Coalgebra Dynamics for Skolem Machine Computations

(*prospectus*)

John Fisher

jrfisher@cpp.edu

started: October 2010
last: September 11, 2018

Abstract. The Skolem machine is a Turing-complete machine model where the instructions are first-order formulas specified by a *colog* machine language for coherent logic. This note suggests some basic definitions for categories that can be associated with Skolem Machine computations, following some F-coalgebra patterns. We attemp to outline an intuitive *category dynamics* for Skolem machine computations based on model trees and branch morphisms. The issue of an appropriate category framework for model/proof duality is also mentions. These notes are really a prospectus for additional study.

1 Background information

The basic concepts regarding Skolem Machines is presented in the paper [2], and the concepts discussed in that paper form the foundation and motivation for the ideas presented in this note. There is a direct link to a corrected copy of that paper in the reference section. The reader of this note should consult that original paper for the basic definitions of Colog theories, Colog trees, Skolem machines, etc. Colog theories provide the machine code (coherent logic) for Skolem Machines and Colog trees are the object computed by the machine (proofs and models).

The category dynamics for coherent logic proposed here does not exactly correspond to the abstract pattern(s) for F-coalgebras and dynamics associated with state machines that are described in current literature. See the brief notes in the Appendix A regarding the category functor patterns or the examples in [1] to compare. We use the patterns as a guide but modify various details; adequacy seems not yet achieved.

2 Model trees associated with a coherent logic theory

A colog theory $\mathbf{C}$ is a finite sequence of coherent logic rules [2] (Sect. 1).

Definition 1. *The* signature *of colog theory* $\mathbf{C}$ *is* $\Sigma = \Sigma_{\mathbf{C}} = \Sigma_f \cup \Sigma_p \cup \Omega$, *consisting of functions* Σ_f *of* $\mathbf{C}$ *(including constants), predicates* Σ_p *of* $\mathbf{C}$ *and witness symbols* Ω. *Functions and predicates have* arity*, constants are functions of arity* 0*, witness symbols are constants of arity* 0 *and* $\Omega \cap \Sigma_f = \emptyset$. *It is assumed that* **true**, **false** *and* **goal** *all belong to* Σ_p, *even if these reserved predicate symbols do not actually appear in* $\mathbf{C}$.

Many colog theories have the same signatures. The equivalence of coherent logic theories will later be defined in terms of their signatures and on correspondences (semantics) of their model trees.

For example, consider the following colog theory $\mathbf{C}$.

```
true => p(f(X)), q(a).
p(A) => r(A) | s(A).
q(B) => v(B) | w(B).
r(A),v(B) => goal.
```

The signature of $\mathbf{C}$ consists of the constant $a/0$, the function $f/1$ and the predicates $p/1$, $q/1$, $r/1$, $v/1$, and $w/1$.

Definition 2. *A* model tree *over a signature is a multi-branching tree rooted at true whose branch sets consist of grounded instances of the predicates. The groundings can use the constants and functions of the signature or* witness *(new constant) symbols from* Ω.

Technically, there is supposed to be a fixed, countably infinite, set of possible witness symbols, but in the examples here we will just pick any convenient new symbols. For example, Fig.1 illustrates a model tree.

This model tree [1] is actually a colog (aka geolog) tree, as defined in [2], and was built using the rules of the theory. Notice that the witness $sk1$ was generated by an application of the first rule of the theory applied to the initial model tree $\{true\}$.

Model trees do **not** have to be built using the systematic application of the rules of a particular theory. For example, the model tree over the signature of $\mathbf{C}$ depicted in Fig.2 was not generate by the rules of $\mathbf{C}$, but it is still a model tree for the signature of the theory $\mathbf{C}$. (See Lemma 4 below.)

