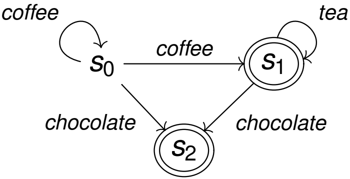
Determinization constructions: from automata to coalgebras

Alexandra Silva
joint work with F. Bonchi, M. Bonsangue and J. Rutten

ACG, November 2011

Motivation (by example)

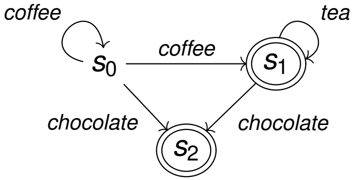
Non-deterministic automata



coffee	✓
chocolate	✓
coffee;chocolate	✓
coffee;tea	✓
coffee;coffee;tea	✓
chocolate;tea	×

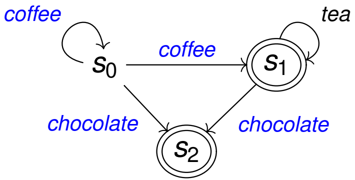
Motivation (by example)

Non-deterministic automata



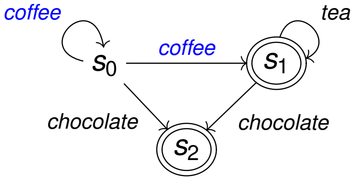
coffee	✓
chocolate	✓
coffee; *chocolate*	✓
coffee; *tea*	✓
coffee; *coffee*; *tea*	✓
chocolate; *tea*	✗

Non-deterministic automata

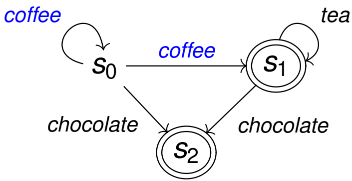


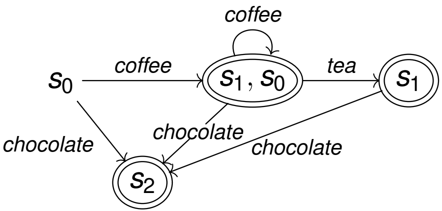
coffee	✓
chocolate	✓
coffee;chocolate	✓
coffee;tea	✓
coffee;coffee;tea	✓
chocolate;tea	×

Non-deterministic automata: how to eliminate *choice*?



Non-deterministic automata: how to eliminate *choice*?





The subset construction

Starting point: Non-deterministic automaton
$\mathcal{A} = (Q, \langle o, \delta \rangle \colon Q \to 2 \times \mathcal{P}(Q)^A)$

Goal: Deterministic automaton
$det(\mathcal{A}) = (\mathcal{P}(Q), \langle \hat{o}, \hat{\delta} \rangle \colon \mathcal{P}(Q) \to 2 \times \mathcal{P}(Q)^A)$

with the property

$L(Q) = \bigcup_{q \in Q} L(q)$ (this will be made more precise later)

The subset construction

Starting point: Non-deterministic automaton
$\mathcal{A} = (Q, \langle o, \delta \rangle \colon Q \to 2 \times \mathcal{P}(Q)^A)$

Goal: Deterministic automaton
$det(\mathcal{A}) = (\mathcal{P}(Q), \langle \hat{o}, \hat{\delta} \rangle \colon \mathcal{P}(Q) \to 2 \times \mathcal{P}(Q)^A)$

with the property

$L(Q) = \bigcup_{q \in Q} L(q)$ (this will be made more precise later)

The subset construction

Starting point: Non-deterministic automaton
$\mathcal{A} = (Q, \langle o, \delta \rangle \colon Q \to 2 \times \mathcal{P}(Q)^A)$

Goal: Deterministic automaton
$det(\mathcal{A}) = (\mathcal{P}(Q), \langle \hat{o}, \hat{\delta} \rangle \colon \mathcal{P}(Q) \to 2 \times \mathcal{P}(Q)^A)$

with the property

$L(Q) = \bigcup_{q \in Q} L(q)$ (this will be made more precise later)

The subset construction

Starting point: Non-deterministic automaton
$\mathcal{A} = (Q, \langle o, \delta \rangle \colon Q \to 2 \times \mathcal{P}(Q)^A)$

