Coalgebraic Determinization of Alternating Automata
M1 Internship Presentation

Meven Bertrand, supervision by Juriaan Rot

September 6 2017
Introduction

Context

Coalgebra: category theory for state based systems

My own motivation

- interest in category theory for some time
- course on coalgebra during the 1st semester
- wanted to go further

Only 1 month of internship due to course schedule

Why this subject?

Determinization: natural problem in coalgebra
Alternating automata: looks like a failure of theory → interesting to study
For me: reasonable background, interesting but not too large question
Introduction

Context

Coalgebra: category theory for state based systems

My own motivation

- interest in category theory for some time
- course on coalgebra during the 1st semester
- wanted to go further

Only 1 month of internship due to course schedule

Why this subject?

Determinization: natural problem in coalgebra
Alternating automata: looks like a failure of theory → interesting to study
For me: reasonable background, interesting but not too large question
Introduction

Context
Coalgebra: category theory for state based systems

My own motivation
- interest in category theory for some time
- course on coalgebra during the 1st semester
- wanted to go further

Only 1 month of internship due to course schedule

Why this subject?
Determinization: natural problem in coalgebra
Alternating automata: looks like a failure of theory → interesting to study
For me: reasonable background, interesting but not too large question
Plan

1. Coalgebra
2. Non-determinism
3. In Search For A Monad
Plan

1. Coalgebra
2. Non-determinism
3. In Search For A Monad
Examples

- stream: $Q \rightarrow \Gamma \times Q$
- automaton: $Q \rightarrow 2 \times Q^\Sigma$
- pushdown automaton: $Q \rightarrow 2 \times ((\Gamma^* \times Q)^\Gamma^*)^\Sigma$
- non-deterministic automaton: $Q \rightarrow 2 \times \mathcal{P}(Q)^\Sigma$
- Moore machine: $Q \rightarrow \Gamma \times Q^\Sigma$
What Is Really a State-Based System?

Two elements: states, and transitions (maybe with observable output).

Transitions

Simplest: state $\rightarrow$ state
Usual ingredients:
- acceptance: $2 \times (-)$
- generic output: $\Gamma \times (-)$
- reading letters: $(-)^\Sigma$
- branching: $\mathcal{P}(-)$
- side-effect: $(M \times (-))^M$

Many more
Examples

- stream: output = \(Q \rightarrow \Gamma \times Q\)
- automaton: acceptance + reading letter = \(Q \rightarrow 2 \times Q^\Sigma\)
- pushdown automaton: acceptance + reading letter + side-effect = \(Q \rightarrow 2 \times ((\Gamma^* \times Q)^\Sigma)^\Sigma\)
- non-deterministic automaton: acceptance + reading letter + branching = \(Q \rightarrow 2 \times \mathcal{P}(Q)^\Sigma\)
- Moore machine: output + reading letter = \(Q \rightarrow B \times Q^\Sigma\)
Why Category Theory?

All the ingredients are examples of **functors**. Functors are a central study subject of category theory, so category theory is a good tool to study state-based systems.

<table>
<thead>
<tr>
<th>State-based systems</th>
<th>Category theory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type of system</td>
<td>Functor $\mathcal{F}$</td>
</tr>
<tr>
<td>Particular instance</td>
<td>coalgebra $Q \rightarrow \mathcal{F}(Q)$</td>
</tr>
<tr>
<td></td>
<td>:</td>
</tr>
<tr>
<td></td>
<td>:</td>
</tr>
</tbody>
</table>
Plan

1. Coalgebra
2. Non-determinism
3. In Search For A Monad
Similar to non-deterministic automata, but with existential/universal alternation.

Coalgebra for $2 \times \mathcal{P}(\mathcal{P}(-))^A$. 
Other Examples of Non-Determinism

Simplest example: non-deterministic automaton, coalgebra for $2 \times \mathcal{P}(-)^A$.
Another example: probabilistic automaton, coalgebra for $2 \times \mathbb{P}(-)^A$. 
Monads For Non-Determinism

There is a common structure:

\[
\text{original functor } \mathcal{F} = \mathcal{G} \circ \mathcal{T} \quad \text{“machine” part} \quad \text{non-determinism}
\]

But \( \mathcal{T} \) has more structure, it is a monad:

**Monad**

A monad has three components:

- a functor \( \mathcal{T} \)
- a unit: collection of \( \eta_X : X \to \mathcal{T}(X) \)
- a multiplication: collection of \( \mu_X : \mathcal{T}(\mathcal{T}(X)) \to \mathcal{T}(X) \)

In this setting, there are nice theorems about transforming a coalgebra \( X \to \mathcal{G}(\mathcal{T}(X)) \) into a coalgebra \( \mathcal{T}(X) \to \mathcal{G}(\mathcal{T}(X)) \) (“determinization”).
Plan

1. Coalgebra
2. Non-determinism
3. In Search For A Monad
For these theorems to work, we need a monad structure for \( \mathcal{PP} \).

The functor \( \mathcal{P} \) is already a monad, so the natural way is to use monad composition.

The main ingredient is a distributive law:

**Distributive law**

A distributive law \( \lambda : T \rightarrow T' \Rightarrow T'T \) is a family of functions \( \lambda_X : T(T'(X)) \rightarrow T'(T(X)) \) respecting some axioms.

But for \( \mathcal{PP} \), the natural constructions fail.
The Failure on $\mathcal{PP}$

The intuitive distribution is

$$(x_1 \lor x_2) \land (y_1 \lor y_2 \lor y_3) \mapsto (x_1 \land y_1) \lor (x_1 \land y_2) \lor (x_1 \land y_3) \lor (x_2 \land y_1) \lor (x_2 \land y_2) \lor (x_2 \land y_3)$$

as formula:

$$\lambda_X : \mathcal{P}(\mathcal{P}(X)) \xrightarrow{S} \mathcal{P}(\mathcal{P}(X)) \quad \ni \quad \{V \subseteq \cup S \mid \forall U \in S, \text{ Card}(V \cap U) = 1\}$$

Changing it to

$$\lambda'_X : \mathcal{P}(\mathcal{P}(X)) \xrightarrow{S} \mathcal{P}(\mathcal{P}(X)) \quad \ni \quad \{V \subseteq \cup S \mid \forall U \in S, \text{ Card}(V \cap U) \geq 1\}$$

is better, but still not correct.
There is still something worth noting: $\lambda'_X (S) = \uparrow \lambda_X (S)$, so order plays a role. Hence, change from sets to ordered sets, and from $\mathcal{P}$ to $\mathcal{U} \mathcal{D}$.

With this change in type,

$$\lambda''_X : \mathcal{D} \mathcal{U} (X) \rightarrow \mathcal{U} \mathcal{D} (X)$$

$$S \mapsto \{ V \subseteq X | \forall U \in S, \text{Card}(V \cap U) \geq 1 \}$$

is a distributive law!

With a little trick to go from sets to ordered sets and back, the problem is solved:
It Works!

Semantics from this monad for $\langle o, \delta \rangle : Q \to 2 \times U Up Dn O(Q)^A$:

- $\text{behaviour}(q)(\varepsilon) = o(q)$
- $\text{behaviour}(q)(a \cdot w) = 1 \iff \exists F \in \delta(q)(a), \forall q' \in F, \text{behaviour}(q')(w) = 1$

As we want.
Conclusion

Interest

- nice and powerful framework for state-based systems
- showing the value of category theory

To do next?

- Study the possibility to give a distributive law for $\mathcal{PP}$, more comparison
- Add negation