

Greedy-like algorithms in modal Kleene algebra

Bernhard Möller, Georg Struth

Angaben zur Veröffentlichung / Publication details:

Möller, Bernhard, and Georg Struth. 2004. "Greedy-like algorithms in modal Kleene algebra." *Lecture Notes in Computer Science* 3051: 202–14.
https://doi.org/10.1007/978-3-540-24771-5_18.



Greedy-Like Algorithms in Modal Kleene Algebra^{*}

Bernhard Möller Georg Struth

Institut für Informatik, Universität Augsburg
Universitätsstr. 14, D-86135 Augsburg, Germany
`moeller, struth@informatik.uni-augsburg.de`

Abstract. This study provides an algebraic background for the formal derivation of greedy-like algorithms. We propose Kleene algebra as a particularly simple alternative to previous approaches such as relation algebra. Instead of converse and residuation we use modal operators that are definable in a wide class of algebras, based on domain/codomain or image/pre-image operations. By abstracting from earlier approaches we arrive at a very general theorem about the correctness of loops that covers particular forms of greedy algorithms as special cases.

Keywords: Idempotent semiring, Kleene algebra, image and preimage operation, modal operators, confluence, Geach formula, program development and analysis.

1 Introduction

This study is concerned with algebraic derivations and correctness proofs of greedy-like algorithms that use a simple loop to calculate a global optimum. We present a fairly general correctness criterion and give sufficient criteria when iteration in a discrete partially ordered problem domain correctly implements the general algorithm scheme. Proper greedy algorithms are further specializations in which the loop steps are driven by an additional local optimality criterion.

Earlier work (e.g. [1,2]) has used relation algebra and other algebraic formalisms (e.g. [4]). We show that modal Kleene algebra allows a particularly simple, concise and more general treatment, avoiding the concepts of converse and residual, as used in relation algebra, and exhibiting the derivation structure more clearly.

The central correctness conditions require that certain optimality criteria semi-commute with certain program statements to achieve global optimality. These semi-commutation properties, of the same shape as confluence properties in rewriting systems, can be expressed without converse using modal operators as in dynamic logic; these can be added to Kleene algebra in a simple way.

^{*} Research partially sponsored by DFG Project InopSys — Interoperability of Calculi for System Modelling

2 Looping for Optimality

Greedy algorithms solve certain optimization problems, proceeding in a stepwise fashion without backtracking. At each step there is a set of choices from which a greedy algorithm always takes the one that seems best at the moment, i.e., it works locally without lookahead to the global optimum to be achieved eventually. Instances of this scheme are shortest path and minimum spanning tree problems in graphs, the construction of Huffman codes and scheduling problems. Of course, the greedy approach only works for certain types of problems: as is well-known from hiking in the mountains, always choosing the steepest path will rarely lead to the highest summit of the whole area. The central correctness requirement for the greedy scheme is that

a local choice must not impair reaching the global optimum.

In this and the following section we derive, within the framework of relational algebra, general conditions under which a loop satisfies this principle; later we abstract to Kleene algebra. It turns out that local optimality is inessential; so we obtain a more general class of loops that we call *greedy-like*.

We start with a specification relation T that connects inputs to admissible outputs and a relation C that compares outputs and captures the notion of (global) optimality. The derivation will exhibit our precise requirements on C .

A relation R *improves* T w.r.t. C if it always relates inputs to outputs that are at least as good as those prescribed by T , in formulas

$$\forall x, y, z : xTy \wedge xRz \Rightarrow y Cz, \quad \text{or, equivalently,} \quad T^\smile; R \subseteq C,$$

where $;$ denotes relational composition and T^\smile is the converse of T . Since \emptyset trivially improves T , we are interested in the greatest improvement and define it by the Galois connection

$$X \subseteq \text{GIMP}(T, C) \stackrel{\text{def}}{\Leftrightarrow} T^\smile; X \subseteq C. \quad (1)$$

Using a residual, this could be expressed as $\text{GIMP}(T, C) = T^\smile \setminus C$. However, we will not need any special properties of residuals.

An implementation of specification T that always produces optimal solutions then is a relation that refines and improves T . So we define

$$\text{OPT}(T, C) \stackrel{\text{def}}{=} T \cap \text{GIMP}(T, C).$$

We now consider a loop program

$$W \stackrel{\text{def}}{=} \text{while } P \text{ do } S \stackrel{\text{def}}{=} (P; S)^* ; \neg P.$$

Here the loop condition P is represented by a subidentity $P \subseteq I$, where I is the identity relation, and $\neg P \stackrel{\text{def}}{=} I - P$ is its relative complement. We want to calculate a sufficient criterion for W to be a refinement of $\text{OPT}(T, C)$, i.e., for

$$W \subseteq T, \quad (2) \quad W \subseteq \text{GIMP}(T, C), \quad (3)$$

where we defer the treatment of (2) to the next section.

