Coalgebraic Semantics

Lecture 2: Bisimulation$^{\text{TM}}$—now with *twice* the simulation!

Hakan Dingenc
hakan@u.northwestern.edu

Pedro Amorim
pamorim@cs.cornell.edu

Siva Somayyajula
ssomayya@cs.cmu.edu

June 17, 2019

Fix an alphabet A.

Definition 1 (Deterministic finite automaton (DFA))**.** ...over A is a tuple (S, o, t):

- A set of states S
- An output function $o : S \to \mathbf{2}$, where for a state s, o(s) = 1 iff s is final
- A transition function $t : S \to S^A$

Remark. It is equivalent to have a set of final states $F \subseteq S$ instead of an output function: let $o = \chi_F$[1]. Likewise, t can also have the type $S \times A \to S$ by uncurrying.

Remark. A^* is inductive but 2^{A^*} (classifiers of strings) is inherently coinductive. What a delicate interplay! This is because acceptors have some notion of non-wellfounded circularity.

Definition 2 (Acceptance)**.** A word w is *accepted* by a state $s \in S$ iff one of the following holds:

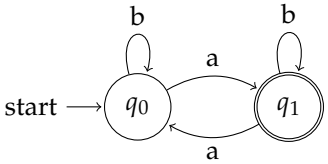
- $w = \epsilon$ and $o(s) = 1$
- $w = au$ and u is accepted by $t(s)(a)$

Remark. It is equivalent to define $\hat{t} : S \to S^{A^*}$ as:

$$\hat{t}(s)(\epsilon) \triangleq o(s)$$
$$\hat{t}(s)(au) \triangleq \hat{t}(t(s)(a))(u)$$

Then, w is accepted by s iff $o(\hat{t}(s)(w)) = 1$.

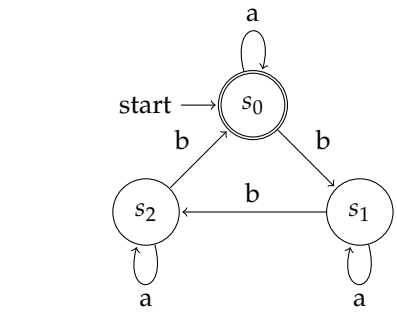
Definition 3 (Language)**.** Let the language of a DFA at state s be $\ell(s) = \{w \in A^* \mid o(\hat{t}(s)(w)) = 1\}$.

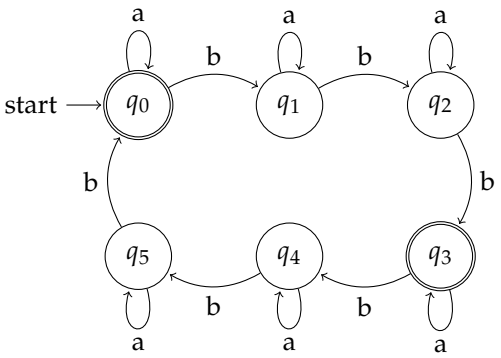


Example 1. Then, $\ell(q_0)$ is the set of strings with odd many a's.

Question 1. Can you write a DFA that accepts strings with a number of b's that's a multiple of three?

[1] $\chi_F(x) = 1$ if $x \in F$ and 0 otherwise





Question 2. How do I prove that two DFAs accept the same language? One way is to go by induction on the length of the word, but some automata (including the one above) have circular, infinite behavior much like streams do. Therefore, a natural choice is coinduction.

Definition 4. Given two automata (S, o_S, t_S) and (T, o_T, t_T) over the alphabet A, a relation $R \subseteq S \times T$ is a bisimulation iff for all $x\ R\ y$:

- $o_S(x) = o_T(y)$
- For $a \in A$, $t_S(x)(a)\ R\ t_T(y)(a)$

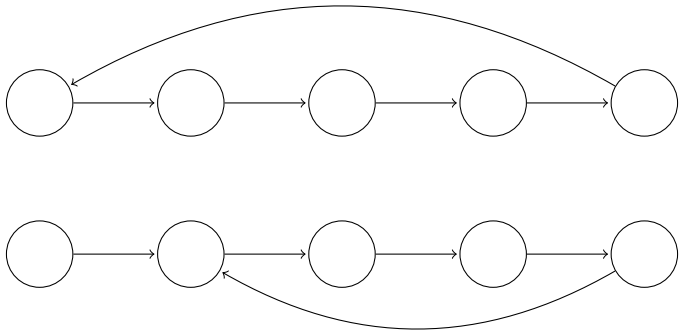
Theorem 1 (Coinduction principle for DFAs)**.** *If there exists a bisimulation R such that $x\ R\ y$, then $\ell(x) = \ell(y)$.*

Proof. The proof follows by induction on the length of the word. □

Example 2. We can prove that the languages accepted by the automata defined in question 1 are equal. It's easy prove it by coinduction using the relation $R = \{(s_0, q_0), (s_0, q_3), (s_1, q_1), (s_1, q_4), (s_2, q_2), (s_2, q_5)\}$ Intuitively, the bisimulation relates states who accept the same languages.