Goal: Deterministic automaton
$det(\mathcal{A}) = (\mathcal{P}(Q), \langle \hat{o}, \hat{\delta} \rangle \colon \mathcal{P}(Q) \to 2 \times \mathcal{P}(Q)^A)$

The maps $\hat{o}$ and $\hat{\delta}$ are defined as

$$\hat{o}(Y) = \begin{cases} 1 & \exists_{y \in Y} o(y) = 1 \\ 0 & \text{otherwise} \end{cases} \qquad \hat{\delta}(Y)(a) = \bigcup_{y \in Y} \delta(y)(a).$$

This construction guarantees that any word which labels a successful path in $\mathcal{A}$ also labels a successful path in $det(\mathcal{A})$ (and vice-versa).

The subset construction

Starting point: Non-deterministic automaton
$\mathcal{A} = (Q, \langle o, \delta \rangle \colon Q \to 2 \times \mathcal{P}(Q)^A)$

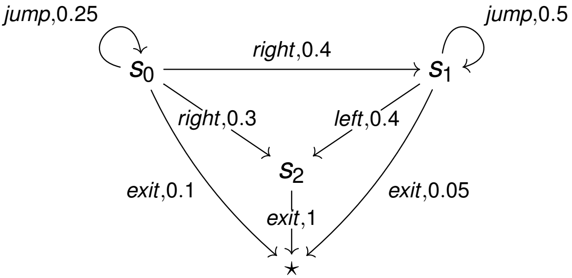
Goal: Deterministic automaton
$det(\mathcal{A}) = (\mathcal{P}(Q), \langle \hat{o}, \hat{\delta} \rangle \colon \mathcal{P}(Q) \to 2 \times \mathcal{P}(Q)^A)$

The maps $\hat{o}$ and $\hat{\delta}$ are defined as

$$\hat{o}(Y) = \begin{cases} 1 & \exists_{y \in Y} o(y) = 1 \\ 0 & \text{otherwise} \end{cases} \qquad \hat{\delta}(Y)(a) = \bigcup_{y \in Y} \delta(y)(a).$$

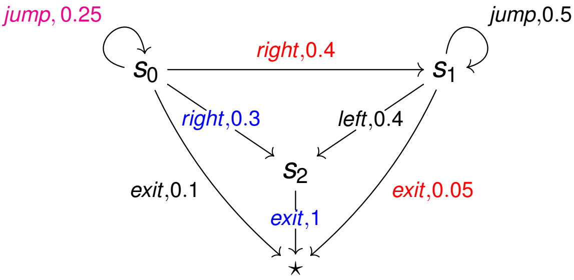
This construction guarantees that any word which labels a successful path in $\mathcal{A}$ also labels a successful path in $det(\mathcal{A})$ (and vice-versa).

Weighted automata



What is the probability that you jump, go right and then exit?

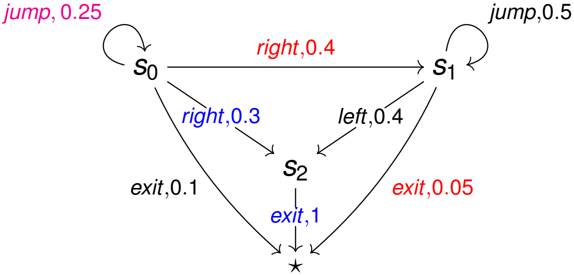
Weighted automata



What is the probability that you jump, go right and then exit?

Motivation (by another example)

Weighted automata

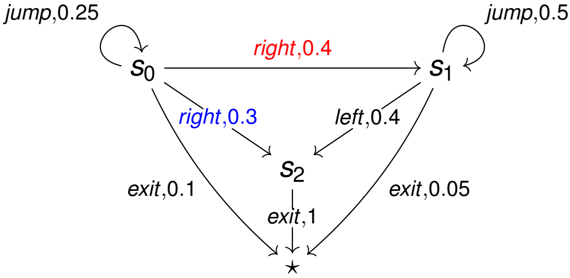


What is the probability that you jump, go right and then exit?

$$0.25\times0.4\times0.05+0.25\times0.3\times1 = 0.08$$

Motivation (by example)

Weighted automata: how to eliminate *choice*?