Spelling out the definitions in (3) results in $T^\sim; (P; S)^*; \neg P \subseteq C$. We abstract a bit and try to answer the question when, for $Q \subseteq I$, we have

$$U; V^*; Q \subseteq C. \quad (4)$$

A standard result from regular algebra (see (15) in Section 5) is the semi-commutation property $W; X \subseteq Y; Z \Rightarrow W; X^* \subseteq Y^*; Z$. Hence (4) can be established given the conditions

$$U; V \subseteq C; U, \quad (5) \quad U; Q \subseteq C, \quad (6)$$

since then

$$U; V^*; Q \subseteq C^*; U; Q \subseteq C^*; C = C^+.$$

If we now assume C to be transitive, which is reasonable for a comparison relation, we have $C^+ \subseteq C$ and can draw the desired conclusion.

How can we, in turn, establish (5) and (6), at least in our special case? Translating back we get the proof obligations

$$T^\sim; P; S \subseteq C; T^\sim, \quad (7) \quad T^\sim; \neg P \subseteq C. \quad (8)$$

Let us interpret these conditions. (7) means that every pass through the loop body preserves the possibility of obtaining a solution that is at least as good as all possible solutions before. (8) means that upon loop termination no possible solution is better than the termination value.

3 Iterating Through the Problem Domain

We now decompose the specification relation T into the iteration of a set E of elementary steps between elements of the problem domain. We admit, as initial approximations, arbitrary inputs but as outputs only terminal elements from which no further elementary steps are possible. Therefore we assume now that T has the special shape

$$T = \text{exhaust } E \stackrel{\text{def}}{=} E^*; \neg E = \text{while } E \text{ do } E, \quad (9)$$

where the *domain* of a relation R is, as usual, defined by

$$\ulcorner R \stackrel{\text{def}}{=} R; R^\sim \cap I. \quad (10)$$

Such a problem structure is found, e.g., in matroids and greedoids [6,7] where it is additionally assumed that T is a discrete strict-order and that all terminal (or maximal) elements, the *bases*, have the same height (also known as *rank* or *dimension*) in the associated Hasse diagram.

We try to calculate an implementation that traverses the problem domain without backtracking, i.e., using elementary steps only forward. This suggests trying $P; S \subseteq E$. Now, by monotonicity of the star operation, proof obligation (2) can be fulfilled if additionally we can achieve $\neg P \subseteq \neg \ulcorner E$ or, equivalently, $\ulcorner E \subseteq P$. Sufficient conditions for these properties are

$$P; S \subseteq E \quad \ulcorner(P; S) \supseteq \ulcorner E . \quad (11)$$

These are reasonable requirements, since they prevent that the iteration blocks at a non-terminal element. They imply even $\ulcorner(P; S) = \ulcorner E$.

Next, we deal with proof obligation (8), assuming (9). We calculate

$$\begin{aligned} T^\smile; \neg \ulcorner E \subseteq C &\Leftrightarrow \neg \ulcorner E; T \subseteq C^\smile \Leftrightarrow \neg \ulcorner E; E^*; \neg \ulcorner E \subseteq C^\smile \Leftrightarrow \\ &\neg \ulcorner E; (I \cup E; E^*); \neg \ulcorner E \subseteq C^\smile \Leftrightarrow \neg \ulcorner E \subseteq C . \end{aligned}$$

Step one employs properties of converse. Step two uses (9). Step three unfolds the star. Step four uses distributivity, $\neg \ulcorner E; E = \emptyset$ and idempotence of $\neg \ulcorner E$.

So (8) is established if C is reflexive on terminal elements. This holds, in particular, if C is fully reflexive, i.e., a pre-order. But in some applications one may choose to leave C partially reflexive. E.g., when constructing a Huffman code, the non-terminal elements are proper forests, for which a comparison relation is not given as easily as for the terminal elements, which are single code trees.

As for proof obligation (7), it is a generic condition that has to be considered individually in each case. Our derivation can be summed up as follows.

Theorem 3.1 *Suppose that $T = \text{exhaust } E$ and that C is reflexive on $\neg \ulcorner E$ and transitive. Then $(7) \wedge (11) \Rightarrow \text{while } \ulcorner E \text{ do } S \subseteq \text{OPT}(T, C)$.*

So far we still have a general scheme that does not specifically mention greediness. But we can refine S further to choose in every step a locally optimal element. To this end we need another pre-order L and stipulate

$$S \subseteq \text{GIMP}(E, L) . \quad (12)$$

This now provides a truly greedy algorithm, the correctness of which is already shown by Theorem 3.1. It corresponds to Curtis's "Best-Global" algorithm [2].

4 From Converses to Modalities

The central step in the above derivation, viz. exhibiting conditions (7) and (8), uses only regular algebra. It is an interesting question whether the whole derivation can be ported to Kleene algebra by eliminating the converse operation in some way. This would generalize the result to a much wider class of models.

In the above formulas converse is used only in a very restricted way reminiscent of the relational formulation of a general diamond (or confluence) property:

$$R^\smile; S \subseteq T; U^\smile . \quad (13)$$

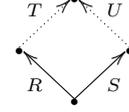
To bring (8) into this form, just compose the right hand side with I^\smile .