The process of bisimulation allows you to find a concrete counterexample if two automata aren't bisimilar. In fact, this can be applied to practical situations such as verifying a complex automaton model of a security protocol given a "monitor" automaton by checking that the two are bisimilar.

Example 3. There are pathologically large bisimulations—in the worst case, it takes $O\left(|S|^2\right)$ time to compute. How can we cull it? Consider *bisimulation up-to equivalence*, where you look at the reflexive transitive closure of a candidate bisimulation, and check if a candidate pair of states is related. If so, then you don't need to add it. This approach, due to Hopcroft and Karp, takes $O\left(|S| \log |S|\right)$ largely by storing the relation in the union-find data structure. The example below illustrates this blow-up.



Definition 5 (Bisimulation up-to equivalence on DFAs)**.** R is a bisimulation up-to equivalence if for $x\ R\ y$

- $o_S(x) = o_T(y)$
- For $a \in A$, $t_S(x)(a)\ R^*\ t_T(y)(a)$ where R^* is the reflexive transitive closure of R.

Theorem 2 (Coinduction principle*)**.** *If there is a bisimulation up to equivalence R such that $x\ R\ y$, then $\ell(x) = \ell(y)$.*

Proof. If we have a bisimulation up to equivalence R then R^* is a bisimulation. Therefore, we can apply the previous coinduction principle. □

In the table below, we summarize the duality between induction and coinduction.

induction	coinduction
proof by induction	bisimulation
least fixpoint	greatest fixpoint
initial algebra	final coalgebra

Example 4. A^* is an example of a least fixpoint and $A^* \cup A^\omega$ is an example of a greatest fixpoint.

Definition 6 (Functor)**.** A (**Set**-valued endo)functor F is a function that…

- Maps sets to sets
- Maps a function $f : A \to B$ to $F(f) : F(A) \to F(B)$

…such that…

- $F(f \circ g) = F(f) \circ F(g)$ for appropriately typed f and g
- $F(\mathrm{id}_A) = \mathrm{id}_{F(A)}$ where id_* is the identity function

Remark. The general definition of categories and functors is out of the scope of these notes; refer to Category Theory by Awodey for more details.

For $f : X \to Y$ and $g : Z \to W$, let $f \times g : X \times Y \to Z \times W$ be given by $(f \times g)(x, y) = (f(x), g(y))$. Dually, $f + g : X + Y \to Z + W$ is given by $(f + g)(x) = f(x)$ for $x \in X$ and $(f + g)(y) = g(y)$ for $y \in Y$. Viewing the Cartesian product $\times$ and disjoint sum $+$ as functions on sets, $\times$ and $+$ are thus functors. Furthermore, any set B induces an eponymous functor which sends every set to B and every function to id_B. The consequent "ring" of functors leads to a natural notion of *polynomial*.

Definition 7 (Polynomial functor)**.** …with "coefficients" x_i (which are sets viewed functors) is the following.

$$F(X) = \sum_{i=0}^{n} x_i \times X^i$$

Example 5. The greatest fixpoint of $X \mapsto \mathbf{1} + a \times X$, as a function on sets, is the set of potentially infinite lists and its least fixpoint is the set of finite lists. Likewise, the greatest fixpoint of $X \mapsto a \times X$ is the set of streams and its least fixpoint is just $\emptyset$.

Definition 8 ((Co)algebras of functors)**.** An *algebra* over a functor F is a particular set X and function $F(X) \to X$. Likewise, a *coalgebra* has $X \to F(X)$.

Example 6. Let $F(X) = 1 + X$. $\mathbb{N}$ with $\mathbf{1} + \mathbb{N} \xrightarrow[[0,\text{succ}]]{} \mathbb{N}$ is an (initial) algebra over F. On the other hand, $\mathbb{N}^\infty \triangleq \mathbb{N} \cup \{\infty\}$ with $\mathbb{N}^\infty \xrightarrow[\text{pred}]{} \mathbf{1} + \mathbb{N}^\infty$ is a (final) coalgebra over F.

The morphisms presented in the algebraic case look like *construction*, whereas in the coalgebraic case look like *destruction*, which suggests that codata privileges destructors.