Radboud University Nijmegen

A coalgebraic perspective on minimization and determinization

Alexandra Silva
joint work with J. Adámek, F. Bonchi, M. Hülsbusch,
S. Milius, B. König

FoSSaCS, March 2012

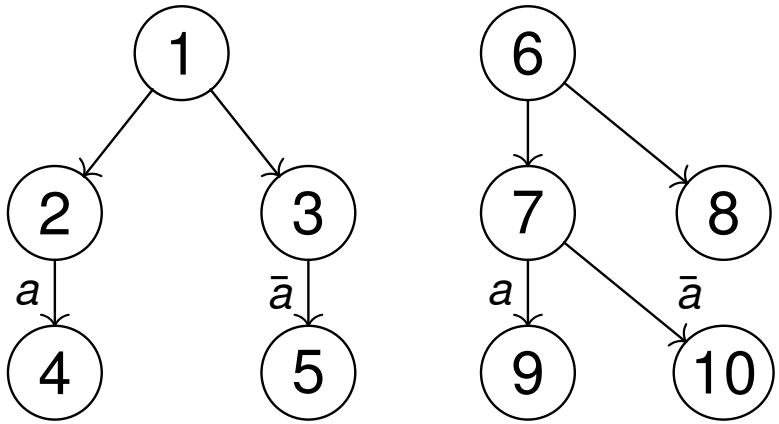
Motivation

- Automata are basic structures in computer science.
- Minimization of automata is not yet fully understood (cf. recent papers [Brzozowski 2011], [Panangaden et al. 2012], [Sakarovitch 2010]);
- Situation gets even more complicated for weighted automata, probabilistic automata, etc.

Motivation

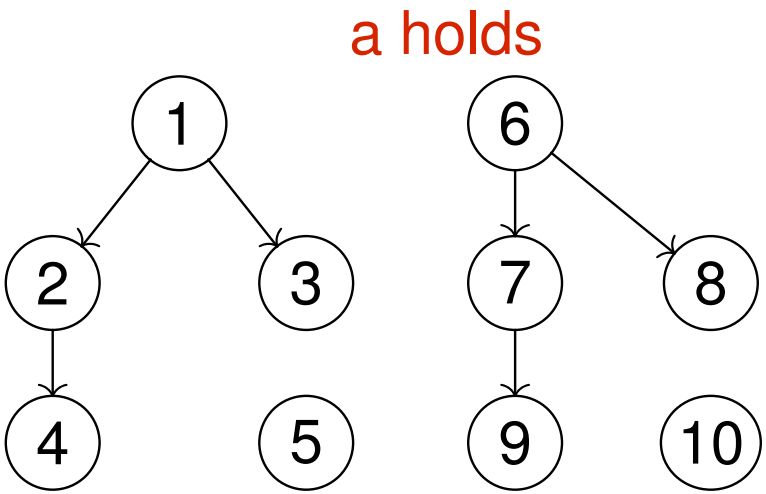
- This talk is about minimization and determinization of generalized labelled transition systems.
- We started from a very concrete example (next slide).
- Via a minimization and determinization construction we will be able to find canonical representatives of several different types of automata modulo an appropriate equivalence.

Example I: CTS



- A transition can only be taken if the guard is true.
- The environment can make one choice: either *a* holds or not.

