Weakest preconditions, predicate transformers, and Hoare logic were relatively early developments. In this lecture we survey some more recent developments in this area:

- Dynamic logic (DL)
- Kleene algebra (KA) and Kleene algebra with tests (KAT)

1 Kleene Algebra (KA)

Kleene algebra (KA) is an algebraic system that captures axiomatically the properties of several natural classes of structures that arise in logic and computer science. It is named for Stephen Cole Kleene (1909–1994), who among his many other achievements invented finite automata and regular expressions, structures of fundamental importance in computer science. Kleene algebra is the algebraic theory of these objects, although it has many other natural and useful interpretations. Kleene algebras arise in various guises in many contexts: relational algebra, semantics and logics of programs, automata and formal language theory, and the design and analysis of algorithms.

Formally, a *Kleene algebra* is an algebraic structure $(K, +, \cdot, ^*, 0, 1)$ consisting of a set K with binary operations $+$ (choice) and $\cdot$ (sequential composition), unary operation * (iteration), and constants 1 (skip) and 0 (fail) satisfying the axioms listed below. We usually omit the operator $\cdot$, writing pq for $p \cdot q$. The order of precedence of the operators is $^* > \cdot > +$; thus $p + qr^*$ should be parsed as $p + (q(r^*))$.

The axioms of KA are

$$\begin{array}{rr} p + (q + r) = (p + q) + r & p(qr) = (pq)r \\ p + q = q + p & 1p = p \\ p + 0 = p & p1 = p \\ p + p = p & \end{array}$$

$$\begin{array}{rr} p(q + r) = pq + pr & 0p = 0 \\ (p + q)r = pr + qr & p0 = 0 \end{array}$$

$$\begin{array}{rr} 1 + pp^* = p^* & q + pr \le r \Rightarrow p^*q \le r \\ 1 + p^*p = p^* & q + rp \le r \Rightarrow qp^* \le r \end{array}$$

where $\le$ refers to the natural partial order on K:

$$p \le q \stackrel{\Delta}{\Leftrightarrow} p + q = q.$$

Instead of the last two equational implications, we might take the equivalent axioms

$$pr \le r \;\Rightarrow\; p^*r \le r \qquad\qquad rp \le r \;\Rightarrow\; rp^* \le r.$$

The axioms not involving *, taken together, say that the structure $(K, +, \cdot, 0, 1)$ is an *idempotent semiring*. The term *idempotent* refers to the axiom $p + p = p$, which implies that $\le$ is a partial order. The remaining axioms involving * say essentially that * behaves like the Kleene asterate operator of formal language theory or the reflexive transitive closure operator of binary relations.

The standard model is the family of *regular sets* over a finite alphabet Σ. Let Σ^* be the set of all finite-length strings over the alphabet Σ, including the null string ε.

The KA operations are defined on subsets of Σ^* as follows. The semiring operations are

$$\begin{aligned} A + B &\triangleq A \cup B & 0 &\triangleq \varnothing \\ A \cdot B &\triangleq \{xy \mid x \in A,\ y \in B\} & 1 &\triangleq \{\varepsilon\}. \end{aligned}$$

The * operation on sets of strings is known as *Kleene asterate*:

$$A^* \triangleq \bigcup_{n \geq 0} A^n = \{x_1 \cdots x_n \mid n \geq 0 \text{ and } x_i \in A,\ 1 \leq i \leq n\},$$

where $A^0 \triangleq \{\varepsilon\}$ and $A^{n+1} \triangleq AA^n$. The powerset of Σ^* forms a KA under these definitions, with $\leq$ the subset relation. The *regular sets* are the smallest subalgebra containing $\{a\}$ for $a \in \Sigma$.

Another KA, more relevant for programming language semantics, is the family of binary relations on a set X, that is, subsets of $X \times X$, with operations

$$\begin{aligned} R + S &\triangleq R \cup S & 0 &\triangleq \varnothing \\ R \cdot S &\triangleq R\,;S = \{(u,w) \mid \exists v\ (u,v) \in R \wedge (v,w) \in S\} & 1 &\triangleq \{(u,u) \mid u \in X\} \\ R^* &\triangleq \bigcup_{n \geq 0} R^n = \text{reflexive transitive closure of } R, \end{aligned}$$

where $R^0 \triangleq \{(u,u) \mid u \in X\}$ and $R^{n+1} \triangleq R^n\,;R$.

One can show that the family of $n \times n$ matrices over a Kleene algebra is a Kleene algebra. Other more unusual interpretations include the (min,+) algebra used in shortest path algorithms and models consisting of convex polyhedra used in computational geometry.

