# Kleene algebra notes

Spencer van Koevering (lightly edited by Dexter)

March 5, 2024

1. Today we introduce a different version of automata through coalgebra
2. Seminal paper will be posted on the website
   Rutten, Universal coalgebra
   Silva thesis
3. Today we introduce coalgebra
4. Terminology relationships between algebra and coalgebra
5. Algebra
   (a) signature
   (b) constructors
   (c) homomorphism
   (d) congruence (binary relation on elements that respects the operations)
   (e) congruence relations are kernels of homomorphisms
   (f) free (initial) algebras
   (g) there are unique homomorphisms out of free algebras
   (h) equations. Can be thought of as a syntactic impediment to injectivity as two elements of the free algebra can get mapped to the same element in the non-free algebra
   (i) terms built from constructors inductively
   (j) induction. Related to least fixed points. Build something then when you're done you have a set of things you've built
6. Coalgebra
   (a) signature
   (b) destructors
   (c) homomorphism
   (d) Bisimulation instead of congruence
   (e) Bisimulations are kernels of homomorphisms

(f) cofree (final) coalgebras

(g) there are unique homorphisms into cofree algebras

(h) coequations. Syntactic impediment to surjectivity of the unique homomorphism to the final coalgebra

(i) coterms. It's an infinite term

(j) coinduction. This is based on the greatest fixed point.

7. What is an algebra? Lets give some definitions

8. Suppose we had a category $C$ and a function $F : C \to C$. For our examples $C$ will be Set, the category of sets and functions.

9. $F$ is an endofuctor. its from a category to itself.

10. an $F$-algebra is a pair $(X, \alpha)$ where $X$ is an object of $C$ and $\alpha$ is a morphism $FX \to X$

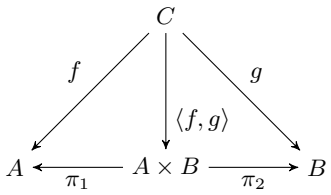
11. $\alpha$ is called the structure map of the algebra

12. Lets make a new category. $F$-Alg. Objects are $(X, \alpha)$ pairs, and the morphisms $h : (X, \alpha) \to (Y, \beta)$ is a morphism of $C$ such that the following diagram commutes. (diagram omitted)

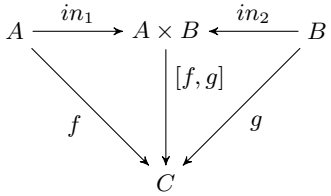
$$\begin{array}{ccc} FX & \xrightarrow{Fh} & FY \\ {\scriptstyle\alpha}\downarrow & & \downarrow{\scriptstyle\beta} \\ X & \xrightarrow[h]{} & Y \end{array}$$

13. Let's fit an example into this framework. Groups howabout. The signature is $(\cdot, ^{-1}, 1)$. A group is a set with some definition of these operations that satisfy closure, associativity, identity, and inverses. Each of these operators has an arity (2,1,0).

14. The functor $F$ in this case is $FX = X^2 + X + 1$. It takes a set $X$ and gives you a new set which is the sum of the cartesian product of $X$ with itself, $X$ is $X$ and 1 is the identity. The plus is coproduct: $X + Y =$ disjoint sum $= \{(1, x) \mid x \in X\} \cup \{(2, y) \mid y \in Y\}$ or $\{in_1(x) \mid x \in X\} \cup \{in_2, y) \mid y \in Y\}$. We can tell which set the elements came from. This is how we are defining these operations in Set. Other categories may have different products. In a general category $C$ a product is an operation $(A, B) \to A \times B$ such there exist $\pi_1 : A \times B \to A$ and $\pi_2 : A \times B \to B$ for any $C$ and morphisms $f : C \to A$ and $g : C \to D$ there is a unique mediating morphism $\langle f, g \rangle : C \to A \times B$ such that the following diagram commutes (omitted).




A coproduct $A + B$ is where we turn the arrows around. we have coprojections from $A, B$ to $A + B$ and then $f, g$ from $A, B$ to $C$ and $[f, g]$ from $A + B$ to $C$. The idea is that the tags let us distinguish which set each element came from.

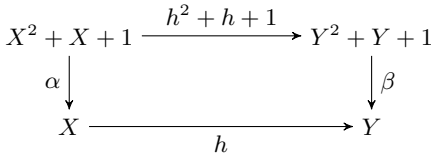



$[f, g]$ is just the match statement in OCaml:
$[f, g](x) =$ match $x$ with
$\mid in_1(a) \to f(a)$
$\mid in_2(b) \to g(b)$

15. $+$ and $\times$ are actually functors from $C^2 \to C$ so we can compose them.

16. Then abstractly a group is $(G, \alpha)$ with $\alpha : X^2 + X + 1 \to X$ but because of the tags, we can distinguish so its really $\alpha_2 : X^2 \to X$, $\alpha_1 : X \to X$, $\alpha_0 : 1 \to X$

17. We have a homomorphism from $(X, \alpha) \to (Y, \beta)$. We need to make the diagram comute (omitted)




but its $\alpha, \beta$ and $h$ and $h^2 + h + 1$. $h^2 : X^2 \to X$, $h^2(x, y) = (h(x), h(y))$, $h^0(1) = 1$. We make it commute and get that $h(ab) = h(a)h(b)$.