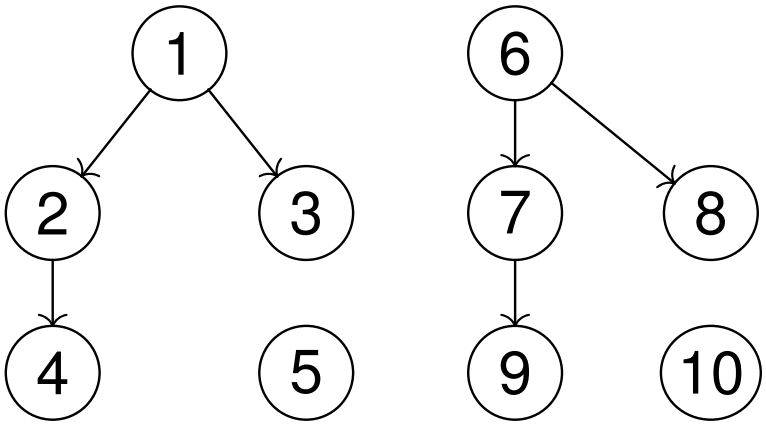
Example I: CTS



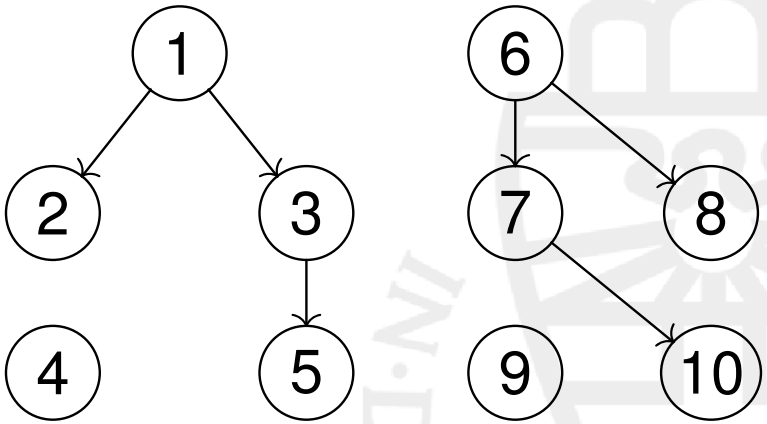
- A transition can only be taken if the guard is true.
- The environment can make one choice: either *a* holds or not.

Example I: CTS

a holds



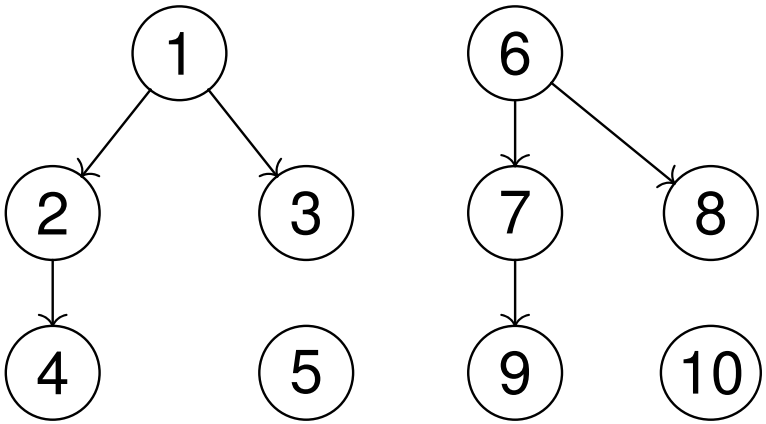
a does not hold



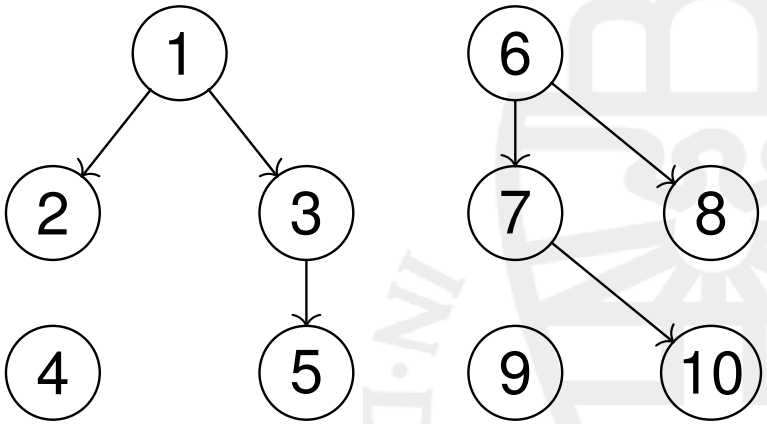
- A transition can only be taken if the guard is true.
- The environment can make one choice: either *a* holds or not.

Example I: CTS

a holds



a does not hold



- A transition can only be taken if the guard is true.
- The environment can make one choice: either *a* holds or not.
- In both cases states 1 and 6 are equivalent (bisimilar in the classical sense).

Example I: CTS

One possible way to solve the question whether two states in a CTS are equivalent:

1. Enumerate all conditions in the guards.
2. Create suitably many instantiations of the transition system.
3. Minimize the (possibly huge) resulting transition system with respect to bisimilarity.

Example II: NDA

This is analogous to the steps of determinization and minimization for non-deterministic automata.

1. Enumerate all possible subsets of the states.
2. Create a powerset automaton (determinization).
3. Minimize the (possibly huge) resulting deterministic automaton with respect to language equivalence.

In a nutshell

In this work...