It is instructive to think about model trees as possible worlds that might or might not be supported by the original coherent theory (or some other coherent theory having the same signature). These possible worlds must only consider the signature of the theory and possible anonymous

[1] Displayed trees were automatically generated by the Colog prover [3]

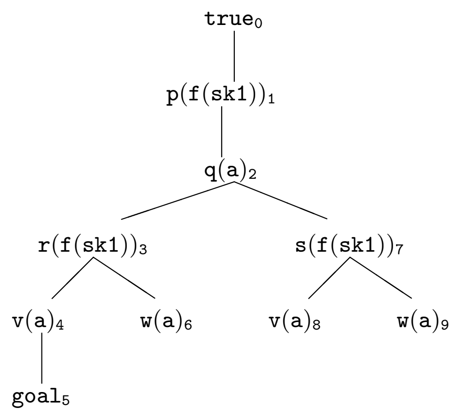

Fig. 1. Model tree T_1

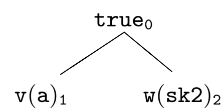

Fig. 2. Model tree T_2, but not colog tree for **C**

witnesses that could arise. The true facts in one of these worlds consists of the set of propositions in the branch. Thus, there will generally be abundantly many model trees that are not colog-theory generated trees. Of course, every **C** tree (colog tree built using rules of **C**) is a model tree.

Definition 3. *A model tree* morphism $m : T_a \to T_b$ *maps branches of* T_a *to branches of* T_b *using a mapping* w *of the witnesses occurring in* T_a *to the witnesses of* T_b *in such a way that if* B *is a branch of* T_a *then* $w(B) \subseteq m(B)$*, and distinct branches of* T_a *are mapped to distinct branches of* T_b*.*

Lemma 1. *If* $m_a : T_a \to T_b$ *uses witness map* w_a *and* $m_b : T_b \to T_c$ *uses witness map* w_b *then we have* $w_a \circ w_b(B) \subseteq m_a \circ m_b(B)$ *and distinct branches of* T_a *are morphed to distinct branches of* T_c*.*

Lemma 1 justifies the natural definition of composition for model tree morphisms via composition of witness maps and branch maps.

Since the branches of a model tree generated by a colog theory can be thought of as "cases", the requirement of Definition 3 that distinct branches of the source tree be mapped to distinct branches of the target tree can be viewed are a "separation of cases" requirement imposed on morphisms. Morphisms map cases to cases, subject to renaming of witnesses. (This supports a kind of topological analogy regarding how one tree could be "morphed" to part of another tree, and this intuition will be supported by dynamics defined in the next section.)

Definition 4. *The morphism $m : T_a \to T_b$ is* invertible *provided that there is a tree morphism $m' : T_b \to T_a$ such that the composition $m \circ m'$ is the identity branch map of T_a and the composition of the corresponding witness maps $w \circ w'$ maps witnesses of T_a to themselves.*

Definition 5. *Given two model trees T_a, T_b in $\mathbf{M}_\Sigma$ we say that T_a and T_b are* similar, *$T_a \approx T_b$, provided there is a morphism $m : T_a \to T_b$ which is invertible.*

The relation $\approx$ is an equivalence relation on the objects of $\mathbf{M}_\Sigma$. Fig. 3 shows a colog tree T_3 similar to the one in Fig.1. T_3 results from application of the rules of $\mathbf{C}$ in a different order than was the case for T_3. Thus, these trees should be similar. Careful inspections of these trees easily reveals that the branches of T_1 correspond to branches in T_3. Let B_{ij} refer to the jth branch of tree i, and let w be the map of witnesses $sk_1 \hookrightarrow sk_5$. Let m be the morphism: $m(B_{11}) = B_{21}$, $m(B_{12}) = B_{33}$, $m(B_{13}) = B_{32}$, $m(B_{14}) = B_{34}$. Then we have the inclusions $w(B_{ij}) \subseteq m(B_{ij})$ for $i = 1 \ldots 2$ and $j = 1 \ldots 3$. The morphism m is invertible and we thus have $T_1 \approx T_3$.

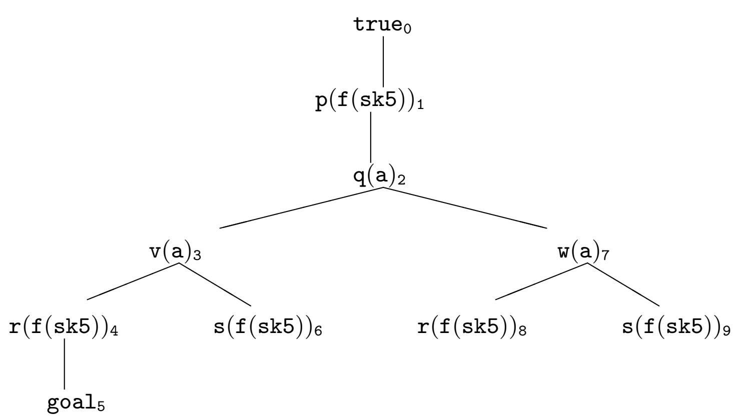

Fig. 3. Model tree T_3

Note that there are no morphisms between T_2 and either T_1 or T_3 because they represent different, exclusive, partial models. It can be tedious to explicitly enumerate morphisms for more complicated theories or corresponding model trees, but the intuitive idea is quite simple.