This observation is the key to success if one also remembers modal correspondence theory (see e.g. [12]), according to which the above formula is equivalent to each of the modal Geach formulas

$$\langle R \rangle [T] P \Rightarrow [S] \langle U \rangle P , \quad \langle S \rangle [U] P \Rightarrow [R] \langle T \rangle P . \quad (14)$$

Now we are in good shape, since the modal operators $\langle X \rangle$ and $[X]$ can be defined as predicate transformers in Kleene algebra, cf. [11].

We shall use many instances of these formulas. Since one can easily confuse the rôles of the relations involved in the modal formulation, we shall illustrate these formulas by the type of diagram shown on the right hand side. When read as a confluence-type diagram, the solid arrows and their end points symbolize given elements and relations between them, whereas the dotted ones stand for a quantifier stipulating existence of a further element and appropriate relations to given elements. If one of the arrows is an identity, the diagram shrinks to a triangle.



5 Abstracting to Modal Kleene Algebra

We now abstract from the relational setting to the more general formalism of modal Kleene algebra.

First, a *Kleene algebra* (KA) [8] is a structure $(K, +, \cdot, *, 0, 1)$ such that $(K, +, \cdot, 0, 1)$ is an idempotent semiring and $*$ is a unary operation axiomatized by the (Horn) identities

$$1 + aa^* \leq a^* , \quad (*-1) \quad b + ac \leq c \Rightarrow a^*b \leq c , \quad (*-3)$$

$$1 + a^*a \leq a^* , \quad (*-2) \quad b + ca \leq c \Rightarrow ba^* \leq c , \quad (*-4)$$

for all $a, b, c \in K$. Here, \leq denotes the natural ordering on K defined by $a \leq b$ iff $a + b = b$. An important property that follows from these axioms is the semi-commutation law

$$ab \leq ca \Rightarrow ab^* \leq c^*a . \quad (15)$$

A special case of this establishes (4). A KA is **-continuous* if for all a, b, c we have $ab^*c = \sum_{i \in \mathbb{N}} ab^i c$.

A *Kleene algebra with tests* (KAT) [9] is a KA with a Boolean subalgebra $\text{test}(K) \subseteq K$ of *tests* or *predicates*, in which 1 is the greatest element, 0 is the least element and \cdot coincides with the meet operation. In a KAT K one again defines *while* p *do* a $\stackrel{\text{def}}{=} (pa)^* \neg p$ for $p \in \text{test}(K)$ and $a \in K$.

Finally, a *Kleene algebra with domain* (KAD) [3] is a KAT with an additional operation $\ulcorner : K \rightarrow \text{test}(K)$ such that for all $a, b \in K$ and $p, q \in \text{test}(K)$,

$$a \leq \ulcorner a a , \quad \ulcorner(pa) \leq p , \quad \ulcorner(ab) = \ulcorner(a \ulcorner b) . \quad (\text{dom})$$

Let us explain these axioms. As in the algebra of relations, multiplication with a test from the left or right means domain or range restriction, resp. Now first, since $\ulcorner a \leq 1$ by $\ulcorner a \in \text{test}(K)$, monotonicity of multiplication shows that the first

axiom can be strengthened to an equality expressing that restriction to the full domain is no restriction at all. The second axiom means that after restriction the remaining domain must satisfy the restricting test. The third axiom serves to make the modal operators below well-behaved w.r.t. composition. An important consequence of the axioms is that $\lceil _$ preserves arbitrary existing suprema [11].

Examples of KADs are the algebra of concrete relations, where $\lceil _$ coincides with the operation defined in (10), the algebra of path sets in a directed graph (see e.g. [10]) and Kleene's original algebra of formal languages.

In a KAD, the (forward) modal operators *diamond* and *box* are defined by

$$\langle a \rangle p \stackrel{\text{def}}{=} \lceil (ap) \text{ , } \quad [a]p \stackrel{\text{def}}{=} \neg \langle a \rangle \neg p \text{ ,}$$

resulting in a *modal Kleene algebra*. This is adequate, since it makes the diamond coincide with the inverse image operator. The third axiom in (dom) implies

$$\langle ab \rangle p = \langle a \rangle \langle b \rangle p \text{ , } \quad [ab]p = [a][b]p \text{ .} \quad (16)$$

The modal operators for a test q are given by

$$\langle q \rangle p = qp \text{ , } \quad [q]p = q \rightarrow p \stackrel{\text{def}}{=} \neg q + p \text{ .} \quad (17)$$

For the above-mentioned connection between confluence-type formulas and the Geach formula we introduce the following notion. A KAD *with converse* is a KAD K with an additional operation $\smile : K \rightarrow K$ that is an involution, distributes over $+$, is the identity on tests and is contravariant over \cdot , i.e., satisfies $(ab)^\smile = b^\smile a^\smile$. One can show (see again [3]) that over a KAD with converse the first two axioms of (dom) imply the Galois connection

$$\langle a^\smile \rangle p \leq q \Leftrightarrow p \leq [a]q \text{ .} \quad (18)$$

It follows that in a KAD with converse all predicate transformers $\lambda p. \langle a \rangle p$ are universally disjunctive, i.e., preserve all existing suprema (and that all predicate transformers $\lambda p. [a]p$ are universally anti-disjunctive, i.e., transform all existing suprema into corresponding infima). This generalizes to KADs that also provide a codomain operation, since there one can also define the backward modal operators and replace $\langle a^\smile \rangle$ by the backward diamond of a .