- we study both constructions in a general setting ...
- and also show how they can be combined into a single algorithm.
- For CTSs this means an algorithm that checks if two states are bisimilar under all the possible conditions, without performing all the possible instantiations.

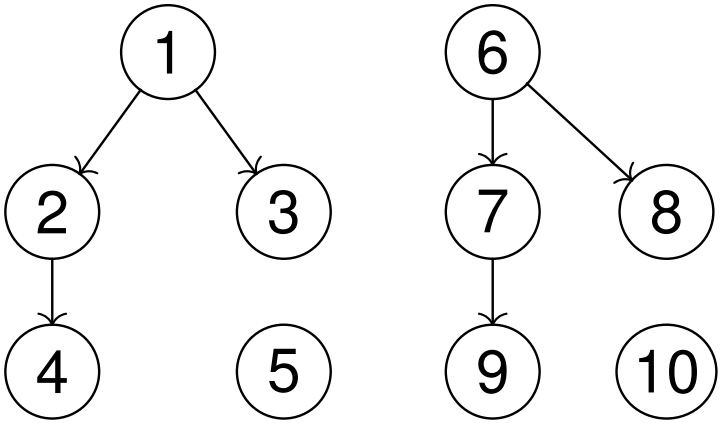
Canonical systems

First, an observation:

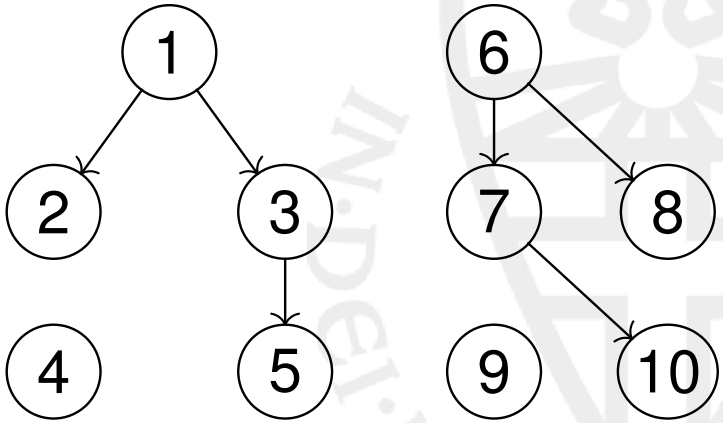
- In both cases above, one obtains a minimal system.
- But in which sense is this minimal system related with the original systems?
- Can we *map* the original systems into the minimal/canonical system?