Trees T_1 and T_3 are both saturated by the example theory. Neither tree can be embedded in any larger colog tree, but both could be embedded in larger model trees: Consider arbitrarily expanding the signature an extend branches, or consider a larger colog theory containing the rules of the current theory. If we were to consider either rooted subtree consisting of nodes 0,1, and 2, then either subtree could be embedded in the corresponding saturated trees, but there would be no inverse morphism.

Lemma 2. *If model tree T_a has some branches extended producing model tree T_b then there is a natural morphism $i : T_a \to T_b$ injecting T_a into T_b.*

A *proof* tree is a model tree each of whose branches contains either **goal** or **false**. A *trimmed* proof tree is a proof tree such that each leaf is either **goal** or **false** and there are no other occurrences of **goal** or **false**.

Lemma 3. *For every proof tree T there is a unique trimmed proof tree T' and a morphism $T' \to T$.*

Proof. Trim the branches of T to the *highest* occurrence of **goal** or **false** in T to produce T', and consider the inclusion map of T' to T □

It might be more accurate to use a term like *potential* or *hypothetical* model tree rather than model tree since, for example, the tree in Fig.2 does not actually have a branch which is a model for theory **C**, but that tree would have branch models for another colog theory having the following rule: `true => v(a) | w(Z).`

Lemma 4. *Any model tree over a given signature can be built using some coherent theory associated with the signature.*

3 Categories of model trees and dynamics

Suppose that Σ is a coherent signature. We define a category $\mathbf{M}_\Sigma$ whose objects are the model trees having signature Σ, and whose morphisms are the model tree branch morphisms defined in the previous section.

The objects of $\mathbf{M}_\Sigma$ are also referred to as *states* or *model states* to emphasize that the trees are partial, or potential world models for the signature. The *final* states are the model trees which are proof trees.

Now let us suppose that **C** is a colog theory having signature Σ. In addition let us suppose that **C** has n rules. The rules of **C** act on objects of $\mathbf{M}_\Sigma$ using a natural transition function:

$$\delta : \mathbf{M}_{\Sigma} \times \mathbf{n} \rightarrow \mathbf{M}_{\Sigma} \tag{1}$$

If T is a model tree object of $\mathbf{M}_{\Sigma}$, and i is the index of a rule in $\mathbf{C}$, then

$$\delta(T, i) = T' \tag{2}$$

where T' is the model tree obtained by extending the branches of T using all applications of the *ith* rule of $\mathbf{C}$. We emphasize that the *ith* rule is *completely* applied to each branch of T, that is, all possible extensions are made. We also assume, as in [2], that an application makes no extension to any branch to which it does not apply or on which the rule is already fully satisfied.

As an illustration, let us suppose that we have the model tree shown in Fig.4.

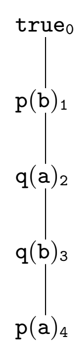

Fig. 4. T

and consider a full application of rule `p(X), q(X) => r(X)` to the tree, result shown in Fig.5. Notice that there is a morphism embedding tree T into tree T'.

If we curry the δ function we have a function $\mathbf{M}_{\Sigma} \rightarrow (\mathbf{n} \rightarrow \mathbf{M}_{\Sigma})$. This function is taken to be the *dynamics* that the rules of $\mathbf{C}$ impose on the category $\mathbf{M}_{\Sigma}$.