Moreover, using (17) and the shunting rule from Boolean algebra we get, even in KADs without converse, for tests p, q, r

$$p \leq [r]q \Leftrightarrow \langle r \rangle p \leq q \text{ .} \quad (19)$$

Finally, using the star induction axioms, one can show the following induction principle for the diamond operator (cf. [3]):

$$\langle a \rangle p + q \leq p \Rightarrow \langle a^* \rangle q \leq p \text{ .} \quad (20)$$

6 Properties of Modal Operators

Our previous relation-algebraic derivation can now be mimicked more abstractly at the level of predicate transformers over an arbitrary KAD. In particular, we do not need the carrier of the algebra to be a complete Boolean algebra as in the case of relation algebra. Although the particular case of greedy algorithms can be treated in the weaker system of power allegories (see [1,2]), we can even avoid residuals and converse.

Assume a KAT $(K, \text{test}(K), +, \cdot, 0, 1, *)$. By a *predicate transformer* we mean a function $f : \text{test}(K) \rightarrow \text{test}(K)$. It is *disjunctive* if $f(p+q) = f(p) + f(q)$ and *conjunctive* if $f(p \cdot q) = f(p) \cdot f(q)$.

Let P be the set of *all* predicate transformers and D the set of strict and disjunctive (and hence monotonic) ones. Under the pointwise ordering

$$f \leq g \stackrel{\text{def}}{\Leftrightarrow} \forall p. f(p) \leq g(p)$$

P forms a lattice in which the supremum $f \oplus g$ and infimum $f \odot g$ of f and g are the pointwise liftings \oplus and \odot of $+$ and \cdot , resp.

We now concentrate on the modal predicate transformers $\langle _ \rangle$ and $[_]$ from KAD. For the rest of the paper we will work as much as possible at the point-free level. To smoothen the notation, we will denote composition of predicate transformers by mere juxtaposition.

In a KAD with converse the test-level Galois connection (18) implies the predicate-transformer-level cancellation laws and Galois connection

$$\langle a^\smile \rangle [a] \leq \langle 1 \rangle \leq [a] \langle a^\smile \rangle, \quad (21)$$

$$\langle a^\smile \rangle f \leq g \Leftrightarrow f \leq [a]g. \quad (22)$$

We will now give an abstract proof of equivalence of the Geach formula (14) and the confluence property (13). We do this in KADs that are *extensional* (or *separable* as they are called in dynamic algebra), i.e., satisfy

$$a \leq b \Leftrightarrow \langle a \rangle \leq \langle b \rangle. \quad (23)$$

Note that only the direction from right to left must be required, the other one holds in KAD by monotonicity.

Theorem 6.1 *In an extensional KAD with converse,*

$$a^\smile b \leq cd^\smile \Leftrightarrow \langle b \rangle [d] \leq [a] \langle c \rangle.$$

Proof. (\Rightarrow) We calculate

$$\begin{aligned} a^\smile b \leq cd^\smile &\Leftrightarrow \langle a^\smile b \rangle \leq \langle cd^\smile \rangle \Leftrightarrow \langle a^\smile \rangle \langle b \rangle \leq \langle c \rangle \langle d^\smile \rangle \Leftrightarrow \\ &\langle b \rangle \leq [a] \langle c \rangle \langle d^\smile \rangle \Rightarrow \langle b \rangle [d] \leq [a] \langle c \rangle \langle d^\smile \rangle [d] \Rightarrow \langle b \rangle [d] \leq [a] \langle c \rangle. \end{aligned}$$

The first step uses (23). The second step employs (16). The third step applies (18). The fourth step uses monotonicity. The fifth step follows by (21).

(\Leftarrow) Let $\langle b \rangle [d] \leq [a] \langle c \rangle$. Then $\langle b \rangle \leq \langle b \rangle [d] \langle d^\smile \rangle \leq [a] \langle c \rangle \langle d^\smile \rangle$. Now the proof continues like for (\Rightarrow), read upside down. \square

The Geach formula has the desired interpretation even in KAs without converse. To understand this, think of KA elements as describing sets of computation paths leading from initial to final states. The modal operators forget the intermediate states and hence induce a relation-like view of the KA elements, and so the Geach formula carries over its meaning to the more general setting.

It should also be noted that the dual formula $ab^\smile \leq c^\smile d$ cannot be treated in the same way; it requires a KA with codomain rather than domain and the corresponding backward modal operators. So greatest flexibility is achieved in KAs with both domain and codomain.