Canonical systems

a holds

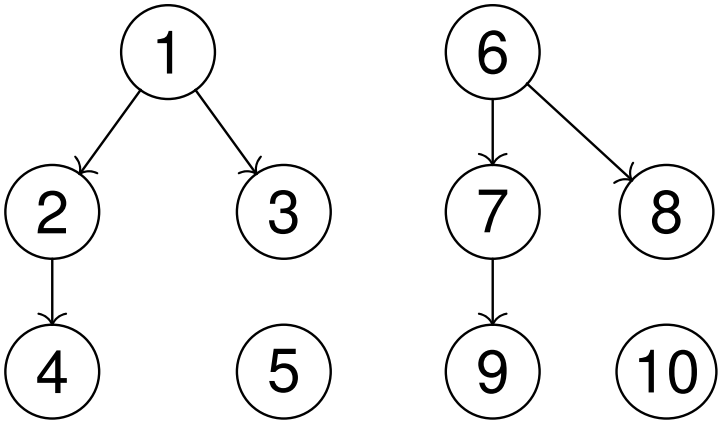


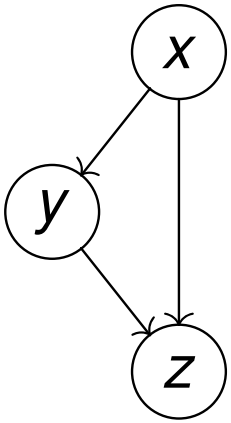
a does not hold



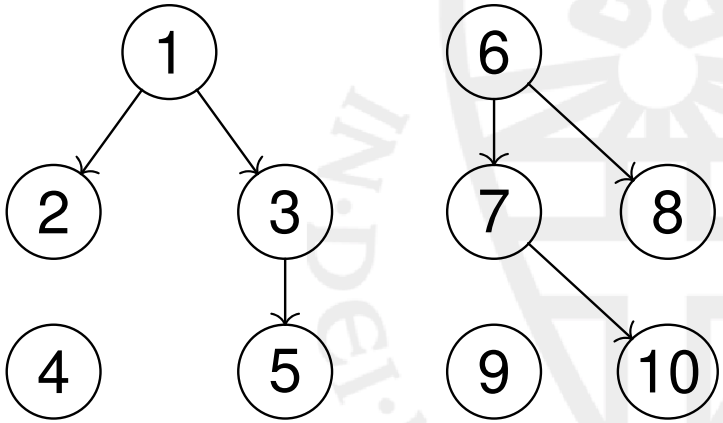
Canonical systems

a holds

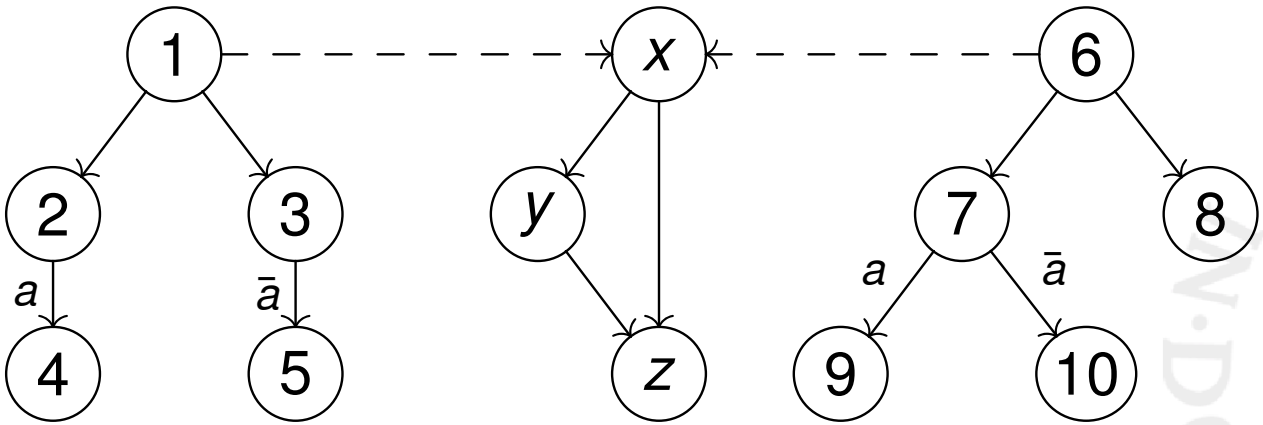




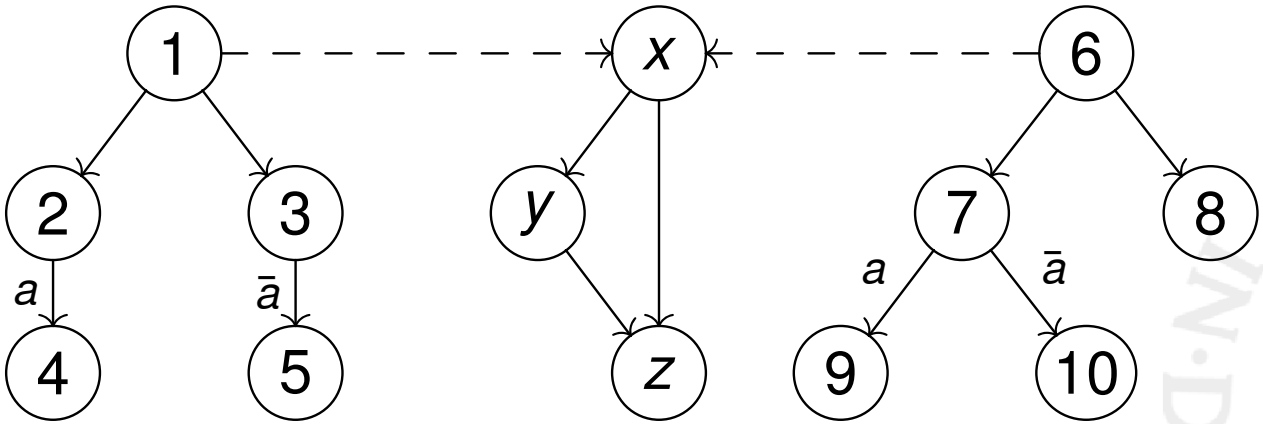
a does not hold



Canonical systems

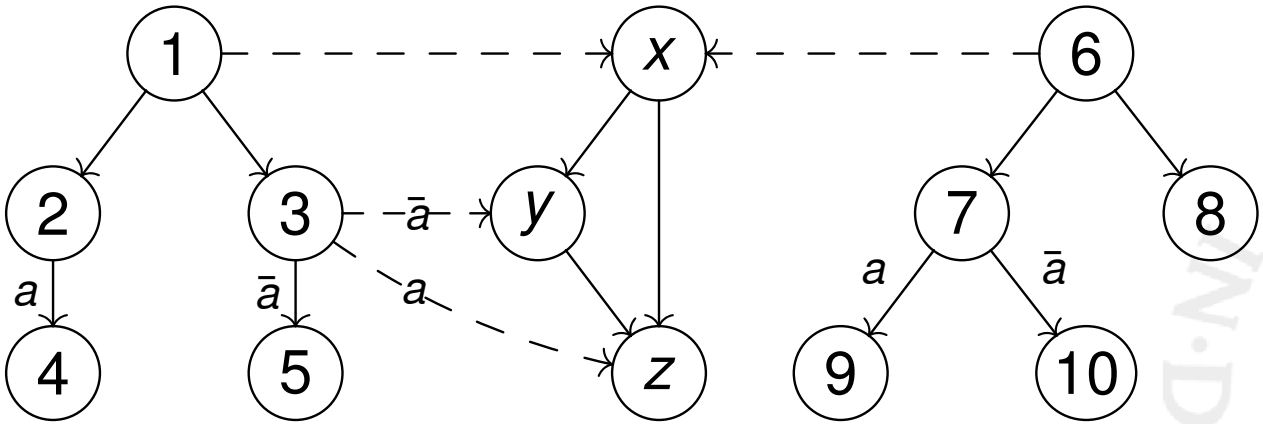


Canonical systems



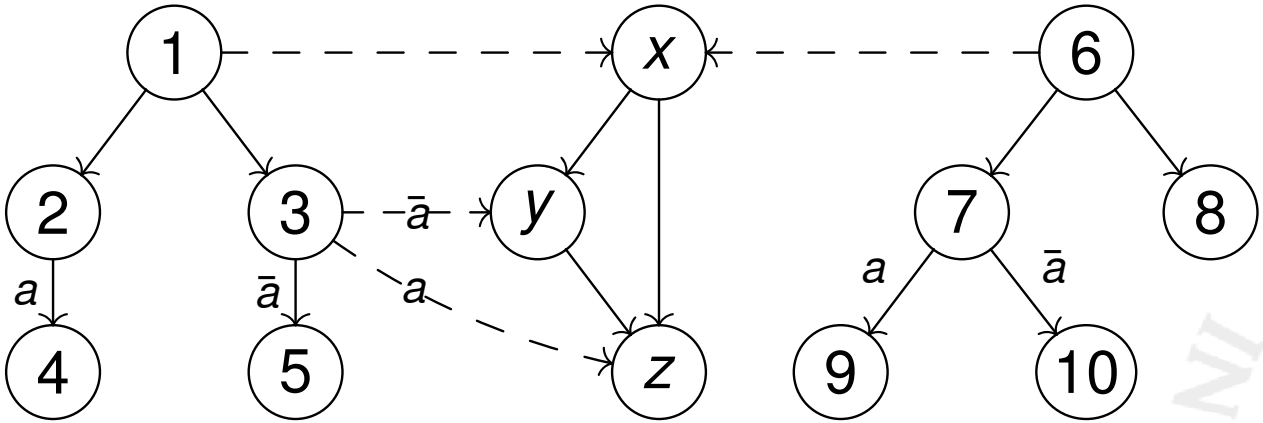
Where do we map 2 and 3?

Canonical systems



Where do we map 2 and 3?

Canonical systems



Where do we map 2 and 3?

We need to work in a setting where we can represent such conditional maps. As we will show in the sequel, by modeling CTS in an appropriate category this will be possible.

How do we do it?

- We use coalgebra.

How do we do it?

- We use coalgebra.
- Coalgebra is a general framework to study several types of transition systems in an uniform way.
- A coalgebra is a pair $(X, f\colon X \to FX)$, where F is the type of the transition system (formally, $F\colon \mathbf{C} \to \mathbf{C}$ is a functor in a category $\mathbf{C}$).

How do we do it?