Definition 6. *An* abstract dynamics *for* $\mathbf{M}_{\Sigma}$ *consists of a finite index ordinal* $\mathbf{n}$ *and a function* $\alpha : \mathbf{M}_{\Sigma} \rightarrow (\mathbf{n} \rightarrow \mathbf{M}_{\Sigma})$*, such that for any morphism of model trees* m*, and any index* i *from* $\mathbf{n}$*, we have a commutative diagram as depicted in Fig.7, for some some model tree morphism* m_x*.*

The intention of the diagram in Fig.7 is to also impose the condition that the composite function $i \circ \alpha$ is a model tree morphism.

$$\begin{tikzcd}
\texttt{true}_0 \arrow[d, dash] \\
\texttt{p(b)}_1 \arrow[d, dash] \\
\texttt{q(a)}_2 \arrow[d, dash] \\
\texttt{q(b)}_3 \arrow[d, dash] \\
\texttt{p(a)}_4 \arrow[d, dash] \\
\texttt{r(b)}_5 \arrow[d, dash] \\
\texttt{r(a)}_6
\end{tikzcd}$$

Fig. 5. $T' = \delta(T, i)$

$$\begin{tikzcd}
T_1 \arrow[r, "m"] \arrow[d, "i \circ \alpha"'] & T_2 \arrow[d, "i \circ \alpha"] \\
T_1' \arrow[r, "\exists\, m_x"'] & T_2'
\end{tikzcd}$$

Fig. 6. Model tree extension m_x after dynamics application

An abstract dynamics only specifies a sequence of n actions produced for each object of $\mathbf{M}_{\Sigma}$. The definition abstracts away any mention of specific coherent logic rules. We wish to study how a coherent theory might be gleaned from such an abstract dynamics. The abstract dynamics amounts to a kind of "coherent" semantics for $\mathbf{M}_{\Sigma}$, but without specification of the rules.

Lemma 5. Suppose that $\mathbf{C}$ is a colog theory and δ is the state-transition function. Then the dynamics associated with δ satisfies the diagram requirement in the definition for abstract dynamics.

Proof. Suppose that we have a branch inclusion morphism $m : T_1 \to T_2$. Consider any branch B of T_1. The rule application $i \circ \alpha$ to T_1 is a model tree morphism which extends B using all possible satisfactions, and similarly for the branch B' in T_2 that corresponds to B under m. Thus, we also have $B_x \subseteq B'_x$, for the extended branches respectively. Since this holds for each branch B_x of T_1, we get a morphism $m : T'_1 \to T'_2$. □

We note that other possible definitions for Skolem machine transition δ might make things so that the Lemma would not hold. For example, consider the identity morphism $id : T \to T$ for the tree in Fig.4. Now suppose that the rule `p(X), q(X) => r(X)` were applied nondeterministically to T, say using the choice $(p(b)@1, q(b)@3)$ on the source tree, and using the choice $(p(a)@4, q(a)@2)$ for the target tree. Then the resulting trees could not be morphed. Even a deterministic definition for δ might not be sufficient. A similar counterexample to the one just described shows that defining δ as the earliest-first satisfaction of the rule (and only one application) would not be sufficient to obtain the Lemma[2].

If $\mathbf{C}$ is a colog theory, we can also consider the category $\mathbf{M_C}$ whose objects are only the model trees which are also colog trees built using $\mathbf{C}$. The morphisms of $\mathbf{M_C}$ are the morphisms between colog trees. Let $\mathbf{H} : \mathbf{M_C} \to \mathbf{M}_\Sigma$ be the inclusion functor. Notice that a Lemma 2 also holds in the category $\mathbf{M_C}$.

4 Similarity of coherent theories

Definition 7. *We say that colog theory* $\mathbf{C}_2$ covers *theory* $\mathbf{C}_1$*, or in symbols*

$$\mathbf{C}_1 \Rightarrow \mathbf{C}_2 \tag{3}$$

provided both theories have the same signature Σ, and for each colog tree T in $M_{\mathbf{C}_1}$ there is a similar tree T' in $M_{\mathbf{C}_2}$, i.e., $T \approx T'$.

Definition 8. *Define $\mathbf{C}_1 \Leftrightarrow \mathbf{C}_2$ to mean that both $\mathbf{C}_1 \Rightarrow \mathbf{C}_2$ and $\mathbf{C}_1 \Rightarrow \mathbf{C}_2$ hold, each theory covers the other.*

Lemma 6. If theory $\mathbf{C}_2$ is a reordering of $\mathbf{C}_1$ then $\mathbf{C}_1 \Leftrightarrow \mathbf{C}_2$

The relation $\Leftrightarrow$ is an equivalence relation for colog theories which have the same signature. A more general formulation could allow different signatures, such as *renamings* of constants, functions and predicates.

A *counter model* T in $\mathbf{M_C}$ is a model tree which is not a proof and has some branch with no extensions under the theory dynamics.