A special case of the Geach formula deals with the domain operator. Its relational definition (10) implies $\ulcorner R \subseteq R ; R^\smile$, admitting a certain relaxation of domain constraints. Using the Geach formula, this translates into the modal Kleene formula $\langle \ulcorner a \rangle [a] \leq \langle a \rangle$, which holds in *all* KADs, not only in extensional KADs with converse. An easily verified consequence is

$$ab \leq b \Rightarrow \langle \ulcorner a \rangle [b] \leq \langle a \rangle [b] . \quad (24)$$

7 Looping for Optimality in Kleene Algebra

We can now replay our previous derivation in the more abstract setting of KAD. Specifications and implementations are now simply elements of a KAD K with complete test algebra $\text{test}(K)$.

A difference to the relational setting is that we cannot carry over GIMP directly, since in general KADs residuals need not exist. But for our derivation there is no need to internalize the concept of greatest improvement; rather we use a characterizing predicate in which the right hand side of (1) is replaced by the corresponding modal formula and c now plays the rôle of C :

$$\text{IMP}(x, t, c) \stackrel{\text{def}}{\iff} \langle x \rangle \leq [t] \langle c \rangle . \quad (25)$$

Now we need to find a sufficient criterion for $\text{IMP}(\text{while } p \text{ do } s, t, c)$, which spells out to $\langle (ps)^* \neg p \rangle \leq [t] \langle c \rangle$.

As in Section 2, we abstract and want to achieve, for $v \in K$ and $q \in \text{test}(K)$, that $\langle v^* q \rangle \leq [t] \langle c \rangle$, which by (20) is implied by $\langle v \rangle [t] \langle c \rangle p + qp \leq [t] \langle c \rangle p$, in equivalent point-free notation

$$\langle q \rangle \leq [t] \langle c \rangle \wedge \langle v \rangle [t] \langle c \rangle \leq [t] \langle c \rangle . \quad (26)$$

The second conjunct, in turn, follows from

$$\langle v \rangle [t] \leq [t] \langle c \rangle , \quad (27)$$

provided $\langle c \rangle \langle c \rangle \leq \langle c \rangle$. In this case we call c *weakly transitive*. This is a much weaker requirement than transitivity $cc \leq c$. To see this, view the Kleene elements again as sets of computation paths. If c consists of paths with exactly two states each (i.e., is isomorphic to a binary relation on states) then cc consists of

paths with exactly three states, and so $cc \leq c$ holds only if $cc = 0$. But c is still weakly transitivity if it is transitive considered as a binary relation.

The full KA specification reads $\text{KAOPT}(x, t, c) \stackrel{\text{def}}{=} x \leq t \wedge \text{IMP}(x, t, c)$, analogously to Section 2 and yields the proof obligation, abstracted from (2),

$$\text{while } p \text{ do } s \leq t . \quad (28)$$

Assume now, as in (9), that $t = \text{exhaust } e = e^* ; \neg \ulcorner e$. The same derivation as in Section 3 yields that the following abstraction of (11) is sufficient for (28):

$$ps \leq e \wedge \ulcorner (ps) \geq \ulcorner e . \quad (29)$$

Next we note that (27) and the first conjunct of (26) spell out to

$$\langle ps \rangle [t] \leq [t] \langle c \rangle , \quad (30) \quad \langle \neg p \rangle \leq [t] \langle c \rangle . \quad (31)$$

Again, (29) implies (31) if $\langle \neg \ulcorner e \rangle \leq \langle c \rangle$. Then we call c *weakly reflexive* on $\neg \ulcorner e$.

Summing up, we have the following KA variant of Theorem 3.1:

Theorem 7.1 *Suppose that c is weakly reflexive on $\neg \ulcorner e$ and weakly transitive and that $t = \text{exhaust } e$. Then $(29) \wedge (30) \Rightarrow \text{KAOPT}(\text{while } \ulcorner e \text{ do } s, t, c)$.*

In [1,2] it is pointed out that under certain circumstances the above proof obligations will not succeed for the full c . Rather one has to add “context information”, restricting c by intersecting it with an element of the form $d \tilde{d}$. E.g., the relation *perm*, holding between lists that are permutations of each other, can be expressed as *bagify bagify*[~], where *bagify* transforms lists into bags. Although an element of the shape $d \tilde{d}$ can be mimicked by a predicate transformer when the backward modal operators are available, we offer an alternative here. Using the Geach formula again, we calculate, in an extensional KAD,

$$x \leq d \tilde{d} \Leftrightarrow 1 \tilde{x} \leq d \tilde{d} \Leftrightarrow \langle x \rangle [d] \leq [1] \langle d \rangle \Leftrightarrow \langle x \rangle [d] \leq \langle d \rangle .$$

So the context restriction yields another conjunct in the proof obligations.

8 Classifying Greedy Algorithms

We now demonstrate that modal Kleene algebra is indeed convenient for further applications. We demonstrate this in an abstract reconstruction of Curtis’s classification of Greedy algorithms in [2] to which we also refer the reader for concrete examples of the various types of algorithms. The modal operators again lead to considerably more concise proofs than the original relational ones.