The following are some typical consequences of the axioms:

$$(p^*q)^*p^* = (p+q)^* \qquad p(qp)^* = (pq)^*p \qquad (pq)^* = 1 + p(qp)^*q \qquad p^* = (pp)^*(1+p).$$

The axioms are complete for the equational theory of the regular set model. That is, all true identities between regular expressions, interpreted as regular sets of strings, are provable. The axioms are also complete for the equational theory of relational models.

2 Kleene Algebra with Tests (KAT)

A *Kleene algebra with tests* (KAT) is just a Kleene algebra with an embedded Boolean subalgebra. That is, it is a two-sorted structure $(K, B, +, \cdot, ^*, \bar{}, 0, 1)$ such that

- $(K, +, \cdot, ^*, 0, 1)$ is a Kleene algebra,
- $(B, +, \cdot, \bar{}, 0, 1)$ is a Boolean algebra, and
- $(B, +, \cdot, 0, 1)$ is a subalgebra of $(K, +, \cdot, 0, 1)$.

The Boolean complementation operator $\bar{}$ is defined only on B. Elements of B are called *tests*. The letters p, q, r, s denote arbitrary elements of K and a, b, c denote tests.

This deceptively simple definition actually carries a lot of information in a concise package. The operators $+, \cdot, 0, 1$ each play two roles: applied to arbitrary elements of K, they refer to nondeterministic choice, composition, fail, and skip, respectively; and applied to tests, they take on the additional meaning of Boolean

disjunction, conjunction, falsity, and truth, respectively. These two usages do not conflict—for example, sequential testing of b and c is the same as testing their conjunction—and their coexistence admits considerable economy of expression.

The programming constructs of the IMP language are encoded as follows:

$$p\,;q \triangleq pq \qquad \text{if } b \text{ then } p \text{ else } q \triangleq bp + \overline{b}q \qquad \text{while } b \text{ do } p \triangleq (bp)^*\overline{b}.$$

For applications in program verification, the standard interpretation would be a Kleene algebra of binary relations on a set and the Boolean algebra of subsets of the identity relation. There are also *trace models*, in which the Kleene elements are sets of traces (sequences of states) and the Boolean elements are sets of states (traces of length 0). As with KA, one can form the algebra of $n \times n$ matrices over a KAT (K, B); the Boolean elements of this structure are the diagonal matrices over B. There is also a language-theoretic model that plays the same role in KAT that the regular sets of strings over a finite alphabet play in KA, namely the family of regular sets of *guarded strings* over a finite alphabet Σ with guards from a set B. The equational theory of this structure is exactly the set of all equational consequences of the KAT axioms. Moreover, KAT is complete for the equational theory of relational models.

2.1 Encoding Hoare Logic

As we have seen, Hoare logic uses a specialized syntax involving *partial correctness assertions* (PCAs) of the form $\{b\}p\{c\}$ and a deductive apparatus consisting of a system of specialized rules of inference. The PCA $\{b\}p\{c\}$ is encoded in KAT in any one of the following three equivalent ways:

$$bp \le pc \qquad bp = bpc \qquad bp\overline{c} = 0.$$

Intuitively, the last of these says that the program p with preguard b and postguard $\overline{c}$ has no halting execution, and the second says that the test c after executing p with preguard b is always redundant.

Hoare-style inference rules of the form

$$\frac{\{b_1\}p_1\{c_1\},\ \ldots,\ \{b_n\}p_n\{c_n\}}{\{b\}p\{c\}} \tag{1}$$

become equational implications (universal Horn formulas)

$$b_1p_1 \le p_1c_1 \wedge \ldots \wedge b_np_n \le p_nc_n \ \Rightarrow\ bp \le pc$$

in KAT. The variables are implicitly universally quantified. The Hoare rules (see Lecture 17) then take the following form:

$$\begin{aligned}
bp \le pc \wedge cq \le qd &\ \Rightarrow\ bpq \le pqd && \text{(sequential composition)}\\
bcp \le pd \wedge \overline{b}cq \le qd &\ \Rightarrow\ c(bp + \overline{b}q) \le (bp + \overline{b}q)d && \text{(conditional)}\\
bcp \le pc &\ \Rightarrow\ c(bp)^*\overline{b} \le (bp)^*\overline{b}\,\overline{b}c && \text{(while)}\\
b' \le b \wedge bp \le pc \wedge c \le c' &\ \Rightarrow\ b'p \le pc' && \text{(weakening)}
\end{aligned}$$

Theorem The KAT encodings of the Hoare rules above are all theorems of KAT.