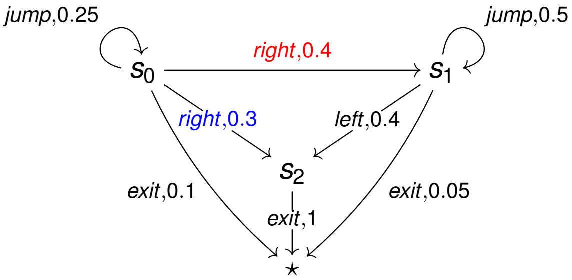


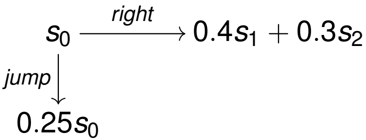
$$s_0 \xrightarrow{right} 0.4s_1 + 0.3s_2$$

$$s_0 \xrightarrow{jump} 0.25s_0$$

Motivation (by example)

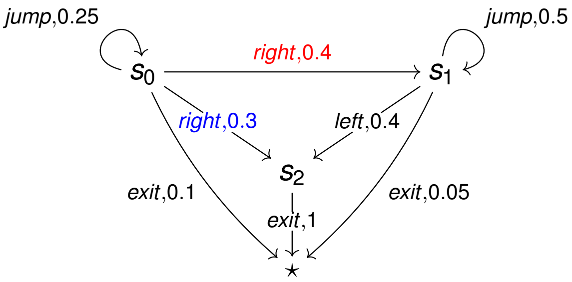
Weighted automata: how to eliminate *choice*?





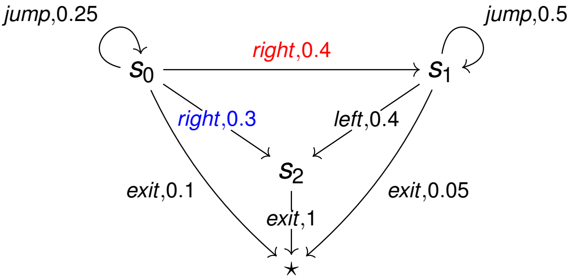
Motivation (by example)

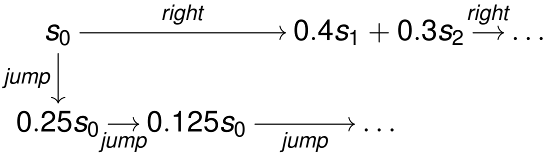
Weighted automata: how to eliminate *choice*?



Motivation (by example)

Weighted automata: how to eliminate *choice*?





The linearization construction

Starting point: weighted automaton $\mathcal{A} = (Q, \langle o, \delta \rangle \colon Q \to \mathbb{R} \times V(Q)^A)$, $V(S) = \mathbb{R} \to S =$ linear combinations of S.

Goal: Deterministic automaton
$lin(\mathcal{A}) = (V(Q), \langle \hat{o}, \hat{\delta} \rangle \colon V(Q) \to \mathbb{R} \times V(Q)^A)$

with the property

$L(rq_1 + sq_2) = rL(q_1) + sL(q_2)$ (this will be made more precise later)

The linearization construction

Starting point: weighted automaton $\mathcal{A} = (Q, \langle o, \delta \rangle \colon Q \to \mathbb{R} \times V(Q)^A)$,
$V(S) = \mathbb{R} \to S =$ linear combinations of S.

Goal: Deterministic automaton
$lin(\mathcal{A}) = (V(Q), \langle \hat{o}, \hat{\delta} \rangle \colon V(Q) \to \mathbb{R} \times V(Q)^A)$

with the property

$L(rq_1 + sq_2) = rL(q_1) + sL(q_2)$ (this will be made more precise later)

The linearization construction

Starting point: weighted automaton $\mathcal{A} = (Q, \langle o, \delta \rangle \colon Q \to \mathbb{R} \times V(Q)^A)$, $V(S) = \mathbb{R} \to S =$ linear combinations of S.