Throughout this section we assume the following. First, c and l are pre-orders that model global and local comparison, respectively. Second, $t = \text{exhaust } e$ is the construction operation that completes initial approximative solutions to terminal ones using elementary steps e . Third, $g \leq e$ is supposed to be a greedy step that satisfies, analogously to (11) and (12),

$$\ulcorner g = \ulcorner e , \quad (32)$$

$$\text{IMP}(g, e, l), \text{ i.e., } \langle g \rangle \leq [e] \langle l \rangle \quad (\Leftrightarrow e \tilde{g} \leq l) . \quad (33)$$

In the following theorems we will always list the conditions both in modal notation as obtained by the Geach formula from Theorem 6.1 and in the one of KADs with converse and illustrate them by diagrams.

Immediately from Theorem 7.1 we obtain the following description of the first class of greedy algorithms:

Theorem 8.1 (Best-Global) $\text{KAOPT}(\text{exhaust } g, t, c)$ follows from

$$\begin{array}{c}
 \begin{array}{ccc}
 & c & \\
 & \nearrow & \nwarrow \\
 t & & t \\
 \nwarrow & & \nearrow \\
 & g & \\
 & \searrow & \swarrow \\
 & &
 \end{array}
 \end{array}
 \quad \langle g \rangle [t] \leq [t] \langle c \rangle \quad (\Leftrightarrow t^\smile g \leq ct^\smile) . \quad (34)$$

The next class is characterized by

Theorem 8.2 (Better-Global) $\text{KAOPT}(\text{exhaust } g, t, c)$ follows from

$$\begin{array}{c}
 \begin{array}{ccc}
 & c & \\
 & \nearrow & \nwarrow \\
 t & & t \\
 \nwarrow & & \nearrow \\
 & l & \\
 & \searrow & \swarrow \\
 & &
 \end{array}
 \end{array}
 \quad \langle l \rangle [t] \leq [t] \langle c \rangle \quad (\Leftrightarrow t^\smile l \leq ct^\smile) . \quad (35)$$

This condition says that for any pair of local choices the locally better one has a completion at least as good as any completion of the locally worse one.

Proof. We show that the assumptions imply condition (34) of Theorem 8.1.

$$\langle g \rangle [t] \leq [e] \langle l \rangle [t] \leq [e] [t] \langle c \rangle = [\ulcorner e] [t] \langle c \rangle .$$

The first step uses (33), the second one (35) and the third one the definition of $t = \text{exhaust } e$. But by (19) and (32) this is equivalent to the claim. \square

The third class of greedy algorithms has a more elaborate set of preconditions.

Theorem 8.3 (Best-Local) If we assume *-continuity, $\text{KAOPT}(\text{exhaust } g, t, c)$ follows from

$$\begin{array}{c}
 \begin{array}{ccc}
 & l & \\
 & \nearrow & \nwarrow \\
 e^n & & g^n \\
 \nwarrow & & \nearrow \\
 & &
 \end{array}
 \end{array}
 \quad \forall n \in \mathbb{N} : \langle g^n \rangle \leq [e^n] \langle l \rangle \quad (\Leftrightarrow \forall n \in \mathbb{N} : (e^n)^\smile g^n \leq l) , \quad (36)$$

$$\begin{array}{c}
 \begin{array}{ccc}
 & l & \\
 & \nearrow & \nwarrow \\
 e & & l \neg e \\
 \nwarrow & & \nearrow \\
 & &
 \end{array}
 \end{array}
 \quad \langle l \neg e \rangle \leq [e] \langle l \rangle \quad (\Leftrightarrow e^\smile l \neg e \leq l) , \quad (37)$$

$$\begin{array}{c}
 \begin{array}{ccc}
 & c & \\
 & \nearrow & \nwarrow \\
 & & t \\
 \nwarrow & & \nearrow \\
 & \neg e l & \\
 & \searrow & \swarrow \\
 & &
 \end{array}
 \end{array}
 \quad \langle l \rangle [t] \leq [\neg e] \langle c \rangle \quad (\Leftrightarrow \neg e l \leq ct^\smile) . \quad (38)$$

Here the local choice is made depending on the history of choices before. The first of these conditions says that each step produces an approximation to the final optimum that is optimal among the approximations that can be obtained with the same number of elementary steps. The other two conditions state that once the sequence of greedy steps finishes, completions of other approximations cannot improve the result any more.

Proof. First we note that, by an easy induction using idempotence of tests, in particular $\langle \neg e \rangle = \langle \neg e \rangle \langle \neg e \rangle$, condition (37) generalizes to

$$n \geq 0 \Rightarrow \langle l \rangle \langle \neg e \rangle \leq [e^n] \langle l \rangle . \quad (39)$$

The proof proper is performed by showing that the assumptions imply condition (34) of Theorem 8.1, i.e., $\langle g \rangle [t] \leq [t] \langle c \rangle$.

Using *-continuity we get $[\sum_{n \in \mathbb{N}} e^n \neg e] = \bigcirc_{n \in \mathbb{N}} [e^n \neg e]$, so that the claim reduces to

$$\forall n \in \mathbb{N} : \langle g \rangle [t] \leq [e^n] [\neg e] \langle c \rangle .$$

For $n = 0$, we use idempotence of tests and (19) to see that

$$\langle g \rangle [t] \leq [\neg e] \langle c \rangle \Leftrightarrow \langle \neg e \rangle \langle g \rangle [t] \leq [\neg e] \langle c \rangle .$$

Now we calculate, using (32),

$$\langle \neg e \rangle \langle g \rangle [t] = \langle \neg g \rangle \langle g \rangle [t] = \langle \neg g g \rangle [t] = \langle 0 \rangle [t] = \langle 0 \rangle ,$$

and the claim is shown.

