vector (AODV) routing, RFC 3561 (Experimental), 2003. URL: http://www.ietf.org/rfc/rfc3561.txt, http://www.ietf.org/rfc/rfc3561.txt.
[23] R. Backhouse, B. Carré, Regular algebra applied to path-finding problems, Journal of the Institute of Mathematics and Applications (1975).
[24] D. Kozen, A completeness theorem for Kleene algebras and the algebra of regular events, Information and Computation 110 (1994) 366–390.
[25] G. Schmidt, T. Ströhlein, Relation algebras: Concept of points and representability, Discrete Mathematics 54 (1985) 83 – 92.
[26] D. Kozen, Kleene algebra with tests, ACM Transactions on Programming Languages and Systems 19 (1997) 427–443.
[27] W. W. McCune, Prover9 and Mace4, <http://www.cs.unm.edu/~mccune/prover9>, . (accessed December 2, 2013).
[28] C. Bolduc, J. Desharnais, Static analysis of programs using omega algebra with tests, in: W. MacCaull, M. Winter, I. Düntsch (Eds.), Relational Methods in Computer Science, volume 3929 of *Lecture Notes in Computer Science*, Springer, 2006, pp. 60–72.
[29] J. Desharnais, B. Möller, G. Struth, Modal Kleene algebra and applications — A survey, Journal of Relational Methods in Computer Science 1 (2004) 93–131.
[30] J. Desharnais, B. Möller, G. Struth, Kleene algebra with domain, ACM Transactions on Computational Logic 7 (2006) 798–833.
[31] J. Sobrinho, Algebra and algorithms for QoS path computation and hop-by-hop routing in the internet, IEEE/ACM Trans. Networking 10 (2002) 541–550.
[32] J. Sobrinho, Network routing with path vector protocols: Theory and applications, in: Applications, technologies, architectures, and protocols for computer communications, SIGCOMM '03, ACM Press, 2003, pp. 49–60.
[33] T. Griffin, J. Sobrinho, Metarouting, SIGCOMM Comp. Com.. Rev. 35 (2005) 1–12.
[34] B. Carré, Graphs and Networks, Oxford Applied Mathematics & Computing Science Series, Oxford University Press, 1980.
[35] C. Anderson, N. Foster, A. Guha, J.-B. Jeannin, D. Kozen, C. Schlesinger, D. Walker, NetKAT: Semantic foundations for networks, in: Symposium on Principles of Programming Languages (POPL 14), ACM, 2014, pp. 113–126.
[36] H. Leiß, Kleene modules and linear languages, Journal of Logic and Algebraic Programming 66 (2006) 185 – 194.

[37] I. Copilowish, Matrix development of the calculus of relations, Journal of Symbolic Logic 13 (1948) 193–203.
[38] R. Maddux, Relation Algebras, volume 150 of *Studies in Logic and the Foundations of Mathematics*, Elsevier, 2006.

Appendix A. Relation Algebra

Description of binary relations and their basic operations can be found in most introductory courses on discrete mathematics. There are many textbooks were more information can be found (e.g. [4, 38]).

A binary (homogeneous) relation R over a set M is a subset of $M \times M$—a set of ordered pairs. Since relations are sets, unions $R \cup S$, intersections $R \cap S$ and complements $\overline{R}$ of relations can be easily defined. In this case the set of all binary relations forms a Boolean algebra. For two relations R and S, the *relational product* $R;S$ is the set of all pairs (x,y) such that there exists a $z \in M$ with $(x,z) \in R$ and $(z,y) \in S$. The *converse* R^{T} of a relation R is the set of all pairs (y,x) with $(x,y) \in R$. The *identity relation* $\mathbb{I}_M$ on M is the set containing all pairs (x,x) with $x \in M$. The structure $(\mathcal{P}(M \times M), \cup, ;, ^-, ^{\mathsf{T}}, \mathbb{I}_M)$ is called *concrete relation algebra* of all binary relations over M. A representation as 0-1-matrix, as done in this paper, is straight forward.

To define relation algebras more abstractly, binary relations are replaced by arbitrary elements of some carrier set M and a set of equational axioms is given. A *(abstract) relation algebra* is a structure $(M, +, ;, ^-, ^{\mathsf{T}}, \mathbb{I})$ satisfying the axioms

$$(R+S)+T = R+(S+T)\ , \qquad R+S = S+R\ , \qquad R = \overline{\overline{R}+\overline{S}}+\overline{\overline{R}+S}\ ,$$
$$(R;S);T = R;(S;T)\ , \qquad (R+S);T = R;T+S;T\ , \qquad R;\mathbb{I} = R\ ,$$
$$R^{\mathsf{T}^{\mathsf{T}}} = R\ , \qquad (R+S)^{\mathsf{T}} = R^{\mathsf{T}}+S^{\mathsf{T}}\ , \qquad R^{\mathsf{T}};\overline{R;S}+\overline{S} = \overline{S}\ .$$

Since relation algebras are Boolean algebras, they form partially ordered sets with respect to $R \leq S \Leftrightarrow R+S = S$; the meet of Boolean algebra is defined as $R \cap S = \overline{\overline{R}+\overline{S}}$, the greatest elements $\top = R+\overline{R}$ and least element $\mathbf{0} = R \cap \overline{R}$.[8]

In this appendix, we freely use well-known facts about relation algebra, such as isotonicity of all operators (except complement, which is antitone). Moreover, if needed we freely switch between matrix representation, sets and algebraic reasoning (whatever is more convenient).

Schmidt and Ströhlein define the concept of points in [25]. A *point* is a relation R satisfying the following three properties.

$$R = R;\top\ , \qquad R \neq \mathbf{0}, \qquad R;R^{\mathsf{T}} \subseteq \mathbb{I}\ .$$

We want to relate the concepts of pre-nodes and points in relation algebra.

Theorem Appendix A.1. *In a concrete relation algebra over a non-empty-set M, the concepts of point and node are closely related.*

(1) *R is a point $\Rightarrow$ $R \cap \mathbb{I}_M$ is a node.*

(2) *R is a node $\Rightarrow$ $R;\top_M$ is a point.*[9]

Proof: Assume a concrete relation algebra over a (non-empty) set M.

(1) $R \cap \mathbb{I}_M \subseteq \mathbb{I}_M$ is trivial; since the Tarski rule holds for concrete relation algebra, it remains to show join-irreducibility. In the concrete relation algebra, a point is a matrix with one row of 1's (see e.g. [4]). The intersection with $\mathbb{I}_M$ yields a matrix with exactly one entry equals to 1, which is join-irreducible.

[8] Schmidt often includes the Tarski rule $a \neq \mathbf{0} \Rightarrow \top;a;\top = \top$ as an axiom, whereas Maddux does not. Here, we use the more general case and skip this rule.

[9] The greatest element in a concrete relation algebra is denoted by $\top_M$.

(2) $R; \top_M = R; \top_M; \top_M$ is obvious. By $\top_M; R; \top_M = \top_M$, we have that $R \neq \mathbf{0}$, and hence $R; \top_M \neq \mathbf{0}$. The proof of $(R; \top_M); (R; \top_M)^{\mathsf{T}} \subseteq \mathbb{I}_M$ on matrix-level (similar as before): by join-irreducibility R has only one entry that is not equal $\mathbf{0}$. Hence $R; \top_M$ is a matrix with one row of 1's, for this, by the definition of matrix multiplication, the claim follows. □

For abstract relation algebra the claims do not hold; automated counter example generators such as Mace4 [27] can generate easily examples.