Goal: Deterministic automaton
$lin(\mathcal{A}) = (V(Q), \langle \hat{o}, \hat{\delta} \rangle \colon V(Q) \to \mathbb{R} \times V(Q)^A)$

with the property

$L(rq_1 + sq_2) = rL(q_1) + sL(q_2)$ (this will be made more precise later)

The linearization construction

Starting point: weighted automaton $\mathcal{A} = (Q, \langle o, \delta \rangle \colon Q \to \mathbb{R} \times V(Q)^A)$, $V(S) = \mathbb{R} \to S =$ linear combinations of S.

Goal: Deterministic automaton
$lin(\mathcal{A}) = (V(Q), \langle \hat{o}, \hat{\delta} \rangle \colon V(Q) \to \mathbb{R} \times V(Q)^A)$

The maps $\hat{o}$ and $\hat{\delta}$ are defined as

$$\hat{o}(r_1 q_1 + \ldots + r_n q_n) = r_1 o(q_1) + \ldots + r_n o(q_n)$$
$$\hat{\delta}(r_1 q_1 + \ldots + r_n q_n)(a) = r_1 \delta(q_1)(a) + \ldots + r_n \delta(q_n)(a)$$

This construction guarantees that the weight of any word which labels a path in $\mathcal{A}$ is the same of as the weight in $lin(\mathcal{A})$.

The linearization construction

Starting point: weighted automaton $\mathcal{A} = (Q, \langle o, \delta \rangle \colon Q \to \mathbb{R} \times V(Q)^A)$, $V(S) = \mathbb{R} \to S =$ linear combinations of S.

Goal: Deterministic automaton
$lin(\mathcal{A}) = (V(Q), \langle \hat{o}, \hat{\delta} \rangle \colon V(Q) \to \mathbb{R} \times V(Q)^A)$

The maps $\hat{o}$ and $\hat{\delta}$ are defined as

$$\hat{o}(r_1 q_1 + \ldots + r_n q_n) = r_1 o(q_1) + \ldots + r_n o(q_n)$$
$$\hat{\delta}(r_1 q_1 + \ldots + r_n q_n)(a) = r_1 \delta(q_1)(a) + \ldots + r_n \delta(q_n)(a)$$

This construction guarantees that the weight of any word which labels a path in $\mathcal{A}$ is the same of as the weight in $lin(\mathcal{A})$.

Summary

- In non-deterministic automata we go from a finite automaton to a finite deterministic automaton where states are sets of the original states.
- In weighted automata we go from a finite automaton to an infinite deterministic automaton where states are linear combinations of the original states.
- in both cases we go from a branching semantics (moment of choice) to a linear (or language) semantics.

Summary

- In non-deterministic automata we go from a finite automaton to a finite deterministic automaton where states are sets of the original states.
- In weighted automata we go from a finite automaton to an infinite deterministic automaton where states are linear combinations of the original states.
- in both cases we go from a branching semantics (moment of choice) to a linear (or language) semantics.

Summary

- In non-deterministic automata we go from a finite automaton to a finite deterministic automaton where states are sets of the original states.
- In weighted automata we go from a finite automaton to an infinite deterministic automaton where states are linear combinations of the original states.
- in both cases we go from a branching semantics (moment of choice) to a linear (or language) semantics.

What do these constructions have in common?

- First: what do NDA and WA have in common?

Non-deterministic automata: $(S, S \to 2 \times \mathcal{P}(S)^A)$.

Weighted automata: $(S, S \to \mathbb{R} \times V(S)^A)$.

What do these constructions have in common?

- First: what do NDA and WA have in common?

Non-deterministic automata: $(S, S \to 2 \times \mathcal{P}(S)^A)$.

Weighted automata: $(S, S \to \mathbb{R} \times V(S)^A)$.

.

What do these constructions have in common?

- First: what do NDA and WA have in common?

Non-deterministic automata: $(S, S \to 2 \times \mathcal{P}(S)^A)$.

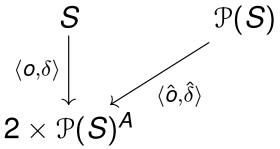
Weighted automata: $(S, S \to \mathbb{R} \times V(S)^A)$.

Coalgebras: $(S, S \to T(S))$.