For fixed $n > 0$ we split the greedy step g into the part $g^\Gamma(g^{n-1})$ that admits at least $n-1$ further greedy steps, and its relative complement $g \neg^\Gamma(g^{n-1})$, and show separately $\langle g^\Gamma(g^{n-1}) \rangle [t] \leq r$ and $\langle g \neg^\Gamma(g^{n-1}) \rangle [t] \leq r$, where $r \stackrel{\text{def}}{=} [e^n] [\neg e] \langle c \rangle$. For the first part we calculate

$$\langle g \rangle \langle g^\Gamma(g^{n-1}) \rangle [t] \leq \langle g \rangle \langle g^{n-1} \rangle [t] \leq [e^n] \langle l \rangle [t] \leq [e^n] [\neg e] \langle c \rangle .$$

The first step uses $g^{n-1} t \leq t$ and (24). The second step joins powers and uses (36). The final step employs (38).

For the second part we want to use (24) again and so have to replace $\neg^\Gamma(g^{n-1})$ by a positive domain expression. We calculate, for arbitrary i ,

$$\neg^\Gamma(g^i) = \neg^\Gamma(g^i \neg^\Gamma g) + \neg^\Gamma(g^i \neg^\Gamma g) = \neg^\Gamma(g^i \neg^\Gamma g) \neg^\Gamma(g^i \neg^\Gamma g) = \neg^\Gamma(g^{i+1}) \neg^\Gamma(g^i \neg^\Gamma g) .$$

Using only the \geq half of this equality and shunting we obtain

$$\neg^\Gamma(g^{i+1}) \leq \neg^\Gamma(g^i) + \neg^\Gamma(g^i \neg^\Gamma g) ,$$

and an easy induction shows $\neg^\Gamma(g^{n-1}) \leq \sum_{i < n-1} \neg^\Gamma(g^i \neg^\Gamma g)$. By disjointivity of $\langle \cdot \rangle$ our claim is thus established if $\langle g \rangle \langle \neg^\Gamma(g^i \neg^\Gamma g) \rangle [t] \leq r$ for all $i < n-1$. We calculate

$$\begin{aligned} \langle g \rangle \langle \neg^\Gamma(g^i \neg^\Gamma g) \rangle [t] &\leq \langle g \rangle \langle g^i \neg^\Gamma g \rangle [t] = \langle g^{i+1} \rangle \langle \neg e \rangle [t] \leq \\ &[e^{i+1}] \langle l \rangle \langle \neg e \rangle [t] \leq [e^{i+1}] [e^{n-i-1}] \langle l \rangle [t] \leq [e^n] [\neg e] \langle c \rangle . \end{aligned}$$

The first step uses $g^i \neg^\Gamma g t \leq t$ and (24). The second step joins powers and uses (32). The third step employs condition (36). The fourth step uses (39) and $n > i+1$. The final step joins powers and employs condition (38). \square

The final class of algorithms is given by

Theorem 8.4 (Better-Local) Under $*$ -continuity, KAOPT(exhaust g, t, c) follows from

$$\begin{array}{c}
 \begin{array}{ccc}
 & l & e \\
 & \swarrow & \searrow \\
 e & & l \ulcorner e \\
 & \searrow & \swarrow \\
 & &
 \end{array}
 \end{array}
 \quad \langle l \rangle \langle \ulcorner e \rangle [e] \leq [e] \langle l \rangle \quad (\Leftrightarrow e \smile l \ulcorner e \leq l e \smile), \quad (40)$$

$$\begin{array}{c}
 \begin{array}{ccc}
 & l & \\
 & \swarrow & \searrow \\
 e & & l \lrcorner e \\
 & \searrow & \swarrow \\
 & &
 \end{array}
 \end{array}
 \quad \langle l \rangle \langle \lrcorner e \rangle \leq [e] \langle l \rangle \quad (\Leftrightarrow e \smile l \lrcorner e \leq l), \quad (41)$$

$$\begin{array}{c}
 \begin{array}{ccc}
 & c & t \\
 & \swarrow & \searrow \\
 & & \lrcorner e l \\
 & \searrow & \swarrow \\
 & &
 \end{array}
 \end{array}
 \quad \langle l \rangle [t] \leq [\lrcorner e] \langle c \rangle \quad (\Leftrightarrow \lrcorner e l \leq c t \smile). \quad (42)$$

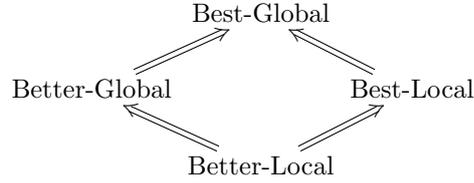
This essentially says that for any two local choices and any one-step extension of the locally worse one there is a locally better one-step extension of the locally better one.