Proof sketch. We just do the while rule. By trivial simplifications it suffices to show $cbp \le bpc \Rightarrow c(bp)^* \le (bp)^*c$. Taking $q = bp$, it suffices to show $cq \le qc \Rightarrow cq^* \le q^*c$. Assume $cq \le qc$. By the axiom $c + xq \le x \Rightarrow cq^* \le x$ of Kleene algebra, we need only show $c + q^*cq \le q^*c$. But

$$\begin{aligned}
c + q^*cq &\le c + q^*qc && \text{by the assumption } cq \le qc \text{ and monotonicity}\\
&\le (1 + q^*q)c && \text{by distributivity}\\
&\le q^*c && \text{by the axiom } 1 + q^*q = q^*.
\end{aligned}$$

□

More importantly,

Theorem 18.1. KAT *is complete for all relationally valid Hoare rules of the form* (1)*; that is, all such rules that are true in all relational interpretations are theorems of* KAT*.*

This is trivially false for Hoare logic. For example, the rule

$$\frac{\{b\}\,\mathsf{if}\ d\ \mathsf{then}\ p\ \mathsf{else}\ p\{c\}}{\{b\}p\{c\}}$$

is relationally valid, but not provable in Hoare logic for the simple reason that all the Hoare rules go from shorter programs above the line to longer programs below the line. However, its translation

$$b(dp + \overline{d}p) \leq (dp + \overline{d}p)c \ \Rightarrow\ bp \leq pc$$

is easily provable in KAT.

3 Dynamic Logic (DL)

Dynamic logic (DL) is a logic of programs based on *modal logic*, the logic of *possibility* and *necessity*. Modal logic has formulas $\Diamond\varphi$, read "φ is possible" or just "diamond φ," and $\Box\varphi$, read "φ is necessary" or just "box φ." These operators are dual to each other in the sense that $\Diamond\varphi \Leftrightarrow \neg\Box\neg\varphi$; intuitively, φ is possibly true if and only if it is not necessarily false.

In DL, there is a separate modality for each program p, and we can write $[p]\varphi$ or $\langle p\rangle\varphi$. In propositional dynamic logic (PDL), programs are usually taken to be *regular programs* formed with the regular expression operators as in Kleene algebra. There is also a test operator ? that turns a proposition into a program.

Programs are interpreted as binary relations on a set of states, and propositions are interpreted as sets of states. The formula $[p]\varphi$ is semantically equivalent to the weakest liberal precondition $\mathsf{wlp}\ p\ \varphi$. Thus the Hoare partial correctness assertion $\{\varphi\}p\{\psi\}$ is equivalent to $\varphi \Rightarrow [p]\psi$.

The axioms and rules of inference of PDL are

1. axioms for propositional logic
2. $[p](\varphi \Rightarrow \psi) \quad\Rightarrow\quad ([p]\varphi \Rightarrow [p]\psi)$
3. $[p](\varphi \wedge \psi) \quad\Leftrightarrow\quad [p]\varphi \wedge [p]\psi$
4. $[p + q]\varphi \quad\Leftrightarrow\quad [p]\varphi \wedge [q]\varphi$
5. $[p\,;\,q]\varphi \quad\Leftrightarrow\quad [p][q]\varphi$
6. $[\psi?]\varphi \quad\Leftrightarrow\quad (\psi \Rightarrow \varphi)$
7. $\varphi \wedge [p][p^*]\varphi \quad\Leftrightarrow\quad [p^*]\varphi$
8. $\varphi \wedge [p^*](\varphi \Rightarrow [p]\varphi) \quad\Rightarrow\quad [p^*]\varphi$ (induction axiom)
9. $\dfrac{\varphi \quad \varphi \Rightarrow \psi}{\psi}$ (modus ponens)

10. $\dfrac{\varphi}{[p]\varphi}$ (modal generalization)

Axioms 1–3 and the rules of inference 9 and 10 are not particular to PDL, but come from modal logic.

Axiom 8 is called the PDL *induction axiom*. Intuitively, it says: "Suppose φ is true in the current state, and suppose that after any number of iterations of p, if φ is still true, then it will be true after one more iteration of p. Then φ will be true after any number of iterations of p." In other words, if φ is true initially, and if the truth of φ is preserved by the program p, then φ will be true after any number of iterations of p.

The axioms are complete over all relational interpretations.