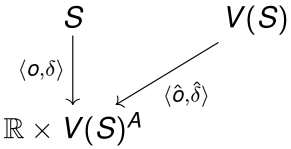
What do these constructions have in common?

- The constructions

Non-deterministic automata



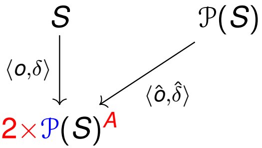
Weighted automata



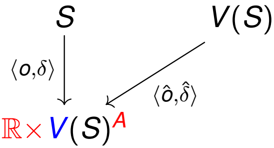
What do these constructions have in common?

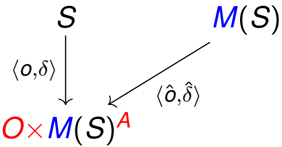
- The constructions

Non-deterministic automata

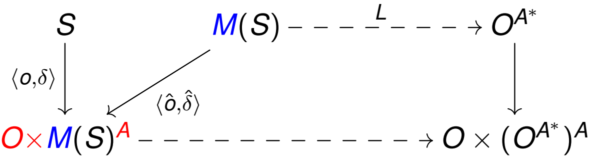


Weighted automata





Recall from Jan's talk: semantics by finality

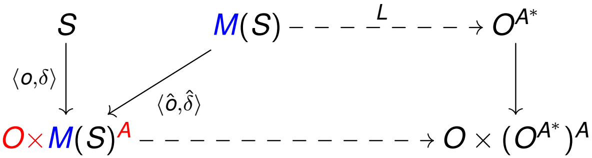


How to tie things together: construction and semantics?

$$L(Q) = \bigcup_{q \in Q} L(q) \qquad L(rq_1 + sq_2) = rL(q_1) + sL(q_2)$$

Language semantics

Recall from Jan's talk: semantics by finality

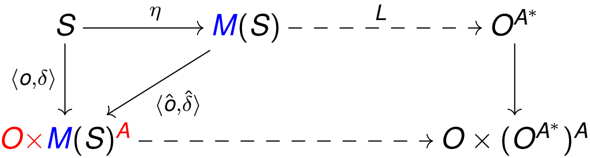


How to tie things together: construction and semantics?

$$L(Q) = \bigcup_{q \in Q} L(q) \qquad\qquad L(rq_1 + sq_2) = rL(q_1) + sL(q_2)$$

Language semantics

Recall from Jan's talk: semantics by finality   M is a monad



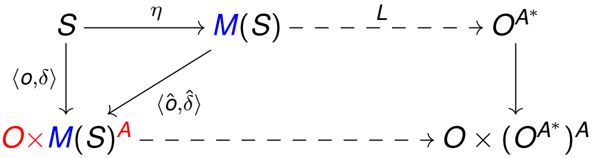
How to tie things together: construction and semantics?

$$L(Q) = \bigcup_{q \in Q} L(q) \qquad\qquad L(rq_1 + sq_2) = rL(q_1) + sL(q_2)$$

The commutativity of the diagram above is precisely these conditions!

Language semantics

Recall from Jan's talk: semantics by finality   M is a monad



How to tie things together: construction and semantics?

$$L(Q) = \bigcup_{q \in Q} L(q) \qquad\qquad L(rq_1 + sq_2) = rL(q_1) + sL(q_2)$$

The commutativity of the diagram above is precisely these conditions!

Now some abstract non-sense

For any monad M and functor F such that FTX is an algebra of the monad M (or: F as a lifting to Set^M) and F has a final coalgebra, we can generalize the construction and semantics:

$$
\begin{array}{ccccc}
S & \xrightarrow{\ \eta\ } & M(S) & \overset{L}{\dashrightarrow} & \Omega \\
{\scriptstyle \langle o,\delta \rangle}\downarrow & \swarrow {\scriptstyle \langle \hat{o},\hat{\delta} \rangle} & & & \downarrow \\
FM(S) & & \dashrightarrow & & F\Omega
\end{array}
$$

This gives rise to determinization constructions for many transitions systems: Mealy machines, structured Moore automata, Pushdown automata, . . .

Conclusions

What I hope you take home. . .

- Coalgebra is not only about semantics but also about algorithms
- Coalgebra is about unifying and generalizing