[2] Hint: consider the tree of Fig.4 and another tree with the branch propositions reordered; clearly each morphs to the other, but earliest-first rule application could produce distinct extensions.

Lemma 7. If $\mathbf{C}_1 \Leftrightarrow \mathbf{C}_2$ then proofs (resp. counter models) for one theory correspond to proofs (resp. counter models) for the other.

Proof. ... details ???

5 Colog theory representations for abstract dynamics

The natural question to ask would be: Is every abstract dynamics on the category $\mathbf{M}_\Sigma$ determined by some colog theory with the same signature?

First thoughts regarding this question suggest that a *geometric* theory $\mathbf{G}$ could be constructed by using grounded rules that represent all of the extensions $T \to (i \circ \alpha)(T)$ for all $T \in \mathbf{M}_\Sigma$. (Consult Fig.7.)

$$\mathbf{G} = \{A(T) \Rightarrow C((i \circ \alpha)(T)) \mid T \to (i \circ \alpha)(T)\} \tag{4}$$

where $A(-)$ are the branch antecedents and $C(-)$, the branch consequents, are determined by the added propositions in the extended tree specified by the dynamics. This is possibly an infinite geometric theory. Many of the rules might be characterized by the introduction of variables using some coherent rule. Disjunctions in consequents of rules arise from the branchings for the trees.

For example, let us create a small dynamics and then compile a colog theory from the dynamics so that the colog theory generates the same dynamics.

We seek some reasonable way to stipulate a finiteness or chaining condition that would result in a (finite) coherent theory.

6 Coherent topos with dynamics

... products of model trees, conjunction ?
coproducts of model trees, disjunction ?
how *dynamics* using topos versions ?

References

1. Jiri Adamek, *Introduction to coalgebra* `http://www.tac.mta.ca/tac/volumes/14/8/14-08abs.html`
2. John Fisher and Marc Bezem, Skolem Machines, *Fundamenta Informaticae*, 91 (1) 2009, pp.79-103. Copy with minor corections: `https://SkolemMachines.ORG/reports/SkolemMachines.pdf`
3. `https://SkolemMachines.org/`

4. John Fisher and Marc Bezem, Skolem Machines and Geometric Logic. In C.B. Jones, Z. Liu and J. Woodcock, *Proc. ICTAC 2007 The 4th International Colloquium on Theoretical Aspects of Computing*, Macao SAR, China, September 26-28, 2007. Springer LNCS vol. 4711, pp. 201-215.
5. John Fisher and Marc Bezem, Query Completeness of Skolem Machine Computations. In J. Durand-Losé and M. Margenstern, editors, *Proc. Machines, Computations and Universality '07*, Universite d'Orleans - LIFO, Orleans, France September 10-14, 2007. Springer LNCS vol. 4664, pp. 182-192.
6. https://en.wikipedia.org/wiki/F-coalgebra

A Prototypical F-Coalgebra Dynamics Patterns

Coalgebra dynamics are usually defined with reference to a simple F-coalgebra pattern, where F is a category functor. Attempts to extend the naive pattern requires stressful categorical acrobatics.

Suppose that $F : C \to C$ is a functor on category C. An *F-coalgebra* consists of a pair A, α where $\alpha : A \to FA$, such that if A, α and A, β are F-coalgebras and with $h : A \to B$ a morphism of C, then $Fh \circ \alpha = \beta \circ h$.

$$\begin{tikzcd} A \arrow[r, "h"] \arrow[d, "\alpha"'] & B \arrow[d, "\beta"] \\ FA \arrow[r, "Fh"'] & FB \end{tikzcd}$$

Fig. 7. Coalgebra morphism preservation

Assume that $C = Set$ (sets). For a transition machine depending on current_state and input_symbol we might use $FX = X^A$, where $A \in Set$ is set of inputs, and the coalgebra would be $\delta : X \to FX$, or in short curried form $\delta(current\_state, input\_symbol) = next\_state$. For inputs A and outputs B, a functor pattern would be $FX = (X \times B)^A$. Start and final states can be coded in various ways. The *instruction action*, *program* or *inference* of the coalgebra is the mapping $\delta : X \to A \to (X \times B)$.

It is a good exercise to express various Mealy and Moore machines in coalgebra forms – likewise, for Turing machines. Extending or adapting naive coalgebra patterns for more complicated machines requires some stressful categorical acrobatics. See the references in the Wikipedia entry[6].