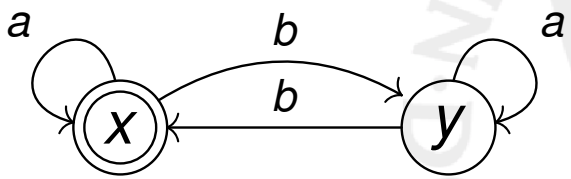
- We use coalgebra.
- Coalgebra is a general framework to study several types of transition systems in an uniform way.
- A coalgebra is a pair $(X, f\colon X \to FX)$, where F is the type of the transition system (formally, $F\colon \mathbf{C} \to \mathbf{C}$ is a functor in a category $\mathbf{C}$).

The power of F

- Behavioural equivalence (intuition: bisimilarity for LTS or language equivalence for DA)
- Universe of behaviors (final coalgebra, intuition: set of all languages)

How do we do it?

Prototypical example of a coalgebra: deterministic automata.
Modelled as $X \to 2 \times X^A$ (**C = Set**)

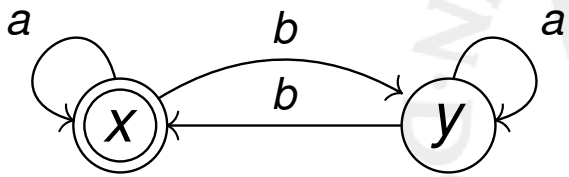


How do we do it?

Prototypical example of a coalgebra: deterministic automata.
Modelled as $X \to 2 \times X^A$ (**C = Set**)

Behaviours: Languages

$$\begin{array}{ccc} X & \xrightarrow{\quad beh \quad} & \mathcal{P}(A^*) \\ \downarrow & & \downarrow \\ 2 \times X^A & \longrightarrow & 2 \times \mathcal{P}(A^*)^A \end{array}$$

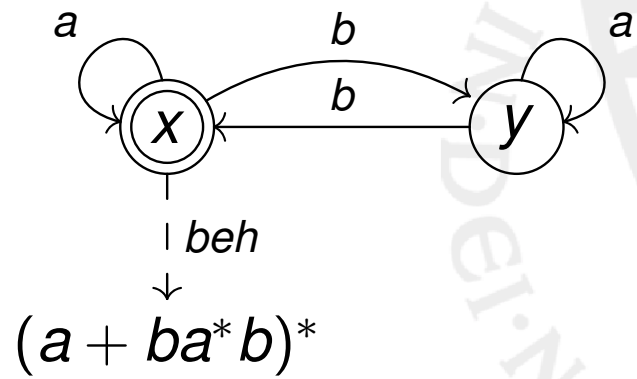


How do we do it?

Prototypical example of a coalgebra: deterministic automata.
Modelled as $X \to 2 \times X^A$ (**C = Set**)

Behaviours: Languages

$$\begin{array}{ccc} X & \xrightarrow{\quad beh \quad} & \mathcal{P}(A^*) \\ \downarrow & & \downarrow \\ 2 \times X^A & \longrightarrow & 2 \times \mathcal{P}(A^*)^A \end{array}$$



How do we do it?

How can we see CTSs and NDAs as coalgebras?

[Hasuo, Jacobs, Sokolova 2007] showed us the way!

- we model NDAs as functions $X \to \mathcal{P}(1 + A \times X)$, which are simply arrows in **Rel**. Behavioural equivalence is language equivalence.
- we model CTSs as functions $X \to (\mathcal{P}X)^A$ (in our example $A = \{a, \overline{a}\}$), which are arrows in $\mathcal{K}\ell((-)^A)$. Behavioural equivalence is bisimilarity after instantiation.

Coalgebras in Kleisli categories

Intuition: The non-deterministic or conditional branching is made implicit and – as a side-effect – ”hidden” underneath the monad.

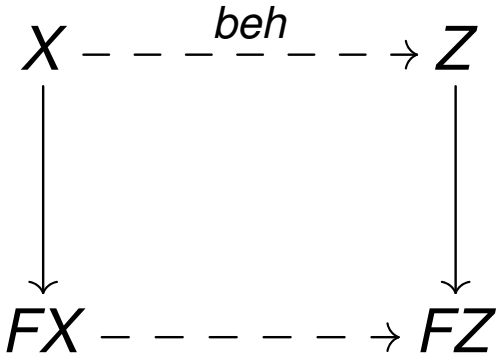
In Kleisli categories coalgebras are functions of the form

$$f : X \to TFX$$

where intuitively F (the endofunctor) describes the explicit branching and T (the monad) describes the implicit branching.