Proof. We show that condition (35) of Theorem 8.2 is satisfied. First, using antitonicity of $[_]$ and distributivity, we obtain

$$\begin{aligned}
 (40) \wedge (41) &\Leftrightarrow \langle l \rangle (\langle \ulcorner e \rangle [e] \oplus \langle \lrcorner e \rangle [1]) \leq [e] \langle l \rangle \Rightarrow \\
 &\langle l \rangle (\langle \ulcorner e \rangle [e + 1] \oplus \langle \lrcorner e \rangle [e + 1]) \leq [e] \langle l \rangle \Leftrightarrow \langle l \rangle [e + 1] \leq [e] \langle l \rangle.
 \end{aligned}$$

Now dualization of (15) together with the equivalence in (14) allows us to infer $\langle l \rangle [e^*] = \langle l \rangle [(e + 1)^*] \leq [e^*] \langle l \rangle$, from which by condition (42) we get $\langle l \rangle [t] = \langle l \rangle [e^*] [t] \leq [e^*] \langle l \rangle [t] \leq [e^*] [\lrcorner e] \langle c \rangle = [t] \langle c \rangle$. \square

Curtis's classification is completed by showing the following relationship between the algorithm classes:



Except for the relation between Better-Local and Best-Local this was established by the proofs of the previous theorems.

Theorem 8.5 *The Better-Local conditions imply the Best-Local conditions.*

Proof. It suffices to show (36). This is done by induction on n . For $n = 0$ the claim follows from $1 \leq l$. For the induction step we calculate

$$\begin{aligned}
 \langle g^{n+1} \rangle &\leq [e^n] \langle l \rangle \langle g \rangle \leq [e^n] \langle l \rangle \langle \ulcorner g \rangle \langle g \rangle \leq \\
 &[e^n] \langle l \rangle \langle \ulcorner g \rangle [e] \langle l \rangle \leq [e^n] [e] \langle l \rangle \langle l \rangle \leq [e^{n+1}] \langle l \rangle.
 \end{aligned}$$

The first step splits a power and uses the induction hypothesis. The second step uses a domain law. The third step employs (33). The fourth step uses (32) and (40). The last step joins powers and uses transitivity of l . \square

9 Conclusion

We have shown that a concise algebraic derivation of a general greedy-like algorithm can be obtained in the framework of Kleene algebra. The more pristine framework avoids detours through residuals and leads to a simpler correctness proof than in [2,4].

The treatment has exhibited an interesting relation with semi-commutation properties as known from rewriting and allegories [5]. The connection to KA has already been explored in [13].

Omitting converse has led us into the interesting and very well-behaved algebra of predicate transformers. In it we can prove properties such as $\langle a^* \rangle = \langle a \rangle^*$ that cannot even be expressed in dynamic logic. We are therefore convinced that this algebra will have many further applications; for an example see [11].

Acknowledgement We are grateful to S. Curtis for an enlightening discussion on greedy algorithms. Helpful remarks were provided by R. Backhouse, H. Bherer, J. Desharnais, T. Ehm, O. de Moor and M. Winter.

References

1. R.S. Bird, O. de Moor: Algebra of programming. Prentice Hall 1997
2. S.A. Curtis: A relational approach to optimization problems. D.Phil. Thesis. Technical Monograph PRG-122, Oxford University Computing Laboratory 1996
3. J. Desharnais, B. Möller, and G. Struth: Kleene algebra with domain. Technical Report 2003-07, Universität Augsburg, Institut für Informatik, 2003
4. J.E. Durán: Transformational derivation of greedy network algorithms from descriptive specifications. In: E.A. Boiten, B. Möller (eds.): Mathematics of program construction. Lecture Notes in Computer Science **2386**. Springer 2002, 40–67
5. P. Freyd, A. Scedrov: Categories, allegories. North-Holland 1990
6. P. Helman, B.M.E. Moret, H.D. Shapiro: An exact characterization of greedy structures. SIAM Journal on Discrete Mathematics **6**, 274-283 (1993)
7. B. Korte, L. Lovász, R. Schrader: Greedoids. Heidelberg: Springer 1991
8. D. Kozen: A completeness theorem for Kleene algebras and the algebra of regular events. Information and Computation **110**, 366–390 (1994)
9. D. Kozen: Kleene algebra with tests. Transactions on Programming Languages and Systems **19**, 427–443 (1997)
10. B. Möller: Derivation of graph and pointer algorithms. In: B. Möller, H.A. Partsch, S.A. Schuman (eds.): Formal program development. Lecture Notes in computer science **755**. Springer 1993, 123–160
11. B. Möller and G. Struth: Modal Kleene algebra and partial correctness. Technical Report 2003-08, Universität Augsburg, Institut für Informatik, 2003
12. S. Popkorn: First steps in modal logic. Cambridge University Press 1994
13. G. Struth: Calculating Church-Rosser proofs in Kleene algebra. In: H.C.M. de Swart (ed.): Relational Methods in Computer Science, 6th International Conference. Lecture Notes in Computer Science **2561**. Springer 2002, 276–290