18. So what is a coalgebra? We turn the arrows around

19. An $F$ coalgebra over a category $C$ is a pair $(X, \gamma)$ such that $\gamma : X \to FX$ is a morphism of $C$. $F$-coalgebras are objects of a category $F$-coalg where the morphisms are maps $h : (X, \gamma) \to (Y, \delta)$ such that we flip the direction of the functor $F$ applications in the previous diagram.

20. The simplest example is streams over $\Sigma$. They are like infinite lists of characters. This is a coalgebra $(\Sigma^\omega, \text{hd}, \text{tl})$ (raised to the power omega means infinite). $\text{hd} : \Sigma^\omega \to \Sigma$ and $\text{tl} : \Sigma^\omega \to \Sigma^\omega$. Head gives the first element, tail gives the rest of the stream once the head is removed. $(\text{hd}, \text{tl}) : \Sigma^\omega \to \Sigma \times \Sigma^\omega$. This is a coalgebra for the functor $FX : \Sigma \times X$. In general $(X, \gamma) = \quad \gamma : X \to \Sigma \times X.$ or $(X, (obs, cont))$, $\gamma = (obs, cont)$. Were $obs$ is a single observation $X \to \Sigma$, and $cont$ is a continuation $X \to X$

21. Automata are like this. The observation is a check if you are in a final state, and the continuation is the rest of the computation.

22. This is a final coalgebra. For any other coalgebra there is a unique coalg morphism to $\Sigma^\omega$

23. Given an arbitrary coalgebra $(X, obs, cont)$, the morphism to the final coalgegra is $\theta : (X, obs, cont) \to (\Sigma^\omega, hd, tl)$ such that $\theta(x) = (obs(x), obs(cont(x)), obs(cont^2(x))...)$. This is a coalgebra morphism. To show this we have to show that the previous diagram commutes.

$$\begin{array}{ccc} X & \xrightarrow{\quad\theta\quad} & \Sigma^\omega \\ {\scriptstyle (obs,\, cont)}\downarrow & & \downarrow{\scriptstyle (hd,\, tl)} \\ \Sigma \times X & \xrightarrow[id_\Sigma \times \theta]{} & \Sigma \times \Sigma^\omega \end{array}$$

24. So what are automata over the alphabet $\Sigma$? They are coalgebras for the functor $FX = 2 \times X^\Sigma$. The coalgebras actually don't care about the start state, and $\Sigma$ is implicit. So only 3 operators are needed. We instead write it as $(Q, (\epsilon, \delta))$ where $(\epsilon, \delta) : Q \to 2 \times Q^\Sigma$ and $\epsilon : Q \to 2 = \{0, 1\}$. $\epsilon(f) = \begin{cases} 1 & q \in f \\ 0 & q \notin F \end{cases}$. And $\delta : Q \to (\Sigma \to Q)$ or $\delta = \Sigma \to (Q \to Q)$ or we could write $\delta_a : Q \to Q$. All are equivalent. This is a coalgebra for the functor.

25. There is a final coalgebra for this functor. What is it? What is the most information we can get about a state? The behavior of a state is the set of strings $\{x \mid x \text{ would be accepted if } s \text{ were the start state}\} = \{x \in \Sigma^* \mid \hat{\delta}(s, x) \in F\}$. So the final coalgebra is $(2^{\Sigma^*}, e, d)$ where $e : 2^{\Sigma^*} \to 2$ and $d_a : 2^{\Sigma^*} \to 2^{\Sigma^*}$. Then $\forall A \subseteq \Sigma^*\ e(\theta(s)) = \begin{cases} 1 & \epsilon \in \theta(s) \\ 0 & else \end{cases}$ and $d(A) = \{x \in \Sigma^* \mid ax \in A\}$. SO then we want to show that $d_a(\theta(s)) = \theta(\delta_a(s))$ making the diagram commute. We need to show that $\epsilon$ is preserved by $\theta$. $e(\theta(s)) = \epsilon(s)$ this is saying $\epsilon$ is accepted at $s$ iff $s$ is a final state, which is true! For the delta we have $\theta(\delta_a(s)) = \{x \mid \hat{\delta}(\delta(s, a), x) \in F\}$ and $d_a(\theta(s)) = \{x \mid \hat{\delta}(s, a, x) \in F\}$

26. last time we went from automata to algebras via the matrix construction

27. what this is saying that the Kleene Algebra functor is related to the automata functor.

28. Suppose $G$ were to the automata functor and $F$ is kleene algebra functor. To go int he opposite direction from last time we show that every kleene algebra is an automaton is done by showing the free algebra gives an automaton. The regular expressions also carry a coalgebra structure $(Exp\Sigma, E, D)$ called the Brzozowski derivative. The behavior of an expression is the regular set that it represents. So there will be a mapping $R_\Sigma : Exp\Sigma \to 2^{\Sigma^*}$ and this is the unique coalgebra morphism to the final coalgebra. We will present this next time.