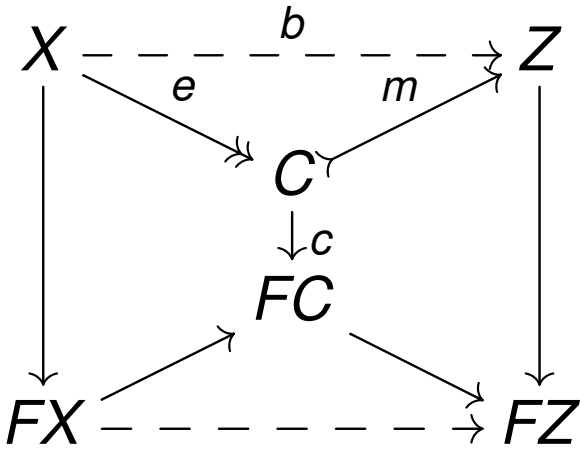
Minimization algorithm

- The image of an F-coalgebra (X, α) under the unique morphism is its minimal/canonical representative (C, c).



Minimization algorithm

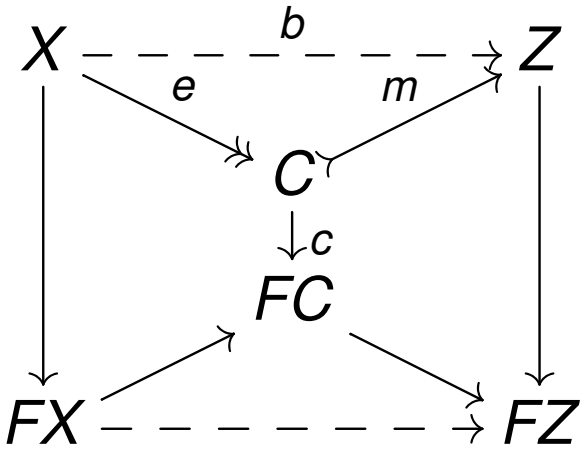
- The image of an F-coalgebra (X, α) under the unique morphism is its minimal/canonical representative (C, c).



- We can then factorize *beh* and obtain C.

Minimization algorithm

- The image of an F-coalgebra (X, α) under the unique morphism is its minimal/canonical representative (C, c).



- We can then factorize *beh* and obtain C.
- In the paper, we give an explicit construction of the minimal representative (similar to partition refinement).

Minimization algorithm

Can we apply the algorithm to the examples in Kleisli categories? (CTS, NDA)

Not directly, since we have no suitable factorization structures in these categories. . .

This is related to the fact that there is no unique miniminal non-deterministic automaton.

Minimization algorithm

Can we apply the algorithm to the examples in Kleisli categories? (CTS, NDA)

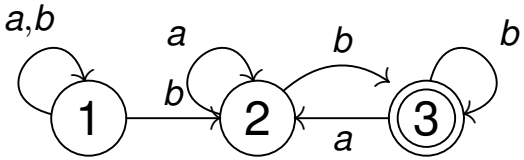
Not directly, since we have no suitable factorization structures in these categories...

This is related to the fact that there is no unique miniminal non-deterministic automaton.

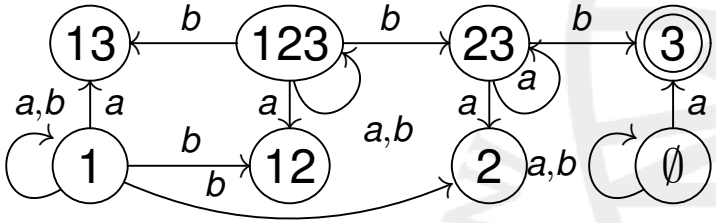
Idea: look for a suitable (reflective) subcategory with a nice factorization structure

Reflections for CTSs and NDAs

For NDA, reflection results in backwards-determinization:



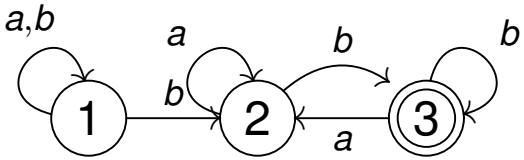
is transformed into



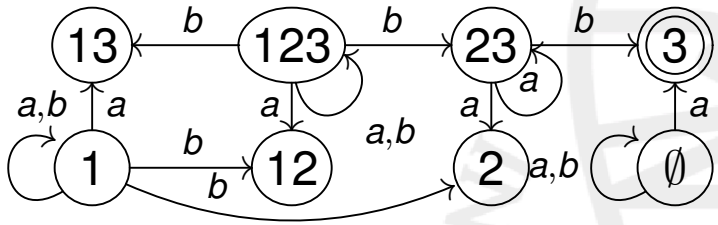
This is equivalent to reverse the arrows and then determinize the automaton. Precisely the first part of Brzozowski's algorithm to minimize NDAs.

Reflections for CTSs and NDAs

For NDA, reflection results in backwards-determinization:



is transformed into



This is equivalent to reverse the arrows and then determinize the automaton. Precisely the first part of Brzozowski's algorithm to minimize NDAs.

For CTS, we get precisely all the instantiations, which exactly the intuition to derive the right equivalence.

Another way...

- After the determinization (reflection), we can minimize as before.
- We explored how to combine both determinization and miniminization by defining pseudo-factorizations for categories with no suitable factorization structure (details in the paper).

Two constructions

Theorem

The following two constructions produce the same canonical/minimal representative for a given coalgebra α:

1. *Reflect α into the subcategory* **S** *and minimize with the factorization structure of* **S**.
2. *Minimize α directly using pseudo-factorizations.*

Two constructions

Theorem

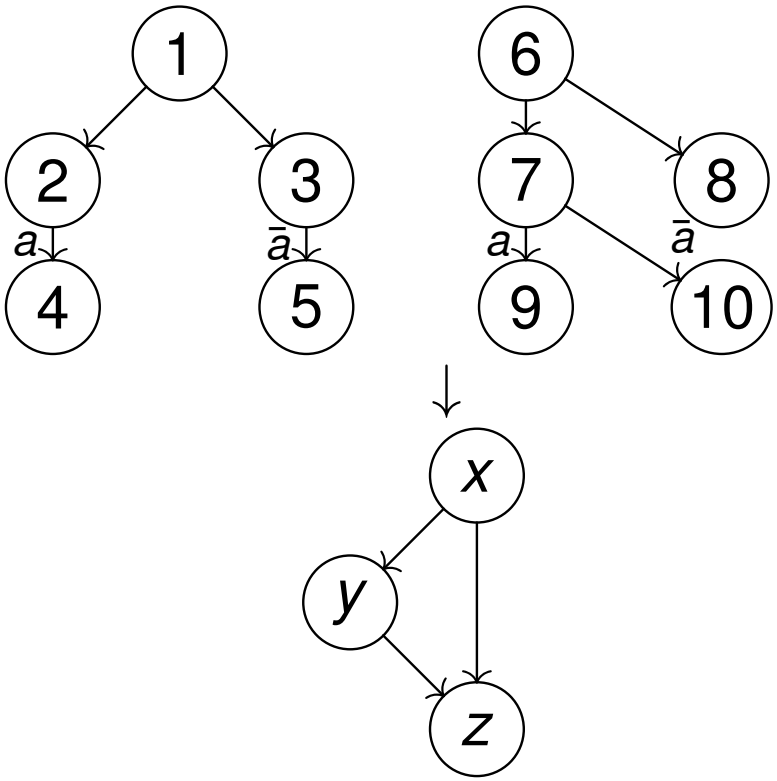
The following two constructions produce the same canonical/minimal representative for a given coalgebra α:

1. *Reflect α into the subcategory* **S** *and minimize with the factorization structure of* **S**.
2. *Minimize α directly using pseudo-factorizations.*

The second procedure above allows us to minimize CTS without having to explicitly construct all instantiations.

Pseudo-factorizations

Pseudo factorizations give us a combined determinization/minimization algorithm.



Conclusions

What I hope you take home...

- Minimization of automata is still keeping many people busy...
- Coalgebra is a suitable framework to derive algorithms to obtain canonical representatives of generalized transition systems uniformly.
- Wish to read the paper :-) where we have many more examples, including weighted systems and an interesting relationship to *átomata* [Brzozowski 2011].

Conclusions

What I hope you take home...

- Minimization of automata is still keeping many people busy...
- Coalgebra is a suitable framework to derive algorithms to obtain canonical representatives of generalized transition systems uniformly.
- Wish to read the paper :-) where we have many more examples, including weighted systems and an interesting relationship to *átomata* [Brzozowski 2011].

Thanks!